

Ali Kaveh
Kiarash Biabani Hamedani

Advanced Metaheuristic Algorithms and Their Applications in Structural Optimization

Studies in Computational Intelligence

Volume 1059

Series Editor

Janusz Kacprzyk, Polish Academy of Sciences, Warsaw, Poland

The series “Studies in Computational Intelligence” (SCI) publishes new developments and advances in the various areas of computational intelligence—quickly and with a high quality. The intent is to cover the theory, applications, and design methods of computational intelligence, as embedded in the fields of engineering, computer science, physics and life sciences, as well as the methodologies behind them. The series contains monographs, lecture notes and edited volumes in computational intelligence spanning the areas of neural networks, connectionist systems, genetic algorithms, evolutionary computation, artificial intelligence, cellular automata, self-organizing systems, soft computing, fuzzy systems, and hybrid intelligent systems. Of particular value to both the contributors and the readership are the short publication timeframe and the world-wide distribution, which enable both wide and rapid dissemination of research output.

Indexed by SCOPUS, DBLP, WTI Frankfurt eG, zbMATH, SCImago.

All books published in the series are submitted for consideration in Web of Science.

Ali Kaveh · Kiarash Biabani Hamedani

Advanced Metaheuristic Algorithms and Their Applications in Structural Optimization



Springer

Ali Kaveh
School of Civil Engineering
Iran University of Science and Technology
Tehran, Iran

Kiarash Biabani Hamedani
School of Civil Engineering
Iran University of Science and Technology
Tehran, Iran

ISSN 1860-949X ISSN 1860-9503 (electronic)
Studies in Computational Intelligence
ISBN 978-3-031-13428-9 ISBN 978-3-031-13429-6 (eBook)
<https://doi.org/10.1007/978-3-031-13429-6>

© The Editor(s) (if applicable) and The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

This work is subject to copyright. All rights are solely and exclusively licensed by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors, and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

The main purpose of the present book is to develop a general framework for population-based metaheuristic algorithms based on some basic concepts of set theory. The basic idea of the framework is to divide the population of individuals into a number of subpopulations of identical sizes. Then, in each iteration of the search process, different subpopulations independently explore the search space and do not communicate with each other. The division process is carried out in such a way that the diversity in each subpopulation is maintained automatically. Furthermore, subpopulations are close to each other in terms of their average fitness values. The reason for this is that individuals are fairly distributed among the subpopulations. Once an iteration is completed, subpopulations are merged to constitute the population of the next generation. However, before the next iteration starts, the population is redivided into subpopulations, and then the search process continues. In this way, a high diversity is maintained in the subpopulations throughout the search process. The main aim of the framework is to maintain an appropriate balance between global exploration and local exploitation abilities during the search process. It has been recognized for many years that a successful metaheuristic algorithm should be able to perform a wide exploration in the early stages and a deep exploitation during the final stages of the search process. The proposed framework makes it possible that different subpopulations independently explore the search space at the same time. As a result, in the early stages of the search process, the candidate solutions are scattered all over the search space instead of focusing on a small region. This guarantees that different regions of the search space are evenly explored and that the search is not limited to a small region, which significantly reduces the possibility of getting trapped in local optima and premature convergence. Therefore, exploration is favored in the early stages of the search process. In addition, because of the presence of different subpopulations, as the search process continues, the search is guided towards different promising regions of the search space rather than concentrating on the most promising region obtained so far. Therefore, exploitation is promoted during the final stages of the search process. The number of subpopulations can change with the number of iterations.

Chapter 1 explains the purpose of the book and provides an overview of the remaining chapters. In Chap. 2, the set-theoretical shuffled shepherd optimization algorithm is introduced and applied to the optimal design of reinforced concrete cantilever retaining walls. Chapter 3 introduces the set-theoretical variants of the teaching-learning-based optimization algorithm for structural optimization with frequency constraints. In Chap. 4, enhanced versions of the shuffled shepherd optimization algorithm are developed for structural optimization. In Chap. 5, a number of set-theoretical metaheuristic algorithms are applied to reliability-based design optimization of truss structures. In Chap. 6, optimal analysis is used in the service of frequency-constrained optimization of cyclic symmetric structures with set-theoretical Jaya algorithm. Discrete structural optimization with set-theoretical Jaya algorithm is discussed in Chap. 7. In Chap. 8, enhanced forensic-based investigation algorithm is introduced and its application to structural optimization with frequency constraints is examined. In Chap. 9, improved slime mould algorithm is developed for structural optimization with frequency constraints. Finally, in Chap. 10, improved arithmetic optimization algorithm is proposed for discrete structural optimization.

We would like to take this opportunity to acknowledge a deep sense of gratitude to a number of colleagues and friends who have helped us in different ways in the process of writing this book. Our special thanks are due to Dr. Thomas Ditzinger, the Editorial Director of Interdisciplinary and Applied Sciences and Engineering from Springer, for his constructive comments and suggestions during the preparation of this book. Our sincere appreciation is extended to our Springer colleagues who prepared the layout design of this book. We would also like to thank our colleagues, Dr. Mohammad Kamalinejad, Mr. Ataollah Zaerreza, and Mr. Ali Joudaki, for their contribution to our shared knowledge. Finally, we especially appreciate the support and patience of our wives, Mrs. L. Kaveh and Mrs. M. Bakhshian, during the preparation of this book.

We would like to thank the publishers who permitted some of our papers to be utilized in the preparation of this book, consisting of Springer, Elsevier, and Budapest University of Technology and Economics.

Every effort has been made to render this book error-free. However, the authors would appreciate any remaining errors being brought to his attention through their email addresses: alikaveh@iust.ac.ir (Ali Kaveh) and kiarashbiabani@yahoo.com (Kiarash Biabani Hamedani).

Tehran, Iran
June 2022

Ali Kaveh
Kiarash Biabani Hamedani

Contents

1	Introduction	1
1.1	Introduction	1
1.2	Organization of the Present Book	3
	References	6
2	Set-Theoretical Shuffled Shepherd Optimization Algorithm for Optimal Design of Reinforced Concrete Cantilever Retaining Wall Structures	9
2.1	Introduction	9
2.2	Shuffled Shepherd Optimization Algorithm (SSOA)	11
2.3	Set-Theoretical Shuffled Shepherd Optimization Algorithm (ST-SSOA)	12
2.4	Definition of the Optimization Problem	16
2.5	Analysis of Cantilever Retaining Walls	20
2.5.1	Active and Passive Earth Pressure Coefficients	20
2.5.2	Stability Analysis of Cantilever Retaining Walls	22
2.6	Results and Discussion	24
2.7	Concluding Remarks	40
	References	41
3	Set-Theoretical Variants of the Teaching–Learning-Based Optimization Algorithm for Structural Optimization with Frequency Constraints	43
3.1	Introduction	43
3.2	Teaching–Learning-Based Optimization (TLBO) Algorithm	45
3.3	Set-Theoretical Variants of the Teaching–Learning-Based Optimization Algorithm	47
3.3.1	Ordered Set-Theoretical Teaching–Learning-Based Optimization (OST-TLBO) Algorithm	48

3.3.2	Set-Theoretical Multi-Phase Teaching–Learning-Based Optimization (STMP-TLBO) Algorithm	49
3.4	Formulation of Truss Optimization Problem with Frequency Constraints	54
3.5	Numerical Examples	55
3.5.1	A 37-Bar Planar Truss	56
3.5.2	A 52-Bar Dome Truss	59
3.5.3	A 120-Bar Dome Truss	64
3.5.4	A 200-Bar Planar Truss	75
3.6	Concluding Remarks	80
	References	83
4	Enhanced Set-Theoretical Versions of the Shuffled Shepherd Optimization Algorithm for Structural Optimization	85
4.1	Introduction	85
4.2	Overview of the Shuffled Shepherd Optimization Algorithm (SSOA)	88
4.3	Parameter-Free Shuffled Shepherd Optimization Algorithm (PF-SSOA)	90
4.4	Set-Theoretical Multi-phase Shuffled Shepherd Optimization Algorithm (STMP-SSOA)	95
4.5	Formulation of the Optimization Problems	99
4.5.1	Size Optimization of Truss Structures with Frequency Constraints	99
4.5.2	Discrete Size Optimization of Steel Frame Structures	101
4.6	Numerical Examples	103
4.6.1	A 120-Bar Dome-Like Truss	104
4.6.2	A 200-Bar Planar Truss	108
4.6.3	A 3-Bay 15-Story Steel Frame Structure	114
4.6.4	A 3-Bay 24-Story Steel Frame Structure	122
4.7	Concluding Remarks	138
	References	139
5	Set-Theoretical Metaheuristic Algorithms for Reliability-Based Design Optimization of Truss Structures	141
5.1	Introduction	141
5.2	Set-Theoretical Variants of the Population-Based Optimization Algorithms	144
5.2.1	Set-Theoretical Variants of the Teaching–Learning-Based Optimization Algorithm	144
5.2.2	Set-Theoretical Variant of the Shuffled Shepherd Optimization Algorithm	148

5.2.3	Set-Theoretical Variant of the Jaya Algorithm	150
5.3	System Reliability Analysis of Truss Structures	151
5.3.1	Generation of Safety Margins for Truss Structures	153
5.3.2	The Branch and Bound Method	154
5.3.3	Evaluation of the System Reliability	156
5.4	System Reliability-Based Design Optimization of Truss Structures	156
5.5	Numerical Examples	157
5.5.1	Statically Indeterminate 16-Member Planar Truss	158
5.5.2	Statically Indeterminate 65-Member Truss Bridge	159
5.5.3	Statically Indeterminate 67-Member Truss Bridge	162
5.6	Concluding Remarks	164
	References	166
6	Optimal Analysis in the Service of Frequency-Constrained Structural Optimization with Set-Theoretical Jaya Algorithm	169
6.1	Introduction	169
6.2	Free Vibration Analysis of Structures	172
6.3	Efficient Free Vibration Analysis of Cyclic Symmetric Structures	173
6.3.1	Structural Matrices in the Cartesian and Cylindrical Coordinate Systems	174
6.3.2	Efficient Eigensolution Method for Free Vibration Analysis of Cyclic Symmetric Structures	178
6.4	Mathematical Formulation of the Optimization Problem	181
6.5	Optimization Algorithms	183
6.5.1	Jaya Algorithm	183
6.5.2	Set-Theoretical Jaya Algorithm	185
6.6	Results and Discussion	185
6.6.1	A 600-Bar Single-Layer Dome-Like Truss	187
6.6.2	A 1410-Bar Double-Layer Dome-Like Truss	189
6.7	Concluding Remarks	198
	References	200
7	Discrete Structural Optimization with Set-Theoretical Jaya Algorithm	203
7.1	Introduction	203
7.2	Structural Optimization with Discrete Design Variables	205
7.3	Classical Jaya Algorithm (JA)	207
7.4	Set-Theoretical Jaya Algorithm (ST-JA)	208
7.5	Numerical Examples	211
7.5.1	A 72-Bar Spatial Truss Structure	212
7.5.2	A 47-Bar Planar Power Line Tower	217
7.5.3	A 52-Bar Planar Truss Structure	223
7.5.4	A 160-Bar Spatial Truss Structure	228
7.6	Concluding Remarks	236
	References	242

8	Enhanced Forensic-Based Investigation Algorithm	245
8.1	Introduction	245
8.2	Forensic-Based Investigation (FBI) Algorithm	247
8.2.1	Forensic Investigation Process	247
8.2.2	Mathematical Model	248
8.3	Enhanced Forensic-Based Investigation (EFBI)	252
8.4	Formulation of the Optimization Problem	257
8.5	Numerical Examples	259
8.5.1	A 52-Bar Dome-Like Truss	260
8.5.2	A 120-Bar Dome-Like Truss	261
8.5.3	A 600-Bar Dome-Like Truss	265
8.6	Concluding Remarks	271
	References	276
9	Improved Slime Mould Algorithm	279
9.1	Introduction	279
9.2	Overview of the Slime Mould Algorithm (SMA)	282
9.3	Proposed Improved Slime Mould Algorithm (ISMA)	286
9.4	Formulation of the Optimization Problem	288
9.5	Numerical Examples	291
9.5.1	A 600-Bar Dome-Like Truss	292
9.5.2	A 1180-Bar Dome-Like Truss	299
9.5.3	A 1410-Bar Dome-Like Truss	304
9.6	Concluding Remarks	320
	References	321
10	Improved Arithmetic Optimization Algorithm	323
10.1	Introduction	323
10.2	Overview of the Arithmetic Optimization Algorithm (AOA)	325
10.2.1	Initialization Phase	326
10.2.2	Exploration Phase	327
10.2.3	Exploitation Phase	328
10.3	Proposed Improved Arithmetic Optimization Algorithm (IAOA)	329
10.4	Structural Optimization with Discrete Design Variables	333
10.5	Numerical Examples	335
10.5.1	A 72-Bar Space Truss	335
10.5.2	A 384-Bar Double-Layer Barrel Vault	340
10.5.3	A 3-Bay 15-Story Steel Frame	351
10.6	Application to High-Dimensional Structural Optimization Problems	354
10.7	Concluding Remarks	358
	References	360

Chapter 1

Introduction



1.1 Introduction

Nature-inspired metaheuristic algorithms have gained considerable popularity in recent years in a wide range of engineering applications [1–3]. This is due to the fact that these algorithms: (1) do not require gradient information; (2) are based on relatively simple concepts and are easy to be implemented; (3) are able to escape from local optima; and (4) can be utilized in a wide range of problems from various disciplines [4]. In the last few decades, tremendous research efforts have been devoted to the development of new metaheuristics [5]. Meanwhile, many studies have focused on the improvement of existing metaheuristics [6]. Additionally, hybridization of metaheuristics algorithms is currently an interesting field of research that offers the opportunity to develop very powerful metaheuristic algorithms [7]. Despite significant growth in research on metaheuristic algorithms, there are still many important questions and challenges to be addressed. The most important problem seems to be the gap between theory and practice. Indeed, although metaheuristic algorithms have effectively tackled a wide variety of complex real-world optimization problems, but mathematical analysis of basic aspects of these algorithms is still an ongoing research topic [8]. For example, convergence analysis has not been carried out yet for the majority of metaheuristic algorithms. As another example, achieving the optimal balance between exploration and exploitation is an extremely challenging task and still remains an open problem. In addition, the proper tuning of the algorithm-specific parameters is usually a tedious and problem-dependent task.

Based on the source of inspiration, nature-inspired metaheuristic algorithms can be categorized into four main groups: evolution-based, physics-based, swarm-based, and human-based [4]. Although these algorithms draw their inspiration from a wide variety of natural phenomena or behaviors, they all should have two major components, namely exploration (diversification) and exploitation (intensification) [9]. Exploration refers to the ability of a metaheuristic algorithm to explore the search space on a global scale, and thus it makes it possible to search for new solutions far

from the current ones [8]. In this way, the diversity of the population is maintained during the search process. The advantage of exploration is that it is less prone to be trapped in local optima, and the global optimum can be more likely approached. However, its disadvantages are slow convergence rate and the waste of computational resources. This is due to the fact that many generated solutions may be quite far from the global optimum. On the other hand, exploitation refers to the ability of a metaheuristic algorithm to intensify the search in promising regions of the search space. The advantage of exploitation is that it usually leads to high convergence rates. However, it may easily get trapped in local optima. It has been recognized for many years that an appropriate balance between exploration and exploitation is necessary for the efficient and effective operation of any metaheuristic algorithm. Too much exploration and too little exploitation mean the search process is more like a random search, which can significantly slow down the convergence rate, and even lead to divergence. On the other hand, too much exploitation too little exploration may cause the algorithm to be trapped in local optima, which makes it very difficult or even impossible to find the global optimum. In such a situation, the algorithm behaves like a local search operator. Therefore, the trade-off between exploration and exploitation is critical for good performance of metaheuristic algorithms. However, as mentioned earlier, achieving such a balance is not an easy task. In fact, we are faced with a hyper-optimization problem (i.e., the optimization of an optimization algorithm). Such a balance may depend on many factors such as the optimization problem at hand, the working mechanism of the algorithm, its setting of parameters, tuning and control of these parameters, etc. [8].

The main purpose of the present book is to develop a general framework for population-based metaheuristic algorithms based on some basic concepts of set theory. The basic idea behind this framework is to divide the population of candidate solutions into a predefined number of smaller subpopulations of the same size. Therefore, in each iteration of the search process, different subpopulations independently explore the search space and do not communicate with each other. The division is carried out in such a way that diversity is maintained in the subpopulations. It is expected that subpopulations are close to each other in terms of their average fitness values. This is due to the fact that candidate solutions are fairly distributed among the subpopulations. After each iteration is completed, the subpopulations are merged into a single population to form the population of the next generation. However, at the beginning of the next iteration, the population is again divided into smaller subpopulations, and then the search process continues. This aims to keep a high diversity in the subpopulations throughout the search process. The main objective of the framework is to provide a fine balance between exploration and exploitation during the search process. It is desired for metaheuristic algorithms to favor exploration in the early stages and exploitation during the final stages of the search process. Following the proposed framework, different subpopulations explore the search space simultaneously but independently. As a result, during the first iterations of the search process, the candidate solutions are scattered over the whole search space instead of focusing on a specific region. This ensures that all regions of the search space are

evenly explored and that the search is not confined to a limited region, which significantly reduces the possibility of being trapped in local optima and premature convergence. Therefore, exploration is promoted in the early stages of the search process. In addition, because of the presence of multiple subpopulations, as the search process progresses, the search is directed towards different promising regions of the search space rather than concentrating on the most promising region found so far. Therefore, exploitation is promoted during the final stages of the search process. The framework is of a general nature and can be applied to almost all population-based metaheuristic algorithms. However, based on the description given above, the framework seems to be best suited for population-based metaheuristics where the population is guided towards the best search agent. Examples of such metaheuristics are teaching-learning-based optimization (TLBO) algorithm, Jaya algorithm (JA), and particle swarm optimization (PSO). In teacher phase of the standard TLBO, the best student in the class is considered as the teacher who shares his or her knowledge with other students in the class. The basic principle of the classical JA is to approach the best solution and move away from the worst solution. In the conventional PSO, each particle moves towards its best previous position and towards the best particle in the swarm. It should be noted that the proposed framework makes it possible to design different variants of a population-based metaheuristic. For example, instead of using a predefined number of subpopulations, the number of subpopulations can change with the number of iterations.

1.2 Organization of the Present Book

The remaining chapters of the book are organized into two major sections. The first section entitled “**Set-Theoretical Framework for Population-Based Metaheuristic Algorithms**” contains six chapters which deal with the development of the general set-theoretical framework for population-based metaheuristic algorithms. The framework is applied to a number of well-known population-based metaheuristic algorithms such as shuffled shepherd optimization algorithm (SSOA), teaching-learning-based optimization (TLBO) algorithm, Jaya algorithm (JA), etc. Finally, to demonstrate the applicability and effectiveness of the proposed framework, the developed metaheuristics are applied to a wide variety of structural optimization problems including the cost optimization of reinforced concrete cantilever retaining walls, optimum design of truss structures with frequency constraints, optimum design of frame structures, reliability-based design optimization of truss structures, and discrete optimization of truss structures. In the second section entitled “**Algorithm-Specific Modifications of Metaheuristic Algorithms**”, some algorithm-specific modifications are suggested to the standard versions of three state-of-the-art metaheuristic optimization algorithms, namely forensic-based investigation (FBI), arithmetic optimization algorithm (AOA) and slime mould algorithm

(SMA), aiming to overcome their inherent drawbacks. The efficiency and effectiveness of the proposed metaheuristics are examined through some benchmark structural optimization problems. The remaining chapters are structured as follows:

Chapter 2 employs some basic concepts of set theory to generalize the description of the shuffled shepherd optimization algorithm (SSOA). As a result, the set-theoretical shuffled shepherd optimization algorithm (ST-SSOA) is introduced. The set-theoretical framework suggested for the description of ST-SSOA is of a general nature, i.e., it can be applied to almost all population-based metaheuristics. The main concept of the set-theoretical framework is to divide the set of candidate solutions of a population-based metaheuristic into a number of smaller subsets having the same number of candidate solutions. To demonstrate the effectiveness and efficiency of ST-SSOA, it is employed to solve the cost optimization problem of reinforced concrete cantilever retaining walls under static and seismic loading conditions, and the results are compared with several recently developed metaheuristic algorithms.

Chapter 3 extends the application of the set-theoretical framework described in the previous chapter to the teaching-learning-based optimization (TLBO) algorithm. As a result, two different set-theoretical versions of TLBO, namely ordered set-theoretical TLBO (OST-TLBO) and set-theoretical multi-phase TLBO (STMP-TLBO), are proposed. The main idea behind these two improved versions of TLBO is to divide the class of learners into a number of specific subclasses having the same number of learners, aiming to improve the exploration and exploitation abilities of the standard TLBO and make a proper balance between them. In order to examine the performance of the proposed framework, OST-TLBO and STMP-TLBO are applied on various benchmark truss optimization problems with frequency constraints, and the results are compared with those of the standard TLBO and some other optimization methods in the literature.

Chapter 4 first deals with the development of a simplified version of the shuffled shepherd optimization algorithm (SSOA), namely parameter-free SSOA (PF-SSOA). As compared to the classical SSOA, the proposed PF-SSOA requires fewer algorithm-specific parameters, and thus it is easier to be implemented. Subsequently, a multi-phase version of PF-SSOA, namely set-theoretical multi-phase SSOA (STMP-SSOA), is developed. The main idea behind the proposed STMP-SSOA is to divide the optimization process into different phases, in each of which a self-contained PF-SSOA is executed with a different number of subpopulations. The number of subpopulations decreases as the optimization process proceeds. The decreasing rate of the number of subpopulations allows a wide exploration of the search space at the beginning of the search process, and deeper exploitation of the promising regions of the search space at the end of the search process. Finally, the efficiency and effectiveness of the proposed algorithms are demonstrated through four engineering problems of sizing optimization of skeletal structures. The results obtained by the proposed methods are compared with those of the classical SSOA and some other metaheuristic algorithms in the literature.

Chapter 5 presents the application of four set-theoretical metaheuristic algorithms to system reliability-based design optimization (SRBDO) of truss structures. The four

metaheuristic algorithms used in this study are ordered set-theoretical TLBO (OST-TLBO) and set-theoretical multi-phase TLBO (STMP-TLBO) algorithms, which were presented in Chap. 3, set-theoretical multi-phase SSOA (STMP-SSOA), which was presented in Chap. 4, and set-theoretical Jaya algorithm (ST-JA), which will be presented in this chapter. In the SRBDO approach proposed in this study, the branch-and-bound method is employed to identify the dominant failure modes of structural systems, and then to evaluate the system reliability using dominant failure modes. Additionally, the first-order second-moment (FOSM) method is applied to calculate the failure probabilities. Finally, the applicability and efficiency of the proposed SRDBO approach is demonstrated through three truss optimization problems with system reliability constraints.

Chapter 6 employs an efficient method of free vibration analysis of cyclic symmetric structures for the optimal design of large-scale dome truss structures with cyclic symmetry subject to frequency constraints. Through the present method, by applying a block-diagonalization technique, the initial free-vibration eigenvalue problem is decomposed into a number of independent sub-eigenproblems with significantly smaller dimensions, and thus the computational time required to solve the problem is reduced dramatically. In order to perform the optimization task, the set-theoretical Jaya algorithm (ST-JA) which was presented in the previous chapter is used. Finally, two large-scale dome optimization problems with frequency constraints are presented to demonstrate the efficiency and effectiveness of the proposed method. The results show that ST-JA can improve both the global exploration and local exploitation abilities of the original JA.

Chapter 7 proposes a discrete version of the set-theoretical Jaya algorithm (ST-JA) to solve structural optimization problems with discrete search spaces. The classical Jaya algorithm (JA) is based on the simple concept that search agents try to improve their positions by moving towards the best search agent and getting away from the worst one. However, in ST-JA, which is a multi-population variant of the classical JA, each subpopulation has its own specific best and worst search agents. This promotes the global exploration in the initial stages of optimization and local exploitation in the later stages of optimization. Four benchmark truss optimization problems with discrete design variables are presented to examine the performance of the proposed ST-JA. The results obtained by ST-JA are compared with those of the classical JA and some other state-of-the-art optimization methods existing in the literature.

Chapter 8 deals with the development of an enhanced version of the forensic-based investigation (FBI) algorithm, called enhanced forensic-based investigation (EFBI) algorithm, and its application to optimal design of dome truss structures with multiple frequency constraints. In the standard FBI, which is inspired by the suspect investigation-location-pursuit process, two teams of search agents (i.e., investigation and pursuit teams) are considered to perform exploration and exploitation of search space, respectively. However, the standard FBI suffers from the lack of coordination between the investigation and pursuit teams, which may result in various drawbacks such as low convergence rate and trapping in local optima. The proposed EFBI algorithm aims to reinforce the relationship between exploration and exploitation strategies of the standard FBI. The performance of the proposed algorithm is tested

on three benchmark dome truss optimization problems with frequency constraints. The results obtained by the proposed EFBI are compared with those of the standard FBI and some other optimization algorithms reported in the literature.

Chapter 9 presents an improved version of the slime mould algorithm (SMA) named improved SMA (ISMA) to deal with the sizing optimization problem of truss structures with frequency constraints. The classical SMA suffers from slow convergence and often converges prematurely to non-optimal solutions. To overcome the drawbacks mentioned above, two main modifications are made in the classical SMA. First, instead of the generational replacement strategy of the classical SMA, an elitist strategy is applied within the replacement phase of the proposed ISMA. Second, a slight modification is made to the exploration phase of the classical SMA to ensure a vast exploration of the search space. Finally, three numerical examples on size optimization of large-scale dome truss structures with frequency constraints are presented to demonstrate the efficiency and robustness of the proposed ISMA, and the numerical results are compared with those of the classical SMA and some other state-of-the-art metaheuristic algorithms in the literature.

Chapter 10 proposed an improved version of the arithmetic optimization algorithm (AOA), named improved arithmetic optimization algorithm (IAOA), to solve discrete structural optimization problems. The standard AOA often leads to the loss of population diversity, which may result in poor exploration of the search space and premature convergence to non-optimal solutions. The following modifications are made in the standard AOA to overcome the inherent drawbacks mentioned above. First, the position update rule of the standard AOA is modified to enhance the exploration and exploitation capabilities. Second, the proposed IAOA has less algorithm-specific parameters than the standard AOA. To investigate the performance of the proposed IAOA, four benchmark discrete structural optimization problems are presented. The results obtained by the proposed IAOA are compared with those of the standard AOA and some other optimization methods in the literature.

References

1. Yang XS (2010) Engineering optimization: an introduction with metaheuristic applications. Wiley
2. Kaveh A (2017) Applications of metaheuristic optimization algorithms in civil engineering, 1st edn. Springer Cham, Switzerland. <https://doi.org/10.1007/978-3-319-48012-1>
3. Yang XS, Gandomi AH, Talatahari S, Alavi AH (2012) Metaheuristics in water, geotechnical and transport engineering. Newnes. <https://doi.org/10.1016/C2011-0-07801-8>
4. Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 95:51–67. <https://doi.org/10.1016/j.advengsoft.2016.01.008>
5. Kaveh A (2021) Advances in metaheuristic algorithms for optimal design of structures, 3rd edn. Springer, Switzerland. <https://doi.org/10.1007/978-3-319-05549-7>
6. Ezugwu AE, Shukla AK, Nath R, Akinyelu AA, Agushaka JO, Chironma H, Muhuri PK (2021) Metaheuristics: a comprehensive overview and classification along with bibliometric analysis. Artif Intell Rev 54(6):4237–4316. <https://doi.org/10.1007/s10462-020-09952-0>
7. Blum C, Raidl GR (2016) Hybrid metaheuristics: powerful tools for optimization. 1st edn. Springer Cham, Switzerland. <https://doi.org/10.1007/978-3-319-30883-8>

8. Yang XS (2010) Nature-inspired metaheuristic algorithms. 2nd edn. Luniver Press
9. Talbi EG (2009) Metaheuristics: from design to implementation. Wiley

Chapter 2

Set-Theoretical Shuffled Shepherd Optimization Algorithm for Optimal Design of Reinforced Concrete Cantilever Retaining Wall Structures



2.1 Introduction

In this chapter, some basic concepts of set theory are employed to generalize the description of the shuffled shepherd optimization algorithm (SSOA) [1]. This may be very helpful for better understanding the search mechanism in SSOA. Furthermore, it makes an attempt to present a general set-theoretical framework for population-based metaheuristics. Finally, SSOA is applied to the minimum cost design of reinforced concrete cantilever retaining wall structures under both static and dynamic loading conditions.

Retaining wall structures are designed and constructed for the purpose of supporting vertical or near-vertical slopes of soil or other loose materials [2]. This kind of structure is suitable for design situations in which it is desired to maintain a difference in ground level at points located so close to each other that a natural slope would occupy too much space. Retaining walls are among the most common geotechnical structures and have been widely used in many structural applications, including railway embankments, roadsides, culverts, bridge abutments, etc. Therefore, it is very important to design safe and cost-efficient retaining wall structures. Retaining walls can be classified into two main groups of gravity and cantilever retaining walls. Gravity retaining walls are made of plain concrete and rely solely on their own weight to maintain stability against sliding and overturning. Where walls up to 2 m in height are needed, it is generally economical to choose a gravity retaining wall [3]. On the other hand, cantilever retaining walls are made of reinforced concrete and generally composed of a vertical stem, a base slab, and a shear key. Several different types of forces act on a cantilever retaining wall: self-weight of the wall, surcharge load, lateral pressure due to soil and surcharge, etc. The design process of a cantilever retaining wall can be divided into two main steps. The first step is focused on the estimation of forces acting on the wall. In this step, the wall is checked for stability against sliding, overturning, and bearing capacity failure. In the second step, each component of the wall is checked for strength and the

required reinforcement is calculated. Thus, proper design of a cantilever retaining wall requires accurate estimation of the acting forces, especially lateral earth pressures. The Coulomb and Rankine theories are well-known classical theories for the evaluation of lateral earth pressures under static loading conditions. When dynamic loading conditions are considered, the Mononobe-Okabe method is the first choice for evaluating lateral earth pressures [4].

Optimization approaches can be divided into two main groups of mathematical programming techniques and metaheuristic algorithms. Mathematical programming techniques are analytical approaches and apply differential calculus for finding the optimal solution [5]. It should be mentioned that these techniques are useful only in cases where the objective function is continuous and differentiable. Moreover, they may easily get trapped in local optima and gradients are not easily calculated. In order to overcome the limitations of mathematical programming techniques, a group of stochastic global optimization methods called metaheuristic algorithms have been developed over the past few decades. Metaheuristic algorithms, which are conceptually different from mathematical programming techniques, try to integrate rule-based theories and randomization with the aim of exploring effectively the search space of the optimization problem [6].

Over the past decades, a wide range of metaheuristic algorithms have been applied to the optimal design problem of cantilever retaining wall structures. Camp and Akin [7] used the big bang-big crunch (BB-BC) algorithm for design optimization of reinforced concrete cantilever retaining walls with shear keys. Kaveh and Behnam [8] employed the charged system search (CSS) algorithm to for the design optimization of reinforced concrete cantilever retaining walls under static loading conditions. Gandomi et al. [9] explored the efficiency of accelerated particle swarm optimization (APSO), cuckoo search (CS), and firefly algorithm (FA) for the optimal design of reinforced concrete cantilever retaining walls. Kaveh and Farhoudi [10] used the dolphin echolocation optimization (DEO) algorithm to find the optimal design of reinforced concrete cantilever retaining wall structures. Tumer and Bekdas [11] applied the teaching–learning-based optimization (TLBO) algorithm to optimize the design of reinforced concrete cantilever retaining walls. Yepes et al. [12] employed the simulated annealing (SA) algorithm for the optimum design of cantilever earth-retaining walls. Recently, Kaveh et al. [13] applied eight different population-based metaheuristic algorithms namely artificial bee colony (ABC), BB-BC, TLBO, imperialist competitive algorithm (ICA), CS, CSS, ray optimization (RO), tug of war optimization (TWO), water evaporation optimization (WEO), vibrating particles system (VPS), and cyclical parthenogenesis algorithm (CPA) to the cost optimization of reinforced concrete cantilever retaining walls.

The shuffled shepherd optimization algorithm (SSOA) is a simple population-based metaheuristic algorithm developed by Kaveh and Zaerreza [14]. This algorithm is inspired by the way a shepherd guides a flock of sheep towards a target. In a recent research, Kaveh and Zaerreza [15] applied SSOA to truss optimization problems and showed its competitive performance in terms of accuracy and convergence speed.

In this chapter, the shuffled shepherd optimization algorithm is applied to find the optimal design of reinforced concrete cantilever retaining wall structures. The

optimization objective is to minimize the cost of cantilever retaining walls while satisfying stability and strength requirements. The structural design is performed according to the Building Code Requirements for Structural Concrete (ACI 318–05) [16]. In order to demonstrate the performance of SSOA, a numerical example of the cost optimization of a reinforced concrete cantilever retaining wall is studied. The example is investigated under both static and dynamic loading conditions. The Coulomb and Rankine theories are used for the evaluation of lateral earth pressures under static loading conditions, and the Mononobe-Okabe method is used for dynamic loading conditions.

The rest of this chapter is organized as follows: In Sect. 2.2, the shuffled shepherd optimization algorithm (SSOA) is overviewed briefly. In Sect. 2.3, SSOA is expressed based on set-theoretical concepts and set-theoretical SSOA (ST-SSOA) is presented. In Sect. 2.4, the optimization problem is formulated. In Sect. 2.5, the structural analysis of cantilever retaining wall is done. In Sect. 2.6, the optimization results are discussed in detail. Finally, Sect. 2.7 concludes the chapter.

2.2 Shuffled Shepherd Optimization Algorithm (SSOA)

Over time, humans have learned to domesticate a large variety of animals such as horses, dogs, cows, etc. to make use of the physical work which these animals can provide. For example, nomadic pastoralists rely on their horses to move their flocks and herds across the great grassland steppes, and to carry their shelter. As another example, the breeding of herding dogs and livestock guardian dogs play an important role in the facilitation of hard work of shepherds and enhancement of their efficiency. Many shepherds use animals like herding dogs and horses to help them to handle herds, protect herds, and look for left behind and lost animals. The basic inspiration of the shuffled shepherd optimization algorithm (SSOA) is based on the way that a shepherd tends and rears sheep [14]. Like most of the population-based metaheuristic algorithms, SSOA starts with a random initial population of solutions. In this algorithm, a herd of sheep is considered as the population of solutions. It should be noted that it is a common practice for two or more herds to graze a common pasture. Accordingly, the initial population of SSOA (the main herd) is divided into a predefined number of smaller subpopulations (subherds) having the same number of solutions (sheep). For this purpose, the initial population is evaluated and the solutions are sorted in ascending order of their penalized objective function values. Next, through a few simple steps that will be explained in detail in Sect. 2.3, the division process is carried out. The subpopulations are formed in such a way that they are close to each other in terms of their average penalized objective function values. In addition, the solutions in the subpopulations are in ascending order of their penalized objective function values. In the next step, the position update of each sheep in each subherd is determined. For instance, let us consider a sheep in one of the subherds and take it as the shepherd. It is obvious that the sheep under consideration (shepherd) belongs to one of the subherds. The subherd to which the shepherd belongs

contains other sheep, some of which are more fit than the shepherd (i.e., have better penalized objective function values), while others are less so (i.e., have worse penalized objective function values). However, this is not true for the first and last sheep of each subherd. This is because the first sheep of each subherd is the best sheep of the subherd and there is no better sheep than it in the subherd. On the other hand, the last sheep of each subherd is the worst sheep of the subherd and there is no worse sheep than it in the subherd. The better and worse sheep are called horses and sheep, respectively. A horse and a sheep are selected randomly from the subherd to which the shepherd belongs. In order to determine the new position of the shepherd, the shepherd moves from his current position towards the positions of both the horse and the sheep. Therefore, the new position of the shepherd is obtained. After the position update, a position check is carried out to make sure that none of the newly generated candidate solutions have moved out of the search space boundaries. Next, a greedy replacement strategy is employed to form the subherds of the next generation. For this purpose, if the new position is better than the current position according to the penalized objective function value, then the new position will replace the current one. The above process is carried out for the sheep of all subherds. Next, the subherds are merged together and the herd of sheep of the next generation is formed. The sheep are sorted in ascending order of their penalized objective function values, and the herd is divided into a predefined number of smaller subherds having the same number of sheep. The search process continues until the termination condition of the algorithm is satisfied.

2.3 Set-Theoretical Shuffled Shepherd Optimization Algorithm (ST-SSOA)

In this section, some basic concepts of set theory are employed to generalize the description of SSOA. Taking a look at the shuffled shepherd optimization algorithm, it can be seen that there are similarities between some concepts of set theory and SSOA. The population of candidate solutions (i.e., the herd of sheep) can be viewed as a set of elements. The weight of an element is defined as the value of its penalized objective function. Also, the weight of a subset is defined as the sum of the weights of its elements. At each iteration of the algorithm, the elements are sorted in ascending order of their weight values. Next, the set of elements is divided into a predefined number of smaller subsets of the same cardinality. The subsets are formed in such a way that their weight values are close to each other. For each element of each subset, the new position is determined as follows: Let us consider an element in one of the subsets. A better element (i.e., an element with a lower weight value) and a worse element (i.e., an element with a higher weight value) are chosen randomly for the element under consideration. It should be noted that the better and worse elements are chosen from the subset to which the element under consideration belongs. After determining the better and worse elements, the new position of the element is achieved

based on the position update rule. Next, a position check is carried out to prevent the new position from moving out of the search space boundaries. Next, a greedy replacement strategy is applied to determine the subsets of the next generation. If the new position of an element has a better weight than its corresponding current position, then the new position will replace the current one. Once the subsets of new elements are obtained, they are merged to form the set of elements of the next generation. The search process continues until the termination condition is fulfilled. The set-theoretical shuffled shepherd optimization algorithm (ST-SSOA) is stated in five steps as follows:

Step one (initialization): Like most of the population-based metaheuristic algorithm, SSOA starts with a set of randomly generated initial elements. These elements are evaluated and their weight values are calculated. For instance, let us consider the set EL of nEL elements. The set of elements is generated as follows:

$$EL(:, j) = Lb(j) + rand(nEL, 1). * (Ub(j) - Lb(j)), \forall j \in \{1, 2, \dots, nV\} \quad (2.1)$$

where nEL is the number of elements; nV is the number of design variables; $Lb(j)$ and $Ub(j)$ are the lower and upper bounds of the j -th design variable, respectively; EL is the set of elements; and $rand(nEL, 1)$ is a vector of random numbers uniformly distributed in the range of $[0, 1]$.

Step two (forming the subsets): At the first step, the elements are sorted in ascending order of their weight values ($PFit$). Next, the set of elements is divided into a predefined number, say k , of smaller subsets of the same cardinality. For this purpose, let us consider k null subsets. In the first step of forming the subsets, the first k elements of the sorted set of elements are chosen and one element is randomly placed in each subset. Once the first step is completed, each of the subsets has one element. In the next step of forming the subsets, the next k elements of the sorted set of elements are chosen and one element is randomly placed in each subset. Once the second step is completed, each of the subsets has two elements. This process continues until no more element remains in the sorted set of elements. At the end of the process, all the subsets have the same number of $m = nEL/k$ elements. Therefore, it can be said that

$$nEL = m \times k \quad (2.2)$$

where nEL is the number of elements; k is the number of subsets; and m is the number of elements of each subset.

As mentioned earlier, the subsets are close to each other in terms of their weight values. Also, the elements of each subset are in ascending order of their weight values. It is clear that the first and last elements of each subset have the lowest and highest weight values as compared to other elements of the subset, respectively. The average weight value of the elements of the i -th subset ($mean_{s_i}$) can be calculated as:

$$mean_{s_i} = \frac{1}{m} (P\text{Fit}_{i,1} + P\text{Fit}_{i,2} + \dots + P\text{Fit}_{i,m}); i = 1, 2, \dots, k \quad (2.3)$$

where k is the number of subsets; m is the number of elements of each subset; and $P\text{Fit}_{i,j}$ is the value of weight of the j -th element of the i -th subset.

Step three (position update rule): In this step, the position update rule is applied to the elements of the subsets. For this purpose, let us consider an element in one of the subsets. A better element (i.e., an element with a lower weight value) and a worse element (i.e., an element with a higher weight value) are chosen randomly for the element under consideration. It is noted again that the better and worse elements are chosen from the subset to which the element under consideration belongs. For instance, for the j -th element of a subset, there exist $j - 1$ better elements and $m - j$ worse elements. It is clear that there is no better element for the first element of a subset. On the other hand, there is no worse element for the last element of a subset. The position update rule for each element of each subset is as follows:

$$newEL_{i,j} = stepsize_{i,j} + EL_{i,j}; i = 1, 2, \dots, k \text{ and } j = 1, 2, \dots, m \quad (2.4)$$

$$stepsize_{i,j} = \beta \times rand_1 \times (EL_B_{i,j} - EL_{i,j}) + \alpha \times rand_2 \times (EL_W_{i,j} - EL_{i,j}) \quad (2.5)$$

where $EL_{i,j}$ and $newEL_{i,j}$ are the current and new positions of the j -th element of the i -th subset, respectively; $stepsize_{i,j}$ is the step size of the j -th element of the i -th subset; $EL_B_{i,j}$ is the position of one of the elements of the i -th subset which are better than the j -th element of the i -th subset; $EL_W_{i,j}$ is the position of one of the elements of the i -th subset which are worse than the j -th element of the i -th subset; $rand_1$ and $rand_2$ are uniformly distributed random numbers from the interval $[0, 1]$; and α and β are the parameters of the position update rule, which are provided to maintain the balance between exploration and exploitation.

A careful examination of Eq. (2.5) indicates that for the first element of each subset, the first term of the equation becomes zero. Also, for the last element of each subset, the second term of Eq. (2.5) becomes zero. Figure 2.1 illustrates the movement of an element from its current position to its new position. When an element moves towards a better element, exploitation of the search space is achieved. On the other hand, when an element moves towards a worse element, exploration of the search space is encouraged. Therefore, the parameters α and β control exploration and exploitation of the algorithm, respectively. It is generally recognized that a successful metaheuristic algorithm focuses more on exploration in the early stages of the search process, while the exploitation is preferred in the later stages [6]. Accordingly, the parameters α and β are defined as decreasing and increasing functions of the iteration number, respectively.

$$\alpha = \alpha_0 - \frac{\alpha_0}{MaxIter} \times Iter \quad (2.6)$$

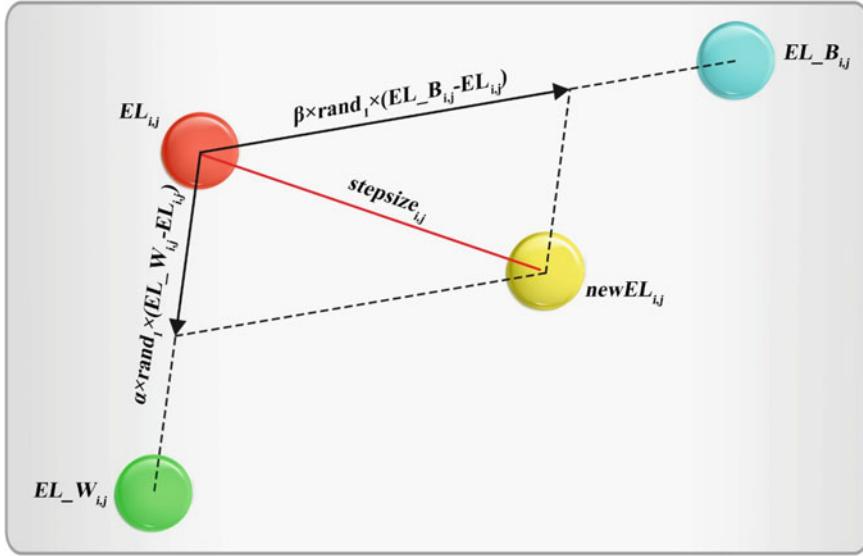


Fig. 2.1 Position update of an element of ST-SSOA

$$\beta = \beta_0 + \frac{\beta_{max} - \beta_0}{MaxIter} \times Iter \quad (2.7)$$

where $Iter$ is the current iteration number; $MaxIter$ is the maximum number of iterations; α is the initial value of the parameter α ; and β_0 and β_{max} are the initial and final values of the parameter β .

Once new positions of the elements are obtained, a position check is carried out to ensure that the new positions are located inside the search space boundaries.

Step four (replacement strategy): In order to form the subsets of the next generation, a greedy replacement strategy is applied. For this purpose, each element is evaluated according to its new position. If the new position of an element has a better weight than its corresponding current position, then the new position replaces the current position of the element. In this way, the positions of the elements are updated. After applying the replacement strategy, the subsets are merged together to form the set of elements of the next generation.

Step five (termination condition): If the number of iterations reaches the predetermined maximum number of iterations ($MaxIter$), the algorithm terminates and the best element found so far is reported. Otherwise, the algorithm goes to step two.

The pseudocode of the ST-SSOA is provided below:

Initialization:

Set the algorithm parameters: nEL , k , α_0 , β_0 , β_{max} , and $MaxIter$.

Generate the initial set of elements (EL) randomly using Eq. (2.1).

Evaluate the set of elements and calculate the weight of each element (*PFit*).

Iter = 0;

Cyclic body of ST-SSOA:

While *Iter* < *MaxIter*

Step 1: Sort the set of elements in ascending order of their weights.

Step 2: Form the subsets based on the process described in the second step of ST-SSOA.

Step 3: Determine the step size matrix (*stepsize*) using Eqs. (2.5), (2.6), and (2.7).

Step 4: Determine the new positions of elements using Eq. (2.4).

Step 5: Check the newly generated positions to ensure that they are located inside the search space.

Step 6: Evaluate the elements according to their new positions and apply the replacement strategy to update the subsets.

Update the iteration number (*Iter* = *Iter* + 1).

Step 7: Merge the subsets to form the set of elements of the next generation.

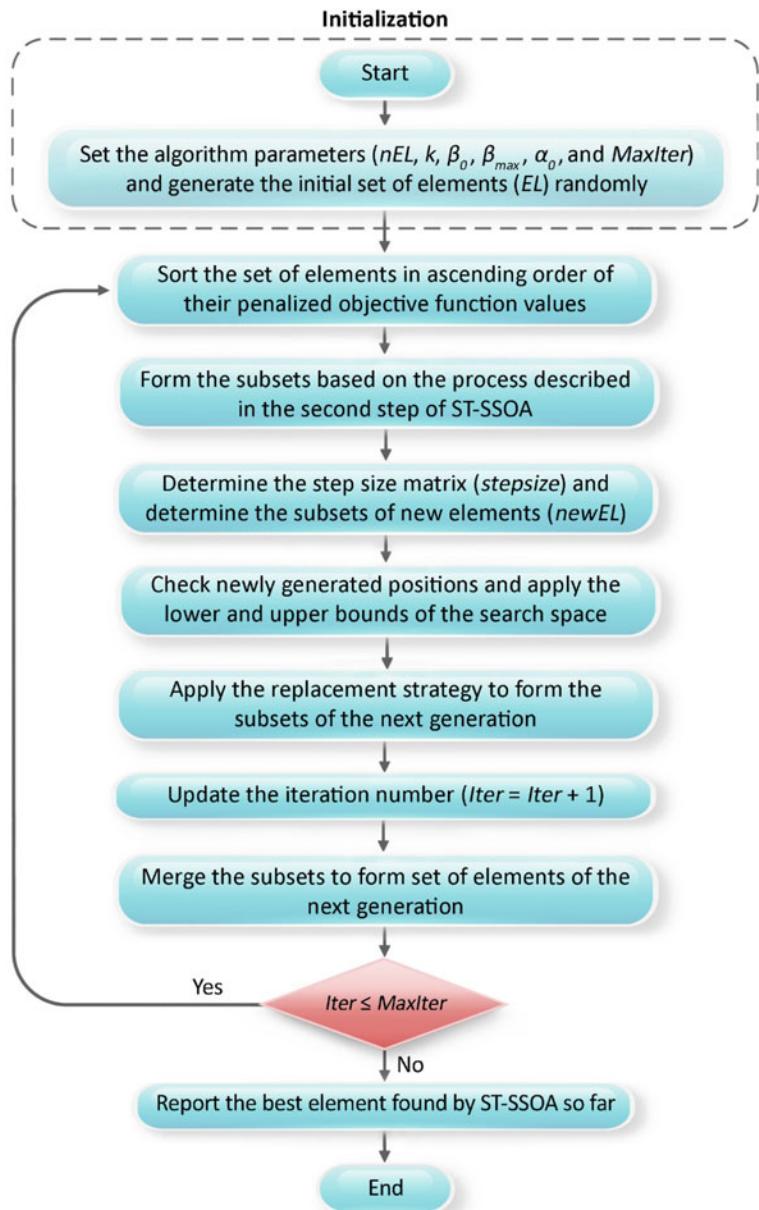
Monitor the best element found by ST-SSOA so far.

End while.

The flowchart of the ST-SSOA is shown in Fig. 2.2. As is clear from the figure, it is not possible to distinguish between exploration and exploitation phases of the algorithm. This is due to the fact that the position update rule for an element of ST-SSOA involves simultaneously two movement components: a movement towards a better element, and a movement towards a worse element. Figure 2.3 shows the elements of ST-SSOA at different steps of the search process.

2.4 Definition of the Optimization Problem

In this section, the cost optimization problem of a reinforced concrete cantilever retaining wall is presented. The objective of optimization is to minimize the total cost of the wall while satisfying structural stability and strength constraints. The total cost is defined as the summation of the concrete cost and the steel reinforcement cost. It is noted that costs include the cost of materials and labor. Figure 2.4 shows the schematic of a reinforced concrete cantilever retaining wall. As can be seen in the figure, there are seven geometric design variables and four structural design variables for the problem. Geometric design variables were chosen with respect to the geometry of the wall and structural design variables were chosen with respect to the critical sections of the wall. Geometric design variables express the sections of the wall, and they include the stem thickness at the top of the wall (L_1), width of the shear key (L_2), width of the toe slab (L_3), width of the heel slab (L_4), height of the top stem (L_5), thickness of the base slab (L_6), and height of the shear key (L_7). Structural design variables express the steel reinforcement of the wall, and they include the area of vertical steel reinforcement in the top stem per unit length of

**Fig. 2.2** Flowchart of the ST-SOA

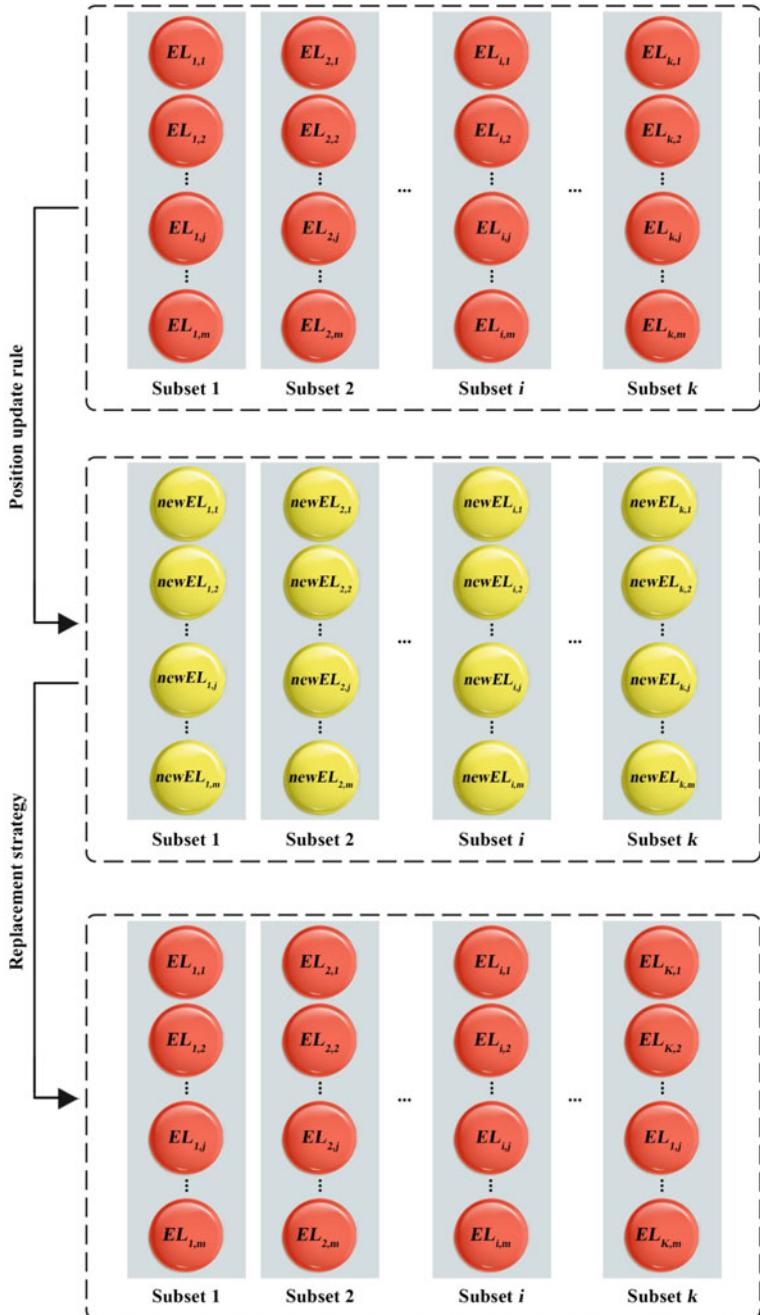
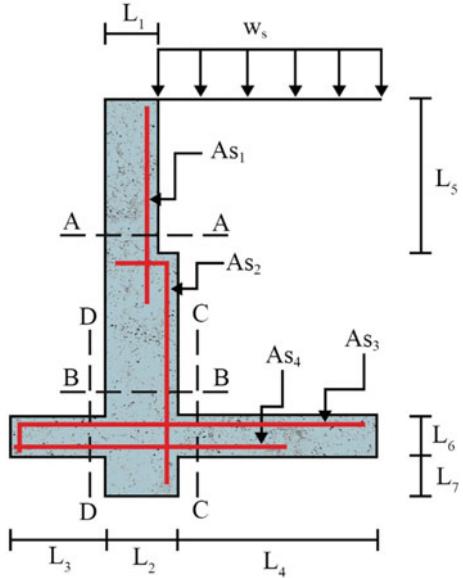


Fig. 2.3 Elements of the ST-SSOA at different steps of the search process

Fig. 2.4 Design variables for the reinforced concrete cantilever retaining wall



the wall (As_1), area of vertical steel reinforcement in the bottom stem per unit length of the wall (As_2), area of horizontal steel reinforcement in the heel slab per unit length of the wall (As_3), and area of horizontal steel reinforcement in the toe slab per unit length of the wall (As_4). For both the steel reinforcement areas and geometric dimensions, continuous design variables are considered. The optimization problem can be formulated as follows [10]:

$$\text{Find: } \{X\} = \{L_1, \dots, L_7, As_1, \dots, As_4\} \quad (2.8)$$

$$\text{to minimize: } Cost(\{X\}) = V_{concrete} + W_{steel} \times \left(\frac{C_3 + C_4}{C_1 + C_2} \right) \quad (2.9)$$

$$\text{subject to: } \begin{cases} SF_{overturning} \geq 1.5 \\ SF_{sliding} \geq 1.5 \\ SF_{bearing capacity} \geq 2 \\ \frac{M_u}{\phi_b M_n} \leq 1 \\ \frac{V_u}{\phi_v V_n} \leq 1 \end{cases} \quad (2.10)$$

where $\{X\}$ is the vector of design variables including geometric and structural design variables; L_i is the i -th geometric design variable; As_j is the j -th structural design variable, which represents the area of steel reinforcement per unit length in the j -th critical section of the wall; $Cost(\{X\})$ is the total cost of the wall including the

cost of materials and labor; $V_{concrete}$ is the volume of concrete per unit length of the wall; W_{steel} is the weight of reinforcement steel per unit length of the wall; C_1 and C_2 are the unit costs of concrete and steel reinforcement, respectively; and C_3 and C_4 are the unit costs of concreting and reinforcing, respectively. It is noted that the values of C_1 , C_2 , C_3 , and C_4 depend on many factors such as the type of structure, economic conditions, work conditions, country, etc. However, it has been shown that the value of $\frac{C_3+C_4}{C_1+C_2}$ is in the range between 0.035 and 0.045 [10]. In this study, the value of this ratio is set to be 0.04, as recommended by Kaveh and Behnam [8]. As shown in Eqs. (2.10), (2.11), (2.12), (2.13), and (2.14), structural stability and strength requirements are considered as the constraints of the problem. $SF_{overturning}$, $SF_{sliding}$, and $SF_{bearing\ capacity}$ are the factors of safety against overturning about the toe, sliding, and bearing capacity failure, respectively. According to the American Association of State Highway and Transportation Officials (AASHTO) standard specifications for highway bridges [17], the factors of safety against sliding and overturning failure under seismic loading conditions may be reduced to 75% of the factors of safety under static loading conditions. In addition, allowable stresses used for the design of the wall facing are permitted to increase by 33% for concrete components of the facing [17]. V_u , V_n , M_u , and M_n are the factored shear force, nominal shear strength, factored moment, and nominal flexural strength, respectively. In addition, ϕ_b and ϕ_v are the strength reduction factors for flexure and shear, respectively. Equations (2.13) and (2.14) should be checked at the critical sections shown in Fig. 2.4.

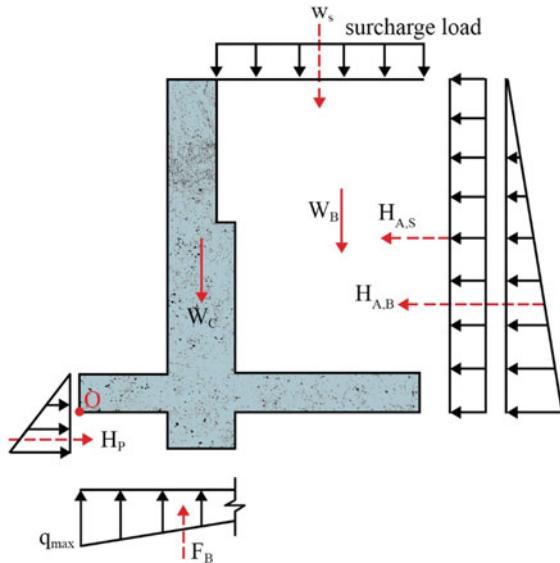
2.5 Analysis of Cantilever Retaining Walls

Figure 2.5 shows all the effective forces acting on the cantilever retaining wall. In the figure, W_C denote the sum of the weight of all the sections of the cantilever retaining wall, W_B is the weight of backfill on the heel of the wall, W_S is the surcharge load, $H_{A,B}$ is the force resulting from the active earth pressure, H_P is the force resulting from the passive earth pressure, $H_{A,S}$ is the force resulting from the surcharge load, and F_B is the force resulting from the bearing stress of the base soil.

2.5.1 Active and Passive Earth Pressure Coefficients

A number of investigations have been performed by several researchers to evaluate the active and passive earth pressures under both static and seismic loading conditions. As mentioned earlier, this study is conducted under both static and seismic loading conditions. The commonly used Rankine and Coulomb theories are used to calculate the active and passive earth pressures acting on the retaining wall under static loading conditions, while the Mononobe-Okabe method is used under seismic loading conditions. Based on the Rankine earth pressure theory, the active and passive earth pressure coefficients are given by the following equations [18]:

Fig. 2.5 Forces acting on the reinforced concrete cantilever retaining wall



$$k_{A,R} = \cos(\alpha) \times \left(\frac{\cos(\alpha) - \sqrt{\cos^2(\alpha) - \cos^2(\phi)}}{\cos(\alpha) + \sqrt{\cos^2(\alpha) - \cos^2(\phi)}} \right) \quad (2.11)$$

$$k_{P,R} = \cos(\alpha) \times \left(\frac{\cos(\alpha) + \sqrt{\cos^2(\alpha) - \cos^2(\phi)}}{\cos(\alpha) - \sqrt{\cos^2(\alpha) - \cos^2(\phi)}} \right) \quad (2.12)$$

where $k_{A,R}$ and $k_{P,R}$ are the Rankine active and passive earth pressure coefficients, respectively; α is the inclination angle of the backfill with respect to the horizontal axis; and ϕ is the angle of internal friction of the backfill soil.

According to the the Coulomb earth pressure theory, the active and passive earth pressure coefficients are calculated by the following equations [18]:

$$k_{A,C} = \frac{\sin^2(\beta + \phi)}{\sin^2(\beta) \times \sin(\beta - \delta) \times \left[1 + \sqrt{\frac{\sin(\phi + \delta) \times \sin(\phi - \alpha)}{\sin(\beta - \delta) \times \sin(\alpha + \beta)}} \right]^2} \quad (2.13)$$

$$k_{P,C} = \frac{\sin^2(\beta - \phi)}{\sin^2(\beta) \times \sin(\beta + \delta) \times \left[1 - \sqrt{\frac{\sin(\phi + \delta) \times \sin(\phi + \alpha)}{\sin(\beta + \delta) \times \sin(\alpha + \beta)}} \right]^2} \quad (2.14)$$

where $k_{A,C}$ and $k_{P,C}$ are the Coulomb active and passive earth pressure coefficients, respectively; α and ϕ are defined similar to those in Eqs. (2.15) and (2.16); β is the angle of the back face of the retaining wall with respect to the horizontal axis; and δ is the angle of friction between the wall and the fill.

According to the Mononobe-Okabe method, the active and passive earth pressure coefficients under seismic loading conditions are calculated by the following equations [19]:

$$k_{A,M} = \frac{\sin^2(\phi + \beta - \theta)}{\cos(\theta) \times \sin^2(\beta) \times \sin(\beta - \delta - \theta) \times \left[1 + \sqrt{\frac{\sin(\phi + \delta) \times \sin(\phi - \alpha - \theta)}{\sin(\beta - \delta - \theta) \times \sin(\alpha + \beta)}} \right]^2} \quad (2.15)$$

$$k_{P,M} = \frac{\sin^2(\phi + \beta - \theta)}{\cos(\theta) \times \sin^2(\beta) \times \sin(\beta - \delta - \theta) \times \left[1 + \sqrt{\frac{\sin(\phi + \delta) \times \sin(\phi - \alpha - \theta)}{\sin(\beta - \delta - \theta) \times \sin(\alpha + \beta)}} \right]^2} \quad (2.16)$$

where $k_{A,M}$ and $k_{P,M}$ are the seismic active and passive earth pressure coefficients according to the Mononobe-Okabe method, respectively; α , ϕ , β , and δ are defined similar to those in Eqs. (2.17) and (2.18); and θ is calculated by the following equation [19]:

$$\theta = \tan^{-1} \left(\frac{k_h}{1 - k_v} \right) \quad (2.17)$$

where k_h and k_v are the horizontal and vertical acceleration coefficients, respectively and they can be expressed as follows [19]:

$$k_h = \frac{\text{horizontal earthquake acceleration component}}{\text{acceleration due to gravity(g)}} \quad (2.18)$$

$$k_v = \frac{\text{vertical earthquake acceleration component}}{\text{acceleration due to gravity(g)}} \quad (2.19)$$

2.5.2 Stability Analysis of Cantilever Retaining Walls

Typically, three failure modes are considered in the analysis of the retaining wall structure: sliding, overturning, and bearing capacity failure. The factor of safety against overturning about the toe, which is shown by point O in Fig. 2.5, is defined as follows [18]:

$$SF_{\text{overturning}} = \frac{\sum M_R}{\sum M_O} \quad (2.20)$$

where $SF_{overturning}$ is the factor of safety against overturning about the toe of the wall; $\sum M_R$ is the sum of the resisting moments about the toe of the wall; and $\sum M_O$ is the sum of the overturning moments about the toe of the wall. It is noted that the weight of the wall, weight of the backfill on the heel of the wall, and surcharge load contribute to the resisting moment, while the lateral forces exerted on the wall due to the soil and surcharge load contribute to the overturning moment.

The factor of safety against sliding is defined as follows [18]:

$$SF_{sliding} = \frac{\sum F_R}{\sum F_D} = \frac{H_P + \sum V \times \tan(\delta)}{H_{A,S} + H_{A,B}} \quad (2.21)$$

where $SF_{sliding}$ is the factor of safety against sliding; $\sum F_R$ is the sum of the horizontal resisting forces; $\sum F_D$ is the sum of the horizontal driving forces; and V is the sum of the vertical forces (i.e., weight of the wall, weight of the backfill on the heel of the wall, and surcharge load). Thus, it can be written that:

$$\sum V = W_B + W_C + W_S \quad (2.22)$$

The factor of safety against bearing capacity failure is defined as follows [18]:

$$SF_{bearingcapacity} = \frac{q_u}{q_{max}} \quad (2.23)$$

where $SF_{bearingcapacity}$ is the factor of safety against bearing capacity failure; q_u is the ultimate bearing capacity of the foundation soil; and q_{max} is the maximum bearing stress acting along the base of the wall (on the toe section). The maximum and minimum bearing stresses acting along the base of the wall can be determined by the following equation [18]:

$$q_{max,min} = \frac{\sum V}{B} \times \left(1 \pm \frac{6e}{B}\right); e \leq \frac{B}{6} \quad (2.24)$$

where q_{max} and q_{min} are the maximum and minimum bearing stresses acting along the base of the wall (on the toe and heel sections, respectively); e is the eccentricity of the resultant vertical load; and B is the width of the base slab. If the eccentricity becomes larger than $\frac{B}{6}$, then q_{min} will be negative, which means there is some tensile stress at the end of the heel section. In such a case, q_{max} is determined as follows [20]:

$$q_{max} = \frac{4 \times \sum V}{3 \times (B - 2e)}; e > \frac{B}{6} \quad (2.25)$$

The eccentricity e is determined as follows:

$$e = \frac{B}{2} - \frac{\sum M_R - \sum M_O}{\sum V} \quad (2.26)$$

2.6 Results and Discussion

In this section, in order to examine the efficiency and effectiveness of the proposed ST-SSOA, the minimum cost design of a reinforced concrete cantilever retaining wall structure is investigated. The retaining wall is designed under both static and seismic loading conditions. In order to calculate the lateral static earth pressures acting on the retaining wall, the Coulomb and Rankine theories are used, and the Mononobe-Okabe method is used for the estimation of the seismic earth pressures acting on the retaining wall. It is noted that four different combinations of horizontal and vertical acceleration coefficients are considered for the design optimization of the retaining wall under seismic loading conditions, as shown in Table 2.1. Therefore, totally six different case studies are investigated here to demonstrate the performance of ST-SSOA. The case studies include the Coulomb theory (case study 1), Rankine theory (case study 2), and Mononobe-Okabe method (case studies 3 to 6). Table 2.2 provides the lower and upper bounds of the design variables of the problem. The problem data including soil parameters, design parameters, and retaining wall parameters are given in Table 2.3.

In order to consider the stochastic nature of the optimization process, 50 independent optimization runs have been performed for each case study, and the results have been presented in Table 2.4. Note that the initial populations were generated randomly. For each case study, the best, worst, mean, and standard deviation of 50 optimized costs that were obtained over 50 independent runs have been reported. It can be seen from Table 2.4 that the cantilever retaining walls designed based on the Rankine earth pressure theory are more expensive than those based on the Coulomb

Table 2.1 Different combinations of horizontal and vertical acceleration coefficients

Case study	Horizontal acceleration coefficient (k_h)	Vertical acceleration coefficient (k_v)
Case study 3	0	0
Case study 4	0.15	0
Case study 5	0.15	0.15
Case study 6	0.15	0.075

Table 2.2 Lower and upper bounds of design variables

Design variable	L_1	L_2	L_3	L_4	L_5	L_6	L_7
Lower bound (cm)	30	30	45	180	150	30	20
Upper bound (cm)	60	60	120	300	610	90	90

Table 2.3 Numerical data of the reinforced concrete cantilever retaining wall problem

Parameter	Symbol	Unit	Value
Factor of safety against overturning failure mode	$SF_{overturning}$	-	1.5
Factor of safety against sliding failure mode	$SF_{sliding}$	-	1.5
Factor of safety against bearing capacity failure mode	$SF_{bearing capacity}$	-	2
Surcharge load	W_S	kN/m ²	10
Stem height	H	m	6.1
Allowable soil pressure	q_a	kN/m ²	300
Concrete cover	d_c	cm	5
Unit weight of concrete	γ_c	kN/m ³	24
Unit weight of reinforcement steel	γ_s	kN/m ³	78
Unit weight of backfill soil	γ_b	kN/m ³	22
Inclination angle of the backfill soil with respect to the horizontal axis	α	Degree (°)	0
Angle of internal friction of the backfill soil	ϕ	Degree (°)	35
Angle of friction between the wall and the backfill soil	δ	Degree (°)	$\frac{1}{2}\phi \leq \delta \leq \frac{2}{3}\phi$
Angle of the back face of the wall with respect to the horizontal axis	β	Degree (°)	90
Base friction coefficient	μ	-	$\tan(\delta)$
Depth of soil in front of the wall	h_p	m	0
Yield strength of reinforcement steel	f_c	MPa	300
Compressive strength of concrete	f_y	MPa	25
Flexural strength reduction factor	ϕ_b	-	0.9
Shear strength reduction factor	ϕ_v	-	0.75

earth pressure theory. This finding was predictable because the Rankine earth pressure theory is more conservative than the Coulomb earth pressure theory, which is due to the fact that the Coulomb theory takes into account the friction between the soil and the base slab, whereas the Rankine theory does not. It can also be concluded from Table 2.4 that the total cost of the wall decreases as the vertical acceleration coefficient increases. On the contrary, it is seen that the increase of the horizontal acceleration coefficient value leads to the increase in the total cost of the wall. Table 2.5 gives the stability and shear capacity ratios of the optimal designs obtained by ST-SSOA for different case studies. It should be noted that the shear capacity ratios are evaluated at four critical sections of the wall (i.e., sections A-A, B-B, C-C, and D-D). An examination of Table 2.5 reveals that for all case studies, the design is controlled by two factors: the shear capacity at the critical section of the toe slab

(section D-D), and the bearing strength of the base soil under the toe of the wall. Through Tables 2.6, 2.7, 2.8, 2.9, 2.10, 2.11, 2.12, 2.13, 2.14, 2.15, 2.16, and 2.17, the optimization results obtained by ST-SSOA for the six case studies mentioned above are compared with those previously reported by Kaveh et al. [13]. Optimization results related to the first and second case studies (i.e., the Coulomb and Rankine theories, respectively) are presented in Tables 2.6 and 2.8, respectively. In addition, Tables 2.10, 2.12, 2.14, and 2.16 present the optimization results related to the third, fourth, fifth, and sixth case studies, respectively. For all case studies, optimized costs obtained by the present study and other methods [13] at five different stages of the optimization process are reported in Tables 2.7, 2.9, and 2.11, 2.13, 2.15, and 2.17. In this study, in order to have a fair comparison, the maximum number of objective

Table 2.4 Comparison of statistical results obtained by ST-SSOA for different case studies

Case study	Best cost	Mean cost	Worst cost	Standard deviation
Coulomb theory (case study 1)	6.941	6.949	7.220	0.042
Rankine theory (case study 2)	7.370	7.760	17.630	1.927
Mononobe-Okabe method (case study 3)	6.219	6.228	6.593	0.053
Mononobe-Okabe method (case study 4)	8.676	10.204	21.759	3.553
Mononobe-Okabe method (case study 5)	7.811	7.821	8.250	0.062
Mononobe-Okabe method (case study 6)	8.241	8.986	18.093	2.504

Table 2.5 Stability and shear capacity ratios of the optimal designs obtained by ST-SSOA for different case studies (best runs)

Case study	Shear capacity (%)				Stability capacity (%)		
	A-A	B-B	C-C	D-D	Overslipping	Sliding	Bearing
Coulomb theory (case study 1)	39.18	67.85	21.95	100.00	41.85	56.61	100.00
Rankine theory (case study 2)	40.49	72.71	22.56	100.00	42.12	62.82	100.00
Mononobe-Okabe method (case study 3)	41.88	67.47	26.55	100.00	35.89	44.11	100.00
Mononobe-Okabe method (case study 4)	41.42	77.14	29.13	100.00	37.10	48.33	100.00
Mononobe-Okabe method (case study 5)	41.35	73.59	28.64	100.00	36.82	47.58	100.00
Mononobe-Okabe method (case study 6)	41.31	75.81	28.93	100.00	36.97	47.98	100.00

Table 2.6 Comparison of optimization results obtained for the retaining wall (Coulomb theory, case study 1)

Table 2.7 Optimized costs obtained for the retaining wall at five different stages of the best optimization run (Coulomb theory, case study 1)

Number of structural analyses	Optimized cost								Present work	
	Kaveh et al. [13]									
	ABC	BB-BC	CS	CSS	ICA	RO	TWO	WEO		
1000	7.136	6.984	7.303	7.105	7.465	7.024	7.147	7.247	6.960	
2000	6.967	6.957	7.004	7.003	7.272	7.000	7.065	7.072	6.942	
3000	6.958	6.950	6.955	6.983	6.980	6.968	6.994	7.012	6.941	
4000	6.954	6.947	6.952	6.983	6.966	6.968	6.962	6.966	6.941	
5000	6.946	6.946	6.951	6.970	6.961	6.968	6.957	6.963	6.941	

function evaluations is used as the termination condition, and it is set to 5000 for all algorithms in all case studies.

A comparison of the results show that for all case studies, ST-SSOA has the best performance in terms of best cost, mean cost, and standard deviation of optimized costs. For example, as can be seen from Table 2.6, the best cost obtained by ST-SSOA is 6.941, while it is 6.946, 6.946, 6.951, 6.970, 6.961, 6.968, 6.957, and 6.963 for ABC, BB-BC, CS, CSS, ICA, RO, TWO, and WEO algorithms, respectively. In addition, the standard deviation of optimized costs obtained by ST-SSOA is 0.171, while it is 1.337, 0.346, 1.161, 0.506, 1.187, 1.071, 1.014, and 0.906 for ABC, BB-BC, CS, CSS, ICA, RO, TWO, and WEO algorithms, respectively. Figures 2.6, 2.7, 2.8, 2.9, 2.10, and 2.11 show a comparison of convergence histories of ST-SSOA and other methods [13] for six different case studies. A magnified view is added to the convergence histories to simplify the comparison of convergence performance. As observed from convergence curves, in all case studies, the proposed ST-SSOA converges much faster than the other eight metaheuristics. It can also be observed from the convergence histories that for all case studies, ST-SSOA has almost converged within the first 2000 objective function evaluations. This conclusion can be confirmed by the results shown in Tables 2.7, 2.9, 2.11, 2.13, 2.15, and 2.17. Figures 2.12, 2.13, 2.14, 2.15, 2.16, and 2.17 show the final costs obtained by ST-SSOA for six different case studies over 50 independent optimization runs. These figures can be used to measure the success rate of ST-SSOA. For example, the success rate of ST-SSOA is 96% for the first and second case studies.

Table 2.8 Comparison of optimization results obtained for the retaining wall (Rankine theory, case study 2)

Table 2.9 Optimized costs obtained for the retaining wall at five different stages of the best optimization run (Rankine theory, case study 2)

Number of structural analyses	Optimized cost								Present work	
	Kaveh et al. [13]									
	ABC	BB-BC	CS	CSS	ICA	RO	TWO	WEO		
1000	7.976	7.399	7.761	7.424	7.519	7.481	7.576	7.539	7.381	
2000	7.618	7.383	7.556	7.380	7.382	7.455	7.449	7.403	7.372	
3000	7.420	7.380	7.452	7.377	7.375	7.424	7.406	7.394	7.370	
4000	7.380	7.375	7.384	7.374	7.374	7.400	7.383	7.390	7.370	
5000	7.376	7.375	7.379	7.374	7.374	7.390	7.375	7.375	7.370	

Table 2.10 Comparison of optimization results obtained for the retaining wall (Mononobe-Okabe method, case study 3)

Table 2.11 Optimized costs obtained for the retaining wall at five different stages of the best optimization run (Mononobe-Okabe method, case study 3)

Number of structural analyses	Optimized cost								Present work	
	Kaveh et al. [13]									
	ABC	BB-BC	CS	CSS	ICA	RO	TWO	WEO		
1000	6.481	6.249	6.393	6.394	6.463	6.310	6.365	6.410	6.226	
2000	6.240	6.240	6.253	6.235	6.250	6.269	6.242	6.247	6.221	
3000	6.223	6.231	6.237	6.233	6.230	6.254	6.224	6.227	6.220	
4000	6.222	6.230	6.230	6.233	6.226	6.228	6.221	6.221	6.219	
5000	6.222	6.229	6.225	6.232	6.226	6.228	6.220	6.220	6.219	

Table 2.12 Comparison of optimization results obtained for the retaining wall (Mononobe-Okabe method, case study 4)

Table 2.13 Optimized costs obtained for the retaining wall at five different stages of the optimization process (Mononobe-Okabe method, case study 4)

Number of structural analyses	Optimized cost								Present work	
	Kaveh et al. [13]									
	ABC	BB-BC	CS	CSS	ICA	RO	TWO	WEO		
1000	8.713	8.721	9.070	8.768	9.018	8.731	9.140	8.713	8.699	
2000	8.681	8.688	8.777	8.730	8.922	8.705	8.769	8.704	8.678	
3000	8.680	8.685	8.700	8.710	8.748	8.705	8.700	8.680	8.677	
4000	8.680	8.683	8.685	8.697	8.718	8.702	8.690	8.678	8.676	
5000	8.679	8.683	8.679	8.697	8.704	8.696	8.682	8.677	8.676	

Table 2.14 Comparison of optimization results obtained for the retaining wall (Mononobe-Okabe method, case study 5)

Table 2.15 Optimized costs obtained for the retaining wall at five different stages of the best optimization run (Mononobe-Okabe method, case study 5)

Number of structural analyses	Optimized cost								Present work	
	Kaveh et al. [13]									
	ABC	BB-BC	CS	CSS	ICA	RO	TWO	WEO		
1000	8.286	7.840	8.427	7.974	8.147	8.000	8.194	8.161	7.818	
2000	7.992	7.827	7.912	7.881	8.003	7.876	7.896	7.992	7.813	
3000	7.870	7.826	7.847	7.834	7.960	7.840	7.833	7.888	7.811	
4000	7.843	7.821	7.835	7.824	7.867	7.828	7.825	7.834	7.811	
5000	7.838	7.817	7.831	7.822	7.858	7.828	7.819	7.828	7.811	

Table 2.16 Comparison of optimization results obtained for the retaining wall (Mononobe-Okabe method, case study 6)

Table 2.17 Optimized costs obtained for the retaining wall at five different stages of the best optimization run (Mononobe-Okabe method, case study 6)

Number of structural analyses	Optimized cost								
	Kaveh et al. [13]								
	ABC	BB-BC	CS	CSS	ICA	RO	TWO	WEO	
1000	8.652	8.277	8.312	8.283	8.462	8.394	8.601	8.789	8.266
2000	8.299	8.277	8.263	8.283	8.297	8.293	8.331	8.449	8.244
3000	8.250	8.254	8.251	8.282	8.268	8.253	8.260	8.322	8.241
4000	8.243	8.249	8.246	8.255	8.258	8.253	8.255	8.261	8.241
5000	8.241	8.247	8.245	8.254	8.253	8.253	8.249	8.260	8.241

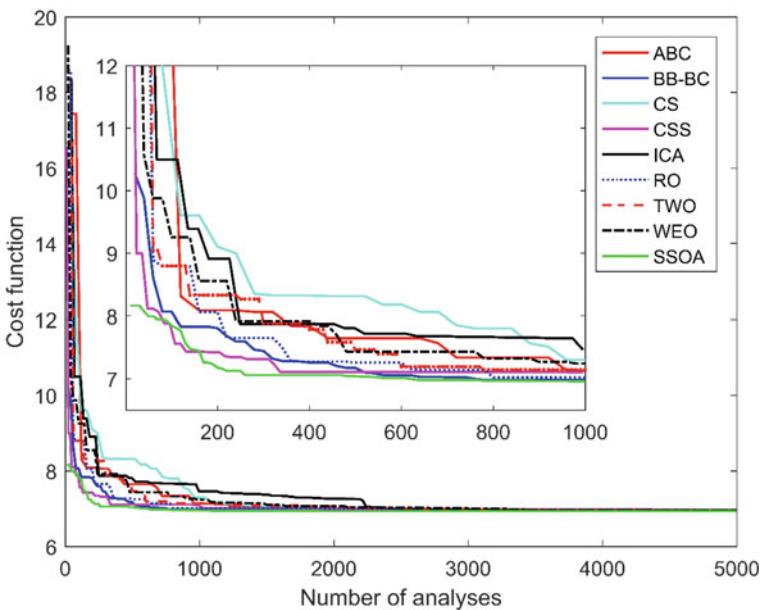


Fig. 2.6 Comparison of convergence histories for the first case study (Coulomb theory)

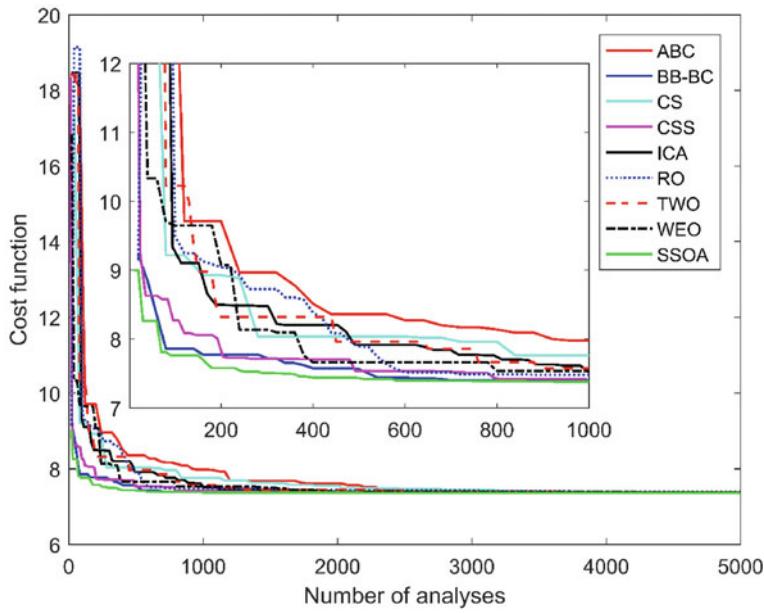


Fig. 2.7 Comparison of convergence histories for the second case study (Rankine theory)

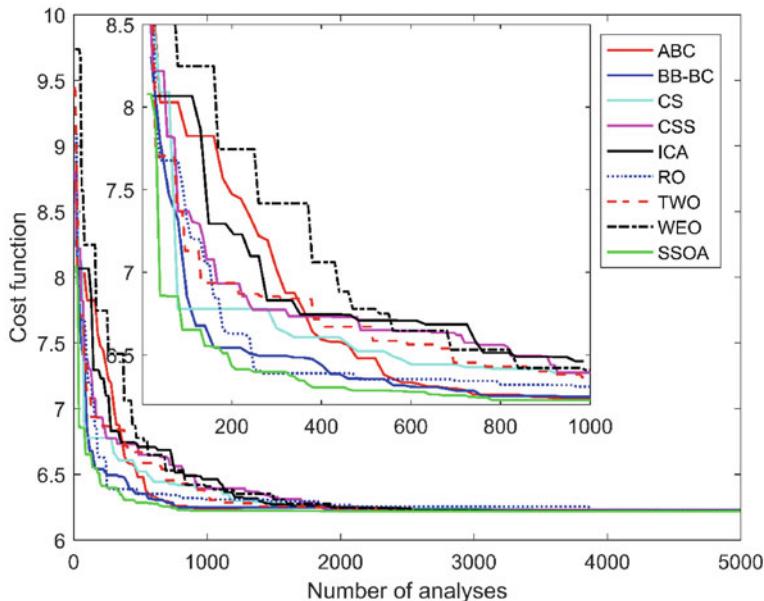


Fig. 2.8 Comparison of convergence histories for the third case study (Mononobe-Okabe method)

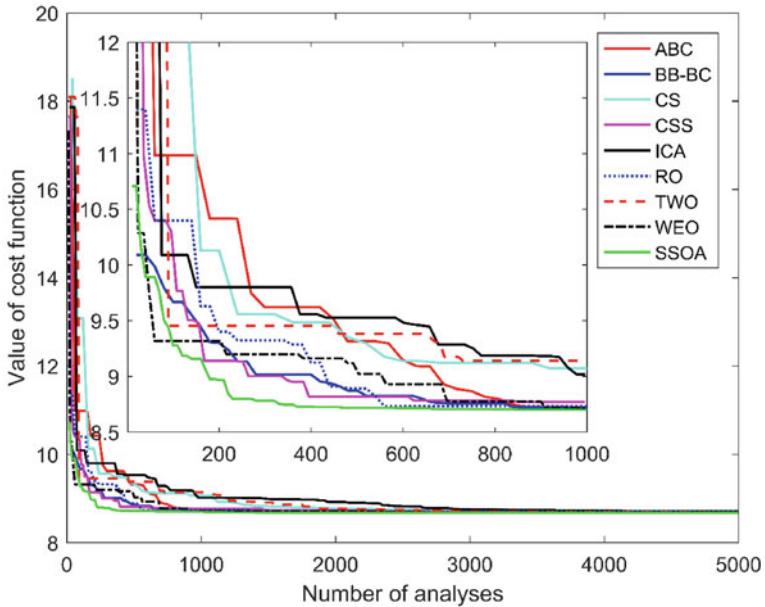


Fig. 2.9 Comparison of convergence histories for the fourth case study (Mononobe-Okabe method)

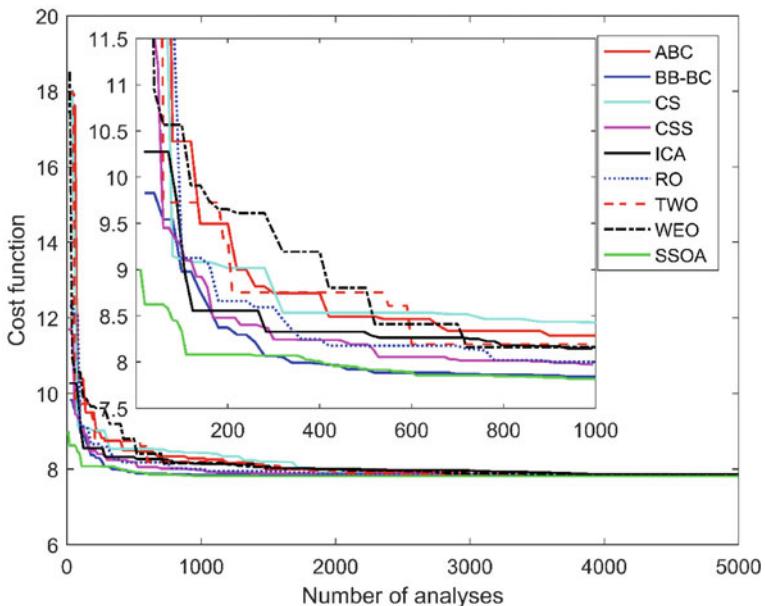


Fig. 2.10 Comparison of convergence histories for the fifth case study (Mononobe-Okabe method)

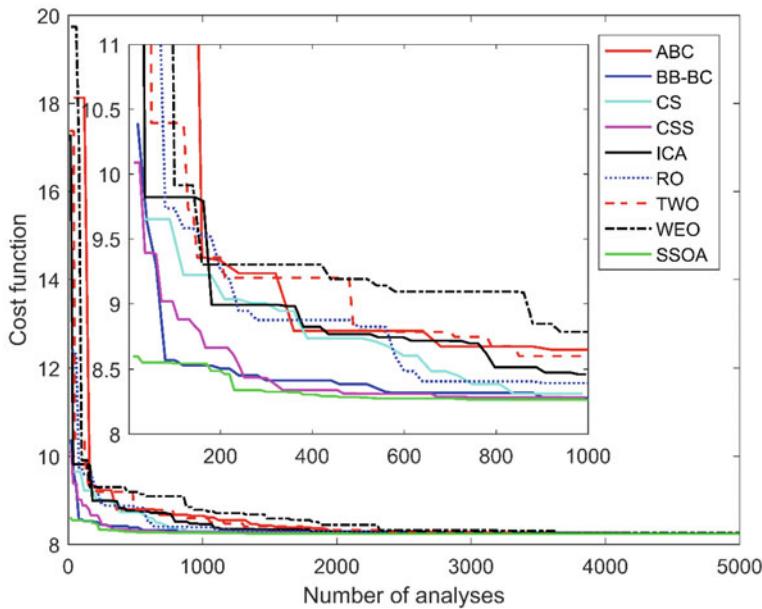


Fig. 2.11 Comparison of convergence histories for the sixth case study (Mononobe-Okabe method)

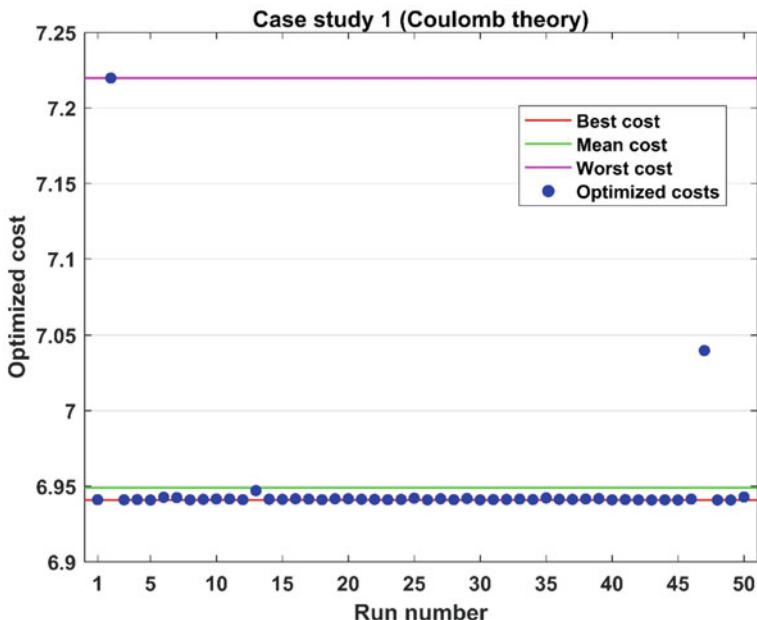


Fig. 2.12 Final costs obtained by ST-SSOA over 50 independent optimization runs for the first case study (Coulomb theory)

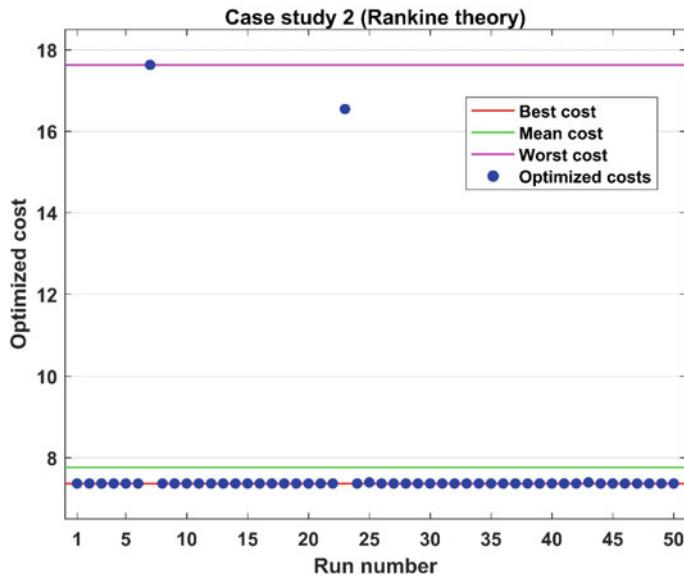


Fig. 2.13 Final costs obtained by ST-SSOA over 50 independent optimization runs for the second case study (Rankine theory)

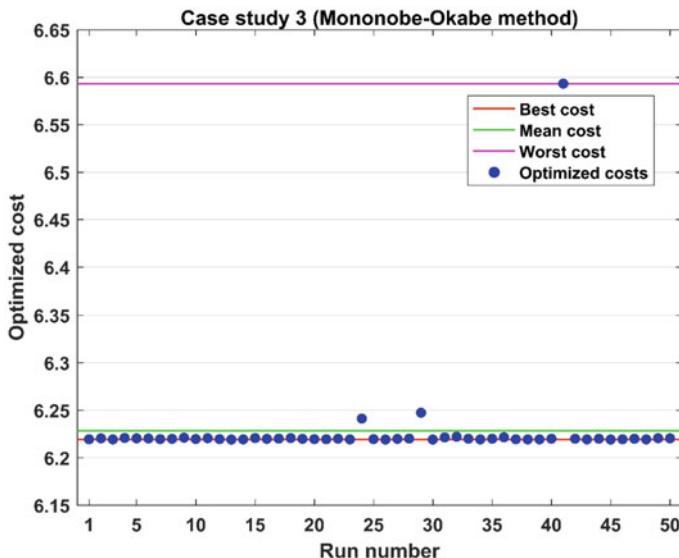


Fig. 2.14 Final costs obtained by ST-SSOA over 50 independent optimization runs for the third case study (Mononobe-Okabe method)

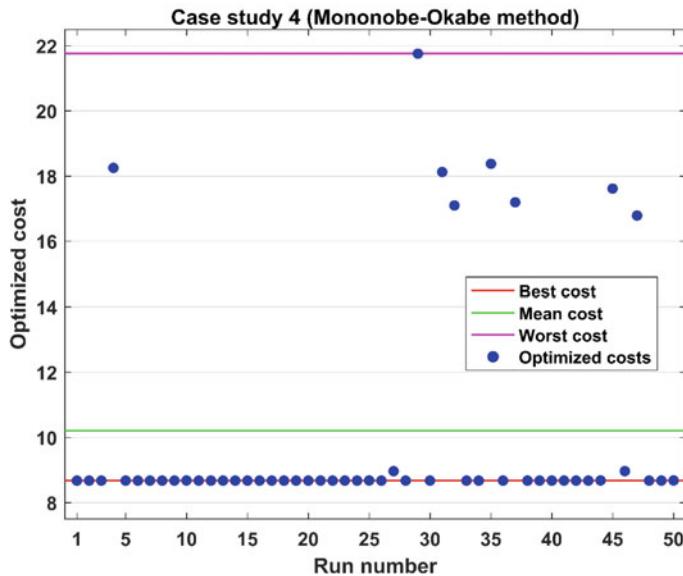


Fig. 2.15 Final costs obtained by ST-SSOA over 50 independent optimization runs for the fourth case study (Mononobe-Okabe method)

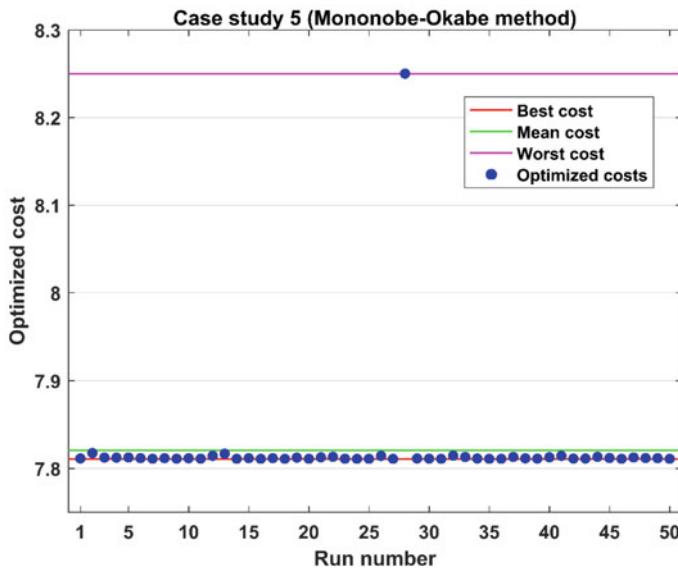


Fig. 2.16 Final costs obtained by ST-SSOA over 50 independent optimization runs for the fifth case study (Mononobe-Okabe method)

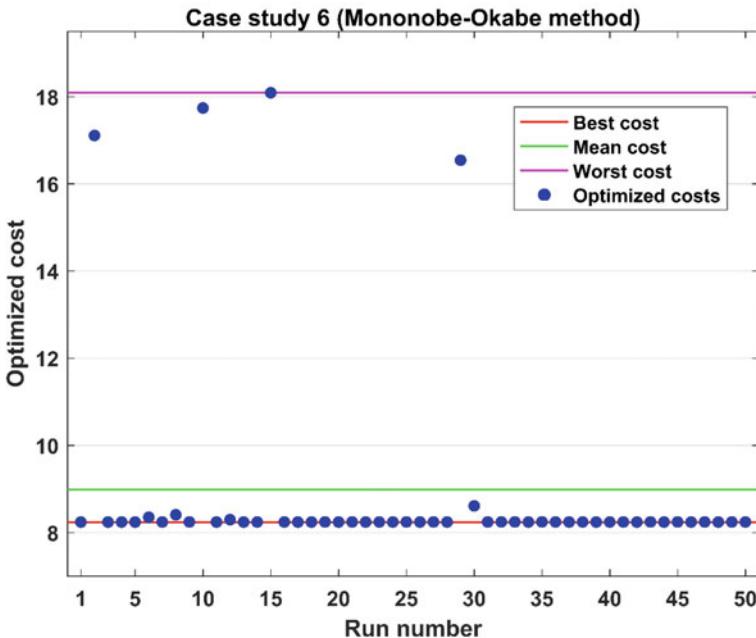


Fig. 2.17 Final costs obtained by ST-SSOA over 50 independent optimization runs for the sixth case study (Mononobe-Okabe method)

2.7 Concluding Remarks

In this study, some basic concepts of set theory are employed to generalize the description of the shuffled shepherd optimization algorithm (SSOA), which leads to the introduction of the set-theoretical SSOA (ST-SSOA). The set-theoretical framework of ST-SSOA is of a general nature and can be applied to almost all population-based metaheuristic algorithms. The main concept of the set-theoretical framework of ST-SSOA is to divide the set of candidate solutions into a number of smaller subsets having the same number of candidate solutions. ST-SSOA is then applied to the optimization of reinforced concrete cantilever retaining walls under both static and seismic loading conditions. The optimization objective is to minimize the total cost of the cantilever retaining wall subjected to structural stability and strength constraints. The Rankine and Coulomb theories are employed for estimating the static lateral earth pressures, and the Mononobe-Okabe method is used for the seismic loading conditions. It is found from the results that the retaining walls designed on the Rankine earth pressure theory are more expensive than those based on the Coulomb earth pressure theory. Furthermore, for case studies of the Mononobe-Okabe method, the total cost of the wall decreases as the vertical acceleration coefficient increases. However, as the horizontal acceleration coefficient increases, the total cost of the wall also increases. For all case studies, the shear capacity at the

critical section of the toe slab and the bearing strength of the base soil under the toe of the wall control the design. Optimization results show the superiority of ST-SSOA as compared to the other considered metaheuristic algorithms and reveal the high capability of ST-SSOA to deal with cost optimization of reinforced concrete cantilever retaining walls.

References

1. Kaveh A, Biabani Hamedani K, Zaerreza A (2021) A set theoretical shuffled shepherd optimization algorithm for optimal design of cantilever retaining wall structures. Eng Comput 37(4):3265–3282. <https://doi.org/10.1007/s00366-020-00999-9>
2. Kaveh A, Mahdavi VR (2015) Colliding bodies optimization: extensions and applications, 1st edn. Springer International Publishing, Switzerland. <https://doi.org/10.1007/978-3-319-19659-6>
3. Arya C (2009) Design of structural elements: concrete, steelwork, masonry and timber designs to British standards and Eurocodes, 3rd edn. CRC Press, London
4. Yazdani M, Azad A, Talatahari S (2013) Extended “mononobe-okabe” method for seismic design of retaining walls. J Appl Math 136132. <https://doi.org/10.1155/2013/136132>
5. Rao SS (2009) Engineering optimization: theory and practice. 4th ed. Wiley, Hoboken, New Jersey. <https://doi.org/10.1002/9780470549124>
6. Kaveh A, Bakhshpoori T (2019) Metaheuristics: outlines, MATLAB codes and examples. 1st ed. Springer Cham, Switzerland. <https://doi.org/10.1007/978-3-030-04067-3>
7. Camp CV, Akin A (2012) Design of retaining walls using big bang-big crunch optimization. J Struct Eng 138(3):438–448. [https://doi.org/10.1061/\(ASCE\)ST.1943-541X.0000461](https://doi.org/10.1061/(ASCE)ST.1943-541X.0000461)
8. Kaveh A, Behnam AF (2013) Charged system search algorithm for the optimum cost design of reinforced concrete cantilever retaining walls. Arab J Sci Eng 38(3):563–570. <https://doi.org/10.1007/s13369-012-0332-0>
9. Gandomi AH, Kashani AR, Roke DA, Mousavi M (2015) Optimization of retaining wall design using recent swarm intelligence techniques. Eng Struct 103:72–84. <https://doi.org/10.1016/j.engstruct.2015.08.034>
10. Kaveh A, Farhoudi N (2016) Dolphin echolocation optimization for design of cantilever retaining walls. Asian J Civ Eng 17(2):193–211
11. Tuner R, Bekdas G (2016) Teaching learning-based optimization for design of cantilever retaining walls. Struct Eng Mech 57(4):763–783. <https://doi.org/10.12989/sem.2016.57.4.763>
12. Yepes V, Alcala J, Perea C, Gonzalez-Vidosa F (2008) A parametric study of optimum earth-retaining walls by simulated annealing. Eng Struct 30(3):821–830. <https://doi.org/10.1016/j.engstruct.2007.05.023>
13. Kaveh A, Biabani Hamedani K, Bakhshpoori T (2020) Optimal design of reinforced concrete cantilever retaining walls utilizing eleven meta-heuristic algorithms: a comparative study. Period Polytech Civ Eng 64:156–168. <https://doi.org/10.3311/PPci.15217>
14. Kaveh A, Zaerreza A (2020) Shuffled shepherd optimization method: a new meta-heuristic algorithm. Eng Comput 37(7):2357–2389. <https://doi.org/10.1108/EC-10-2019-0481>
15. Kaveh A, Zaerreza A (2020) Size/layout optimization of truss structures using shuffled shepherd optimization method. Period Polytech Civ Eng 64(2):408–421. <https://doi.org/10.3311/PPci.15726>
16. American Concrete Institute (ACI) (2005) Building code requirements for structural concrete (ACI 318-05) and commentary (ACI 318R-05), Farmington Hills, Michigan, USA
17. American Association of State Highway and Transportation Officials (AASHTO) (2002) Standard specifications for highway bridges, 17th ed., USA
18. Das BM (2006) Principles of foundation engineering, 6th edn. Thomson India, New York

19. Das BM, Ramana GV (1993) Principles of soil dynamics, 2nd edn. PWS-KENT Publishing Company, Boston
20. McCormac JC, Brown RH (2015) Design of reinforced concrete. 10th ed. Wiley

Chapter 3

Set-Theoretical Variants of the Teaching–Learning-Based Optimization Algorithm for Structural Optimization with Frequency Constraints



3.1 Introduction

In this chapter, two enhanced set-theoretical variants of the teaching–learning-based optimization (TLBO) algorithm are proposed to solve truss optimization problems with multiple frequency constraints [1]. The main idea behind these algorithms, which is derived from set theory, is to divide the population of candidate solutions into a number of well-arranged subpopulations of the same size. In order to demonstrate the performance of the proposed algorithms, they are applied to four benchmark truss optimization problems with frequency constraints and the obtained results are compared with those of the standard TLBO and other optimization methods in the literature.

Set theory is a branch of mathematics that studies the sets and their properties. A set can be defined as a well-defined collection of objects called elements or members. The elements of a set may or may not have mathematical nature. Georg Cantor, one of the creators of the set theory, defined a set as follows [2]: “A set is a gathering together into a whole of definite, distinct objects of our perception or of our thought, which are called elements of the set.” The language of set theory can be used to express almost any mathematical concept. This capability can be regarded as one of the most important achievements of mathematics. Therefore, it can be said that the set theory provides a foundation for mathematics. Set theory has also found many applications in different branches of engineering including structural engineering. An example of this application can be found in the work of Behravesh et al. [3], wherein set theory is employed for configuration processing. In another work, Kaveh [4] employed a set of contours and its transversal in nodal ordering for bandwidth reduction. A recent example can be found in the work presented by Kaveh et al. [5], in which set theory is employed for modeling the shuffled shepherd optimization algorithm (SSOA).

Metaheuristics can be classified based on many different criteria, the most common of which is population-based search versus single-solution-based search. Single-solution-based algorithms manipulate and move a single candidate solution

in the search space, whereas population-based metaheuristics start their own search process with a set of candidate solutions called initial population. Considering this point, the initial population of a population-based metaheuristic could be viewed as a set with a certain number of elements. Consequently, a population-based metaheuristic could be viewed as an iterative improvement of a set of elements. In the first step, the initial population is generated within the specified lower and upper bounds of the design variables. Different strategies can be used to generate the initial population, including random generation, sequential diversification, parallel diversification, heuristic initialization, etc. However, random generation is the most common approach to obtain an initial population in metaheuristics. Next, the search process starts and continues until a predefined stopping criterion is fulfilled. During the search process, higher-level strategies act on the current set of candidate solutions (i.e., current population) and a new set of candidate solutions (i.e., new population) is generated. Next, a replacement strategy involving both current and new populations is applied to determine the population for the next generation. This process is repeated until a predefined stopping criterion is satisfied. The most important point in designing a metaheuristic is to keep a good balance between two conflicting aspects of exploration (global search or diversification) and exploitation (local search or intensification). However, this is not an easy task [6]. Exploration means the ability of the algorithm to explore the search space globally and thereby find diverse solutions, whereas exploitation means focusing the search on the most promising regions of the search space.

In this chapter, inspired by the concepts of set theory, a general set-theoretical framework is proposed for population-based metaheuristics. The main idea of the framework is to divide the population of candidate solutions into a number of well-arranged subpopulations of the same size. This framework aims to strike a good balance between diversification and intensification of the search. The framework, which is applicable to almost all population-based metaheuristics, makes it possible to design various variants of a single population-based metaheuristic. The teaching–learning-based optimization algorithm is one of the most efficient population-based metaheuristics developed by Rao et al. [7] in 2011. In this study, the proposed framework is applied to the standard TLBO and two set-theoretical variants of TLBO called ordered set-theoretical TLBO (OST-TLBO) and set-theoretical multi-phase TLBO (STMP-TLBO) are proposed. The STMP-TLBO algorithm is a multi-phase variant of the proposed OST-TLBO. The performance of the proposed set-theoretical variants of TLBO is demonstrated through four truss optimization problems with multiple natural frequency constraints. The results obtained by the proposed algorithms are compared with those of the standard TLBO and some other methods reported in the literature. The reason for choosing this type of optimization problem is that the objective function and constraints of the problem are often highly nonlinear and non-convex, making it difficult to obtain an optimal solution. The most common problem with frequency-constrained optimization seems to be the high sensitivity of mode shapes to shape modifications, which means that mode shapes and corresponding obtained frequencies may switch during the optimization process. This may

lead to significant changes in natural frequencies, which causes convergence difficulties. In the past few decades, structural optimization with frequency constraints has attracted the attention of many researchers. Bellagamba and Yang [8] were one of the first researchers to study the minimum-mass design of truss structures with natural frequency constraints. Grandhi and Venkayyat [9] presented a design optimization algorithm based on an optimality criterion approach to minimize the structural weight with multiple frequency constraints. Gomes [10] investigated the performance of particle swarm optimization (PSO) algorithm for size and shape optimization of truss structures under frequency constraints. In a similar work, Miguel and Fadel Miguel [11] employed the harmony search (HS) and firefly algorithm (FA) to solve truss shape and size optimization problems with multiple frequency constraints. Kaveh and Javadi [12] proposed a hybridization of PSO, HS and ray optimization (RO) for solving size and shape truss optimization problems with multiple frequency constraints. Kaveh and Ilchi Ghazaan [13] applied improved ray optimization (IRO) algorithm for size and layout optimization of truss structures with multiple natural frequency constraints. Kaveh and Ilchi Ghazaan [14] employed vibrating particles system (VPS) algorithm to solve the shape and size optimization problems for truss structures with frequency constraints. Kaveh and Zolghadr [15] used cyclical parthenogenesis algorithm (CPA) for size and shape optimization of truss structures subjected to frequency constraints.

The rest of this chapter is organized as follows: The standard TLBO algorithm is briefly introduced in Sect. 3.2. In Sect. 3.3, two set-theoretical variants of TLBO are proposed. In Sect. 3.4, the formulation of the truss optimization problem with frequency constraints is presented. In Sect. 3.5, four numerical examples are investigated to demonstrate the effectiveness and accuracy of the proposed algorithms. Finally, the last section concludes the study.

3.2 Teaching–Learning-Based Optimization (TLBO) Algorithm

The basic inspiration of the TLBO algorithm comes from the traditional process of education in schools. The traditional education process in schools includes two distinct phases of teaching and learning. The teaching phase involves the transmission of knowledge from teacher to students, and the learning phase involves the transfer of knowledge between students. Analogously, each iteration of the TLBO algorithm consists of two sequential phases namely teacher phase and learner phase. The teacher phase contributes to the intensification of the search process, while the learner phase favors the diversification of the search process. Similar to almost all population-based metaheuristic algorithms, the TLBO algorithm starts the search process with a set of randomly generated candidate solutions (i.e., a class of students), which constitute the initial population. The knowledge level of the students represents the fitness function values. In each iteration of the TLBO algorithms, students are updated iteratively

based on the teacher and learner phases. In the teacher phase, the best student of the population of the current iteration (i.e., the candidate solution with the best penalized objective function value) is considered as the teacher. Next, the knowledge level of the students is improved by the influence of the teacher's knowledge on the performance of the students in the classroom. On the other hand, in the learner phase, the knowledge level of each student is updated by sharing his/her knowledge with another randomly selected one. It should be noted that after each phase, a replacement strategy is applied to determine the population of the next iteration. The steps of the TLBO algorithm are formulated as follows [6]:

Step one (forming the initial population): The initial population of the TLBO algorithm can be viewed as a classroom with a certain number of students (e.g., nS). The initial population S is generated by the following equation:

$$S = LB + (UB - LB) \times \text{rand}(nS, nV) \quad (3.1)$$

where nS represents the number of students (i.e., the population size); nV is the number of design variables of the problem; LB and UB denote the lower and upper bound vectors of the design variables, respectively; and rand is a random number from the interval $[0, 1]$.

Step two (teacher phase): In the first step, students are evaluated and penalized objective function values ($PFit$) are calculated for all students. Next, the best student of the class (i.e., the student with the best penalized objective function value) is selected as the teacher (T). Afterwards, the students' knowledge level is updated based on the difference between the knowledge level of the teacher and the average knowledge of the students (Ave_S). The teacher phase is formulated as follows:

$$\text{Stepsize}_i = T - TF_i \times Ave_S; i = 1, 2, \dots, nS \quad (3.2)$$

$$NewS_i = S_i + rand_{i,j} \times \text{Stepsize}_i; i = 1, 2, \dots, nS \text{ and } i = 1, 2, \dots, nV \quad (3.3)$$

where Stepsize_i is the step size of movement of the i -th student; S_i is the current position of the i -th student; $NewS_i$ is the new position of the i -th student; $rand_{i,j}$ is a random number chosen from the interval $[0, 1]$; and TF_i is the teacher factor, and its value is either 1 or 2. It is worth mentioning that the teacher factor is considered for controlling the influence of the teacher's knowledge on the average knowledge of the class.

Step three (replacement strategy): In this step, newly generated students are evaluated and replaced with the corresponding current ones in a simple greedy manner. In this way, if a newly generated student has a better penalized objective function value than the corresponding current one, then the newly generated student is preferred to the old one. Therefore, a new class with nS students is formed.

Step four (learner phase): In the learner phase, students learn through interaction and communicating with each other. In the first step, each student S_i is randomly paired with another student (S_{rs}) in the class to interact with ($i \neq rs$). Next, each student tries to improve his/her knowledge level by learning from the paired one. For this purpose, the penalized objective function values of S_i and S_{rs} are compared with each other. If the penalized objective function value of the student S_i is better than that of the student S_{rs} (i.e., $PFit_i < PFit_{rs}$ for a minimization optimization problem), then the student S_{rs} moves toward the student S_i , and vice versa. The learner phase is formulated as follows:

$$Stepsize_i = \begin{cases} S_i - S_{rs}, & PFit_i < PFit_{rs}; \\ S_{rs} - S_i, & PFit_i \geq PFit_{rs} \end{cases}; i = 1, 2, \dots, nS \quad (3.4)$$

$$NewS_i = S_i + rand_{i,j} \times StepSize_i; i = 1, 2, \dots, nS \text{ and } i = 1, 2, \dots, nV \quad (3.5)$$

Step five (replacement strategy): The replacement strategy is applied again.

Step six (termination criteria): If the termination criterion of the algorithm is satisfied, the optimization process is terminated. Otherwise, go to step two.

3.3 Set-Theoretical Variants of the Teaching–Learning-Based Optimization Algorithm

Like other population-based metaheuristic algorithms, the standard TLBO starts the search process with a set of randomly generated initial solutions called learners. The initial population of the TLBO algorithm can be viewed as a set of elements. Therefore, each element is equivalent to a learner. Each element has a penalized objective function value. The weight of each element is defined as the value of its penalized objective function [5]. Therefore, a unique weight is assigned to each element. As mentioned earlier, the main idea of the proposed set-theoretical framework is to divide the set of elements (here class of learners) into well-arranged subsets (here subclasses). In set theory, the number of elements of a set A is defined as the cardinality of the set, usually denoted by $|A|$. When the numbers of a set of numbers are ordered from the smallest number to the largest one, the numbers are said to be in an ascending order. On the other hand, when the numbers are ordered from the largest number to the smallest one, the numbers are said to be in a descending order. Inspired by the concepts of set theory and based on the definitions introduced above, set-theoretical variants of the TLBO algorithm are proposed as follows.

3.3.1 Ordered Set-Theoretical Teaching–Learning-Based Optimization (OST-TLBO) Algorithm

The ordered set-theoretical teaching–learning-based optimization (OST-TLBO) algorithm is based on the idea of dividing the population of solutions into well-arranged subpopulations of the same cardinality. The OST-TLBO algorithm is stated in the three following steps:

Step one (forming the subsets): Let us consider an initial population of nE candidate solutions. In the first step, based on a procedure proposed by Kaveh et al. [5], the set of elements is divided into a certain number of smaller well-arranged subsets (e.g., m subsets) of the same cardinality. For this purpose, first, m null subsets are considered. Next, the elements (i.e., initial candidate solutions) are evaluated and sorted in ascending order of weight (i.e., penalized objective function value). In the first step of forming the subsets, the first m elements of the ordered set of elements are selected and each element is placed in a different null subset randomly. As a result, at the end of the first step, each subset contains one element. In the second step, the next m elements of the ordered set are selected and each element is placed in a different subset randomly. At the end of this step, each subset contains two elements. This process continues until all elements of the ordered set are selected and placed in the subsets. At the end of the last step, all subsets have an equal number of elements. It is obvious that the best and worst elements of each subset are the first and last ones, respectively. It is worth mentioning the subsets obtained through this procedure are close to each other in terms of average weight [5]. If nS is the number of elements of each subset, it can be said that:

$$nE = m \times nS \quad (3.6)$$

Step two (main body of the TLBO algorithm): The main body of the TLBO algorithm is performed for each subset separately. For this purpose, steps two to five of the TLBO algorithm, as described in Sect. 3.2, are executed once for each subset separately. Therefore, m new subsets, each of which contains nS new elements (i.e., nS new candidate solutions), are generated. These subsets collectively form the set of new elements.

Step three (termination criteria): If the termination criterion of the algorithm is satisfied, the algorithm is terminated. Otherwise, go to step one. In this research, the maximum number of objective function evaluations ($MaxNFEs$) is considered as the termination criterion of the optimization process.

The first step of the OST-TLBO algorithm ensures both the exploration and exploitation phases of the search process at the same time; thus, it is difficult to distinguish between these two phases in the first step. The division of the population of solutions into smaller well-arranged subpopulations encourages exploration of the search space. On the other hand, the procedure employed for dividing the population increases the exploitation of the search space. Hence, it is expected that the OST-TLBO algorithm demonstrates a slow convergence rate in the early stages of the

search process, but shows a fast convergence rate in the later stages of search. This is due to the fact that as the search process progresses, the number of subsets decreases (i.e., subpopulations become larger), which leads to strengthening the exploitation capability of the algorithm. The pseudocode of the OST-TLBO algorithm is provided as follows:

Initialization:

Set the algorithm parameters: m , nS , and MaxNFEs .

Generate a set of initial solutions or elements (E) randomly.

Evaluate the set of elements.

$NFEs = 0$;

Cyclic body of the OST-TLBO algorithm:

While $NFEs < \text{MaxNFEs}$

Step 1: Sort the set of elements in ascending order of weight (i.e., penalized objective function).

Step 2: Form the subsets based on the procedure proposed by Kaveh et al. [5].

Step 3: Generate the subsets of new elements ($newE$) based on the teacher phase.

Step 4: Evaluate the newly generated subsets ($newE$) and apply the replacement strategy to update the subsets.

Update the number of function evaluations ($NFEs = NFEs + nE$).

Step 5: Generate the subsets of new elements ($newE$) based on the learner phase.

Step 6: Evaluate the newly generated subsets ($newE$) and apply the replacement strategy to update the subsets.

Update the number of function evaluations ($NFEs = NFEs + nE$).

Step 7: Collect the subsets and form the set of elements of the next iteration.

Monitor the best element found by the algorithm so far.

End while.

The flowchart of the OST-TLBO algorithm is shown in Fig. 3.1.

3.3.2 Set-Theoretical Multi-Phase Teaching–Learning-Based Optimization (STMP-TLBO) Algorithm

In this section, based on the OST-TLBO algorithm (presented in Sect. 3.3.1), a multi-phase variant of the OST-TLBO algorithm called the set-theoretical multi-phase teaching–learning-based optimization (STMP-TLBO) algorithm is proposed. The STMP-TLBO algorithm operates in several phases (e.g., k phases). Within each phase of the STMP-TLBO algorithm, a self-contained OST-TLBO algorithm is executed with a specific number of subsets. The first phase of the STMP-TLBO algorithm

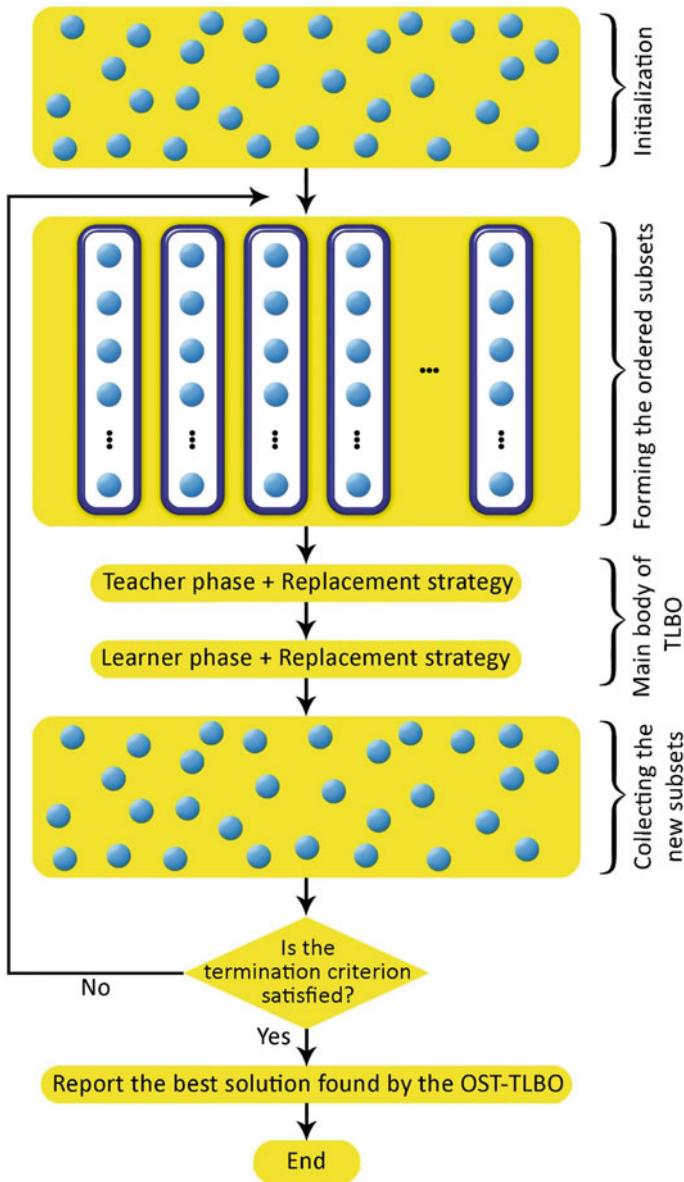


Fig. 3.1 Flowchart of the OST-TLBO algorithm

uses the initial population as input. In the following phases, the input is the output of the previous phase. In other words, different phases of the STPM-TLBO algorithm are executed in a sequence without any cooperation. Each phase continues until the number of objective function evaluations ($NFEs$) reaches a preset threshold. It is noted that different phases differ only in the number of subsets they work with. The number of subsets decreases nonlinearly in a few steps from the integer n_0 ($2 < n_0 \leq nE$) at the first phase to two at the k -th phase. The STMP-TLBO algorithm has three parameters: population size (nE), number of subsets of the first phase (n_0), and maximum number of objective function evaluations ($maxNFEs$). It should be noted that for the STMP-TLBO algorithm, nE needs to be an even number. Furthermore, n_0 needs to be one of the divisors of nE ($2 < n_0 \leq nE$). The number of phases of the STMP-TLBO algorithm (k) is determined as follows: Let D denote the set of all divisors of nE which are greater than one and less than $n_0 + 1$. Thus,

$$D = \{n | n \in N; 2 \leq n \leq n_0; mod(nE, n) = 0\} \quad (3.7)$$

$$k = |D| \quad (3.8)$$

where N is the set of natural numbers.

For example, let us consider an initial population of size 30, which is an even number. Additionally, the number of subsets of the first phase is considered to be 10, which is greater than two and less than 31 and is a divisor of 30. The set D is defined as: $D = \{2, 3, 5, 6, 10\}$. Therefore, the number of phases of the algorithm will be equal to: $k = |D| = 5$. Thus, the STMP-TLBO algorithm will be executed in five different phases. Table 3.1 shows different phases of the STMP-TLBO algorithm for $nE = 30$ and $n_0 = 10$.

In the early phases of the STMP-TLBO algorithm, the search space is explored by a relatively large number of subpopulations. This may provide a larger exploration of the search space at the beginning of the search process. As a result, in the early stages of the search process, the convergence rate of the STMP-TLBO algorithm is expected to be slower than that of the standard TLBO. In the next

Table 3.1 Different phases of the STMP-TLBO algorithm for $nE = 30$ and $n_0 = 10$

STMP-TLBO	$nE = 30$ and $n_0 = 10$				
Phase number	1	2	3	4	5
Number of subsets	10	6	5	3	2
Cardinality of subsets	3	5	6	10	15
Number of objective function evaluations	$maxNFEs/5$	$maxNFEs/5$	$maxNFEs/5$	$maxNFEs/5$	$maxNFEs/5$

phases, the number of subpopulations decreases gradually, whereas subpopulations become larger. This may improve the exploitation of the search space. It is noted that the number of subpopulations of the OST-TLBO algorithm is considered to be a constant value during the search process. However, the number of subpopulations of the STMP-TLBO algorithm decreases gradually. This may improve the convergence performance of the STMP-TLBO algorithm compared to the OST-TLBO algorithm. The pseudocode of the STMP-TLBO algorithm is provided below:

Initialization:

Set the algorithm parameters: nE , n_0 , and MaxNFEs .

Generate a set of initial solutions or elements (E) randomly.

Evaluate the set of elements.

Obtain the set D and its cardinality (k).

$i = 0$; $NFEs = 0$;

Cyclic body of the STMP-TLBO algorithm:

While $i < k$

$i = i + 1$;

While $NFEs < i \times \text{MaxNFEs}/k$

Step 1: Sort the set of elements in ascending order of weight (i.e., penalized objective function).

Step 2: Form the subsets of the i -th phase based on the procedure proposed by Kaveh et al. [5].

Step 3: Generate the subsets of new elements ($newE$) based on the teacher phase.

Step 4: Evaluate the newly generated subsets ($newE$) and apply the replacement strategy to update the subsets.

Update the number of function evaluations ($NFEs = NFEs + nE$).

Step 5: Generate the subsets of new elements ($newE$) based on the learner phase.

Step 6: Evaluate the newly generated subsets ($newE$) and apply the replacement strategy to update the subsets.

Update the number of function evaluations ($NFEs = NFEs + nE$).

Step 7: Collect the subsets and form the set of elements of the next iteration.

Monitor the best element found by the STMP-TLBO algorithm so far.

End while

End while.

The flowchart of the STMP-TLBO algorithm is presented in Fig. 3.2.

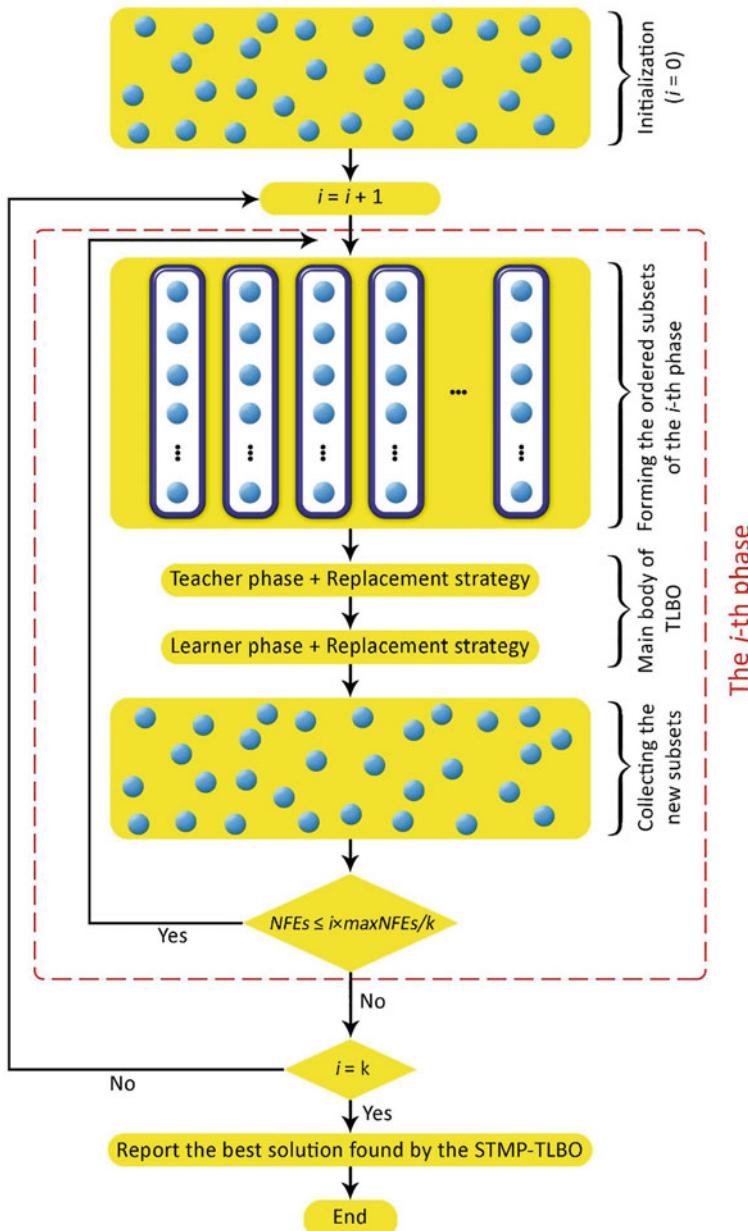


Fig. 3.2 Flowchart of the STMP-TLBO algorithm

3.4 Formulation of Truss Optimization Problem with Frequency Constraints

In this study, the proposed set-theoretical algorithms are applied to solve size and shape optimization of truss structures with multiple frequency constraint. In a truss optimization problem with frequency constraints, the objective is to minimize the weight of the structure while satisfying some constraints on natural frequencies. The cross-sectional areas of the members and the nodal coordinates are considered as continuous design variables. However, it is assumed that the topology of the structure is predefined and kept fixed during the optimization process. The truss optimization problem with multiple frequency constraints is formulated mathematically as follows [15]:

$$\text{Find: } \{X\} = [x_1, x_2, \dots, x_{nDV}], x_i \in S_i; i = 1, 2, \dots, nDV \quad (3.9)$$

$$\text{to minimize: } PFit(\{X\}) = W(\{X\}) \times f_{penalty}(X) \quad (3.10)$$

$$\text{subject to: } \begin{cases} \omega_j \geq \omega_j^* \text{ for some natural frequencies } j \\ \omega_k \leq \omega_k^* \text{ for some natural frequencies } k \\ x_i^L \leq x_i \leq x_i^U i = 1, 2, \dots, nDV \end{cases} \quad (3.11)$$

where

$$W(\{X\}) = \sum_{i=1}^{nE} \rho_i \times A_i \times L_i \quad (3.12)$$

In the above equation, $\{X\}$ is the vector of design variables including the cross-sectional areas of the members and the nodal coordinates; x_i represents the i -th design variable; nDV is the number of design variables; S_i is the allowable range of the i -th design variable; $W(\{X\})$ is the objective function which is the weight of the truss structure; $f_{penalty}(X)$ is the penalty function that is employed to handle the constraints; $PFit(\{X\})$ is the penalized objective function to be minimized; nE is the number of truss members; ρ_i , A_i , and L_i are the material density, cross-sectional area, and length of the i -th truss member, respectively; x_i^L and x_i^U are the lower and upper bounds of i -th design variable, respectively; ω_j is the j -th natural frequency of the structure and ω_j^* is its lower bound; and ω_k is the k -th natural frequency of the structure and ω_k^* is its upper bound. The design variable x_i can vary continuously within a predefined range which is denoted by S_i :

$$S_i = \{x_i | x_i \in [x_i^L, x_i^U]\} \quad (3.13)$$

The free vibration analysis of a structural system leads to a generalized eigenvalue problem which can be formulated mathematically in the following form:

$$K\phi_n = \gamma_n M\phi_n; n = 1, 2, \dots, N \quad (3.14)$$

where K is the stiffness matrix of the structural system; M is the mass matrix of the structural system; and ϕ_n is the n -th natural mode of vibration of the system; γ_n is the n -th eigenvalue of the system; and N is the number of degrees of freedom of the system. It is worth mentioning that the n -th circular frequency (ω_n) and the n -th period (T_n) are related to the n -th eigenvalue (γ_n) by the following equation:

$$\gamma_i = \omega_i^2 = (2\pi/T_i)^2 \quad (3.15)$$

In this study, a well-known penalty function is employed to handle the frequency constraints. In penalizing strategies, infeasible solutions are penalized to force the search towards feasible regions. The penalty function is defined as follows [15]:

$$f_{penalty}(X) = (1 + \varepsilon_1 \times v)^{\varepsilon_2}; v = \sum_{i=1}^{nC} v_i \quad (3.16)$$

where nC is the number of frequency constraints; v denotes the sum of the constraint violations; v_i represents the degree of violation of the i -th frequency constraint; and ε_1 and ε_2 are the parameters of the penalty function. These parameters should be tuned in such a way that a good balance between exploration and exploitation is achieved. In this study, the parameter ε_1 is set to unity. However, the value of the parameter ε_2 is set to 1.5 at the beginning of the optimization process and it grows to 3 at the end of the optimization process. If the i th constraint is fulfilled, then the value of v_i is set to zero; otherwise, it is calculated by the severity of violation. This can be expressed as follows:

$$v_i = \begin{cases} \left| 1 - \frac{\omega_i}{\omega_i^*} \right|, & \text{the } i\text{-th frequency constraint is violated} \\ 0, & \text{otherwise} \end{cases} \quad (3.17)$$

3.5 Numerical Examples

In this section, to demonstrate the efficiency and effectiveness of the proposed algorithms, four well-known design optimization problems of truss structures with multiple frequency constraints are provided as follows: size and shape optimization of a 37-bar planar truss with 19 design variables (five shape variables and 14 sizing variables), size and shape optimization of a 52-bar dome truss with 13 design

variables (five shape variables and eight sizing variables), size optimization of a 120-bar dome-like truss with eight design variables, and size optimization of a 200-bar planar truss with 29 design variables. The results obtained by the OST-TLBO and STMP-TLBO algorithms are compared with those of the standard TLBO and other methods in the literature. For each problem, five different population sizes of 10, 20, 30, 40, and 50 are considered for the standard TLBO algorithm, and the best result is reported. The finite element models and the optimization algorithms are implemented in the Matlab environment.

3.5.1 A 37-Bar Planar Truss

A simply supported 37-bar planar truss is considered as the first size and shape optimization problem. Figure 3.3 shows the initial layout of the structure. A non-structural mass of 10 kg is attached to all free nodes of the lower chord. Table 3.2 provides the material properties, variable bounds, and frequency constraints of the problem. The elements of the lower chord are modeled as bar elements with constant rectangular cross-sectional area of $4 \times 10^{-3} \text{ m}^2$. However, the cross-sectional areas of the other elements of the truss can be changed during the optimization process. As shown in Table 3.3, these elements are categorized into 14 groups of elements according to the symmetry of the structure with respect to the y-axis. All nodes of the upper chord are allowed to move in the y-axis direction in a symmetric way. For this reason, the nodes of the upper chord are categorized into five groups according to the symmetry of the structure with respect to the y-axis. Therefore, there are 14 sizing variables and five shape variables. As shown in Table 3.2, the frequency constraints are imposed on the first three natural frequencies of the structure. This problem has been previously investigated by many researchers using various optimization methods: Gomes [10] using particle swarm optimization (PSO) algorithms, Miguel and Fadel Miguel [11] using harmony search (HS) and firefly algorithm (FA), Kaveh and Ilchi Ghazaan [13] utilizing improved ray optimization (IRO) algorithm, and Kaveh and Zolghadr [15] using cyclical parthenogenesis algorithm (CPA).

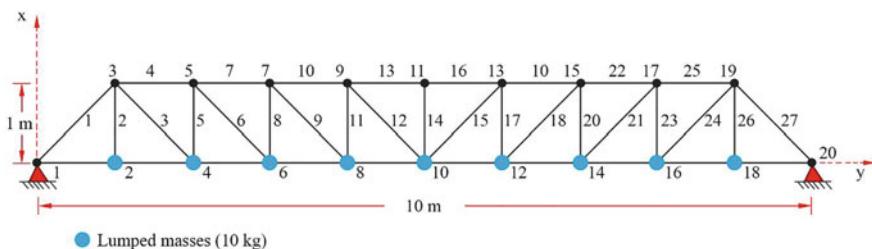


Fig. 3.3 Schematic of the initial layout of the 37-bar planar truss

Table 3.2 Material properties, cross-sectional area bounds, and frequency constraints of the 37-bar planar truss problem

Property (unit)	37-bar planar truss problem
Elasticity modulus (GPA)	210
Material density (kg/m^3)	7800
Added mass (kg)	10
Cross-sectional area bounds (cm^2)	$1 \leq A_i \leq 10$
Frequency constraints (Hz)	$\omega_1 \geq 20, \omega_2 \geq 40, \omega_3 \geq 60$

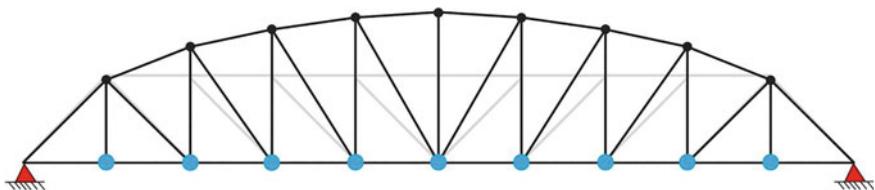
Table 3.3 provides a comparison of the results obtained by the OST-TLBO and STMP-TLBO algorithms with those of the standard TLBO and some other methods in the literature. Due to the stochastic nature of the optimization process, 30 independent runs are conducted for each algorithm, and the optimal design of the best run is reported. It is noted that the initial population of the standard TLBO, OST-TLBO and STMP-TLBO are generated randomly. From Table 3.3 it is found that the best weight obtained by the standard TLBO is slightly better than those of the OST-TLBO and STMP-TLBO, while the results obtained by the OST-TLBO and STMP-TLBO had smaller standard deviations and average weights than those obtained by the standard TLBO. Table 3.4 lists the first five natural frequencies of the best designs obtained by the present algorithms and other compared methods for the 37-bar truss structure. It is seen that none of the methods violate the frequency constraints. Figure 3.4 shows the optimized layout of the 37-bar planar truss structure found by the OST-TLBO. The optimized weights found by the standard TLBO and its set-theoretical variants at five different stages of the optimization process are listed in Table 3.5. As the table demonstrates, the STMP-TLBO has a slower rate of convergence than the standard TLBO in the early iterations, however it converges more effectively than the standard TLBO in the next iterations. The optimized weights of the 37-bar planar truss obtained by the standard TLBO, OST-TLBO, and STMP-TLBO over 30 independent runs are depicted in Figs. 3.5, 3.6, and 3.7, respectively. A close examination of the figures indicates that the OST-TLBO and STMP-TLBO perform better than the standard TLBO in terms of average weight and worst weight. Figure 3.8 compares the convergence histories of the best solutions found by the standard TLBO, OST-TLBO, and STMP-TLBO for the 37-bar planar truss. A magnified view is added to the figure to show the details of the convergence histories. It can be seen from the figure that in the early stages of the optimization process (i.e., within about the first 5000 structural analyses), the STMP-TLBO has a slower convergence rate compared to the standard TLBO, which can be attributed to the high exploration ability of the STMP-TLBO in the early iterations. However, after about the first 5000 structural analyses, the STMP-TLBO has a higher convergence rate than the standard TLBO.

Table 3.3 Comparison of optimal results obtained for the 37-bar planar truss

Design variable: Y (m); A (cm^2)	PSO [10]	HS [11]	FA [11]	IRO [13]	CPA [15]	This study		
						TLBO	OST-TLBO	STMP-TLBO
Y_3, Y_{19}	0.9637	0.8415	0.9392	0.9641	0.9592	0.9605	0.9435	0.9703
Y_5, Y_{17}	1.3978	1.2409	1.3270	1.3490	1.3480	1.3327	1.3249	1.3614
Y_7, Y_{15}	1.5929	1.4464	1.5063	1.5422	1.5236	1.5225	1.5261	1.5318
Y_9, Y_{13}	1.8812	1.5334	1.6086	1.6719	1.6617	1.6585	1.6553	1.6602
Y_{11}	2.0856	1.5971	1.6679	1.7466	1.7431	1.7310	1.7239	1.7404
A_1, A_{27}	2.6796	3.2031	2.9838	2.9082	2.9166	2.9437	2.9580	2.9972
A_2, A_{26}	1.1568	1.1107	1.1098	1.0494	1.0089	1.0037	1.0000	1.0490
A_3, A_{25}	2.3476	1.1871	1.0091	1.0020	1.0000	1.0000	1.0013	1.0000
A_4, A_{24}	1.7182	3.3281	2.5955	2.6153	2.3965	2.5828	2.5521	2.5917
A_5, A_{23}	1.2751	1.4057	1.2610	1.0915	1.3489	1.2249	1.1537	1.1576
A_6, A_{22}	1.4819	1.0883	1.1975	1.2766	1.2240	1.1317	1.2163	1.2046
A_7, A_{21}	4.6850	2.1881	2.4264	2.7346	2.5091	2.5539	2.6149	2.5445
A_8, A_{20}	1.1246	1.2223	1.3588	1.4154	1.2656	1.3762	1.4155	1.4090
A_9, A_{19}	2.1214	1.7033	1.4771	1.5225	1.4866	1.5434	1.5514	1.4821
A_{10}, A_{18}	3.8600	3.1885	2.5648	2.2575	2.5584	2.5360	2.3686	2.4796
A_{11}, A_{17}	2.9817	1.0100	1.1295	1.3206	1.1977	1.2413	1.2898	1.1702
A_{12}, A_{16}	1.2021	1.4074	1.3199	1.2462	1.4003	1.3021	1.2827	1.3042
A_{13}, A_{15}	1.2563	2.8499	2.9217	2.2398	2.5323	2.4440	2.5001	2.3958
A_{14}	3.3276	1.0269	1.0004	1.0000	1.0000	1.0000	1.0022	1.0000
Weight (kg)	377.20	361.50	360.05	359.97	359.93	359.842	359.888	359.854
Worst weight (kg)	N/A	N/A	N/A	N/A	N/A	361.443	360.398	360.261
Average weight (kg)	381.2	362.04	360.37	360.85	360.93	360.218	360.084	360.055
Standard deviation (kg)	4.26	0.52	0.26	0.78	0.65	0.329	0.110	0.097
Maximum number of structural analyses	12,500	20,000	50,000	18,000	12,800	20,000	20,000	20,000
Number of runs	5	5	5	20	50	30	30	30

Table 3.4 First five natural frequencies (Hz) of the 37-bar planar truss evaluated at optimal designs

Frequency number	PSO [10]	HS [11]	FA [11]	IRO [13]	CPA [15]	This study		
						TLBO	OST-TLBO	STMP-TLBO
1	20.0001	20.0037	20.0024	20.0004	20.0000	20.0052	20.0107	20.0055
2	40.0003	40.0050	40.0019	40.0351	40.0002	40.0046	40.0232	40.0015
3	60.0001	60.0082	60.0043	60.0013	60.0024	60.0046	60.0075	60.0306
4	73.0440	77.9753	77.2153	76.3818	77.3492	76.1458	76.4611	76.0899
5	89.8240	99.2564	96.9900	96.7195	96.4671	96.3219	96.6636	96.2735

**Fig. 3.4** Optimized layout of the 37-bar planar truss structure found by the OST-TLBO algorithm**Table 3.5** Optimized weights for the 37-bar planar truss at five different stages of the optimization process (best runs)

Number of structural analyses	Optimized weight (kg)		
	TLBO	OST-TLBO	STMP-TLBO
2000	383.091	377.069	386.843
4000	364.889	361.534	365.464
6000	361.287	360.890	360.795
8000	360.497	360.628	360.133
10,000	360.119	360.034	359.947

3.5.2 A 52-Bar Dome Truss

The second problem considers simultaneous size and shape optimization problem of a 52-bar dome truss structure. The initial layout of the structure is shown in Fig. 3.9. The cross-sectional areas of the members and the coordinates of the free nodes are considered as design variables. A non-structural mass of 50 kg is attached to all free nodes of the dome. The material properties, variable bounds, and frequency constraints of the problem are provided in Table 3.6. According to the symmetry of the structure, all members of the dome are categorized into eight groups of elements as shown in Table 3.7. All free nodes of the dome are allowed to move ± 2 m from their initial positions in a symmetrical way. Therefore, this is a design optimization problem with five shape variables and eight sizing variables. Frequency constraints of the problem are imposed on the first two natural frequencies of the structure. The lower and upper bounds of the cross-sectional areas are 1 cm^2 and 10 cm^2 , respectively. This problem was previously studied by several researchers using various

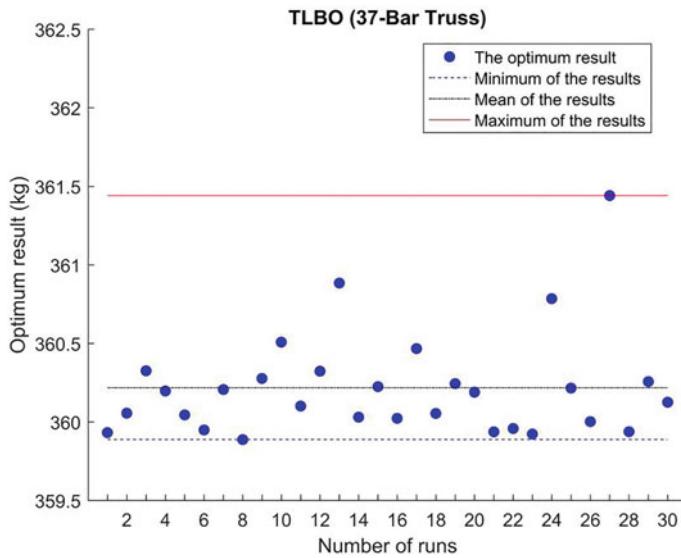


Fig. 3.5 Optimized weights obtained by the standard TLBO for the 37-bar planar truss over 30 runs

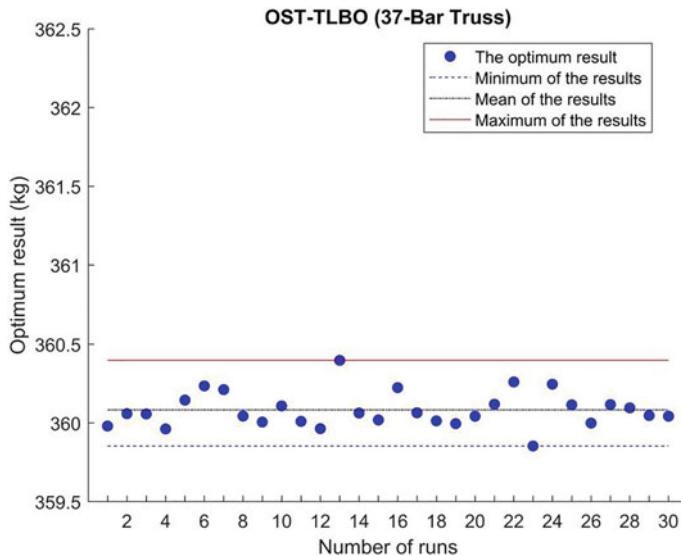


Fig. 3.6 Optimized weights obtained by OTS-TLBO for the 37-bar planar truss over 30 runs

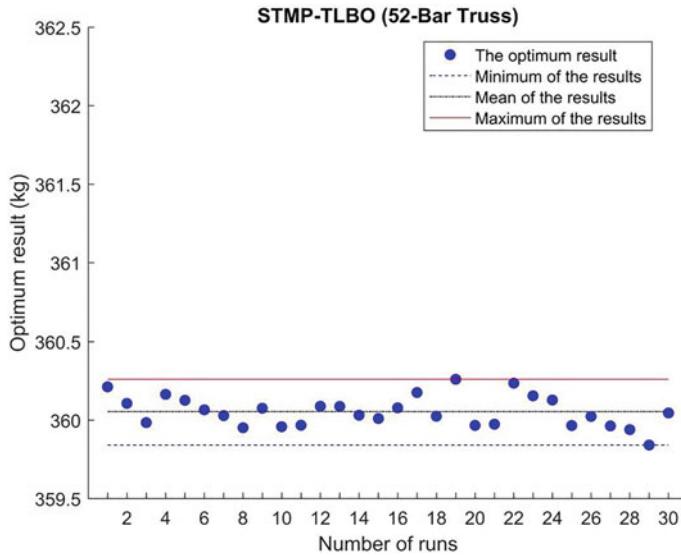


Fig. 3.7 Optimized weights obtained by STMP-TLBO for the 37-bar planar truss over 30 runs

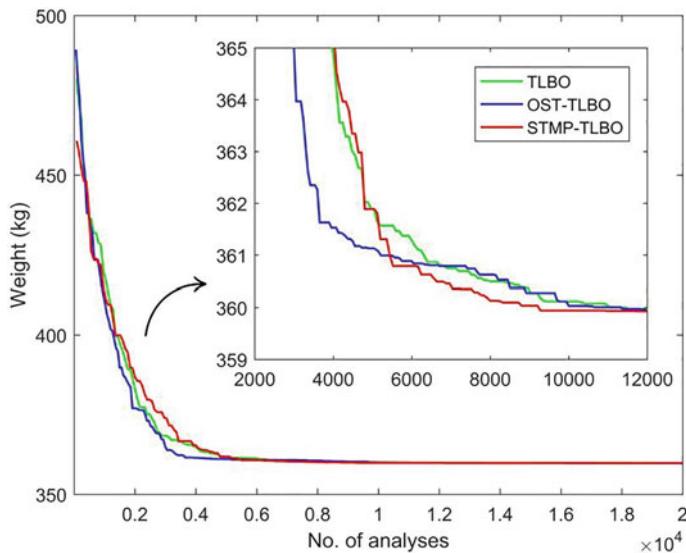


Fig. 3.8 Convergence histories of the best solutions of the standard TLBO, OST-TLBO, and STMP-TLBO for the 37-bar planar truss

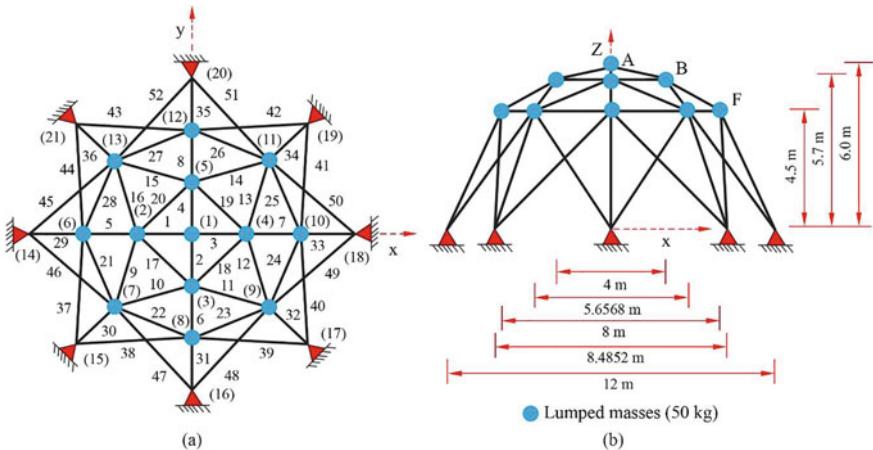


Fig. 3.9 Schematic of the initial layout of the 52-bar dome truss: **a** top view, **b** side view

Table 3.6 Material properties, cross-sectional area bounds, and frequency constraints of the 52-bar dome truss problem

Property (unit)	52-bar dome truss problem
Elasticity modulus (GPA)	210
Material density (kg/m^3)	7800
Added mass (kg)	50
Cross-sectional area bounds (cm^2)	$1 \leq A_i \leq 10$
Frequency constraints (Hz)	$\omega_1 \leq 15.916, \omega_2 \geq 28.648$

optimization methods: Miguel and Fadel Miguel [11] with HS and FA, Kaveh and Ilchi Ghazaan [13] employing the IRO algorithm, Kaveh and Zolghadr [15] using the CPA, etc.

Table 3.7 provides a comparison between the optimal design results obtained in the present study for the 52-bar dome truss and those reported in the literature. For each algorithm, 30 independent runs are performed with random initial populations and the obtained results of the best run are reported. It can be seen from the table that the best weight obtained by the standard TLBO is 193.540 kg, which is slightly heavier than those obtained by the OST-TLBO and STMP-TLBO (193.529 and 193.432 kg, respectively). Moreover, it is observed that both the OST-TLBO and STMP-TLBO algorithms outperform the standard TLBO in terms of average and worst weights. Table 3.8 provides the first five natural frequencies of the best designs of the 52-bar dome truss obtained by different algorithms. As expected, none of the frequency constraints are violated. The results of the table show that the constraint on the second natural frequency controls the design process. Table 3.9 lists the optimized weights obtained by the standard TLBO and the proposed variants at five different stages of the optimization process. The results indicate that the OST-TLBO and STMP-TLBO algorithms have faster convergence rates than the standard

Table 3.7 Comparison of optimal results obtained for the 52-bar dome truss

Design variable: Z and X (m); A (cm^2)	HS [11]	FA [11]	IRO [13]	CPA [15]	This study		
					TLBO	OST-TLBO	STMP-TLBO
Z_A	4.7374	6.4332	5.7600	5.9227	5.9328	5.9649	6.0207
X_B	1.5643	2.2208	2.1959	2.3048	2.2069	2.2420	2.3090
Z_B	3.7413	3.9202	3.7150	3.7061	3.9667	3.9758	4.0113
X_F	3.4882	4.0296	3.9269	3.9768	3.7425	3.7359	3.7424
Z_F	2.6274	2.5200	2.5000	2.5001	2.5002	2.5000	2.5000
A_{1-A_4}	1.0085	1.0050	1.0000	1.0000	1.0003	1.0003	1.0005
A_{5-A_8}	1.4999	1.3823	1.1889	1.1077	1.1998	1.1767	1.1005
$A_{9-A_{16}}$	1.3948	1.2295	1.2411	1.1988	1.2370	1.2433	1.1881
$A_{17-A_{20}}$	1.3462	1.2662	1.4422	1.4899	1.4477	1.4450	1.4705
$A_{21-A_{28}}$	1.6776	1.4478	1.3932	1.3991	1.4002	1.4141	1.4212
$A_{29-A_{36}}$	1.3704	1.0000	1.0000	1.0001	1.0000	1.0000	1.0000
$A_{37-A_{44}}$	1.4137	1.5728	1.6929	1.5998	1.4755	1.4690	1.4751
$A_{45-A_{52}}$	1.9378	1.4153	1.3569	1.4135	1.4785	1.4676	1.4714
Weight (kg)	214.94	197.53	195.38	194.826	193.540	193.529	193.432
Worst weight (kg)	N/A	N/A	N/A	N/A	206.769	202.320	202.113
Average weight (kg)	229.88	212.80	196.43	198.81	199.984	197.708	197.228
Standard deviation (kg)	12.44	17.98	1.81	3.71	3.739	3.980	3.698
Maximum number of structural analyses	20,000	10,000	17,000	12,800	20,000	20,000	20,000
Number of runs	5	5	20	50	30	30	30

TLBO. The optimized layout of the 52-bar dome truss obtained by the STMP-TLBO algorithm is shown in Fig. 3.10. The optimized weights of 30 independent runs of the standard TLBO, OST-TLBO, and STMP-TLBO are shown in Figs. 3.11, 3.12, and 3.13, respectively. As the figures illustrate, the proposed set-theoretical variants of the TLBO algorithm have a better ability to avoid local optima compared to the standard TLBO. In other words, the proposed set-theoretical framework can improve

Table 3.8 First five natural frequencies (Hz) of the 52-bar dome truss evaluated at optimal designs

Frequency number	HS [11]	FA [11]	IRO [13]	CPA [15]	This study		
					TLBO	OST-TLBO	STMP-TLBO
1	12.2222	11.3119	11.4437	11.736	11.2665	11.4406	11.7075
2	28.657	28.6529	28.6529	28.648	28.6485	28.6483	28.6480
3	28.65	28.6529	28.7014	28.648	28.6487	28.6483	28.6480
4	28.661	28.8030	28.7179	28.654	28.6494	28.6510	28.6505
5	30.0997	28.8030	29.1276	28.690	28.6574	28.6946	28.6518

Table 3.9 Optimized weights for the 52-bar dome truss at five different stages of the optimization process (best runs)

Number of structural analyses	Optimized weight (kg)		
	TLBO	OST-TLBO	STMP-TLBO
2000	297.954	266.151	287.791
4000	241.759	218.811	206.245
6000	206.091	197.076	196.316
8000	195.984	195.261	194.302
10,000	194.997	194.420	193.956

the ability to escape from local optima. In this problem, due to the existence of local optimum solutions, the standard deviation is not a good measure of uncertainty. From the figures it can be seen that set-theoretical variants of the TLBO algorithm perform better than the standard TLBO in terms of average and worst weights. Figure 3.14 compares the convergence histories of the best solutions obtained by the standard TLBO, OST-TLBO, and STMP-TLBO for the 52-bar dome truss. As the figure shows, set-theoretical variants of the TLBO have faster convergence rates than the standard TLBO.

3.5.3 A 120-Bar Dome Truss

The 120-bar dome truss shown in Fig. 3.15 is considered as the third example. Non-structural masses are added to all free nodes of the dome as follows: 3000 kg at node 1, 500 kg at nodes 2 to 13, and 100 kg at the rest of the nodes. Table 3.10 lists the material properties, variable bounds, and frequency constraint of the problem. Constraints are imposed on the first two natural frequencies of the dome. Considering the symmetry of the structure, all members of the dome are categorized into seven groups of elements, as shown in Fig. 3.15. The lower and upper bounds of the cross-sectional areas are set equal to 1 cm^2 and 129.3 cm^2 , respectively. It is noted that the layout of the structure is not unchanged during the optimization process. Thus, this is a sizing optimization problem with seven design variables. This problem has been previously reported by many researchers using a variety of optimization methods:

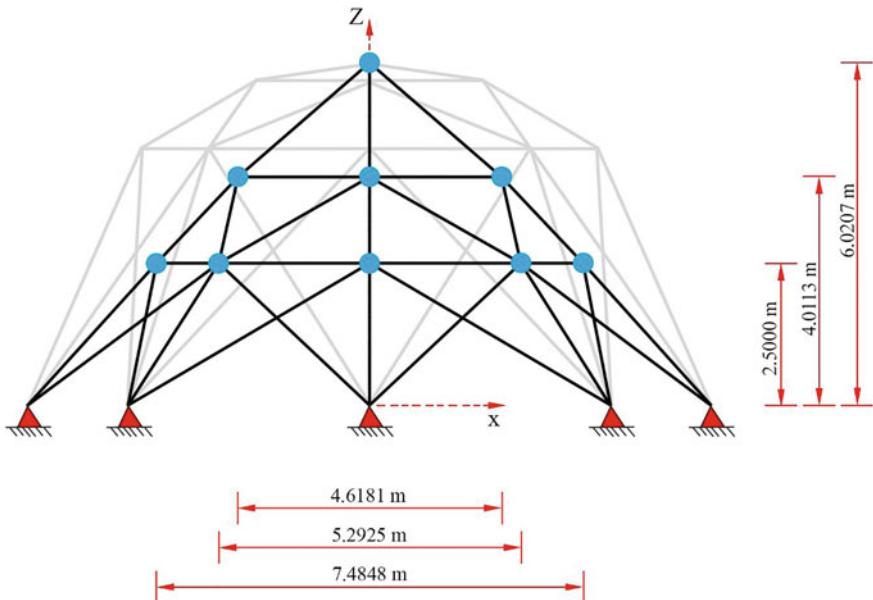


Fig. 3.10 Optimized layout of the 52-bar dome truss structure found by the STMP-TLBO algorithm

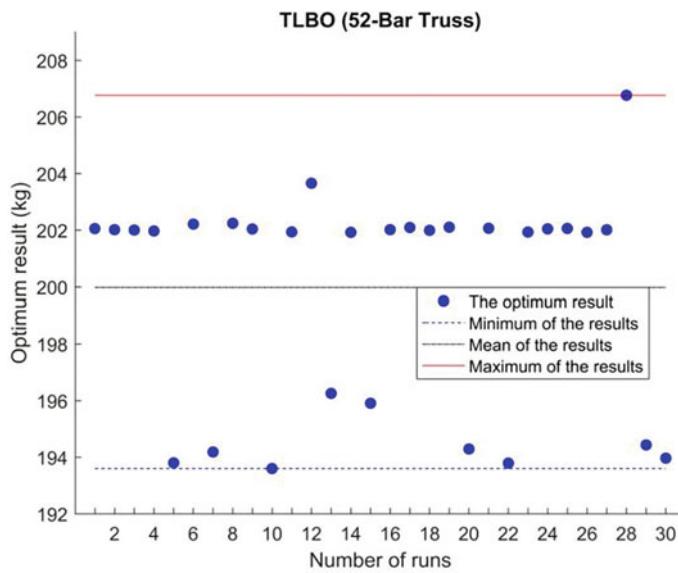


Fig. 3.11 Optimized weights obtained by the standard TLBO for the 52-bar dome truss over 30 runs

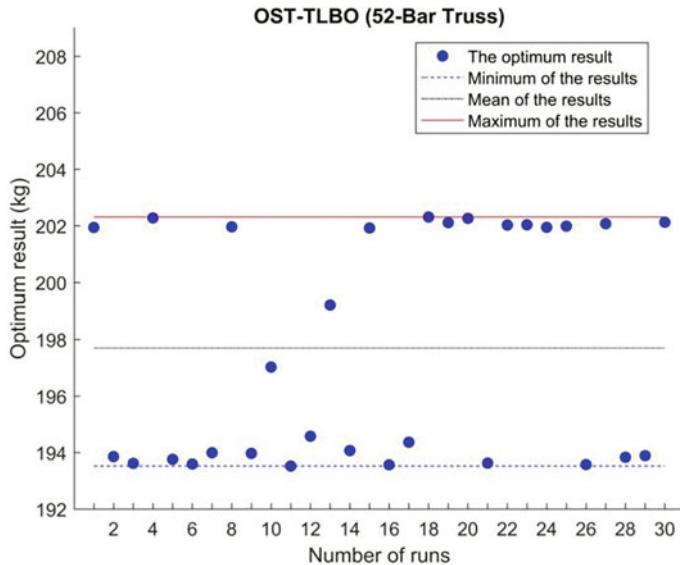


Fig. 3.12 Optimized weights obtained by OTS-TLBO for the 52-bar dome truss over 30 runs

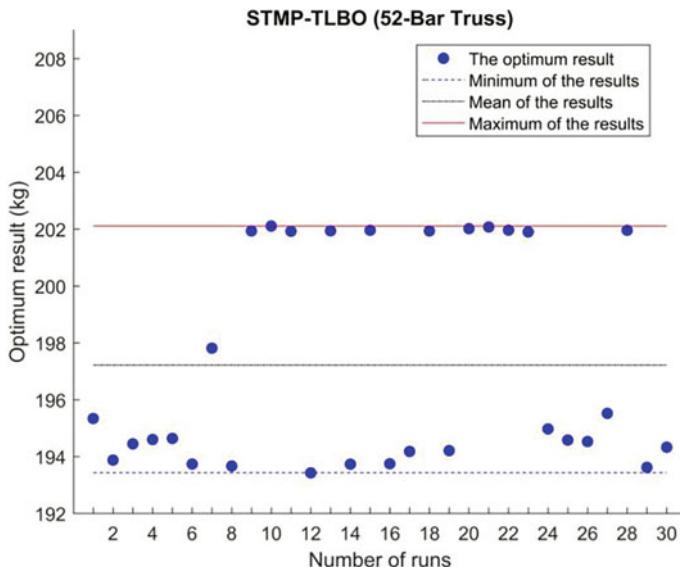


Fig. 3.13 Optimized weights obtained by STMP-TLBO for the 52-bar dome truss over 30 runs

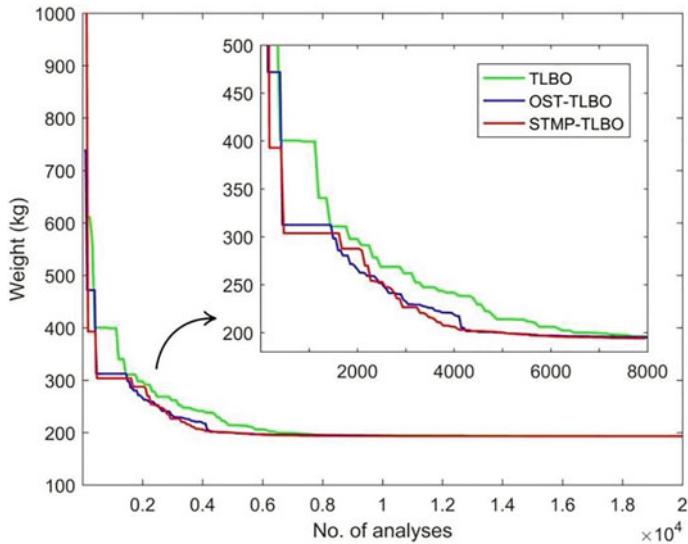


Fig. 3.14 Convergence histories of the best solutions of the standard TLBO, OST-TLBO, and STMP-TLBO for the 52-bar dome truss

Kaveh and Ilchi Ghazaan [16] using hybrid metaheuristic algorithms, Taheri and Jalili [17] employing enhanced biogeography-based optimization (EBBO) algorithm, Dede et al. [18] using Jaya algorithm (JA), etc.

Table 3.11 provides a comparison between the results obtained in the present study with those of other methods in the literature. From the table it can be seen that set-theoretical variants of the TLBO algorithm outperform the standard TLBO algorithm in terms of average and worst weights as well as standard deviation. However, the best weight of the standard TLBO is very close to those of the OST-TLBO and STMP-TLBO. Table 3.12 presents the first five natural frequencies of the 120-bar dome truss evaluated at the optimal designs obtained by the present study and those reported in the literature.

It is obvious that none of the frequency constraints are violated. Table 3.13 presents the optimized weights obtained by the standard TLBO, OST-TLBO, and STMP-TLBO at five different stages of the optimization process. The final weights of 30 independent runs of the TLBO, OST-TLBO, and STMP-TLBO are shown in Figs. 3.16, 3.17, and 3.18, respectively. As is evident from the figures, set-theoretical variants of the TLBO have obtained better results than the standard TLBO in terms of average and worst weights. The best and average convergence histories of the standard TLBO, OST-TLBO, and STMP-TLBO are shown in Figs. 3.19, 3.20, and 3.21, respectively. Moreover, Fig. 3.22 compares the convergence histories of the average results obtained by the standard TLBO, OST-TLBO, and STMP-TLBO over 20 independent runs. A close examination of the figure shows that in the early stages of the optimization process (i.e., within about the first 5000 structural analyses), the

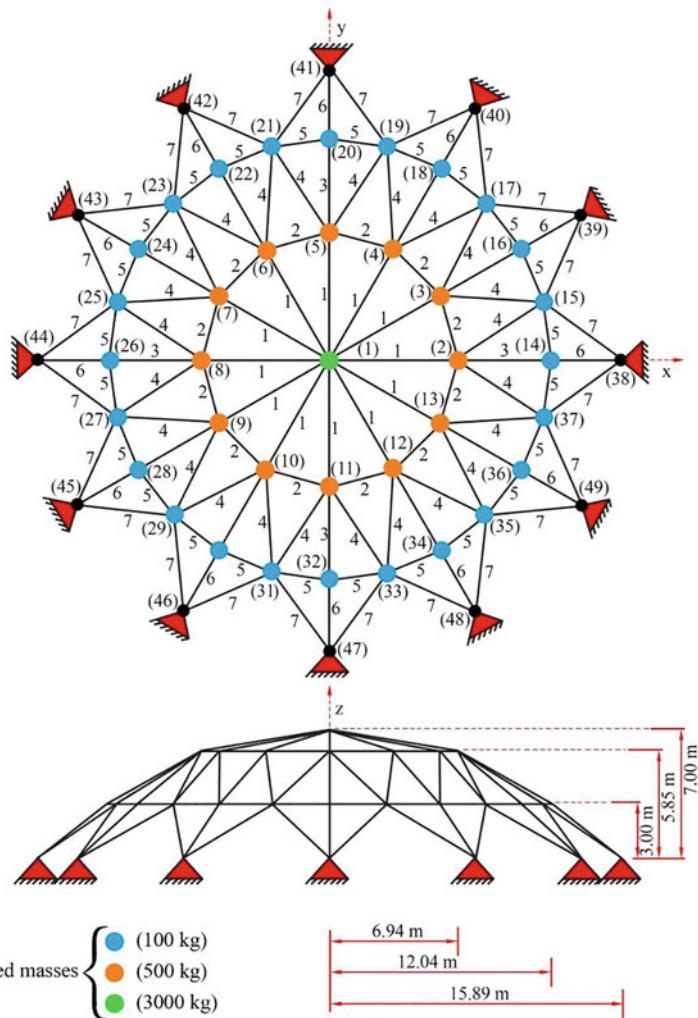


Fig. 3.15 Schematic of the 120-bar dome truss

Table 3.10 Material properties, cross-sectional area bounds, and frequency constraints of the 120-bar dome truss problem

Property (unit)	120-bar dome truss problem
Elasticity modulus (GPA)	210
Material density (kg/m^3)	7971.81
Added mass (kg)	3000 kg at node 1, 500 kg at nodes 2 to 13, 100 kg at the rest of the nodes
Cross-sectional area bounds (cm^2)	$1 \leq A_i \leq 129.3$
Frequency constraints (Hz)	$\omega_1 \geq 9, \omega_2 \geq 11$

Table 3.11 Comparison of optimal results obtained for the 120-bar dome truss

Design variable: A (cm^2)	PSS [16]	ALC-PSO [16]	HALC-PSO [16]	EBBO [17]	JA [18]	This study
	TLBO	OST-TLBO	STMP-TLBO	TLBO	OST-TLBO	STMP-TLBO
A_1	18.4132	19.5316	19.309	19.8878	19.309	19.5254
A_2	47.8316	41.5725	40.763	39.8248	40.763	40.2271
A_3	15.6585	11.3712	10.791	10.5496	10.791	10.6672
A_4	28.7868	21.6754	21.272	21.0929	21.272	21.1988
A_5	9.1114	9.8078	9.943	9.4245	9.943	9.8735
A_6	15.1059	12.7670	11.695	11.6648	11.695	11.7786
A_7	14.4374	14.7140	14.579	15.1282	14.579	14.7480
Weight (kg)	10,163.99	8890.70	8889.96	8711.95	8709.35	8708.902
Worst weight (kg)	N/A	N/A	N/A	N/A	8713.901	8711.685
Average weight (kg)	11,134.77	8900.68	8900.39	8718.5	8713.21	8710.631
Standard deviation (kg)	526.67	8.81	6.38	7.15	2.97	1.027
Maximum number of structural analyses	20,000	20,000	20,000	30,000	18,000	20,000
Number of runs	30	30	30	100	20	20

Table 3.12 First five natural frequencies (Hz) of the 120-bar dome truss evaluated at optimal designs

Frequency number	PSS [16]	ALC-PSO [16]	HALC-PSO [16]	EBBO [17]	JA [18]	This study		
						TBO	OST-TLBO	STMP-TLBO
1	9.067	9.000	9.000	9.0000	9.0000	9.0001	9.0001	9.0004
2	11.199	11.000	11.000	11.0000	11.0002	11.0001	11.0001	11.0001
3	11.214	11.000	11.000	11.0002	11.0002	11.0001	11.0001	11.0001
4	11.695	11.009	11.010	11.0008	11.0008	11.0007	11.0003	11.0001
5	11.726	11.048	11.050	11.0657	11.0674	11.0675	11.0666	11.0669

Table 3.13 Optimized weights for the 120-bar dome truss at five different stages of the optimization process (best runs)

Number of structural analyses	Optimized weight (kg)		
	TLBO	OST-TLBO	STMP-TLBO
2500	8837.905	9566.518	9217.955
5000	8717.148	8749.663	8723.009
7500	8710.248	8720.225	8711.721
10,000	8710.248	8714.799	8710.910
12,500	8710.006	8714.454	8709.294

average weights obtained by the STMP-TLBO are heavier than those of the standard TLBO. In the subsequent stages of optimization, however, the average weights obtained by the STMP-TLBO are better than those of the standard TLBO. The main reason is the high exploration ability of the STMP-TLBO algorithm in the early stages of the optimization process.

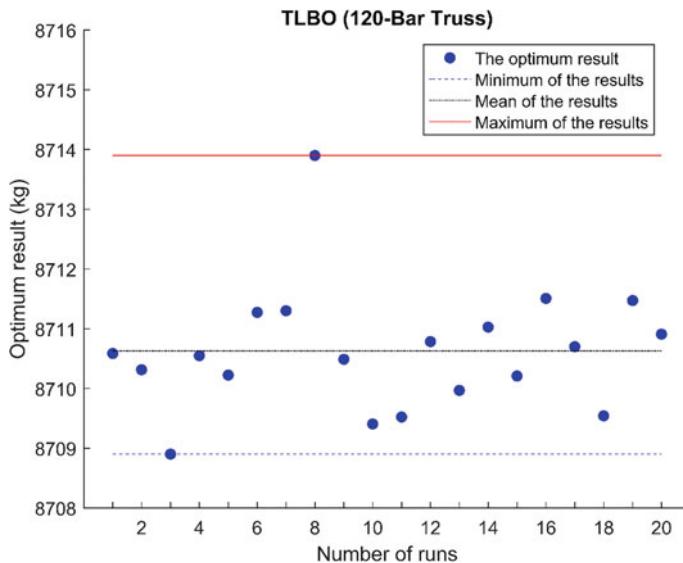


Fig. 3.16 Optimized weights obtained by the standard TLBO for the 120-bar dome truss over 20 runs

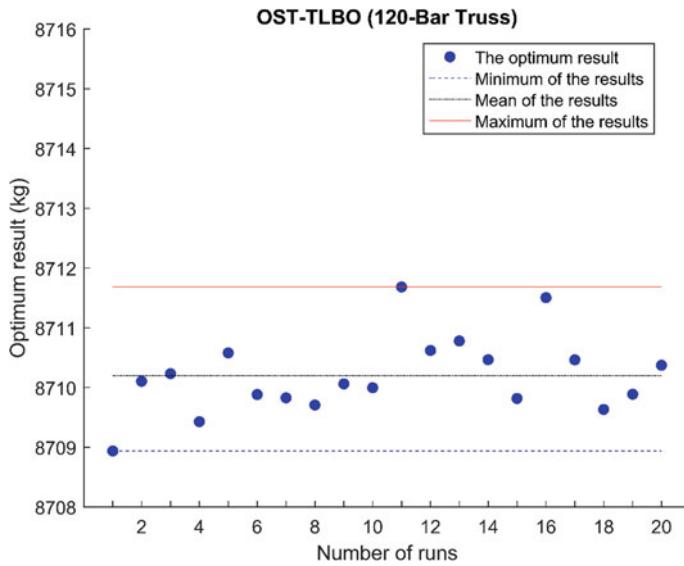


Fig. 3.17 Optimized weights obtained by OST-TLBO for the 120-bar dome truss over 20 runs

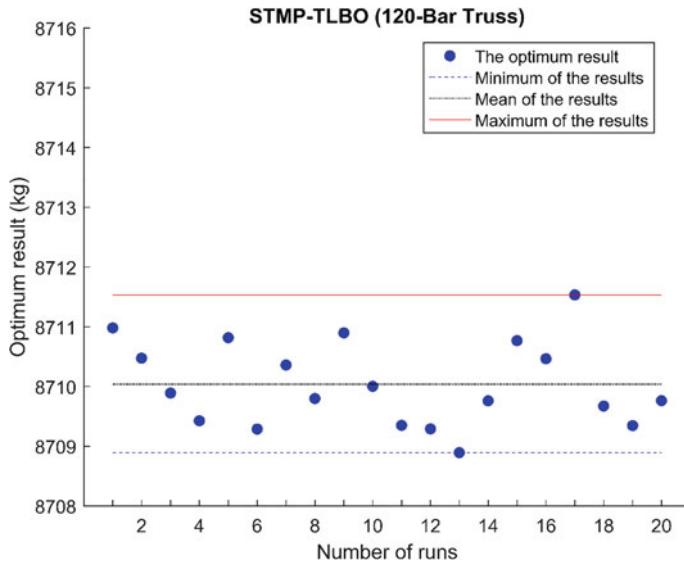


Fig. 3.18 Optimized weights obtained by STMP-TLBO for the 120-bar dome truss over 20 runs

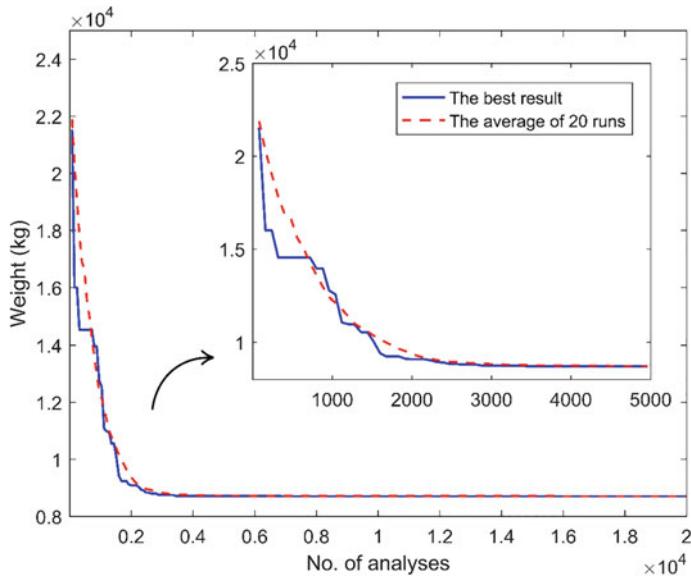


Fig. 3.19 Convergence histories of the standard TLBO for the 120-bar dome truss

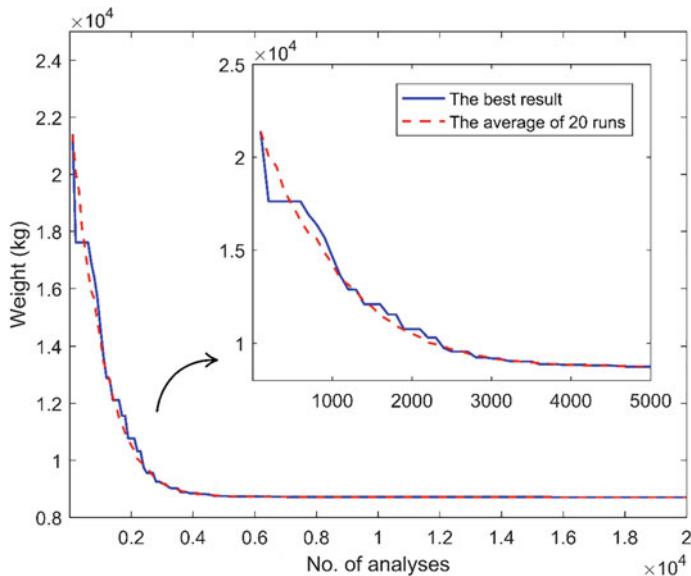


Fig. 3.20 Convergence histories of the OST-TLBO for the 120-bar dome truss

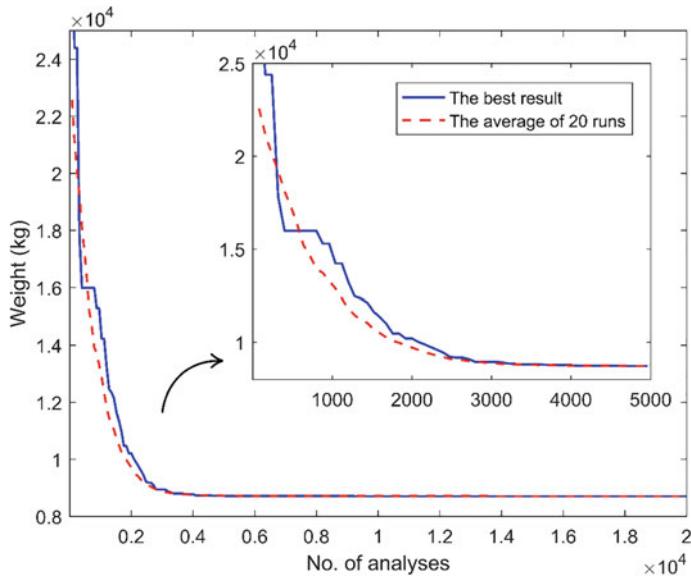


Fig. 3.21 Convergence histories of the STMP-TLBO for the 120-bar dome truss

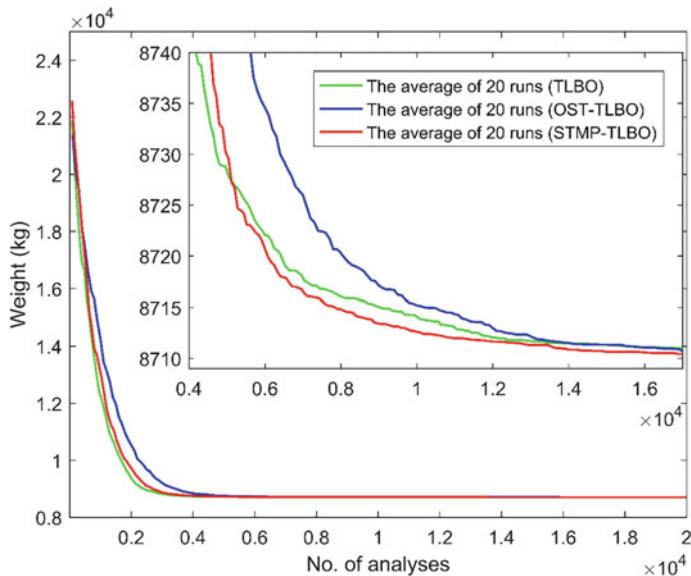


Fig. 3.22 Average convergence histories for the 120-bar dome truss obtained by the standard TLBO, OST-TLBO, and STMP-TLBO

3.5.4 A 200-Bar Planar Truss

The 200-bar planar truss shown in Fig. 3.23 is considered as the last example. A non-structural mass of 100 kg is attached to the upper nodes of the structure. Table 3.14 summarizes the material properties, variable bounds, and frequency constraints of the problem. As shown in the table, the first three natural frequencies of the structure are constrained. With respect to the symmetry of the structure, the 200 members of the truss are grouped into 29 groups of elements, as shown in Table 3.15. The layout of the structure is not changed during the optimization process. Therefore, this is a sizing optimization problem with 29 design variables. This problem has been previously solved by many authors using various optimization methods: Kaveh and Ilchi Ghazaan [16] using hybrid metaheuristic algorithms, Taheri and Jalili [17] employing the EBO algorithm, Kaveh and Ilchi Ghazaan [19, 20] using different variants of the colliding bodies optimization (CBO) algorithm, etc.

Table 3.15 presents the optimal design results obtained by the standard TLBO and its set-theoretical variants and other methods in the literature. It can be seen that the best weight obtained by the STMP-TLBO algorithm is 2157.714 kg, which is better than that of the standard TLBO and OST-TLBO (2157.799 kg and 2157.91 kg, respectively) and is only slightly heavier than that of the UECBO (2157.65 kg). Moreover, it is clear from the table that the STMP-TLBO algorithm outperforms both the standard TLBO and OST-TLBO in terms of best weight, average weight, worst weight, and standard deviation. The standard deviation of the OST-TLBO is less than that of the standard TLBO. However, the standard TLBO performs better than the OST-TLBO in terms of best and average weights. Table 3.16 lists the first five natural frequencies of the 200-bar truss evaluated at the optimal designs obtained by the present study and those reported in the literature. It can be seen that all frequency constraints are satisfied. Table 3.17 presents the optimized weights obtained for the 200-bar truss at seven different stages of the optimization process. As the table indicates, in the final stages of the optimization process, the OST-TLBO and STMP-TLBO algorithms have significantly higher convergence rates than the standard TLBO. The final weights of 20 independent runs of the standard TLBO, OST-TLBO, and STMP-TLBO algorithms are depicted in Figs. 3.24, 3.25, and 3.26, respectively. The figures indicate that the best weight and average weight achieved by STMP-TLBO are better than those of the standard TLBO and OST-TLBO, as mentioned earlier. Figures 3.27 and 3.28 show the convergence histories of the best and average results for the 200-bar truss, respectively.

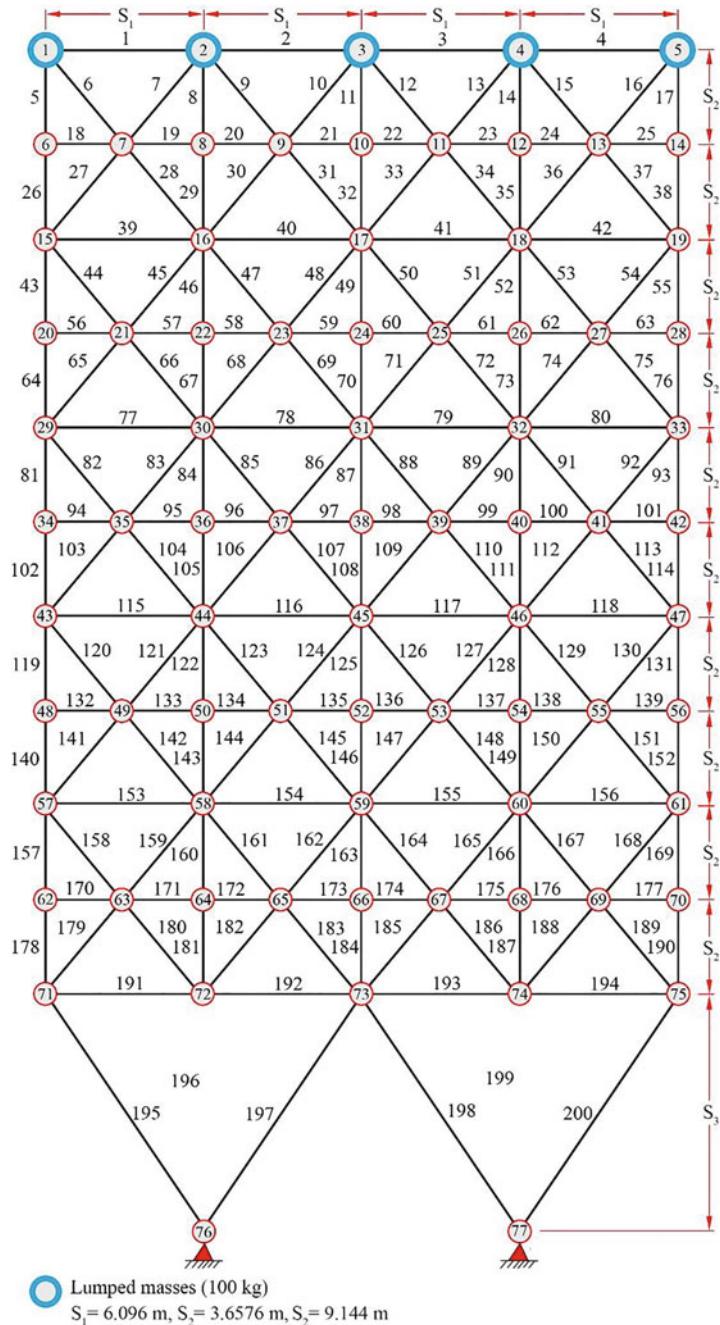


Fig. 3.23 Schematic of the 200-bar planar truss

Table 3.14 Material properties, cross-sectional area bounds, and frequency constraints of the 200-bar planar truss problem

Property (unit)	200-bar planar truss problem
Elasticity modulus (GPA)	210
Material density (kg/m^3)	7800
Added mass (kg)	100
Lower bound of the cross-sectional areas (cm^2)	0.1
Frequency constraints (Hz)	$\omega_1 \geq 5, \omega_2 \geq 10, \omega_3 \geq 15$

Table 3.15 Comparison of optimal results obtained for the 200-bar planar truss

Design variable: A (cm^2)	CBO [19]	ECBO [19]	UECBO [20]	This study		
				TLBO	OST-TLBO	STMP-TLBO
Group number	Members of the group					
1	1, 2, 3, 4	0.3059	0.2993	0.3002	0.3126	0.3243
2	5, 8, 11, 14, 17	0.4476	0.4497	0.4890	0.4622	0.4247
3	19, 20, 21, 22, 23, 24	0.1000	0.1000	0.1000	0.1006	0.1008
4	18, 25, 56, 63, 94, 101, 132, 139, 170, 177	0.1001	0.1000	0.1000	0.1001	0.1030
5	26, 29, 32, 35, 38	0.4944	0.5137	0.5277	0.5030	0.5306
6	6, 7, 9, 10, 12, 13, 15, 16, 27, 28, 30, 31, 33, 34, 36, 37	0.8369	0.7914	0.8310	0.8356	0.8395
7	39, 40, 41, 42	0.1001	0.1013	0.1001	0.1059	0.1113
8	43, 46, 49, 52, 55	1.5514	1.4129	1.3841	1.4764	1.4185
9	57, 58, 59, 60, 61, 62	0.1000	0.1019	0.1000	0.1019	0.1023

(continued)

Table 3.15 (continued)

Design variable: A (cm 2)		CBO [19]	ECBO [19]	UECBO [20]	This study		
Group number	Members of the group				TLBO	OST-TLBO	STMP-TLBO
10	64, 67, 70, 73, 76	1.5286	1.6460	1.5912	1.6209	1.5553	1.5969
11	44, 45, 47, 48, 50, 51, 53, 54, 65, 66, 68, 69, 71, 72, 74, 75	1.1547	1.1532	1.1502	1.1446	1.1519	1.1505
12	77, 78, 79, 80	0.1000	0.1000	0.1008	0.1356	0.1433	0.1668
13	81, 84, 87, 90, 93	2.9980	3.1850	3.0023	2.9846	2.9539	2.9503
14	95, 96, 97, 98, 99, 100	0.1017	0.1034	0.1007	0.1007	0.1054	0.1049
15	102, 105, 108, 111, 114	3.2475	3.3126	3.2767	3.2059	3.2897	3.3305
16	82, 83, 85, 86, 88, 89, 91, 92, 103, 104, 106, 107, 109, 110, 112, 113	1.5213	1.5920	1.6017	1.5903	1.5890	1.5777
17	115, 116, 117, 118	0.3996	0.2238	0.2309	0.2695	0.2359	0.2661
18	119, 122, 125, 128, 131	4.7557	5.1227	5.0228	5.0503	4.9473	5.1869
19	133, 134, 135, 136, 137, 138	0.1002	0.1050	0.1057	0.1624	0.1195	0.1014

(continued)

Table 3.15 (continued)

Design variable: A (cm 2)		CBO [19]	ECBO [19]	UECBO [20]	This study		
Group number	Members of the group				TLBO	OST-TLBO	STMP-TLBO
20	140, 143, 146, 149, 152	5.1359	5.3707	5.2667	5.1528	5.4264	5.4297
21	120, 121, 123, 124, 126, 127, 129, 130, 141, 142, 144, 145, 147, 148, 150, 151	2.1181	2.0645	2.1287	2.0845	2.0412	2.0687
22	153, 154, 155, 156	0.9200	0.5443	0.7337	0.6749	0.7127	0.7319
23	157, 160, 163, 166, 169	7.3084	7.6497	7.9257	7.7495	7.5703	7.4680
24	171, 172, 173, 174, 175, 176	0.1185	0.1000	0.1000	0.1242	0.1392	0.1227
25	178, 181, 184, 187, 190	7.6901	7.6754	8.1735	8.0147	7.8170	7.7060
26	158, 159, 161, 162, 164, 165, 167, 168, 179, 180, 182, 183, 185, 186, 188, 189	3.0895	2.7178	2.7758	2.7438	2.8329	2.8544
27	191, 192, 193, 194	10.6462	10.8141	10.1047	10.7496	10.5864	10.0465
28	195, 197, 198, 200	20.7190	21.6349	21.2172	21.3426	21.4892	21.4945
29	196, 199	11.7463	10.3520	10.9900	10.5208	10.4272	10.9668
Weight (kg)		2161.15	2158.08	2157.65	2157.799	2157.91	2157.714
Worst weight (kg)		N/A	N/A	N/A	2167.884	2167.288	2164.578
Average weight (kg)		2447.52	2159.93	2161.36	2161.577	2162.041	2160.366

(continued)

Table 3.15 (continued)

Design variable: A (cm^2)	CBO [19]	ECBO [19]	UECBO [20]	This study		
				TLBO	OST-TLBO	STMP-TLBO
Group number	Members of the group					
Standard deviation (kg)	301.29	1.57	2.72	3.036	2.586	1.806
Maximum number of structural analyses	20,000	20,000	20,000	30,000	30,000	30,000
Number of runs	20	20	20	20	20	20

Table 3.16 First five natural frequencies (Hz) of the 200-bar planar truss evaluated at optimal designs

Frequency number	CBO [19]	ECBO [19]	UECBO [20]	This study		
				TLBO	OST-TLBO	STMP-TLBO
1	5.000	5.000	5.000	5.0000	5.0001	5.0000
2	12.221	12.189	12.260	12.3303	12.3891	12.2160
3	15.088	15.048	15.096	15.1044	15.0947	15.0639
4	16.759	16.643	16.696	16.7742	16.7304	16.7141
5	21.419	21.342	21.452	21.5242	21.5343	21.4112

Table 3.17 Optimized weights for the 200-bar planar truss at seven different stages of the optimization process (best runs)

Number of structural analyses	Optimized weight (kg)		
	TLBO	OST-TLBO	STMP-TLBO
4000	3185.592	3615.868	3459.616
8000	2235.477	2357.701	2295.660
12,000	2172.418	2187.580	2192.810
16,000	2162.233	2172.569	2180.591
20,000	2160.066	2166.047	2168.862
24,000	2159.062	2160.170	2159.136
28,000	2158.127	2158.282	2157.869

3.6 Concluding Remarks

In this study, inspired by the concepts of set theory, a general set-theoretical framework was proposed for population-based metaheuristic algorithms. The main idea of the proposed framework is to divide the population of solutions into smaller well-arranged subpopulations of the same size. The framework is of a general nature and

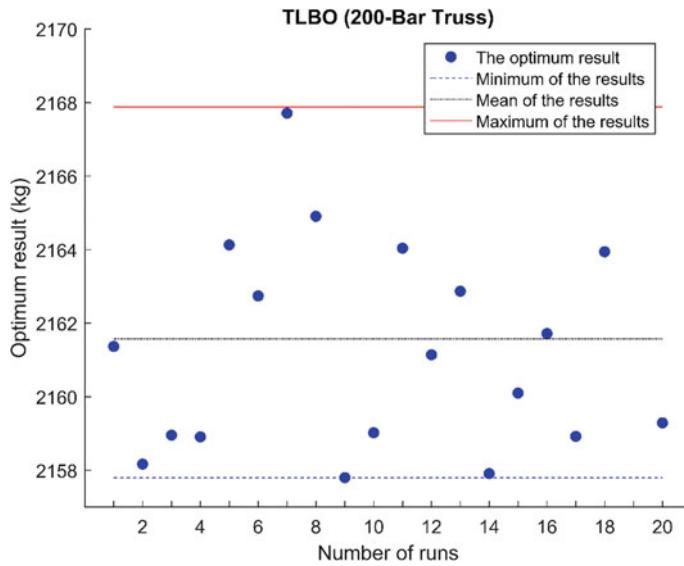


Fig. 3.24 Optimized weights obtained by the standard TLBO for the 200-bar planar truss over 20 runs

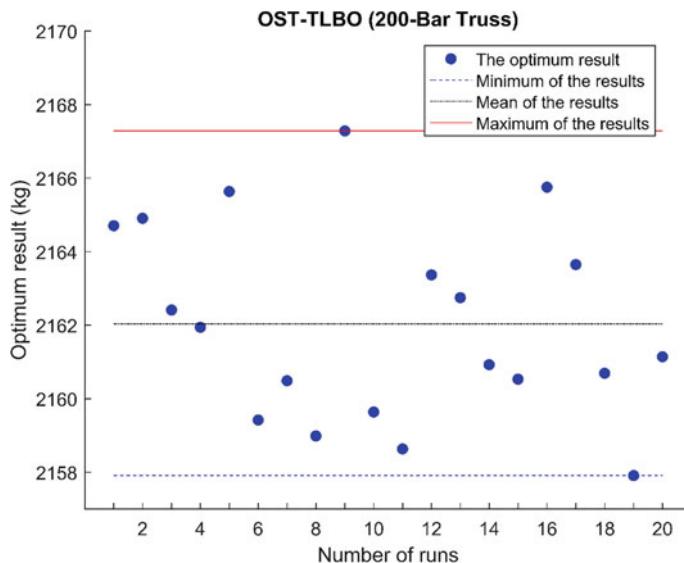


Fig. 3.25 Optimized weights obtained by OST-TLBO for the 200-bar planar truss over 20 runs

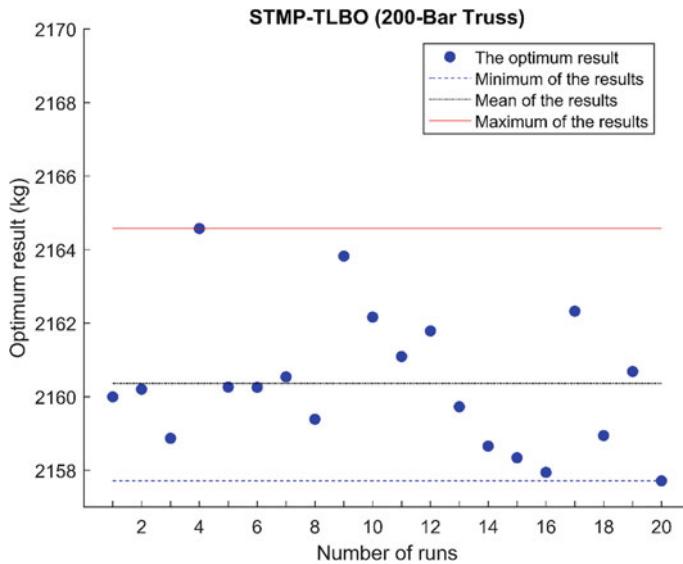


Fig. 3.26 Optimized weights obtained by STMP- TLBO for the 200-bar planar truss over 20 runs

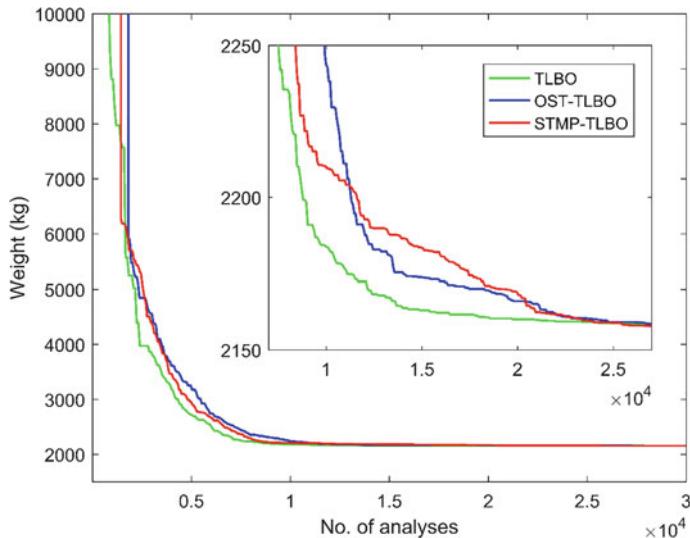


Fig. 3.27 Convergence histories of the best solutions of the standard TLBO, OST-TLBO, and STMP-TLBO for the 200-bar planar truss

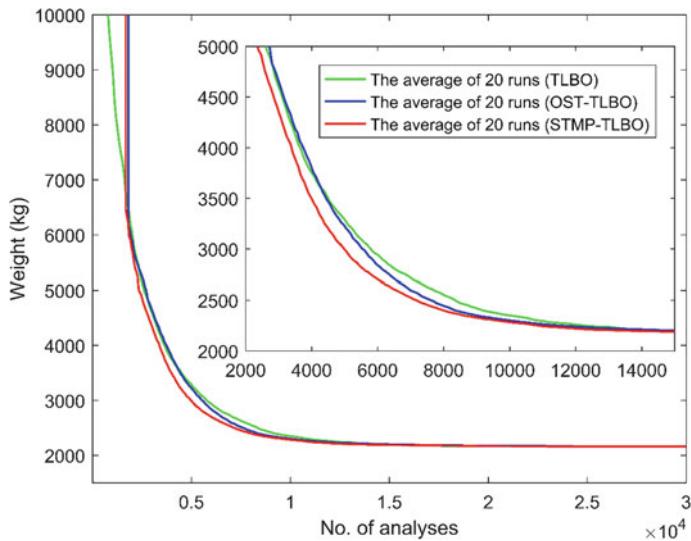


Fig. 3.28 Average convergence histories for the 200-bar planar truss obtained by the standard TLBO, OST-TLBO, and STMP-TLBO

is applicable to almost all population-based metaheuristic algorithms. The proposed framework aims to enhance the balance between exploration and exploitation of the search space. Then, the framework was applied to the standard TLBO algorithm, and two set-theoretical variants of the TLBO algorithm called ordered set-theoretical TLBO (OST-TLBO) and set-theoretical multi-phase TLBO (STMP-TLBO) were proposed. To demonstrate the efficiency and effectiveness of the proposed algorithms, some truss optimization problems with frequency constraints were investigated. The results showed that the proposed set-theoretical variants of the TLBO algorithm had better convergence characteristics compared to the standard TLBO algorithm. Moreover, it was verified from the optimization results that the OST-TLBO and STMP-TLBO algorithms outperformed the standard TLBO, especially in terms of average weight, worst weight, and standard deviation.

References

1. Kaveh A, Biabani Hamedani K, Kamalinejad M (2020) Set theoretical variants of the teaching-learning-based optimization algorithm for optimal design of truss structures with multiple frequency constraints. *Acta Mech* 231(9):3645–3672. <https://doi.org/10.1007/s00707-020-02718-3>
2. Cantor G (1915) Contributions to the founding of the theory of transfinite numbers. Open Court Publishing, Chicago
3. Behravesh A, Kaveh A, Nani M, Sabet S (1988) A set theoretical approach for configuration processing. *Comput Struct* 30(6):1293–1302. [https://doi.org/10.1016/0045-7949\(88\)90194-0](https://doi.org/10.1016/0045-7949(88)90194-0)

4. Kaveh A (2004) Structural mechanics: graph and matrix methods, 3rd ed. Research Studies Press (John Wiley), Baldock, Hertfordshire
5. Kaveh A, Biabani Hamedani K, Zaerreza A (2021) A set theoretical shuffled shepherd optimization algorithm for optimal design of cantilever retaining wall structures. *Eng Comput* 37(4):3265–3282. <https://doi.org/10.1007/s00366-020-00999-9>
6. Kaveh A, Bakhshpoori T (2019) Metaheuristics: outlines, MATLAB codes and examples, 1st edn. Springer, Basel
7. Rao RV, Savsani VJ, Vakharia DP (2011) Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems. *Comput Aided Des* 43(3):303–315. <https://doi.org/10.1016/j.cad.2010.12.015>
8. Bellagamba L, Yang TY (1981) Minimum-mass truss structures with constraints on fundamental natural frequency. *AIAA J* 19(11):1452–1458. <https://doi.org/10.2514/3.7875>
9. Grandhi R, Venkayyat VB (1988) Structural optimization with frequency constraints. *AIAA J* 26(7):858–866. <https://doi.org/10.2514/3.9979>
10. Gomez HM (2011) Truss optimization with dynamic constraints using a particle swarm algorithm. *Expert Syst Appl* 38:957–968. <https://doi.org/10.1016/j.eswa.2010.07.086>
11. Miguel LF, Miguel LF (2012) Shape and size optimization of truss structures considering dynamic constraints through modern metaheuristic algorithms. *Expert Syst Appl* 39(10):9458–9467. <https://doi.org/10.1016/j.eswa.2012.02.113>
12. Kaveh A, Javadi SM (2014) Shape and size optimization of trusses with multiple frequency constraints using harmony search and ray optimizer for enhancing the particle swarm optimization algorithm. *Acta Mech* 225(6):1595–1605. <https://doi.org/10.1007/s00707-013-1006-z>
13. Kaveh A, Ilchi Ghazaan M (2015) Layout and size optimization of trusses with natural frequency constraints using improved ray optimization algorithm. *IJST Trans Civ Eng* 39(C2+):395–408. <https://doi.org/10.22099/ijstc.2015.3509>
14. Kaveh A, Ilchi Ghazaan M (2017) Vibrating particles system algorithm for truss optimization with multiple natural frequency constraints. *Acta Mech* 228(1):307–322. <https://doi.org/10.1007/s00707-016-1725-z>
15. Kaveh A, Zolghadr A (2017) Cyclical parthenogenesis algorithm for layout optimization of truss structures with frequency constraints. *Eng Optim* 49(8):1317–1334. <https://doi.org/10.1080/0305215X.2016.1245730>
16. Kaveh A, Ilchi Ghazaan M (2015) Hybridized optimization algorithms for design of trusses with multiple natural frequency constraints. *Adv Eng Softw* 79:137–147. <https://doi.org/10.1016/j.advengsoft.2014.10.001>
17. Taheri SH, Jalili S (2016) Enhanced biogeography-based optimization: a new method for size and shape optimization of truss structures with natural frequency constraints. *Lat Am J Solids Struct* 13(7):1406–1430. <https://doi.org/10.1590/1679-78252208>
18. Dede T, Grzywiński M, Rao RV (2020) Jaya: a new meta-heuristic algorithm for the optimization of braced dome structures. In: Advanced engineering optimization through intelligent techniques. Springer, Singapore, pp 13–20. https://doi.org/10.1007/978-981-13-8196-6_2
19. Kaveh A, Ilchi Ghazaan M (2015) Enhanced colliding bodies algorithm for truss optimization with frequency constraints. *J Comput Civ Eng* 29(6):04014104. [https://doi.org/10.1061/\(ASCE\)CP.1943-5487.0000445](https://doi.org/10.1061/(ASCE)CP.1943-5487.0000445)
20. Kaveh A, Ilchi Ghazaan M (2016) Truss optimization with dynamic constraints using UECBO. *Adv Comput Des* 1(2):119–138. <https://doi.org/10.12989/acd.2016.1.2.119>

Chapter 4

Enhanced Set-Theoretical Versions of the Shuffled Shepherd Optimization Algorithm for Structural Optimization



4.1 Introduction

This chapter consists of two main parts. The first part is devoted to develop a parameter-free version of the shuffled shepherd optimization algorithm, and the second part presents a multi-phase version of the algorithm proposed in the first part [1]. The main idea of the multi-phase version is to divide the optimization process into several phases each characterized by a different number of subpopulations. The number of subpopulations decreases gradually as the search process continues. In order to validate the efficiency and effectiveness of the proposed algorithms, four optimization problems of skeletal structures are investigated.

Set theory is a field of mathematical logic that focuses on the study of sets and their properties. In set theory, a set is defined as a collection of objects, called members or elements of the set [2]. George Cantor started the study of set theory in the second half of the nineteenth century [3]. Although any type of object can be collected into a set, set theory is often applied to objects that are relevant to mathematics. The language of set theory can be used to express almost any mathematical concept, thus set theory provides a foundation for mathematics. In recent years, set theory has found many applications in engineering. In structural engineering, Behravesh et al. [4] employed set theory for configuration processing. As another example of application of set theory, Kaveh et al. [5] utilized set-theoretical concepts for generalizing the definition of the shuffled shepherd optimization algorithm.

Metaheuristic optimization algorithms can be classified based on various criteria. The most common classification criterion is single-solution-based search versus population-based search. In single-solution-based metaheuristics, a single solution is manipulated and transformed during the iterative search process. Tabu search (TS), simulated annealing (SA), and ant colony algorithm (ACO) are typical examples of single-solution-based algorithms. In population-based metaheuristics, however, a population of solutions is manipulated in each iteration of the search process. Therefore, the population of population-based metaheuristics can be viewed as a set of

elements. So, a population-based metaheuristic can be viewed as an iterative process of manipulating a set of elements. The search process of population-based metaheuristic algorithms starts with a set of candidate solutions called initial population. Different techniques have been proposed to generate an initial population, such as random generation, sequential diversification, parallel diversification, and heuristic initialization [6]. Random generation is the most common technique to generate an initial population. After generation of the initial population, a position updating rule is applied to the current population to generate a new population of solutions. Then, a replacement strategy is applied to update the current population of solutions, and therefore the population of the next generation is constructed. This process continues until a predefined termination condition is satisfied. An essential point in designing a metaheuristic is to provide a good balance between exploration and exploitation of the search space [6].

The shuffled shepherd optimization algorithm (SSOA) is a recent population-based metaheuristic algorithm developed by Kaveh and Zaerreza [7] in 2020. In SSOA, the population of solutions is divided into a number of well-arranged subpopulations of the same size. Then, inspired by the herding behavior of shepherds, the search process is executed for each subpopulation separately. In SSOA, the position updating rule of each search agent involves two components: (1) movement towards a worse solution and (2) movement towards a better solution. The components of the position updating rule are scaled with the parameters α and β , respectively. The parameters α and β control the exploration and exploitation rates of the search process, respectively.

In this chapter first, a parameter-free version of SSOA called parameter-free shuffled shepherd optimization algorithm (PF-SSOA) is proposed. Next, based on the set-theoretical framework proposed by Kaveh et al. [2] for population-based metaheuristic algorithms, a multi-phase version of PF-SSOA called set-theoretical multi-phase shuffle shepherd optimization algorithm (STMP-SSOA) is developed. The main idea of STMP-SSOA is to divide the optimization process of PF-SSOA into several phases. The only difference between the phases is in the number of subpopulations. In other words, different phases are executed with different number of subpopulations. As the optimization process progresses, the number of subpopulations decreases gradually. It should be noted that the framework is applicable to almost all the population-based metaheuristics. In order to demonstrate the performance of the proposed versions of SSOA, four well-known sizing optimization of skeletal structures are considered from the literature. The optimization problems are as follows: size optimization of a 120-bar dome-like truss with multiple frequency constraints, size optimization of a 200-bar planar truss with multiple frequency constraints, and size optimization of two planar frame structures with displacement and stress constraints. The reason for choosing these problems is that they are characterized by different difficulties in discontinuity, non-convexity, and multi-modality of the search space and may stuck in local optima. Frame structures are among the most common types of civil engineering structures. In the past few decades, the optimal design of steel frame structures has been an active area of research in the field of structural optimization and many researchers have been employed various optimization

methods to find optimal steel frame structures: Camp et al. [8] using ACO, Degertekin [9] using harmony search (HS), Kaveh and Bakhshpoori [10] using cuckoo search (CS), Kaveh and Ilchi Ghazaan [11] using colliding bodies optimization (CBO) and enhanced colliding bodies optimization (ECBO), Kaveh and Bakhshpoori [12] using accelerated water evaporation optimization (AWEO), etc.

Structural optimization problems with frequency constraints have a high sensitivity to structural modifications during the optimization process. This is due to the fact that structural size and shape modifications during the optimization process could be contributed to the switching of vibration modes, which can cause significant changes in natural frequencies. In the past few decades, many researchers have been employed a large variety of optimization methods to solve structural optimization problems with frequency constraints. Bellagamba and Yang [13] were among the first researchers who investigated the minimum mass design of truss structures with frequency constraints. Grandhi and Venkayya [14] presented a design optimization algorithm based on an optimality criterion approach for structural weight minimization with multiple frequency constraints. Gomes [15] investigated the performance of particle swarm optimization (PSO) algorithm in solving size and shape truss optimization with frequency constraints. In a similar work, Miguel and Fadel Miguel [16] used HS and firefly algorithm (FA) to solve truss shape and size optimization with multiple natural frequency constraints. Kaveh and Javadi [17] applied an optimization algorithm based on hybridization of PSO, HS, and ray optimization (RO) to solve the size and shape optimization of truss structures with multiple frequency constraints.

In this chapter, two enhanced versions of SSOA are developed to solve sizing optimization of skeletal structures. The performance of the proposed algorithms is tested on two truss optimization problems with frequency constraints and two steel frame optimization problems with strength and displacement constraints. The results obtained by the proposed enhanced versions of SSOA are compared with those of the classical SSOA and other optimization methods in the literature.

The rest of this chapter is organized as follows: The classical shuffled shepherd optimization algorithm is overviewed in Sect. 4.2. In Sect. 4.3, the parameter-free version of the shuffle shepherd optimization algorithm (PF-SSOA) is presented. In Sect. 4.4, the multi-phase version of the shuffle shepherd optimization algorithm (STMP-SSOA) is presented. In Sect. 4.5, a mathematical formulation of the optimization problems is presented. In Sect. 4.6, the proposed algorithms are applied to four benchmark optimization problems and the results are discussed. Finally, the last section concludes the chapter.

4.2 Overview of the Shuffled Shepherd Optimization Algorithm (SSOA)

Kaveh and Zaerreza [7] developed the shuffled shepherd optimization algorithm (SSOA) inspired by the herding behavior of shepherds. Recently, Kaveh et al. [5] employed the concepts of set theory to generalize the statement of the classical SSOA and developed the set-theoretical shuffled shepherd optimization algorithm (ST-SSOA). It should be noted that the only difference between the classical SSOA and ST-SSOA is in the way of defining the algorithms. Therefore, there is no difference between the mathematical formulation of the classical SSOA and ST-SSOA. In the proposed ST-SSOA, based on basic concepts of set theory, the population of candidate solutions is divided into a number of well-arranged subpopulations of the same size. This is done as follows: First, the population of candidate solutions is defined as a set of elements. Therefore, the objective is to divide the set of elements into a finite number, say nS , of well-arranged subsets with the same cardinality. Next, based on the penalized objective function values, a unique weight is assigned to each element. Then, the elements are sorted according to their weight values. In the first step of formation of the subsets, the first nS elements of the sorted set of elements are chosen and one element is placed randomly in each subset. Then, the next nS elements of the sorted set of elements are chosen and one element is placed randomly in each subset. This procedure continues until all elements of the sorted set of elements are placed in the subsets. The cardinality of the subsets, denoted by m , is calculated as the population size divided by the number of subsets (nS). The position updating rule of the ST-SSOA is defined as follows: According to the weight values, for each element i , a better element and a worse element are chosen randomly from the subset to which the element i belongs. In other words, both the better and worse elements must be chosen from the subset to which the element i belongs. The position update of each element is characterized by two movements: (1) movement towards a better element and (2) movement towards a worse element. Therefore, the candidate solutions (i.e., new position of elements) is obtained. After position updating of elements, a greedy replacement strategy is applied to form the subsets of the next generation. If the new position of an element yields a better penalized objective function value than its corresponding current position, then the new position will replace the current one. The above search process is repeated until a termination condition is satisfied. As mentioned above, the new position of elements is affected by two movement components: towards a better element and towards a worse element. The cooperation of the two movement components helps ST-SSOA achieve a proper balance between exploration and exploitation.

It is noted that the movement components are scaled by the parameters α and β . These parameters are taken into account to maintain a suitable balance between exploration and exploitation rates. The set-theoretical shuffled shepherd optimization algorithm (ST-SSOA) is expressed in five steps as follows:

Step one (initialization): The initial population of candidate solutions (EL) is generated randomly within the search space of the design variables as follows:

$$EL(:, j) = Lb(j) + rand(nEL, 1). * (Ub(j) - Lb(j)), \forall j \in \{1, 2, \dots, nV\} \quad (4.1)$$

where $Lb(j)$ and $Ub(j)$ are the lower and upper bounds of the j th design variables, respectively; nV is the number of design variables; nEL is the number of elements (i.e., population size); and EL is the set of elements (i.e., candidate solutions).

Step two (forming the subsets): First, the elements are evaluated and sorted in ascending order of their weight values ($PFit$). Next, the set of elements is divided into nS subsets with the same cardinality. For this purpose, the first nS elements of the sorted set of elements are chosen and one element is randomly placed in each subset. Then, the next nS elements of the sorted set of elements are chosen and one element is randomly placed in each subset. This process continues until no more element remains in the sorted set of elements. At the end of the process, all the subsets have the same number of $m = nEL/nS$ elements. It should be noted that the subsets are close to each other in terms of average penalized objective function values. It is obvious that among the elements of a subset, the first and last elements have the lowest and highest values of penalized objective function, respectively. Equation (4.2) shows the relationship between the population size (nEL) and the number of subsets (nS).

$$nEL = nS \times m \quad (4.2)$$

Step three (position update rule): For each element of each subset, the step size is calculated as follows: For each element, a better element and a worse element are chosen randomly from the subset to which the element under consideration belongs. The step size of ST-SSOA is defined as follows:

$$newEL_{i,j} = stepsize_{i,j} + EL_{i,j}; i = 1, 2, \dots, nS \text{ and } j = 1, 2, \dots, m \quad (4.3)$$

$$stepsize_{i,j} = \beta \times rand_1 \times (EL_B_{i,j} - EL_{i,j}) + \alpha \times rand_2 \times (EL_W_{i,j} - EL_{i,j}) \quad (4.4)$$

where $EL_{i,j}$ is the current position of the j -the element of the i th subset; $newEL_{i,j}$ is the new position of the j -the element of the i th subset; $stepsize_{i,j}$ is the step size of the j th element of the i th subset; $rand_1$ and $rand_2$ are uniformly distributed random numbers from the interval $[0, 1]$; α and β are two parameters to control the balance between exploration and exploitation; $EL_B_{i,j}$ is the position of one of the elements of the i th subset which are fitter than the j th element of the subset; $EL_W_{i,j}$ is the position of one of the elements of the i th subset which are less fit than the j th element of the subset. For example, for the j th element of a subset, there are $j - 1$ better elements and $m - j$ worse elements. However, it is noted that there is no better element for the first element of a subset. Furthermore, there is no worse element for the last element of a subset. Therefore, the first and second terms of Eq. (4.4) are ignored for the best and worst elements of all subsets, respectively.

As mentioned earlier, a successful metaheuristic requires two conflicting search objectives: global exploration of the search space in the early stages of the search process and exploration of promising regions of the search space in the final stages of the search process [6]. In ST-SSOA, the movement towards the position of a worse element is designed to achieve a vast exploration of the search space. On the other hand, the movement towards the position of a better element is designed to facilitate the exploitation of the search space. Therefore, the parameters α and β are defined as decreasing and increasing functions of the iteration number as follows:

$$\beta = \beta_0 + \frac{\beta_{max} - \beta_0}{MaxIter} \times Iter \quad (4.5)$$

$$\alpha = \alpha_0 - \frac{\alpha_0}{MaxIter} \times Iter \quad (4.6)$$

where β_0 and β_{max} are the initial and final values of the parameter β ; α_0 is the initial value of the parameter α ; $Iter$ is the current iteration number; and $MaxIter$ is the maximum number of iterations.

Step four (replacement strategy): A greedy replacement strategy is applied to determine the population of the next generation. For this purpose, the objective function is evaluated at the new positions of the elements. If the new position of an element has a better penalized objective function value than its current position, then the current position is replaced with the new position. In other words, the position with a better weight (i.e., a better penalized objective function value) is preferred. In this way, new subsets of elements are formed. After applying the replacement strategy, all subsets are merged to form the set of elements of the next generation.

Step five (termination criteria): If the termination condition of the algorithm is met, the search process is terminated. Otherwise, the algorithm goes to step two. In this study, the maximum number of objective function evaluations is considered as the stopping criterion.

4.3 Parameter-Free Shuffled Shepherd Optimization Algorithm (PF-SSOA)

The main disadvantages of ST-SSOA are high dependency on parameter tuning and getting trapped in local optima. In other words, the accuracy of the results obtained by ST-SSOA is extremely sensitive to the parameters α and β as well as parameters nEL and nS . In ST-SSOA, the parameters α and β are designed to control the balance between the exploration and exploitation during the search process. The parameter α was defined as a linearly decreasing function of the iteration number and varies from α_0 to zero. Furthermore, the parameter β was defined as a linearly increasing function of the iteration number and varies from β_0 to β_{max} . The increasing rate of change of the parameter β encourages the tendency to move towards better elements,

whereas the decreasing rate of change of the parameter α reduces the tendency to move towards worse elements. The linear change of the parameters α and β with the iteration number may cause the algorithm to be trapped in local optima and the global optimum may not be found. In other words, despite the necessity of a greater diversification of the search space for complicated optimization problems, as the search process continues, ST-SSOA mainly focuses on the intensification of the search process. In order to overcome the shortcomings of ST-SSOA, in this section we propose a parameter-free version of ST-SSOA called the parameter-free shuffled shepherd optimization algorithm (PF-SSOA). The proposed PF-SSOA requires less algorithm-specific parameters compared to ST-SSOA, which makes it easy to be applied. It will be shown that the proposed PF-SSOA can achieve better results compared to ST-SSOA. The only difference between the proposed PF-SSOA and ST-SSOA is in the position updating rule. The position update rule of the proposed PF-SSOA is defined as follows:

$$\text{stepsize}_{i,j} = \beta \times (\text{EL_B}_{i,j} - \text{EL}_{i,j}) + \alpha \times (\text{EL_W}_{i,j} - \text{EL}_{i,j}) \quad (4.7)$$

The parameters α and β are now defined by the following equations:

$$\beta = \text{randi}([1, 2]) * \text{rand}(i, nV) \quad (4.8)$$

$$\alpha = \text{ones}(1, nV) - \beta \quad (4.9)$$

where $\text{randi}([1, 2])$ returns a pseudorandom scalar integer between one and two; and $\text{ones}(1, nV)$ returns a one by nV vector of ones. Equations (4.8) and (4.9) indicate that the parameters α and β are row vectors with nV columns. Therefore, instead of considering a linear variation of the parameters α and β with the iteration number, as shown in Eqs. (4.5) and (4.6), here the parameters α and β are obtained by pseudorandom numbers. As shown in Eq. (4.8), the parameter β can vary between zero and two. As a result, the parameter α can fluctuate between positive and negative one. Figure 4.1 illustrates variation of the parameters α and β with the iteration number in ST-SSOA and PF-SSOA. In this figure, the initial and final values of the parameter β (β_0 and β_{max} , respectively), the initial value of the parameter α (α_0), and the maximum number of iterations ($MaxIter$) are set to be 2.4, 2.8, 0.5, and 200, respectively. A close examination of the figure shows that Eqs. (4.8) and (4.9) can produce pseudorandom numbers for the parameters α and β . Therefore, if the number of samples is large enough, Eqs. (4.8) and (4.9) produce predictable outcomes. The average value of the parameter α is expected to be close to 0.25, and it is close to 0.75 for the parameter β . This means that movement towards better elements is preferred movement towards worse elements. Such a definition of the parameters α and β may help the proposed PF-SSOA to explore the search space more effectively. Furthermore, the sensitivity of PF-SSOA to its parameters (i.e., population size (nEL), the number of subsets (nS), and the maximum number of iterations ($MaxIter$)) is less than ST-SSOA. The flowchart of PF-SSOA is presented

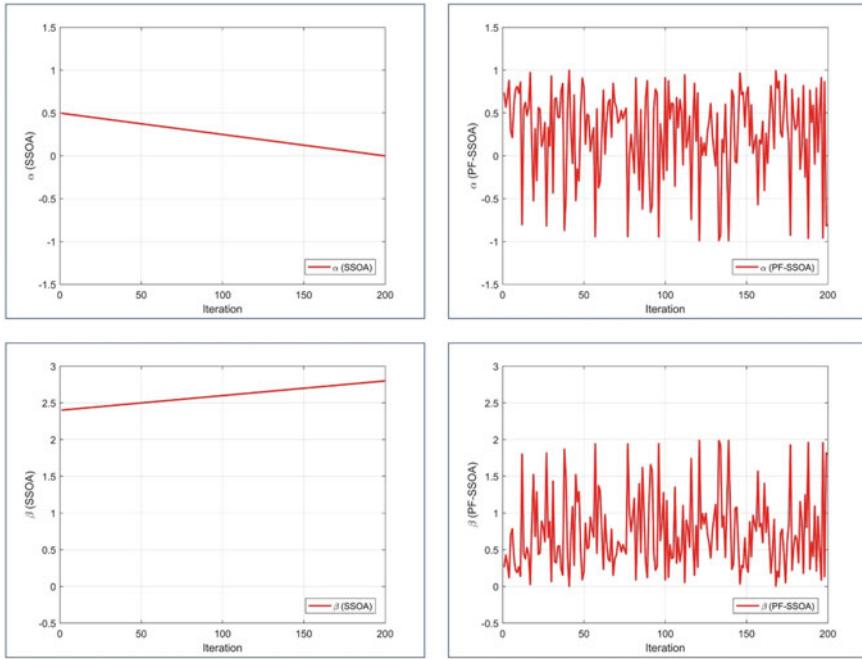


Fig. 4.1 Variation of the parameters α and β with the iteration number in ST-SSOA and PF-SSOA

in Fig. 4.2. Figure 4.3 illustrates the steps of PF-SSOA. The pseudocode of the proposed PF-SSOA is given below:

Initialization:

Set the algorithm parameters: nEL , nS , and $MaxNFEs$.

Generate a set of initial elements (EL) randomly using Eq. (4.1).

Evaluate the set of elements and form the set of penalized objective function values ($PFit$).

$NFEs = 0$;

Cyclic body of PF-SSOA:

While $NFEs < MaxNFEs$

Step 1: Sort the set of elements in ascending order of their penalized objective function values.

Step 2: Form the subsets based on the procedure described in the second step of ST-SSOA.

Step 3: Calculate the step size matrix (*stepsize*) based on Eqs. (4.7), (4.8), and (4.9).

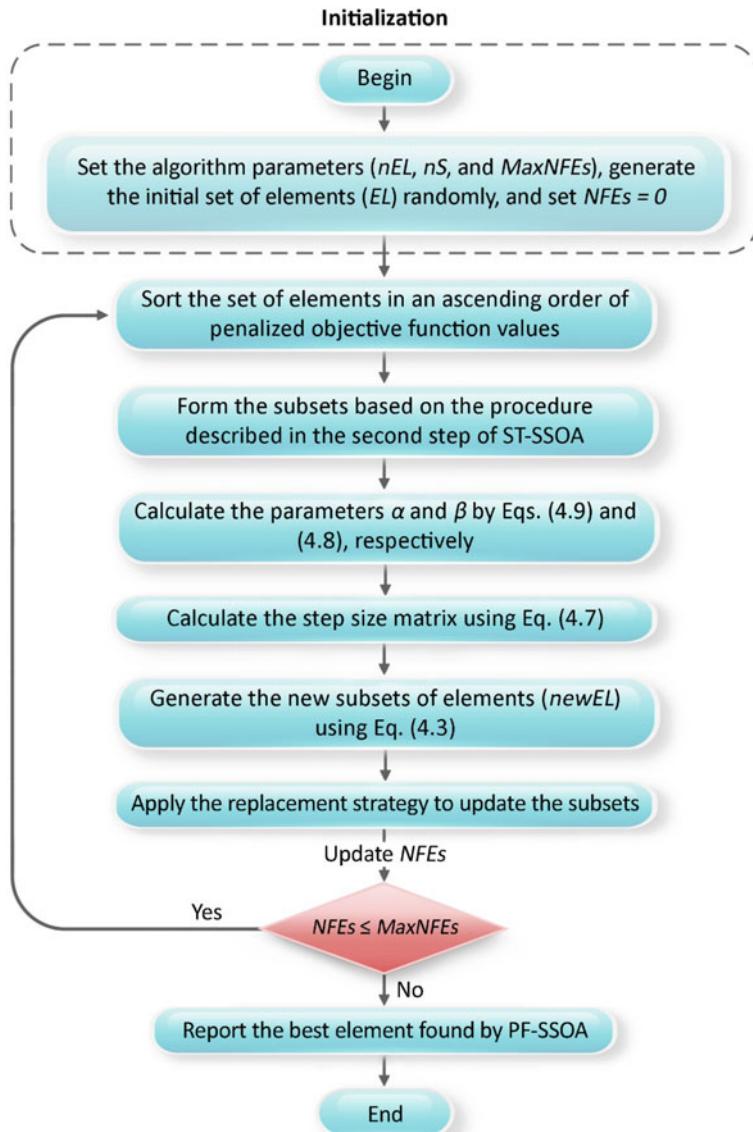


Fig. 4.2 Flowchart of the parameter-free shuffle shepherd optimization algorithm (PF-SSOA)

Step 4: Generate the new subsets of elements based on Eq. (4.3).

Step 5: Evaluate the newly generated subsets ($newEL$) and apply the replacement strategy to update the subsets.

Update the number of objective function evaluations ($NFEs = NFEs + nEL$).

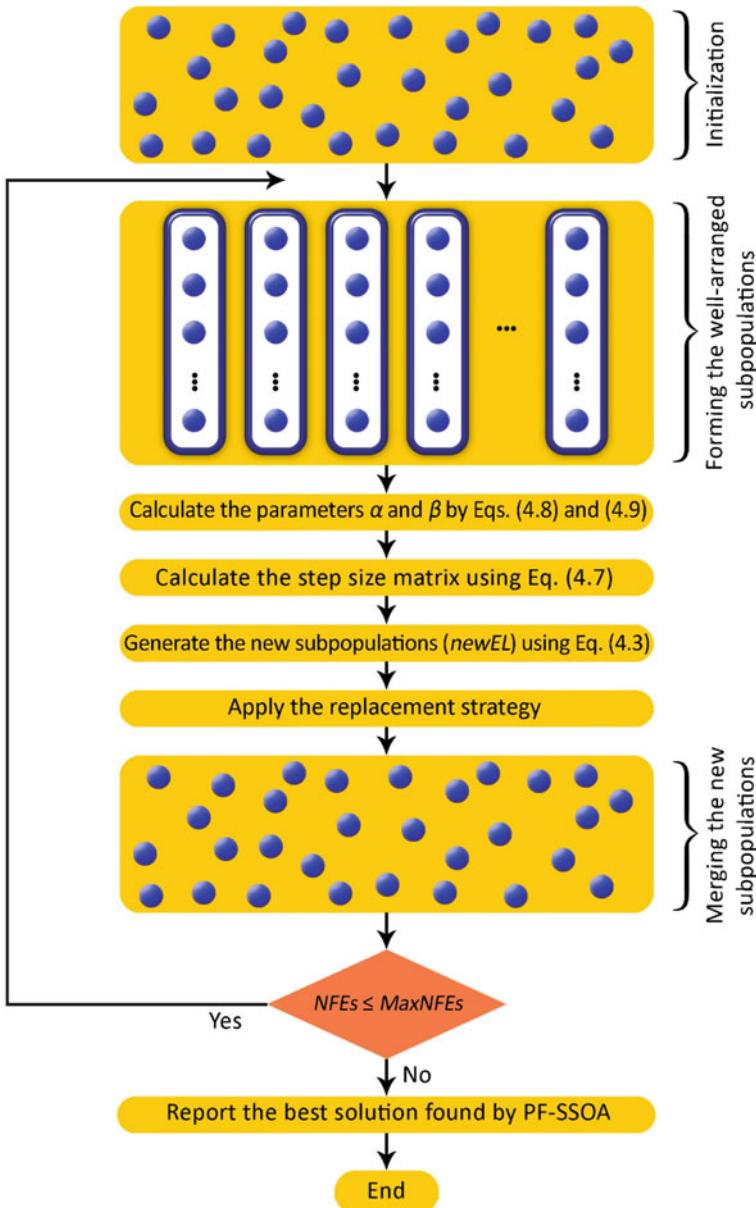


Fig. 4.3 Steps of the parameter-free shuffle shepherd optimization algorithm (PF-SSOA)

Step 6: Merge the subsets and form the set of elements of the next generation.
Monitor the best element found by PF-SSOA so far.

End while

4.4 Set-Theoretical Multi-phase Shuffled Shepherd Optimization Algorithm (STMP-SSOA)

In the previous sections, we have reviewed the set-theoretical SSOA and found that the main idea of ST-SSOA is to divide the set of candidate solutions into a number of well-arranged subsets with the same cardinality. Next, we have developed a parameter-free version of ST-SSOA, aiming to reduce the number of algorithm-specific parameters to be tuned. In this section, based on the proposed PF-SSOA, a multi-phase version of PF-SSOA called the set-theoretical multi-phase shuffle shepherd optimization algorithm (STMP-SSOA) is developed. In STMP-SSOA, the optimization process is divided into several phases. Within each phase of STMP-SSOA, a self-contained PF-SSOA is executed with a specific number of subsets. The first phase of STMP-SSOA uses the initial population as input. In the following phases, the input is the output of the previous phase. Therefore, different phases of STMP-SSOA are executed in a sequence without any cooperation. It is noted that different phases of STMP-SSOA differ only in the number of subsets. As the optimization process progresses, the number of subsets decreases gradually. Each phase continues until the number of objective function evaluations (*NFEs*) reaches a preset threshold. Within each phase of STMP-SSOA, a certain number of objective function evaluations (*MaxNFEs/k*) is executed, where k is the number of phases of STMP-SSOA. The number of phases of STMP-SSOA is calculated as follows: Let D denotes the set of all divisors of nEL which are less than nEL and greater than one. Thus,

$$D = \{n | n \in N; 1 < n < nEL; \text{mod}(nEL, n) = 0\} \quad (4.10)$$

$$\max(k) = |D| \quad (4.11)$$

where N is the set of natural numbers.

The number of subsets of STMP-SSOA can be any element of the set D . Let C denotes a descending combination of the elements of the set D . The elements of the set C are the number of subsets of different phases of STMP-SSOA. The i th element of the set C is the number of subsets of the i th phases of STMP-SSOA ($1 \leq i \leq |C|$). Therefore, the number of phases of STMP-SSOA, denoted by k , is equal to the number of elements of the set C (i.e., $k = |C|$). For example, if the population size is set to 36 (i.e., $nEL = 36$), then the set D is obtained as:

Table 4.1 Four different alternatives of the set C for $nEL = 32$ and $k = 4$

STMP-SSOA	Number of subsets			
Phase number	Alternative 1	Alternative 2	Alternative 3	Alternative 4
1	12	18	9	12
2	6	9	6	9
3	4	3	3	6
4	3	2	2	4
Number of objective function evaluations within each phase	$MaxNFEs/4$	$MaxNFEs/4$	$MaxNFEs/4$	$MaxNFEs/4$

$$D = \{2, 3, 4, 6, 9, 12, 18\}$$

Table 4.1 shows four possible alternatives of the set C for $nEL = 32$ and $k = 4$. For example, the table shows that the third alternative of the set C ($C = \{12, 6, 4, 3\}$) leads to four different phases for STMP-SSOA, in which the first, second, third, and fourth phases have nine, six, three, and two subsets, respectively. In the early iterations of STMP-SSOA, the main population is divided into a large number of small subpopulations. As a result, the subpopulations are scattered all over the search space. Therefore, a large number of subpopulations explore the search space, which allows a better diversification in the whole search space. So, STMP-SSOA has a great ability of global exploration in the early stages of the search process. On the other side, as the search process progresses, the number of subpopulations decreases gradually, and consequently larger subpopulations form. As a result, a greater intensification of the search around promising regions of the search space is obtained. The flowchart of STMP-SSOA is presented in Fig. 4.4. Figure 4.5 illustrates the steps of STMP-SSOA. The pseudocode of STMP-SSOA is provided as follows:

Initialization:

Set the algorithm parameters: nEL , C , and $MaxNFEs$.

Generate a set of initial elements (EL) randomly using Eq. (4.1).

Evaluate the set of elements and form the set of penalized objective function values ($PFit$).

Obtain the set D .

Obtain the set C and calculate its cardinality ($k = |C|$).

$i = 0$;

$NFEs = 0$;

Cyclic body of STMP-SSOA:

While $i < k$

$i = i + 1$

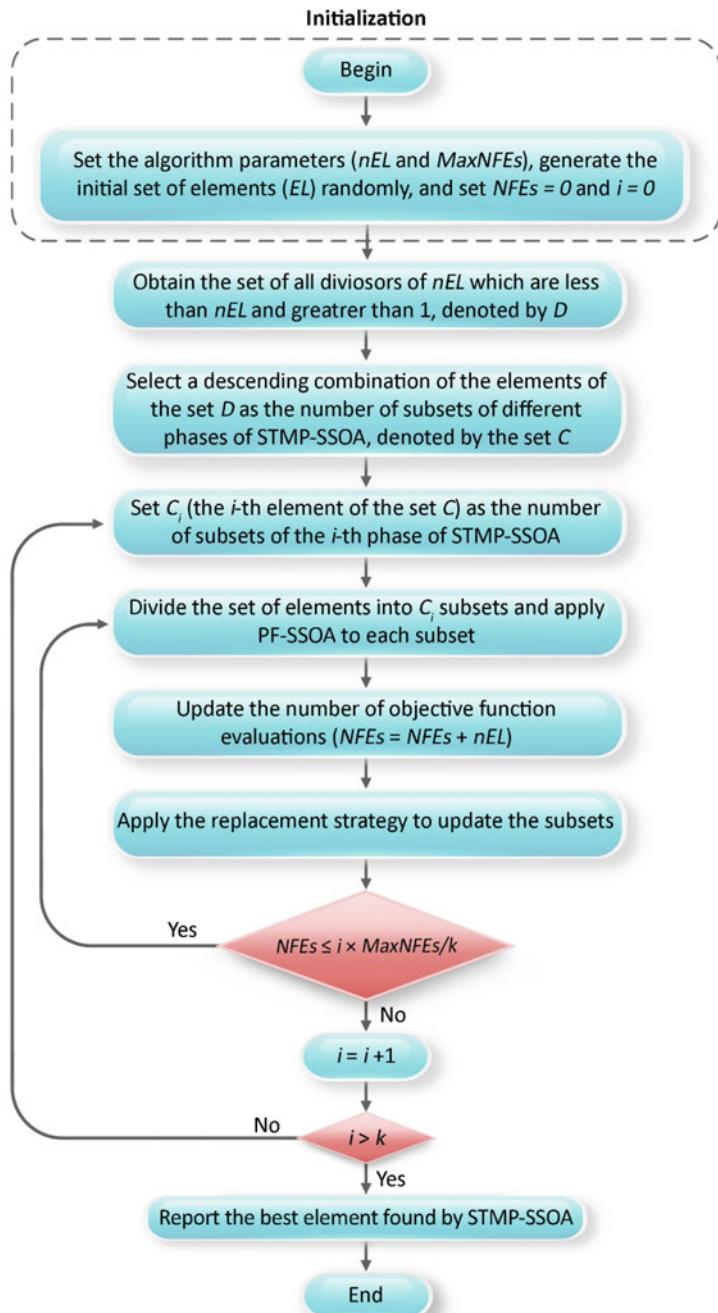


Fig. 4.4 Flowchart of the set-theoretical multi-phase shuffle shepherd optimization algorithm (STMP-SSOA)

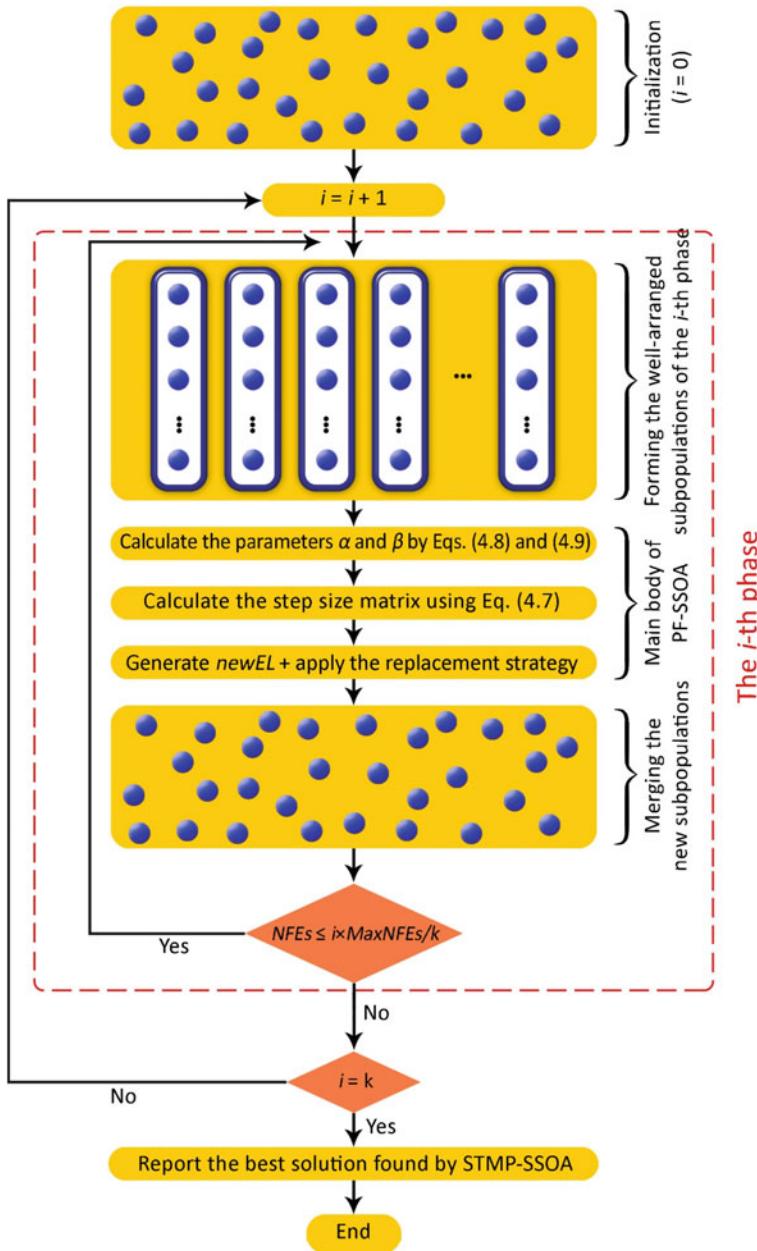


Fig. 4.5 Steps of the set-theoretical multi-phase shuffle shepherd optimization algorithm (STMP-SSOA)

While $NFEs < i \times MaxNFEs/k$

Step 1: Sort the set of elements in ascending order of their penalized objective function values.

Step 2: Form the subsets of the i th phase based on the procedure described in the second step of ST-SSOA.

Step 3: Calculate the step size matrix (*stepsize*) based on Eqs. (4.7), (4.8), and (4.9).

Step 4: Generate the new subsets of elements based on Eq. (4.3).

Step 5: Evaluate the newly generated subsets (*newEL*) and apply the replacement strategy to update the subsets.

Update the number of objective function evaluations ($NFEs = NFEs + nEL$).

Step 6: Merge the subsets and form the set of elements of the next generation.

Monitor the best element found by STMP-SSOA so far.

End while

End while

4.5 Formulation of the Optimization Problems

In this study, in order to demonstrate the performance of the proposed versions of SSOA for solving discrete and continuous sizing optimization of skeletal structures, two benchmark truss sizing optimization problems with continuous design variables under multiple natural frequency constraints and two benchmark steel frame optimization problems with discrete design variables under strength and displacement constraints are considered from the literature. In the next two subsections, the optimization problems are mathematically formulated as follows.

4.5.1 Size Optimization of Truss Structures with Frequency Constraints

Truss optimization with frequency constraints is known as a challenging optimization problem because of highly nonlinear constraints and non-convex search spaces. In a truss optimization problem with frequency constraints, the main objective is usually to minimize the weight of the truss structure while satisfying constraints on natural frequencies. In the case of continuous sizing optimization of truss structures, the cross-sectional areas of the members are considered as design variables, which can vary continuously between the lower and upper bounds of the design variables. The layout of the structure is predetermined and assumed to be unchanged during the

optimization process. The problem of sizing optimization of a truss structure with multiple frequency constraints can be mathematically formulated as follows:

$$\text{Find: } \{D\} = [d_1, d_2, \dots, d_{nD}], d_i \in S_i \quad (4.12)$$

$$\text{to minimize: } PFit(\{D\}) = W(\{D\}) + f_{penalty}(\{D\}) \quad (4.13)$$

$$\text{subject to: } \begin{cases} \omega_j \geq \omega_j^* & \text{for some natural vibration frequencies } j \\ \omega_k \leq \omega_k^* & \text{for some natural vibration frequencies } k \\ d_i^L \leq d_i \leq d_i^U & i = 1, 2, \dots, nD \end{cases} \quad (4.14)$$

where $W(\{D\})$ is the objective function which is considered to be the weight of the whole truss structure, and is given by the following equation:

$$W(\{D\}) = \sum_{i=1}^{nE} \rho_i A_i L_i \quad (4.15)$$

In the above equations, $\{D\}$ is the vector of design variables representing the cross-sectional areas of the members; d_i is the i th design variable (i.e., the cross-sectional area of the i th member group); nD is the number of sizing design variables (i.e., the number of member groups); S_i is the allowable range for the i th design variable; $f_{penalty}(\{D\})$ and $PFit(\{D\})$ are the penalty function and penalized objective function, respectively; nE is the number of truss members; ρ_i , A_i , and L_i are the material density, cross-sectional area, and length of the i th truss member, respectively; d_i^L and d_i^U are the lower and upper bounds of the i th design variable, respectively; ω_j and ω_k are the j th and k th natural frequencies of the structure, respectively; ω_j^* is the lower bound of the j th natural frequency; and ω_k^* is the upper bound of the k th natural frequency. The allowable range for the i th design variable (S_i) can be expressed as follows:

$$S_i = \{x_i \in R^+ | d_i^L \leq x_i \leq d_i^U\} \quad (4.16)$$

where R^+ is the set of positive real numbers.

In order to determine the natural frequencies and modes of vibration of a structural system, it is required to solve the eigenvalue problem of the system stiffness and mass matrices shown in the following equation [2]:

$$K\phi_n = \gamma_n M\phi_n; n = 1, 2, \dots, N \quad (4.17)$$

where K and M denote the stiffness and mass matrices of the structural system, respectively; ϕ_i and γ_i represent the i th eigenvalue and the corresponding eigenvector

(i.e., mode shape) of the structural system, respectively; and n ranges from 1 to N , where N represents the total number of degrees of freedom of the structural system. It is noted that the obtained eigenvalues are the square of the natural frequencies. This can be written as follows:

$$\gamma_i = \omega_i^2 = (2\pi/T_i)^2; n = 1, 2, \dots, N \quad (4.18)$$

where T_i is the i th natural period of the structural system.

One of the well-known constraint handling strategies is the penalizing strategy, in which a penalty function is employed to transfer the constrained optimization problem into an unconstrained one [6]. The main idea of the penalizing strategy is to penalize the infeasible solutions according to the degree of violation of the constraints. In this study, the following penalizing function is adopted to handle the frequency constraints:

$$f_{penalty}(\{D\}) = (1 + \varepsilon_1 \times v)^{\varepsilon_2}; v = \sum_{i=1}^{nC} v_i \quad (4.19)$$

where nC is the number of frequency constraints; v is the sum of the constraint violations; v_i is the violation of the i th frequency constraint; and ε_1 and ε_2 are the parameters of the penalty function. In this study, ε_1 is set to unity, while ε_2 is considered to increase linearly with the iteration number, as shown in the following equation:

$$\varepsilon_2 = 1.5 \times \left(1 + \frac{Iter}{MaxIter} \right) \quad (4.20)$$

If the i th frequency constraint is satisfied, then the value of v_i is set to zero. Otherwise, it is calculated according to the severity of the violation. It can be expressed as follows:

$$v_i = \begin{cases} \left| 1 - \frac{\omega_i}{\omega_i^*} \right|, & \text{the } i\text{-th frequency constraint is violated} \\ 0, & \text{otherwise} \end{cases} \quad (4.21)$$

4.5.2 Discrete Size Optimization of Steel Frame Structures

The frame structures should be designed subject to strength and serviceability constraints as specified by the American Institute of Steel Construction-Load and Resistance Factor Design (AISC-LRFD) specification [18]. The problem can be expressed mathematically as follows [10]:

$$\text{Find: } \{D\} = [d_1, d_2, \dots, d_{nD}], d_i \in R_i \quad (4.22)$$

$$\text{to minimize: } PFit(\{D\}) = W(\{D\}) + f_{penalty}(\{D\}) \quad (4.23)$$

$$\text{subject to: } g_i(\{D\}) \leq 0, i = 1, 2, \dots, nC \quad (4.24)$$

where $W(\{D\})$ is the objective function which is considered to be the weight of the whole frame structure, and is given by the following equation:

$$W(\{D\}) = \sum_{i=1}^{nE} \rho_i A_i L_i \quad (4.25)$$

In the above equations, nD , $PFit(\{D\})$, and $f_{penalty}(\{D\})$ are defined similarly to those in Eqs. (4.12) and (4.13); $\{D\}$ is the vector of design variables; d_i is the i th design variable; nE is the number of members of the frame structure; ρ_i , A_i , and L_i denote the material density, cross-sectional area, and length of the i th frame member, respectively; R_i is the allowable set of values for the design variable d_i ; nC is the total number of problem constraints; and $g_i(\{D\})$ represents the i th design constraint of the problem. The design constraints include strength constraints of the AISC-LRFD specification [18] and displacement constraints. Similar to the truss size optimization problems, the penalty function is defined by Eq. (4.19). The design variable d_i must be selected from a given set of discrete values represented by R_i . The set R_i is given by

$$R_i = \{r_{i,1}, r_{i,2}, r_{i,3}, \dots, r_{i,nV(i)}\} \quad (4.26)$$

where $nV(i)$ is the number of available discrete values for the design variable d_i .

According to the AISC-LRFD specification [18], the design constraints of the problem can be expressed as follows:

(1) Maximum lateral displacement

$$\frac{\Delta_T}{H} - R \leq 0 \quad (4.27)$$

where Δ_T is the maximum lateral displacement at the top of the frame; H is the height of the structure to that level; and R is the maximum allowable drift ratio, which is set to 1/300 for this study.

(2) Inter-story displacements

$$\frac{d_i}{h_i} - R_I \leq 0, i = 1, 2, \dots, ns \quad (4.28)$$

where d_i is the inter-story drift of the i th story; h_i is the height of the i th story; ns represents the total number of stories of the frame; and R_I is the maximum allowable inter-story drift ratio, which is set to 1/300.

(3) Strength constraints

$$\begin{cases} \frac{P_r}{2\phi_c P_n} + \left(\frac{M_{rx}}{\phi_b M_{nx}} + \frac{M_{ry}}{\phi_b M_{ny}} \right) \leq 1, \frac{P_r}{\phi_c P_n} < 0.2 \\ \frac{P_r}{\phi_c P_n} + \frac{8}{9} \left(\frac{M_{rx}}{\phi_b M_{nx}} + \frac{M_{ry}}{\phi_b M_{ny}} \right) \leq 1, \frac{P_r}{\phi_c P_n} \geq 0.2 \end{cases} \quad (4.29)$$

where P_r and P_n represent the required and nominal axial (tensile or compressive) strengths, respectively; ϕ_b is the resistance factor for flexure ($\phi_b = 0.9$); M_{rx} and M_{nx} denote the required and nominal flexural strengths about the x axis, respectively; M_{ry} and M_{ny} are the required and nominal flexural strengths about the y axis, respectively; and ϕ_c is the resistance factor ($\phi_c = 0.9$ for tension, $\phi_c = 0.85$ for compression). It should be noted that for two-dimensional structures, $M_{ny} = 0$.

4.6 Numerical Examples

In this section, in order to demonstrate the performance of the proposed algorithms in sizing optimization of skeletal structures, four benchmark sizing optimization problems of skeletal structures are studied. The optimization problems include a 120-bar dome-like truss with seven sizing design variables, a 200-bar planar truss with 29 sizing design variables, a 3-bay 15-story steel frame structure with 11 sizing design variables, and a 3-bay 24-story steel frame structure with 20 sizing design variables. The frame structures are analyzed by linear static analysis method. The results obtained by PF-SSOA and STMP-SSOA are compared with those of the classical SSOA and some other optimization methods in the literature. For each problem, a comparison of the weight convergence histories obtained by the classical SSOA, PF-SSOA, and STMP-SSOA is presented. A magnified view is added to the convergence histories to better display the convergence characteristics. Furthermore, for each problem, a comparison of the optimized weights found by the algorithms at different stages of the optimization process is provided, which allows a better comparison of the convergence rates of the three algorithms. In order to account for the stochastic nature of the optimization process, each problem was solved 10 times (i.e., 10 independent runs). However, only the optimal design corresponding to the best runs are reported. For all algorithms, the initial population is generated randomly, and the maximum number of objective function evaluations ($MaxNFEs$) is considered as the stopping condition of the optimization process, which is set to 30,000 for the 200-bar planar truss problem and 20,000 for the other ones. In some problems, however, the number of objective function evaluations which is needed to obtain the final solution may be much less than $MaxNFEs$. The required codes are implemented in the Matlab environment.

4.6.1 A 120-Bar Dome-Like Truss

The first design example investigated in this study is the 120-bar dome-like truss shown in Fig. 4.6. Design variables include only the member cross-sectional areas. Therefore, the layout of the structure is assumed to be unchanged during the optimization process. According to the symmetry of the dome, the members are divided into seven different member groups, as shown in Fig. 4.6. Thus, this is a sizing optimization problem with seven design variables. Non-structural masses are attached to all free nodes of the dome as follows: 3000 kg at node 1, 500 kg at nodes 2 to 13, and 100 kg at the rest of the nodes. Material properties, cross-sectional area bounds, and frequency constraints of the problem are listed in Table 4.2. As shown in the table, the frequency constraints are imposed on the first two natural frequencies of the dome. This problem has been previously studied by many researchers using various optimization methods: Kaveh and Ilchi Ghazaan [19] using CBO and ECBO algorithms, Kaveh and Ilchi Ghazaan [20] using hybridization of ECBO with upper bound strategy (UECBO), Kaveh et al. [2] using set-theoretical variants of the TLBO algorithm, etc.

The optimization results obtained by the classical SSOA and PF-SSOA for different combinations of nEL and nS are presented in Tables 4.3 and 4.4, respectively. It can be seen from Table 4.3 that in terms of average weight, the best combination of nEL and nS for the classical SSOA is $nEL = 15$ and $nS = 3$, with an average weight of 8711.97 kg. Also, similar to the classical SSOA, $nEL = 15$ and $nS = 3$ is the best combination of nEL and nS for PF-SSOA in terms of average weight value, with an average weight of 8710.13 kg (please see Table 4.4). Furthermore, for both the classical SSOA and PF-SSOA, the best result in terms of standard deviation of optimized weights was obtained when $nEL = 15$ and $nS = 3$. The best weight of the classical SSOA is 8709.81 kg, while it is 8708.66 kg for PF-SSOA. A comparison of the results of the classical SSOA with those of PF-SSOA shows that PF-SSOA performs better than the classical SSOA, especially in terms of best weight, average weight, and standard deviation of optimized weights. Tables 4.5 and 4.6 provide the first five natural frequencies of the best designs of the 120-bar dome-like truss found by the classical SSOA and PF-SSOA, respectively. As can be seen from the tables, none of the frequency constraints are violated. Table 4.7 provides a comparison of the best results obtained by the present study with those of other methods in the literature. It is noted that the best combinations of nEL and nS for the classical SSOA, PF-SSOA, and STMP-SSOA are considered to be those which have better average weights. As can be seen Table 4.7, STMP-SSOA performs better than the classical SSOA and PF-SSOA in terms of best weight, average weight, and standard deviation of optimized weights. Table 4.8 lists the first five natural frequencies of the best designs obtained by different algorithms for the 120-bar dome-like truss. As expected, all of the two frequency constraints are satisfied. Table 4.9 provides a comparison of the optimized weights obtained by the classical and proposed versions of SSOA at six different stages of the optimization process. As the table demonstrates, during the early stages of the optimization process, the classical SSOA exhibits faster

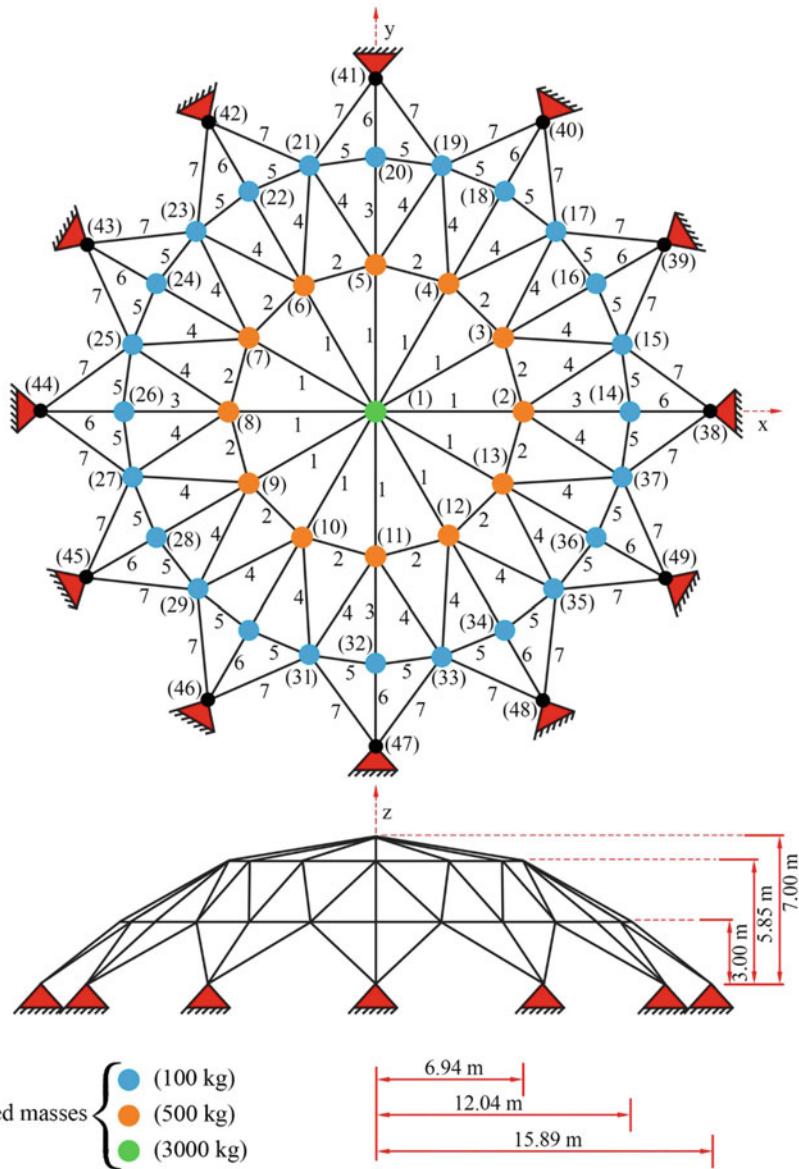


Fig. 4.6 Schematic of the 120-bar dome-like truss structure

Table 4.2 Material properties, cross-sectional area bounds, and frequency constraints of the problems

Property	120-bar dome-like truss	200-bar planar truss
Elasticity modulus (N/m ²)	2.1×10^{11}	2.1×10^{11}
Material density (kg/m ³)	7971.81	7800
Cross-sectional area bounds (cm ²)	$1 \leq A_i \leq 129.3$	$0.1 \leq A_i$
Frequency constraints (Hz)	$\omega_1 \geq 9, \omega_2 \geq 11$	$\omega_1 \geq 5, \omega_2 \geq 10, \omega_3 \geq 15$

convergence rate compared to PF-SSOA and STMP-SSOA, while in the final stages, both the proposed algorithms have significantly higher convergence rates compared to the classical SSOA. Figure 4.7 illustrates the diversity of the optimized weights obtained by the classical SSOA, ST-SSOA, and STMP-SSOA over 10 independent runs. Figure 4.8 shows the convergence histories of the average results obtained by the classical SSOA, ST-SSOA, and STMP-SSOA over 10 independent runs.

Table 4.3 Comparison of optimal results obtained by the classical SSOA for the 120-bar truss

Design variable: A (cm ²)	$nEL = 9$ $nS = 3$	$nEL = 12$ $nS = 3$	$nEL = 12$ $nS = 4$	$nEL = 15$ $nS = 3$	$nEL = 15$ $nS = 5$	$nEL = 16$ $nS = 4$
1	18.8735	19.5340	19.3888	19.2524	19.5530	19.7367
2	42.3496	40.4157	40.7252	40.9097	40.2729	39.7321
3	10.9566	10.3684	10.7372	10.9612	10.6773	10.6318
4	21.1273	20.9316	21.3078	21.1648	21.1977	21.0044
5	10.0104	9.8901	9.8250	9.9664	9.7848	9.7843
6	11.6377	12.0552	11.7766	11.5109	11.7615	11.6369
7	14.4939	14.9986	14.5664	14.6744	14.7783	15.1032
Weight (kg)	8717.65	8710.02	8709.81	8710.84	8710.15	8709.95
Number of FE analyses	13,554	19,932	19,560	19,830	20,000	19,872
Average weight (kg)	8732.69	8715.38	8724.83	8711.97	8714.81	8713.76
Worst weight (kg)	8783.45	8726.59	8769.36	8713.95	8721.67	8730.18
Standard deviation (kg)	20.316	4.896	16.690	1.021	4.113	5.642
Maximum number of FE analyses	20,000	20,000	20,000	20,000	20,000	20,000
Number of runs	10	10	10	10	10	10

Table 4.4 Comparison of optimal results obtained by PF-SSOA for the 120-bar truss

Design variable: A (cm^2)	$nEL = 9$ $nS = 3$	$nEL = 12$ $nS = 3$	$nEL = 12$ $nS = 4$	$nEL = 15$ $nS = 3$	$nEL = 15$ $nS = 5$	$nEL = 16$ $nS = 4$
1	19.4514	19.6134	19.3823	19.6057	19.5016	19.5595
2	40.5017	40.0362	40.4713	40.3226	40.4300	40.1643
3	10.6920	10.6487	10.7731	10.6269	10.7618	10.5010
4	21.1599	21.0862	21.1410	21.1172	21.0472	21.0342
5	9.8592	9.8323	10.0158	9.6692	9.7885	9.9286
6	11.7533	11.7058	11.6822	11.7423	11.5972	11.9095
7	14.7536	14.9294	14.7360	14.8971	14.9334	14.9170
Weight (kg)	8708.66	8708.84	8709.28	8708.82	8709.21	8708.95
Number of FE analyses	13,113	17,976	16,584	12,240	18,240	14,848
Average weight (kg)	8714.45	8710.60	8712.05	8710.13	8710.44	8710.66
Worst weight (kg)	8724.86	8715.81	8720.92	8712.20	8713.13	8711.75
Standard deviation (kg)	4.968	1.888	3.219	1.091	1.135	0.842
Maximum number of FE analyses	20,000	20,000	20,000	20,000	20,000	20,000
Number of runs	10	10	10	10	10	10

Table 4.5 First five natural frequencies (Hz) of the 120-bar dome-like truss evaluated at optimal designs found by the classical SSOA

Frequency number	$nEL = 9$ $nS = 3$	$nEL = 12$ $nS = 3$	$nEL = 12$ $nS = 4$	$nEL = 15$ $nS = 3$	$nEL = 15$ $nS = 5$	$nEL = 16$ $nS = 4$
1	9.0000	9.0003	9.0001	9.0000	9.0005	9.0000
2	11.0000	11.0000	11.0000	11.0000	11.0004	11.0000
3	11.0000	11.0000	11.0000	11.0000	11.0004	11.0000
4	11.0000	11.0000	11.0000	11.0000	11.0022	11.0000
5	11.0664	11.0678	11.0662	11.0666	11.0687	11.0671

Table 4.6 First five natural frequencies (Hz) of the 120-bar dome-like truss evaluated at optimal designs found by PF-SSOA

Frequency number	$nEL = 9$ $nS = 3$	$nEL = 12$ $nS = 3$	$nEL = 12$ $nS = 4$	$nEL = 15$ $nS = 3$	$nEL = 15$ $nS = 5$	$nEL = 16$ $nS = 4$
1	9.0000	9.0001	9.0001	9.0001	9.0001	9.0002
2	11.0001	11.0000	11.0000	11.0000	11.0000	11.0000
3	11.0001	11.0002	11.0000	11.0000	11.0000	11.0000
4	11.0004	11.0002	11.0004	11.0000	11.0010	11.0001
5	11.0671	11.0670	11.0676	11.0663	11.0676	11.0677

Table 4.7 Comparison of optimal results obtained for the 120-bar dome-like truss

Design variable: A (cm ²)	ECBO [19]	UECBO [20]	OST-TLBO [2]	STMP-TLBO [2]	This study		
					SSOA	PF-SSOA	STMP-SSOA
					$nEL = 15$	$nEL = 15$	$nEL = 12$
					$nS = 3$	$nS = 3$	$nS = 4, 3$
1	19.8290	19.5286	19.5034	19.5554	19.2524	19.6057	19.4774
2	41.4037	40.8324	40.6005	40.2398	40.9097	40.3226	40.5159
3	11.0055	11.5304	10.6598	10.5967	10.9612	10.6269	10.6383
4	21.2971	21.6495	21.0981	21.1778	21.1648	21.1172	21.1015
5	9.4718	10.2915	9.7095	9.8356	9.9664	9.6692	9.8216
6	13.0176	12.6229	11.7346	11.8421	11.5109	11.7423	11.7800
7	15.2840	14.7256	14.8614	14.7767	14.6744	14.8971	14.8270
Weight (kg)	8896.50	8894.54	8708.938	8708.894	8710.84	8708.82	8708.35
Number of FE analyses	N/A	N/A	N/A	N/A	19,830	12,240	15,708
Average weight (kg)	8920.16	8936.9	8710.201	8710.040	8711.97	8710.13	8710.07
Worst weight (kg)	N/A	N/A	8711.685	8711.534	8713.95	8712.20	8711.80
Standard deviation (kg)	20.12	29.38	0.634	0.693	1.021	1.091	0.931
Maximum number of FE analyses	20,000	20,000	20,000	20,000	20,000	20,000	20,000
Number of runs	20	20	20	20	10	10	10

4.6.2 A 200-Bar Planar Truss

The 200-bar planar truss structure shown in Fig. 4.9 is considered as the second design example. The layout of the structure is assumed to be fixed during the optimization process. Due to the symmetry of the structure, the members are categorized into 29 different member groups, as shown in Fig. 4.10. Five non-structural masses of 100 kg are attached to upper nodes of the structure, as depicted in Fig. 4.9. Table 4.2 summarizes the cross-sectional area bounds, material properties, and frequency constraints of the problem. As can be seen from the table, the problem constraints are imposed

Table 4.8 First five natural frequencies (Hz) of the 120-bar dome-like truss evaluated at optimal designs found by different algorithms

Frequency number	ECBO [19]	UECBO [20]	OST-TLBO [2]	STMP-TLBO [2]	This study		
					SSOA	PF-SSOA	STMP-SSOA
					$nEL = 15$	$nEL = 15$	$nEL = 12$
					$nS = 3$	$nS = 3$	$nS = 4, 3$
1	9.001	9.001	9.0001	9.0004	9.0000	9.0001	9.0000
2	11.001	11.000	11.0001	11.0001	11.0000	11.0000	11.0001
3	11.003	11.000	11.0001	11.0001	11.0000	11.0000	11.0001
4	11.010	11.010	11.0003	11.0001	11.0000	11.0000	11.0001
5	11.052	11.050	11.0666	11.0669	11.0666	11.0663	11.0668

Table 4.9 Optimized weights for the 120-bar dome truss at six different stages of the optimization process (best runs)

Number of structural analyses	Optimized weight (kg)		
	SSOA	PF-SSOA	STMP-SSOA
	$nEL = 15$	$nEL = 15$	$nEL = 12$
3000	8721.66	8732.23	8734.17
6000	8712.67	8713.83	8715.07
9000	8712.51	8711.06	8710.48
12,000	8711.74	8710.09	8708.74
15,000	8711.49	8710.09	8708.46
18,000	8711.32	8710.08	8708.35

on the first three natural frequencies of the structure. This problem has been previously solved by many researchers using different optimization algorithms: Kaveh and Ilchi Ghazaan [19] using CBO and ECBO, Taheri and Jalili [21] using enhanced biogeography-based optimization (EBBO) algorithm, Kaveh and Ilchi Ghazaan [20] using UECBO, and Kaveh et al. [2] utilizing set-theoretical variants of the TLBO algorithm.

Table 4.10 provides a comparison of the results for the 200-bar planar truss using the classical SSOA and PF-SSOA for four different combinations of nEL and nS . It can be seen from the table that in terms of average weight, the classical SSOA obtained the best result when we use 16 as the nEL value and 4 as the nS value, whereas the average weight obtained by PF-SSOA has the best value when $nEL = 24$ and $nS = 6$. As the table demonstrates, for all combinations of nEL and nS , PF-SSOA has achieved better results than the classical SSOA in terms of best weight, average weight, and worst weight. The first five natural frequencies of the best designs of the 200-bar planar truss found by the classical SSOA and PF-SSOA are summarized in Table 4.11. The results indicate that all of the three frequency constraints are satisfied. It can also be concluded from the Table 4.11 that the constraints on the

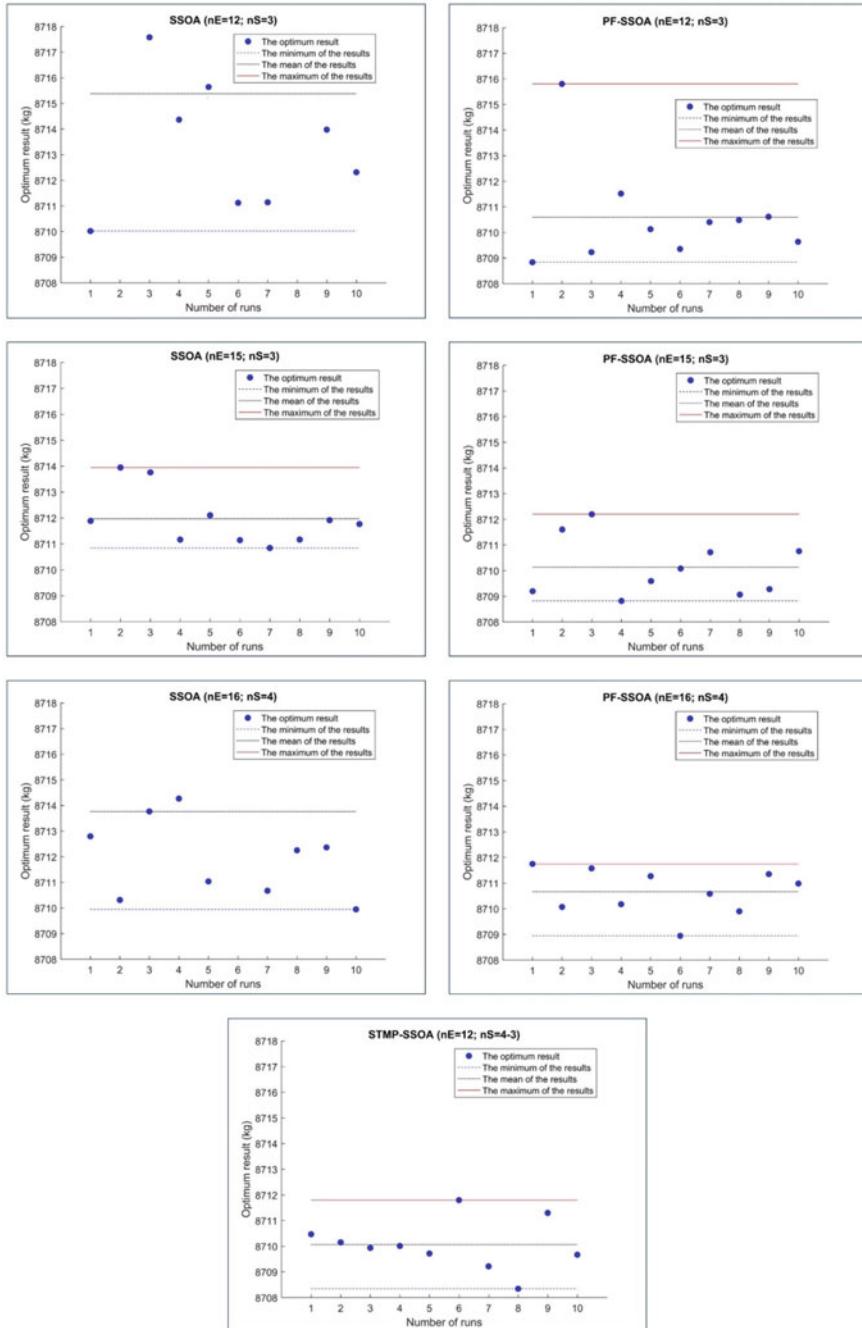


Fig. 4.7 Optimized weights obtained by the classical SSOA, PF-SSOA, and STMP-SSOA for the 120-bar dome-like truss over 10 independent runs

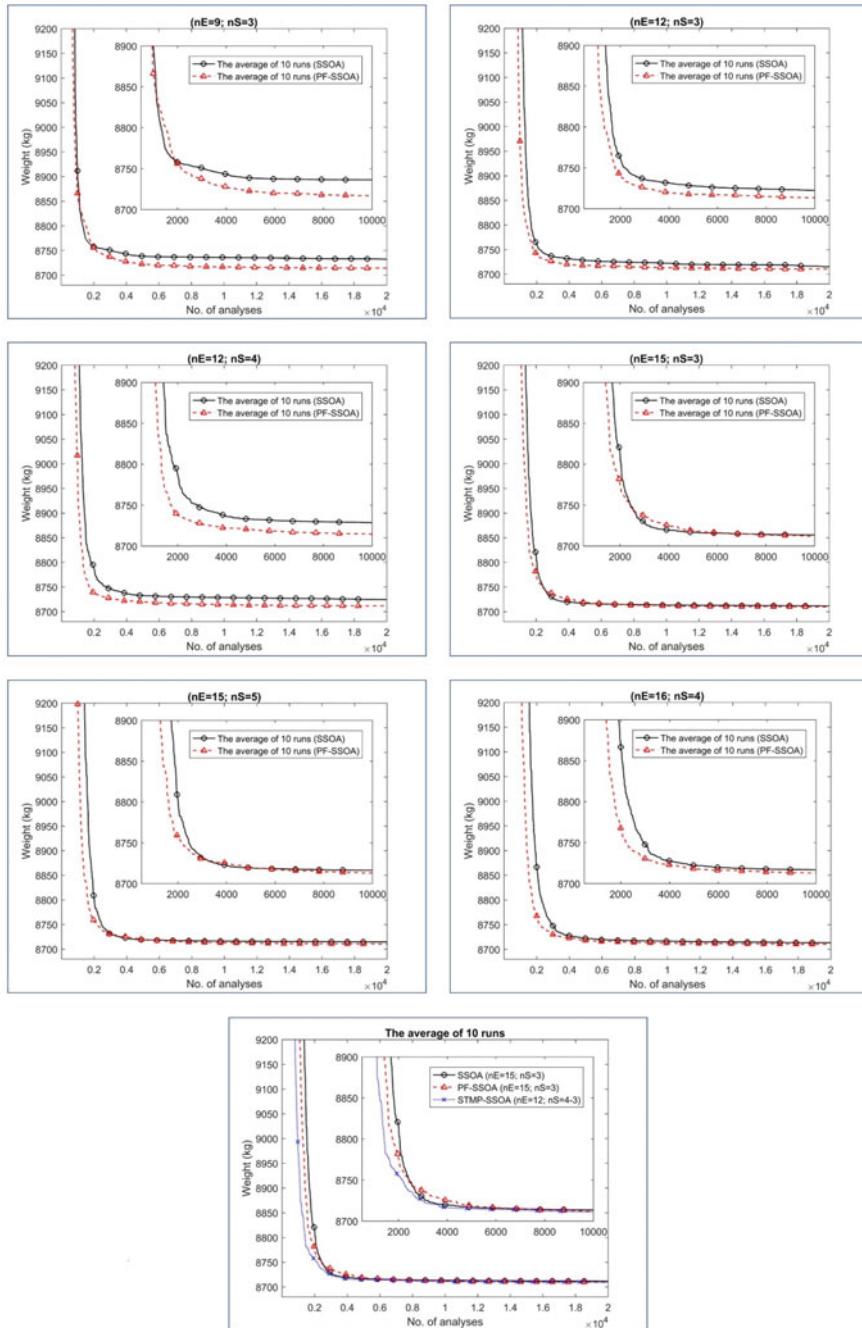


Fig. 4.8 Average weight convergence histories for the 120-bar dome-like truss obtained by the classical SSOA, PF-SSOA, and STMP-SSOA

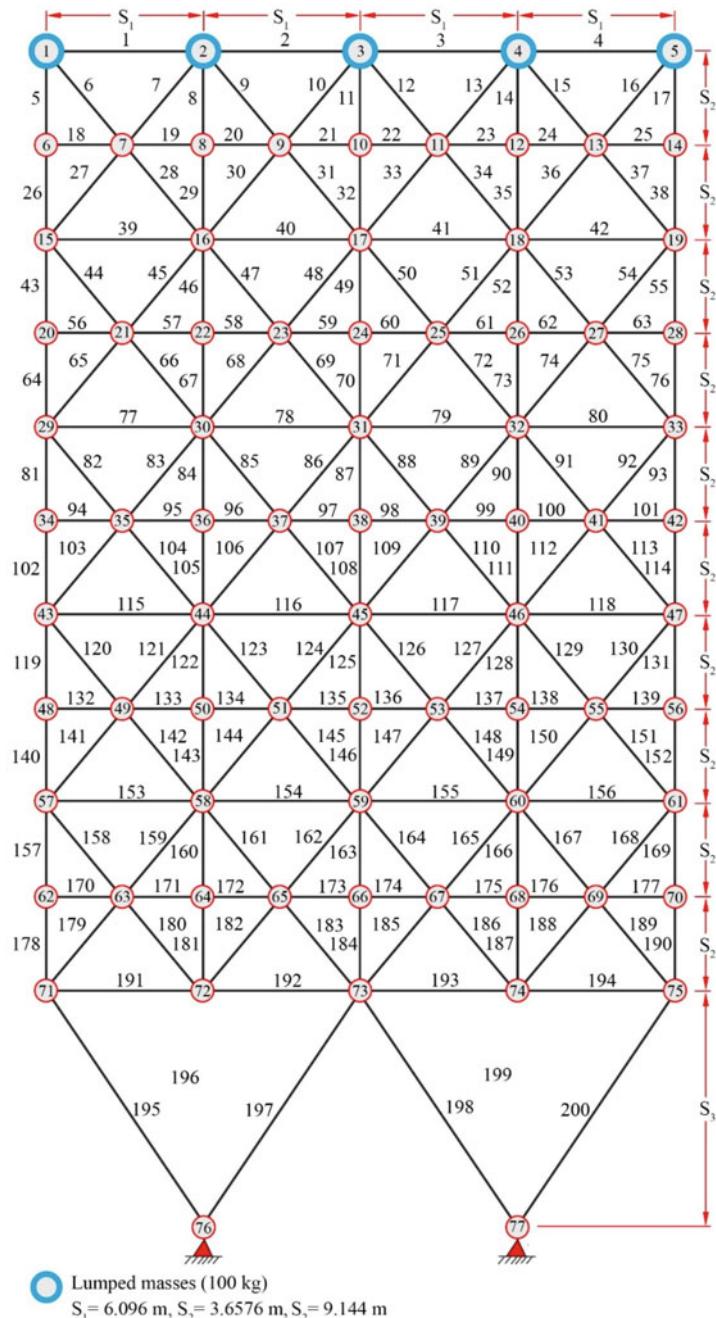


Fig. 4.9 Schematic of the 200-bar planar truss structure

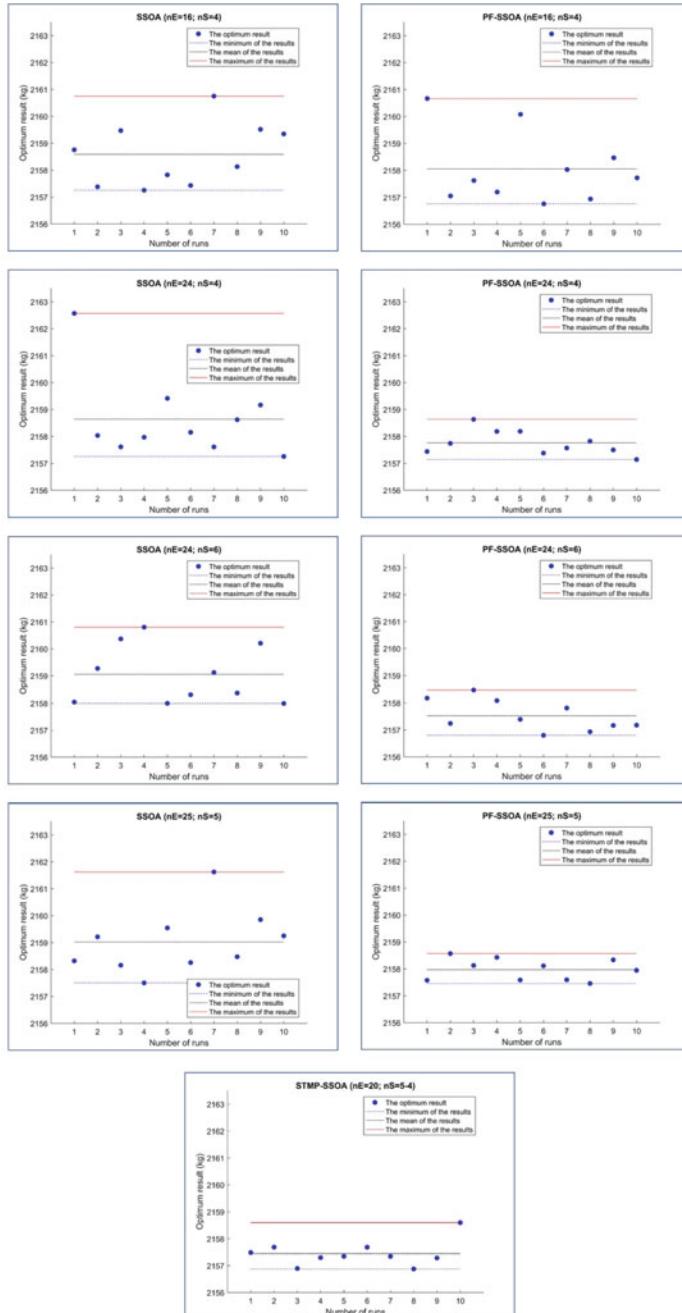


Fig. 4.10 Optimized weights obtained by the classical SSOA, PF-SSOA, and STMP-SSOA for the 200-bar planar truss over 10 independent runs

first and third natural frequencies control the design process. Table 4.12 provides a comparison of the best results obtained by the classical SSOA, ST-SSOA, and STMP-SSOA with those of other methods reported in the literature. Similar to the previous example, the best combination of nEL and nS is considered to be those having better average weight values. It can be seen from the table that both PF-SSOA and STMP-SSOA perform much better than the classical SSOA in terms of best weight, average weight, worst weight, and standard deviation of optimized weights. Table 4.13 lists the first five natural frequencies of the best designs obtained by different optimization algorithms for the 200-bar planar truss. As expected, all of the optimal designs satisfy the frequency constraints. Table 4.14 provides a comparison of the optimized weights found by the classical SSOA, PF-SSOA, and STMP-SSOA at nine different stages of the optimization processes. As can be concluded from the table, during the first stages of the search process, both PF-SSOA and STMP-SSOA has slower convergence rates than the classical SSOA, while during the final stages, PF-SSOA and STMP-SSOA converge faster than the classical SSOA. It can also be seen that STMP-SSOA has a higher convergence rate than PF-SSOA. Figure 4.10 shows the diversity of the optimized weights obtained for the 200-bar planar truss using the classical SSOA, ST-SSOA, and STMP-SSOA over 10 independent runs. The figure confirms that PF-SSOA and STMP-SSOA outperform the classical SSOA in terms of average weight and best weight. Figure 4.11 presents the convergence histories of the average results found by the classical SSOA, PF-SSOA and STMP-SSOA over 10 independent runs.

4.6.3 A 3-Bay 15-Story Steel Frame Structure

The third design example is a 3-bay 15-story steel frame structure shown in Fig. 4.12. The loads applied to the structure and the element grouping numbers are also depicted in the figure. The structure consists of 45 beam elements and 60 column elements. Because of engineering practice demands, the column elements are grouped into 10 different element groups so that the same column section is used for the exterior columns of every three consecutive stories, starting from the foundation. This is also applied for the interior columns of every three consecutive stories. All of the beam elements are also grouped into one beam group. The layout of the structure is kept unchanged during the optimization process. Therefore, this is a sizing optimization problem with 11 design variables (please see Fig. 4.12). The sections of all the column and beam element groups must be selected from all 267 W-shaped sections given by the AISC-LRFD specification [18]. The yield stress of the material is taken as $F_y = 36$ ksi and the modulus of elasticity is taken as $E = 29000$ ksi. The structure is designed subject to strength and serviceability constraints of the AISC-LRFD specification [18], as expressed in Sect. 4.5.2. The maximum lateral displacement of the top story should not exceed 9.25 in [11]. The unbraced length of a beam element is taken to be equal to one-fifth of the span length, and the unbraced length of a column element is equal to the column length. The out-of-plane effective length

Table 4.10 Comparison of optimal results obtained by the classical SSOA and PF-SSOA for the 200-bar planar truss

Design variable: A (cm^2)	Members of the group	SSOA						PF-SSOA					
		$nEL = 16$			$nEL = 24$			$nEL = 24$			$nEL = 25$		
		$nS = 4$	$nS = 4$	$nS = 6$	$nS = 4$	$nS = 6$	$nS = 6$	$nS = 6$	$nS = 6$	$nS = 6$	$nS = 5$	$nS = 5$	$nS = 5$
1	1, 2, 3, 4	0.2883	0.2993	0.2956	0.3053	0.3061	0.3016	0.3114	0.3043				
2	5, 8, 11, 14, 17	0.4647	0.4602	0.4414	0.4588	0.4734	0.4584	0.4422	0.4562				
3	19, 20, 21, 22, 23, 24	0.1001	0.1009	0.1000	0.1003	0.1024	0.1000	0.1011	0.1000				
4	18, 25, 56, 63, 94, 101, 132, 139, 170, 177	0.1003	0.1005	0.1000	0.1009	0.1001	0.1001	0.1009	0.1017				
5	26, 29, 32, 35, 38	0.5200	0.5206	0.5087	0.5329	0.5305	0.5025	0.5003	0.5353				
6	6, 7, 9, 10, 12, 13, 15, 16, 27, 28, 30, 31, 33, 34, 36, 37	0.8247	0.8289	0.8310	0.8143	0.8241	0.8226	0.8158	0.8217				
7	39, 40, 41, 42	0.1008	0.1002	0.1124	0.1000	0.1013	0.1003	0.1022	0.1053				
8	43, 46, 49, 52, 55	1.3902	1.4219	1.4815	1.4212	1.4411	1.4209	1.5016	1.4432				
9	57, 58, 59, 60, 61, 62	0.1042	0.1001	0.1002	0.1000	0.1018	0.1014	0.1003	0.1037				
10	64, 67, 70, 73, 76	1.5943	1.5496	1.5728	1.6224	1.6289	1.5862	1.5337	1.5657				
11	44, 45, 47, 48, 50, 51, 53, 54, 65, 66, 68, 69, 71, 72, 74, 75	1.1436	1.1597	1.1650	1.1664	1.1884	1.1540	1.1579	1.1594				
12	77, 78, 79, 80	0.1350	0.1191	0.1182	0.1269	0.1378	0.1202	0.1212	0.1059				
13	81, 84, 87, 90, 93	2.9892	2.9752	2.9663	2.9684	2.9172	2.9895	3.0054	2.9379				
14	95, 96, 97, 98, 99, 100	0.1010	0.1000	0.1003	0.1257	0.1080	0.1018	0.1226	0.1000				
15	102, 105, 108, 111, 114	3.3066	3.2670	3.2772	3.2173	3.2142	3.2844	3.3105	3.1807				
16	82, 83, 85, 86, 88, 89, 91, 92, 103, 104, 106, 107, 109, 110, 112, 113	1.5686	1.5616	1.5976	1.5610	1.5639	1.5957	1.5749					
17	115, 116, 117, 118	0.3232	0.2581	0.2770	0.2487	0.2933	0.2210	0.1665	0.2376				

(continued)

Table 4.10 (continued)

Design variable: A (cm 2)	Members of the group	SSOA		PF-SSOA		SSOA		PF-SSOA		SSOA		PF-SSOA	
		$nEL = 16$		$nEL = 24$		$nEL = 24$		$nEL = 24$		$nEL = 25$		$nEL = 25$	
		$nS = 4$	$nS = 4$	$nS = 6$	$nS = 6$	$nS = 6$	$nS = 6$	$nS = 6$	$nS = 6$	$nS = 5$	$nS = 5$	$nS = 5$	$nS = 5$
18	119, 122, 125, 128, 131	4.9715	5.1233	5.0935	4.9943	5.0612	5.0891	5.1058	5.1003				
19	133, 134, 135, 136, 137, 138	0.1019	0.1047	0.1003	0.1027	0.1052	0.1000	0.1020	0.1034				
20	140, 143, 146, 149, 152	5.3037	5.4532	5.4563	5.5022	5.5219	5.3135	5.4118	5.4829				
21	120, 121, 123, 124, 126, 127, 129, 130, 141, 142, 144, 145, 147, 148, 150, 151	2.1470	2.1086	2.1093	2.1365	2.0721	2.0764	2.0648	2.1307				
22	153, 154, 155, 156	0.7459	0.6855	0.6153	0.6988	0.7215	0.6469	0.6971	0.5533				
23	157, 160, 163, 166, 169	7.4716	7.6774	7.5803	7.6481	7.4174	7.6932	7.5940	7.5833				
24	171, 172, 173, 174, 175, 176	0.1344	0.1003	0.1486	0.1218	0.2516	0.1582	0.1913	0.1634				
25	178, 181, 184, 187, 190	7.7705	7.9698	7.8331	7.9103	7.7866	7.9229	7.8907	8.2236				
26	158, 159, 161, 162, 164, 165, 167, 168, 179, 180, 182, 183, 185, 186, 188, 189	2.8060	2.7343	2.8526	2.8077	2.9182	2.8018	2.8075	2.7468				
27	191, 192, 193, 194	10.5742	10.5027	10.4871	10.4953	10.1631	10.5570	10.4250	10.5227				
28	195, 197, 198, 200	21.2922	21.3525	21.3057	21.2093	21.2868	21.3845	21.3537	21.3139				
29	196, 199	10.8090	10.7111	10.5743	10.6281	10.8285	10.6574	10.7367	10.7071				
	Weight (kg)	2157.26	2156.76	2157.26	2157.14	2157.99	2156.80	2157.51	2157.46				
	Number of FE analyses	27,856	29,184	29,928	29,160	29,712	27,816	29,850	28,375				
	Average weight (kg)	2158.59	2158.06	2158.64	2157.76	2159.05	2157.53	2159.02	2157.98				
	Worst weight (kg)	2160.75	2160.67	2162.57	2158.64	2160.81	2158.49	2161.63	2158.57				

(continued)

Table 4.10 (continued)

Table 4.11 First five natural frequencies (Hz) of the 200-bar planar truss evaluated at optimal designs found by the classical SSOA and PF-SSOA

Frequency number	SSOA	PF-SSOA	SSOA	PF-SSOA	SSOA	PF-SSOA	SSOA	PF-SSOA
	$nEL = 16$	$nEL = 24$	$nEL = 24$	$nEL = 25$	$nS = 4$	$nS = 6$	$nS = 5$	
1	5.0000	5.0000	5.0001	5.0000	5.0000	5.0000	5.0000	5.0001
2	12.1133	12.2045	12.1523	12.2780	12.3178	12.2006	12.2735	12.2651
3	15.0511	15.0413	15.0346	15.0895	15.1677	15.0533	15.0498	15.0887
4	16.7149	16.6950	16.6975	16.7134	16.7613	16.6684	16.6844	16.6686
5	21.2368	21.3990	21.3414	21.4958	21.5864	21.4189	21.4386	21.4979

factor is specified as $K_y = 1$. For a sway-permitted frame structure, the effective length factors of the members are specified as $K_x \geq 1$ and can be calculated as follows [22]:

$$K_x = \sqrt{\frac{1.6G_A G_B + 4(G_A + G_B) + 7.5}{G_A + G_B + 7.5}} \quad (4.30)$$

where G_B and G_A are stiffness ratios of girders and columns at end joints, B and A , of the column section, respectively. This problem has been previously studied by many researchers using different optimization algorithms: Kaveh and Bakhshpoori using CS [10], Kaveh and Ilchi Ghazaan using CBO and ECBO [11], Kaveh and Bakhshpoori using AWEO [12], etc.

Tables 4.15 and 4.16 provide the results obtained for the 3-bay 15-story frame using the classical SSOA and PF-SSOA for different combinations of nEL and nS , respectively. The results in the tables show that PF-SSOA performs better than the classical SSOA in terms of best weight and average weight for all combinations of nEL and nS . It can also be seen that in terms of average weight, the best combination of nEL and nS for both the classical SSOA and PF-SSOA is $nEL = 24$ and $nS = 6$. Table 4.17 present the best results obtained by the classical SSOA, PF-SSOA, STMP-SSOA, and other optimization methods in the literature. The best combination of nEL and nS is considered to be the one which has better performance in terms of best weight. The results of the table show that STMP-SSOA achieved the best results in terms of average weight, worst weight, and standard deviation of optimized weights. The best weight obtained by both PF-SSOA and STMP-SSOA is 87469 lb, while it is 88366 lb for the classical SSOA. Table 4.18 presents the optimized weights achieved by the classical SSOA, PF-SSOA, and STMP-SSOA at six different stages of the optimization processes. As the table demonstrates, during the early stages of the search process, the classical SSOA exhibits a higher convergence rate than both PF-SSOA and STMP-SSOA, whereas as the search process progresses, PF-SSOA and STMP-SSOA perform better than the classical SSOA in terms of convergence

Table 4.12 Comparison of optimal results for the 200-bar planar truss

Design variable: A (cm^2)	ECBO [19]	UECBO [20]	OST-TLBO [2]	STMP-TLBO [2]	This study	
					SSOA	PF-SSOA
					$nEL = 16$ $nS = 4$	$nEL = 24$ $nS = 6$
1	0.2993	0.3002	0.3243	0.3020	0.2883	0.3016
2	0.4497	0.4890	0.4247	0.4480	0.4647	0.4584
3	0.1000	0.1000	0.1008	0.1000	0.1001	0.1000
4	0.1000	0.1000	0.1030	0.1003	0.1003	0.1001
5	0.5137	0.5277	0.5306	0.5183	0.5200	0.5025
6	0.7914	0.8310	0.8395	0.8055	0.8247	0.8226
7	0.1013	0.1001	0.1113	0.1006	0.1008	0.1003
8	1.4129	1.3841	1.4185	1.4328	1.3992	1.4209
9	0.1019	0.1000	0.1023	0.1003	0.1042	0.1014
10	1.6460	1.5912	1.5553	1.5969	1.5943	1.5862
11	1.1532	1.1502	1.1519	1.1505	1.1436	1.1540
12	0.1000	0.1008	0.1433	0.1668	0.1350	0.1202
13	3.1850	3.0023	2.9539	2.9503	2.9892	2.9895
14	0.1034	0.1007	0.1054	0.1049	0.1010	0.1018
15	3.3126	3.2767	3.2897	3.3305	3.3066	3.2844
16	1.5920	1.6017	1.5890	1.5777	1.5686	1.5639
17	0.2238	0.2309	0.2359	0.2661	0.3232	0.2210
18	5.1227	5.0228	4.9473	5.1869	4.9715	5.0891
19	0.1050	0.1057	0.1195	0.1014	0.1000	0.1038

(continued)

Table 4.12 (continued)

Design variable: A (cm^2)	ECBO [19]	UECBO [20]	OST-TLBO [2]	STMP-TLBO [2]	This study		
					SSOA	PF-SSOA	STMP-SSOA
					$nEL = 16$ $nS = 4$	$nEL = 24$ $nS = 6$	$nEL = 20$ $nS = 5, 4$
20	5.3707	5.2667	5.4264	5.4297	5.3037	5.3135	5.4548
21	2.0645	2.1287	2.0412	2.0687	2.1470	2.0764	2.1055
22	0.5443	0.7337	0.7127	0.7319	0.7459	0.6469	0.6100
23	7.6497	7.9257	7.5703	7.4680	7.4716	7.6932	7.7853
24	0.1000	0.1000	0.1392	0.1227	0.1344	0.1582	0.1368
25	7.6754	8.1735	7.8170	7.7060	7.7705	7.9229	8.0602
26	2.7178	2.7758	2.8329	2.8544	2.8060	2.8018	2.7441
27	10.8141	10.1047	10.5864	10.0465	10.5742	10.5570	10.5319
28	21.6349	21.2172	21.4892	21.4945	21.2922	21.3845	21.3620
29	10.3520	10.9900	10.4272	10.9668	10.8090	10.6574	10.6053
Weight (kg)	2158.08	2157.65	2157.91	2157.714	2157.26	2156.80	2156.97
Number of FE analyses	Approximately 14,700	N/A	N/A	N/A	27,856	27,816	28,960
Average weight (kg)	2159.93	2161.36	2162.041	2160.366	2158.59	2157.53	2157.50
Worst weight (kg)	N/A	N/A	2167.288	2164.578	2160.75	2158.49	2158.41
Standard deviation (kg)	1.57	2.72	2.586	1.806	1.108	0.545	0.400
Maximum number of FEA analyses	20,000	20,000	30,000	30,000	30,000	30,000	30,000
Number of runs	20	20	20	20	10	10	10

Table 4.13 First five natural frequencies (Hz) of the 200-bar planar truss evaluated at optimal designs found by different algorithms

Frequency number	ECBO [19]	UECBO [20]	OST-TLBO [2]	STMP-TLBO [2]	This study		
					SSOA	PF-SSOA	STMP-SSOA
					$nEL = 16$	$nEL = 24$	$nEL = 20$
					$nS = 4$	$nS = 6$	$nS = 5, 4$
1	5.000	5.000	5.0001	5.0000	5.0000	5.0000	5.0000
2	12.189	12.260	12.3891	12.2160	12.1133	12.2006	12.2283
3	15.048	15.096	15.0947	15.0639	15.0511	15.0533	15.0509
4	16.643	16.696	16.7304	16.7141	16.7149	16.6684	16.7068
5	21.342	21.452	21.5343	21.4112	21.2368	21.4189	21.4306

Table 4.14 Optimized weights for the 200-bar planar truss at nine different stages of the optimization process (best runs)

Number of structural analyses	Optimized weight (kg)		
	SSOA	PF-SSOA	STMP-SSOA
	$nEL = 16$	$nEL = 24$	$nEL = 20$
3000	2844.107	3482.931	2942.435
6000	2211.497	2231.742	2220.316
9000	2173.814	2170.192	2170.807
12,000	2166.010	2161.936	2159.323
15,000	2162.166	2159.461	2157.852
18,000	2159.203	2158.444	2157.374
21,000	2158.163	2157.383	2157.270
24,000	2157.690	2156.896	2157.195
27,000	2157.486	2156.812	2157.137

characteristics. The results also indicate that the exploitation capability of both PF-SSOA and STMP-SSOA is significantly better than the classical SSOA. Figure 4.13 illustrates the final weights obtained by the classical SSOA, PF-SSOA, and STMP-SSOA for the 3-bay 15-story frame over 10 independent runs. As is evident from the figure, both PF-SSOA and STMP-SSOA have gained better results compared to the classical SSOA in terms of the average weight and best weight. The convergence histories of the average results obtained by the classical SSOA, PF-SSOA, and STMP-SSOA are shown in Fig. 4.14. Figure 4.15 shows the inter-story drifts of the best designs of the classical SSOA, PF-SSOA, and STMP-SSOA. The stress ratios of the best designs of the mentioned algorithms are also shown in Fig. 4.16. It is clear from the figure that the stress constraints control the design of the 3-bay 15-story steel frame.

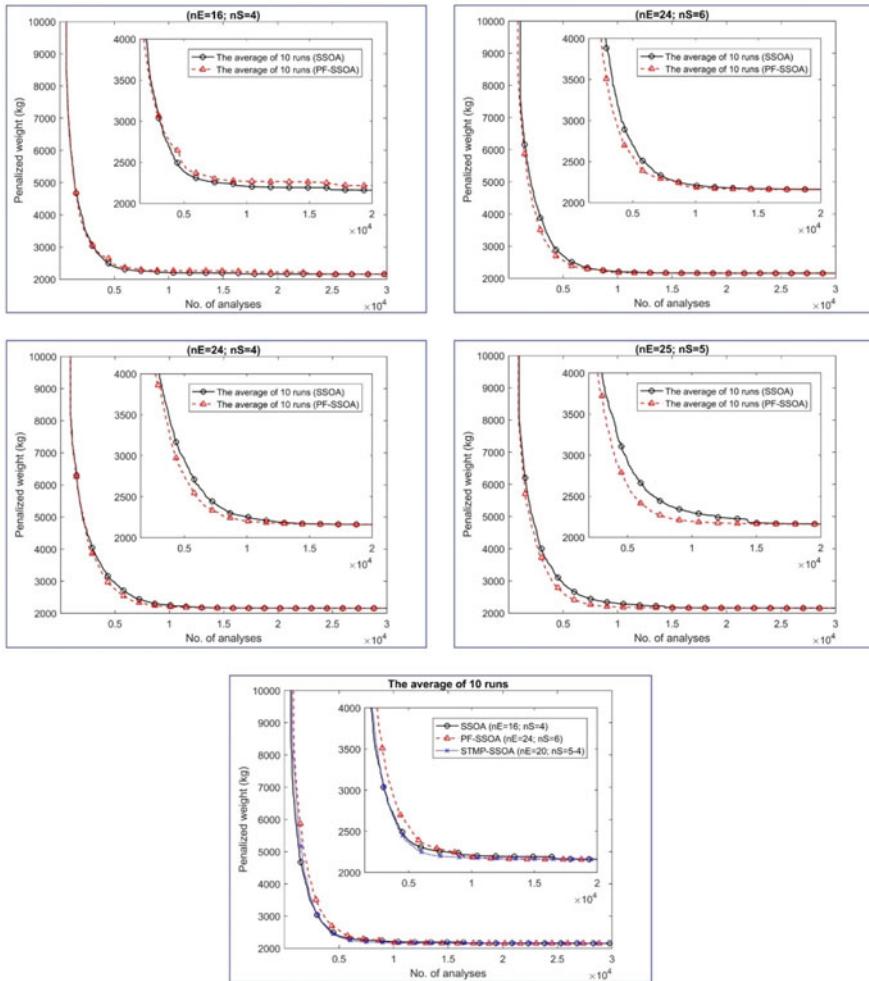


Fig. 4.11 Average weight convergence histories for the 200-bar planar truss obtained by the classical SSOA, PF-SSOA, and STMP-SSOA

4.6.4 A 3-Bay 24-Story Steel Frame Structure

The last design example is a 3-bay 24-story steel frame structure shown in Fig. 4.17. The element grouping numbers and the loads applied to the structure are also shown in the figure. The structure is composed of 72 beam elements and 96 column elements. The column elements are divided into 16 different element groups. Starting from the foundation, the exterior columns of every three consecutive stories are combined in an element group. The same grouping strategy is also applied to the interior columns of every three consecutive stories. The beam elements are also grouped into four

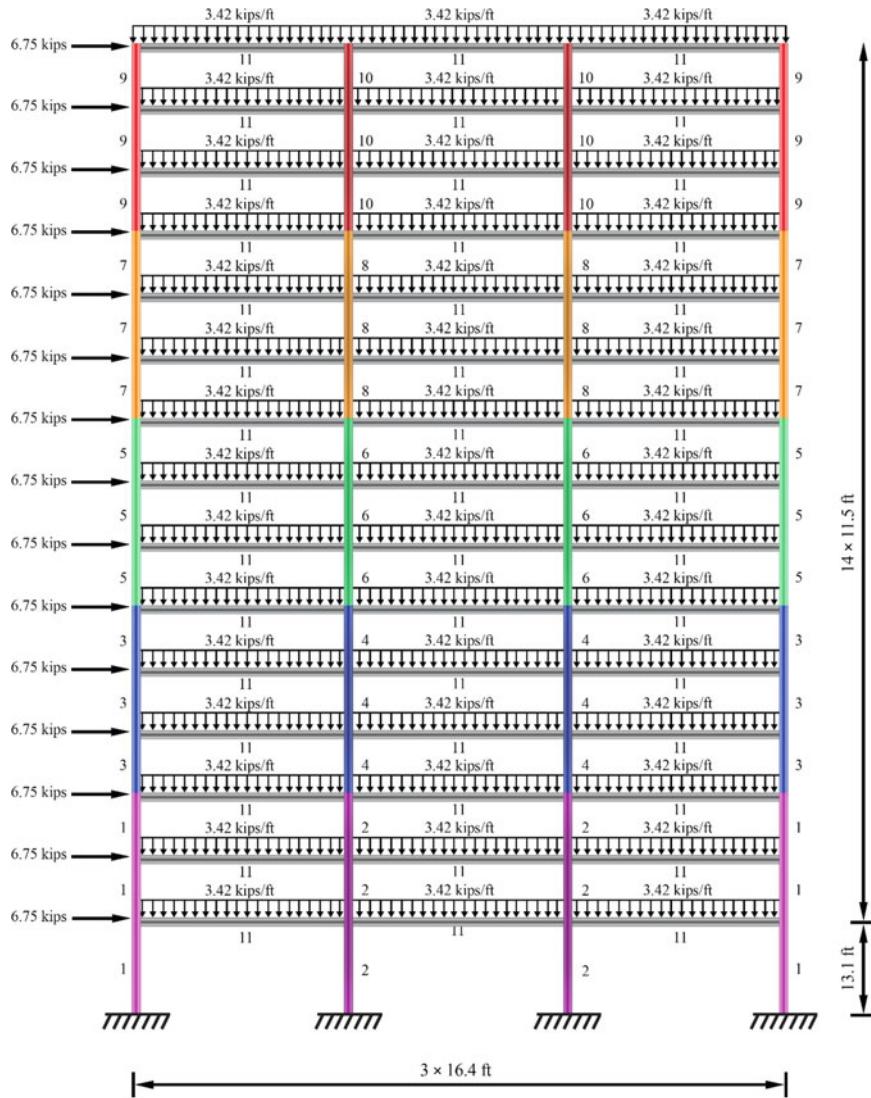


Fig. 4.12 Schematic of the 3-bay 15-story steel frame structure

different element groups. For this purpose, the beams of the exterior bays of all stories except the top one form a beam group. The beams of the interior bay of all stories except the top one form another beam group. Another beam group includes the beams of the exterior bays of the top story, and the last beam group includes the beam of the interior bay of the top story. It is noted that the layout of the structure is predefined and kept fixed during the optimization process. Therefore, this is a sizing optimization problem with 20 design variables. The sections of beam element

Table 4.15 Comparison of optimal results obtained by the classical SSOA for the 3-bay 15-story steel frame structure

Group number	$nEL = 16$ $nS = 4$	$nEL = 20$ $nS = 4$	$nEL = 24$ $nS = 4$	$nEL = 24$ $nS = 6$	$nEL = 25$ $nS = 5$	$nEL = 30$ $nS = 5$
1	W14 × 90	W16 × 89	W24 × 104	W21 × 111	W12 × 106	W14 × 99
2	W36 × 170	W36 × 170	W36 × 160	W36 × 160	W27 × 161	W24 × 162
3	W27 × 84					
4	W27 × 114	W24 × 104	W24 × 104	W24 × 104	W27 × 114	W27 × 114
5	W24 × 68	W21 × 68	W21 × 68	W21 × 68	W24 × 68	W24 × 68
6	W18 × 86	W14 × 90	W30 × 90	W30 × 90	W12 × 87	W24 × 94
7	W21 × 48	W12 × 45	W14 × 53	W18 × 50	W21 × 48	W12 × 50
8	W14 × 68	W21 × 68	W21 × 68	W21 × 68	W24 × 68	W21 × 68
9	W8 × 31	W16 × 40	W10 × 33	W14 × 34	W10 × 33	W14 × 34
10	W10 × 39	W16 × 40	W10 × 39	W10 × 39	W10 × 39	W16 × 40
11	W21 × 44					
Weight (lb)	88,366	88,435	88,724	89,163	89,078	89,404
Number of FE analyses	18,368	19,960	19,440	18,504	19,350	20,000
Average weight (lb)	91,660	91,417	90,916	90,448	90,856	90,989
Worst weight (lb)	96,474	96,405	94,071	93,017	93,431	95,373
Standard deviation (lb)	2813	3021	1841	1356	1726	1603
Maximum number of FE analyses	20,000	20,000	20,000	20,000	20,000	20,000
Number of runs	10	10	10	10	10	10

groups are assumed to be selected from all 267 W-shaped sections given by the AISC-LRFD specification [18]. The column element groups are limited to W14 sections. All beam and column elements are considered as non-braced along their lengths. The effective length factors of the members are calculated same as in Sect. 4.6.3 using Eq. (4.30). The elasticity modulus of the material is $E = 29732$ ksi and the yield stress is $F_y = 33.4$ ksi. The maximum lateral displacement of the top story should not exceed 11.52 in. The maximum inter-story drift ratio is limited to 1/300. This problem has been studied by several researchers using different optimization algorithms: Camp et al. [8] using ant colony optimization (ACO), Degertekin [9] using the HS algorithm, Kaveh and Bakhshpoori using CS [10], Kaveh and Ilchi Ghazaan using CBO and ECBO [11], Kaveh and Bakhshpoori using AWEO [12], etc.

Table 4.16 Comparison of optimal results obtained by PF-SSOA for the 3-bay 15-story steel frame structure

Group number	$nEL = 16$ $nS = 4$	$nEL = 20$ $nS = 4$	$nEL = 24$ $nS = 4$	$nEL = 24$ $nS = 6$	$nEL = 25$ $nS = 5$	$nEL = 30$ $nS = 5$
1	W14 × 99	W14 × 99	W14 × 99	W12 × 106	W14 × 99	W12 × 96
2	W27 × 161	W36 × 170				
3	W12 × 79	W27 × 84	W27 × 84	W27 × 84	W12 × 79	W27 × 84
4	W27 × 114	W24 × 104	W30 × 116	W24 × 104	W30 × 116	W27 × 114
5	W24 × 68	W16 × 67	W12 × 65	W24 × 68	W14 × 68	W24 × 68
6	W14 × 90	W18 × 86	W30 × 90	W18 × 86	W14 × 90	W12 × 87
7	W14 × 34	W8 × 48	W10 × 49	W8 × 48	W18 × 50	W12 × 53
8	W24 × 68	W12 × 65	W14 × 68	W12 × 65	W24 × 68	W12 × 65
9	W8 × 28	W8 × 31	W14 × 34	W8 × 28	W14 × 34	W12 × 30
10	W10 × 39	W10 × 39	W18 × 35	W10 × 39	W10 × 39	W16 × 40
11	W21 × 44					
Weight (lb)	87,745	87,469	88,573	87,767	88,780	89,006
Number of FE analyses	17,504	17,520	16,272	19,320	19,700	17,850
Average weight (lb)	89,449	88,832	90,229	88,666	89,505	89,965
Worst weight (lb)	95,728	90,835	96,336	90,320	92,438	90,919
Standard deviation (lb)	2359	1081	2330	669	1006	531
Maximum number of FE analyses	20,000	20,000	20,000	20,000	20,000	20,000
Number of runs	10	10	10	10	10	10

Table 4.19 provides a comparison of the results obtained by the classical SSOA and PF-SSOA for different combinations of nEL and nS . It is evident from the table that the performance of PF-SSOA is significantly better than the classical SSOA, especially in terms of best weight, average weight, and standard deviation of optimized weights. Table 4.20 presents the best results of the classical SSOA, PF-SSOA, STMP-SSOA, and other optimization algorithm reported in the literature. Similar to the previous example, the best combinations of nEL and nS is considered to be the one which has better performance in terms of best weight. As the table demonstrates, STMP-SSOA and PF-SSOA have better performance than the classical SSOA in terms of average weight (202,151 lb for PF-SSOA, 201,776 lb for STMP-SSOA, and 205,551 lb for the classical SSOA). Moreover, the best weight obtained by both PF-SSOA and STMP-SSOA is 201186 lb, which is lighter than that of the classical SSOA (202,338 lb) and other methods in the literature. Table 4.21 provides

Table 4.17 Comparison of optimal results obtained for the 3-bay 15-story steel frame structure

Group number	CS [10]	ECBO [11]	AWEO [12]	This study		
				SSOA	PF-SSOA	STMP-SSOA
				$nEL = 16$	$nEL = 20$	$nEL = 20$
				$nS = 4$	$nS = 4$	$nS = 5, 4$
1	W14 × 99	W14 × 99	W14 × 99	W14 × 90	W14 × 99	W16 × 89
2	W27 × 161	W27 × 161	W27 × 161	W36 × 170	W27 × 161	W36 × 170
3	W14 × 82	W27 × 84	W27 × 84	W27 × 84	W27 × 84	W12 × 79
4	W24 × 104	W24 × 104	W24 × 104	W27 × 114	W24 × 104	W27 × 114
5	W12 × 65	W14 × 61	W14 × 61	W24 × 68	W16 × 67	W24 × 68
6	W18 × 86	W30 × 90	W30 × 90	W18 × 86	W18 × 86	W18 × 86
7	W18 × 50	W14 × 48	W16 × 50	W21 × 48	W8 × 48	W14 × 48
8	W14 × 61	W14 × 61	W21 × 68	W14 × 68	W12 × 65	W14 × 61
9	W8 × 24	W14 × 30	W14 × 34	W8 × 31	W8 × 31	W12 × 30
10	W12 × 40	W12 × 40	W8 × 35	W10 × 39	W10 × 39	W10 × 39
11	W21 × 44	W21 × 44	W21 × 44	W21 × 44	W21 × 44	W21 × 44
Weight (lb)	86,809	86,986	87,537.96	88,366	87,469	87,469
Number of FE analyses	16,170	9000	10,670	18,368	17,520	19,120
Average weight (lb)	87,784	88,410	88,893.09	91,660	88,832	88,328
Worst weight (lb)	N/A	N/A	N/A	96,474	90,835	90,825
Standard deviation (lb)	942	N/A	N/A	2813	1081	906
Maximum number of FE analyses	19,600	20,000	14,000	20,000	20,000	20,000
Number of runs	50	20	20	10	10	10

Table 4.18 Optimized weights for the 3-bay 15-story steel frame structure at six different stages of the optimization process (best runs)

Number of structural analyses	Optimized weight (lb)		
	SSOA	PF-SSOA	STMP-SSOA
	$nEL = 16$	$nEL = 20$	$nEL = 20$
3000	101,373	108,608	101,811
6000	94,232	95,563	97,215
9000	90,885	91,479	92,617
12,000	89,433	89,988	90,126
15,000	88,881	88,849	88,504
18,000	88,711	87,469	87,952

the optimized weights obtained by the classical SSOA, PF-SSOA, and STMP-SSOA at six different stages of the optimization process. It can be seen from the table that STMP-SSOA and PF-SSOA have significantly higher convergence rates than the classical SSOA. The final weights obtained by the classical SSOA, PF-SSOA, and STMP-SSOA for the 3-bay 24-story steel frame over 10 independent runs are depicted in Fig. 4.18. The figure confirms the robustness of STMP-SSOA compared to the classical SSOA and PF-SSOA. Figure 4.19 shows the convergence histories of the average results obtained by the classical SSOA, PF-SSOA, and STMP-SSOA for the 3-bay 24-story steel frame. Figure 4.20 shows the inter-story drifts of the best designs of the classical SSOA, PF-SSOA, and STMP-SSOA. It can be observed that all of the inter-story drift constraints are satisfied, as expected. Figure 4.21 shows the stress ratios evaluated at the best designs obtained by the classical SSOA, PF-SSOA, and STMP-SSOA for the 3-bay 24-story frame. It can be concluded from Figs. 4.20 and 4.21 that the design of the 3-bay 24-story steel frame is controlled by the displacement constraints.

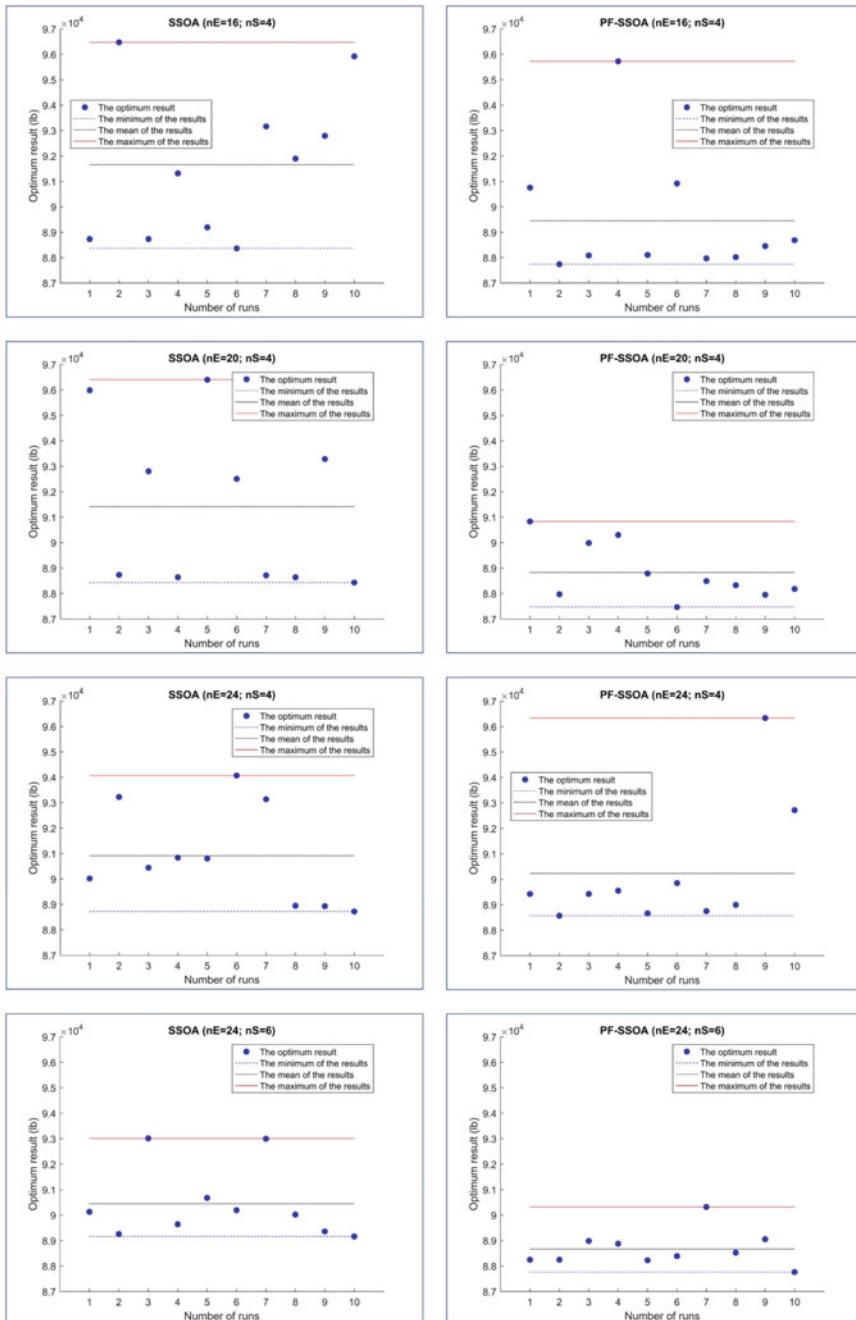
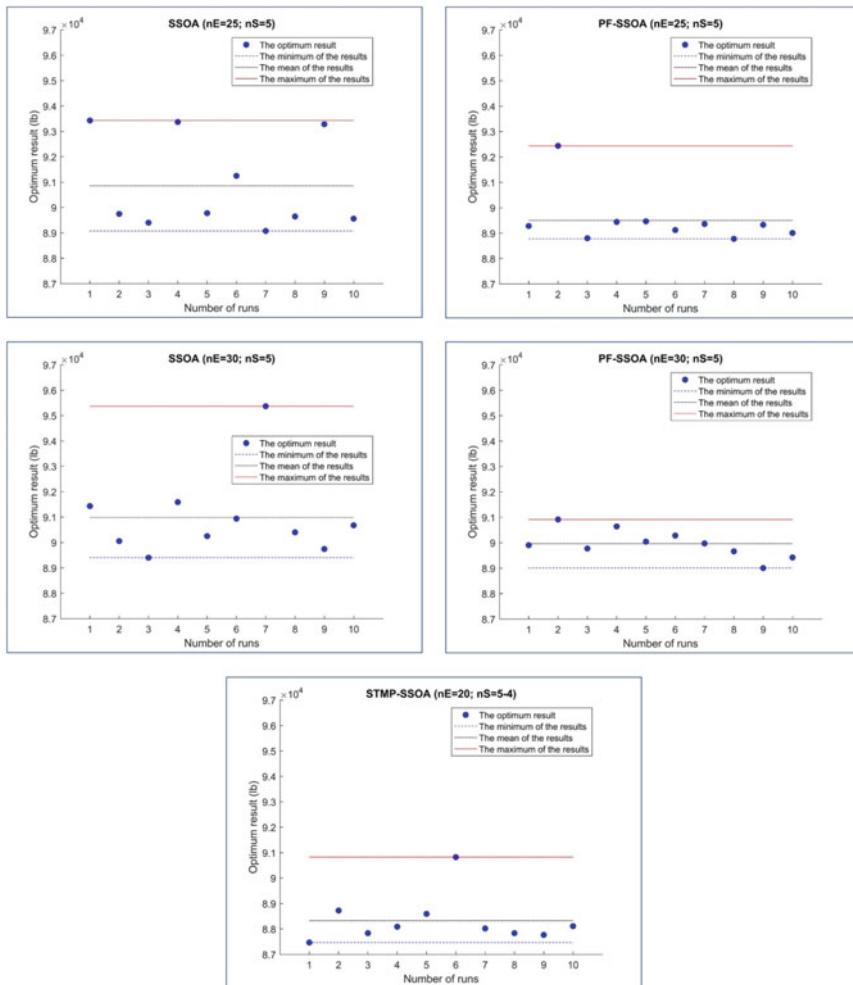


Fig. 4.13 a Optimized weights obtained by the classical SSOA and PF-SSOA for the 3-bay 15-story steel frame structure over 10 independent runs ($nEL = 16, 20, 24$). b Optimized weights obtained by the classical SSOA, PF-SSOA, and STMP-SSOA for the 3-bay 15-story steel frame structure over 10 independent runs ($nEL = 20, 25, 30$)

**Fig. 4.13** (continued)

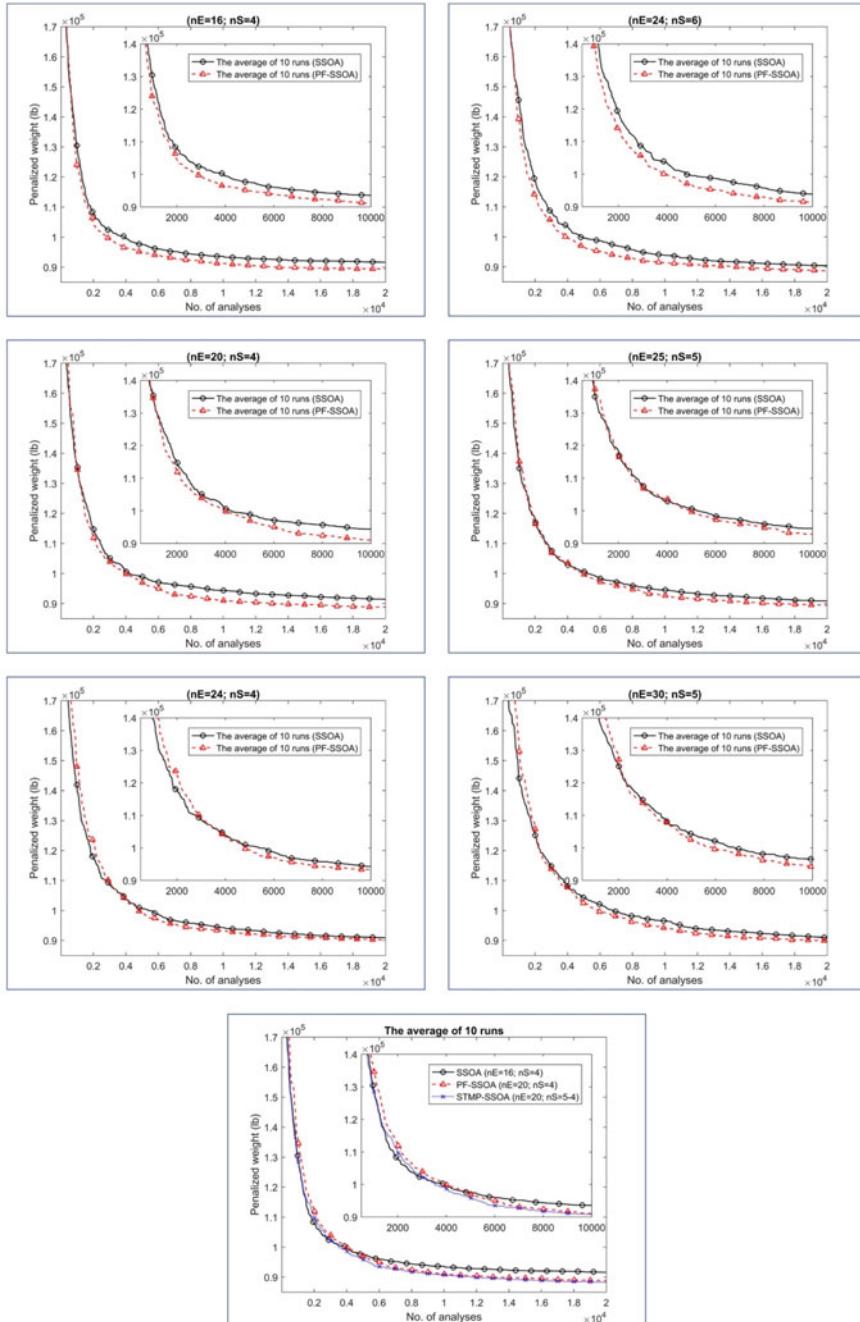


Fig. 4.14 Average weight convergence histories for the 3-bay 15-story steel frame structure obtained by the classical SSOA, PF-SSOA, and STMP-SSOA

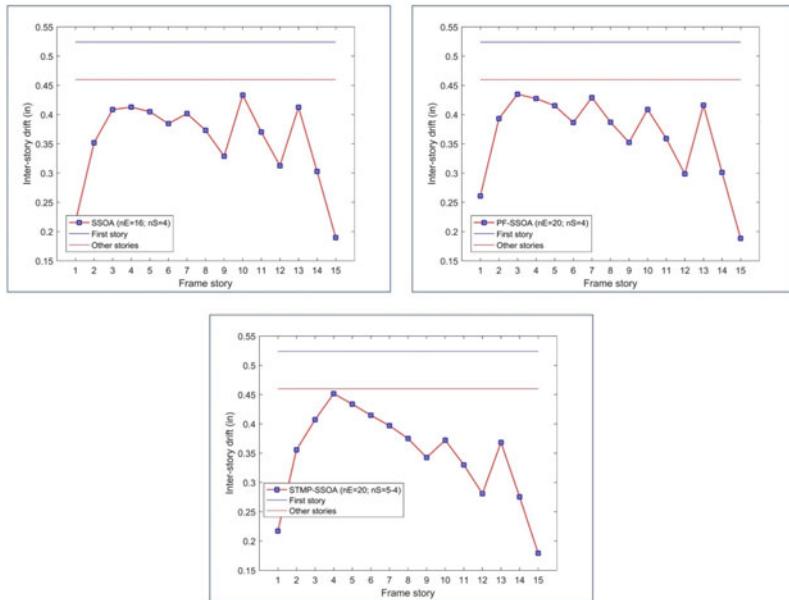


Fig. 4.15 Comparison of the inter-story drifts of the 3-bay 15-story steel frame structure (best runs)

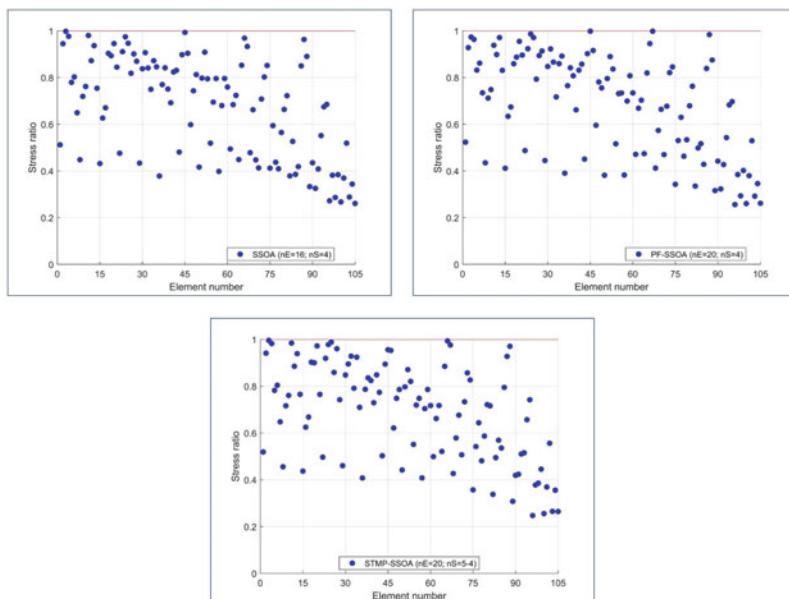


Fig. 4.16 Comparison of the stress ratios of the 3-bay 15-story steel frame structure (best runs)

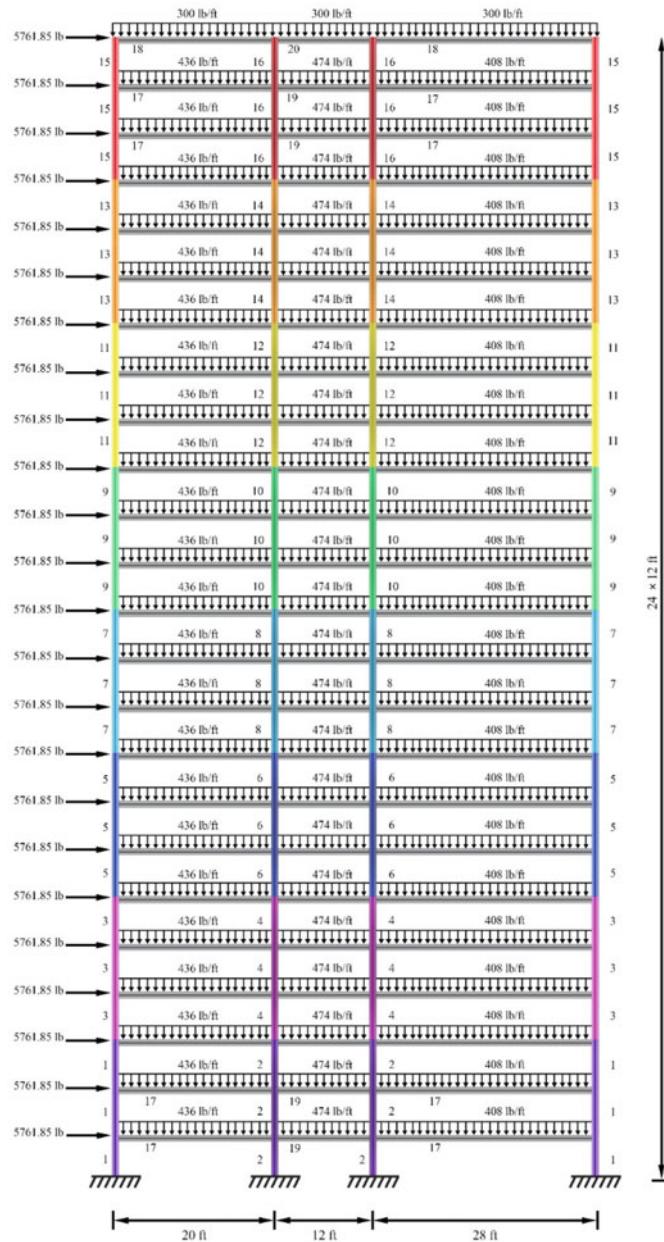


Fig. 4.17 Schematic of the 3-bay 24-story steel frame structure

Table 4.19 Comparison of optimal results obtained by the classical SSOA and PF-SSOA for the 3-bay 24-story steel frame structure

Group number	SSOA	PF-SSOA	SSOA	PF-SSOA	SSOA	PF-SSOA
	$nEL = 24$ $nS = 4$		$nEL = 24$ $nS = 6$		$nEL = 25$ $nS = 5$	
1	W14 × 159	W14 × 159	W14 × 145	W14 × 159	W14 × 132	W14 × 159
2	W14 × 99	W14 × 132	W14 × 109	W14 × 132	W14 × 109	W14 × 109
3	W14 × 120	W14 × 109	W14 × 99	W14 × 109	W14 × 109	W14 × 99
4	W14 × 82	W14 × 74	W14 × 99	W14 × 68	W14 × 99	W14 × 82
5	W14 × 48	W14 × 68	W14 × 68	W14 × 61	W14 × 68	W14 × 68
6	W14 × 53	W14 × 38	W14 × 38	W14 × 43	W14 × 61	W14 × 53
7	W14 × 34	W14 × 34	W14 × 38	W14 × 38	W14 × 30	W14 × 34
8	W14 × 22	W14 × 22	W14 × 22	W14 × 22	W14 × 22	W14 × 22
9	W14 × 90	W14 × 90	W14 × 99	W14 × 90	W14 × 99	W14 × 90
10	W14 × 120	W14 × 99	W14 × 109	W14 × 99	W14 × 120	W14 × 109
11	W14 × 90	W14 × 90	W14 × 109	W14 × 90	W14 × 99	W14 × 99
12	W14 × 90	W14 × 90	W14 × 82	W14 × 99	W14 × 82	W14 × 90
13	W14 × 90	W14 × 68	W14 × 74	W14 × 74	W14 × 74	W14 × 74
14	W14 × 53	W14 × 61	W14 × 68	W14 × 61	W14 × 48	W14 × 53
15	W14 × 34	W14 × 34	W14 × 30	W14 × 30	W14 × 38	W14 × 34
16	W14 × 22	W14 × 22	W14 × 22	W14 × 22	W14 × 22	W14 × 22
17	W30 × 90	W30 × 90	W30 × 90	W30 × 90	W30 × 90	W30 × 90
18	W8 × 18	W8 × 18	W6 × 15	W8 × 18	W6 × 15	W6 × 15
19	W24 × 55	W24 × 55	W24 × 55	W24 × 55	W24 × 55	W24 × 55
20	W6 × 8.5	W6 × 8.5	W6 × 8.5	W6 × 8.5	W6 × 8.5	W6 × 8.5
Weight (lb)	202,338	201,186	202,554	201,546	202,626	201,546
Number of FE analyses	15,984	17,304	19,392	10,824	19,975	13,875
Average weight (lb)	205,551	202,151	203,865	202,201	204,997	202,356
Worst weight (lb)	212,568	203,244	205,710	204,066	213,864	203,778
Standard deviation (lb)	2982	685	1021	684	3088	698
Maximum number of FE analyses	20,000	20,000	20,000	20,000	20,000	20,000
Number of runs	10	10	10	10	10	10

Table 4.20 Comparison of optimal results obtained for the 3-bay 24-story steel frame structure

Group number	CS [10]	ECBO [11]	AWEO [12]	This study		
				SSOA	PF-SSOA	STMP-SSOA
				$nEL = 24$	$nEL = 24$	$nEL = 24$
				$nS = 4$	$nS = 4$	$nS = 6, 4, 3$
1	W14 × 159	W14 × 145	W14 × 159	W14 × 159	W14 × 159	W14 × 159
2	W14 × 132	W14 × 132	W14 × 132	W14 × 99	W14 × 132	W14 × 132
3	W14 × 99	W14 × 99	W14 × 99	W14 × 120	W14 × 109	W14 × 109
4	W14 × 74	W14 × 90	W14 × 109	W14 × 82	W14 × 74	W14 × 74
5	W14 × 61	W14 × 74	W14 × 68	W14 × 48	W14 × 68	W14 × 68
6	W14 × 53	W14 × 38	W14 × 38	W14 × 53	W14 × 38	W14 × 38
7	W14 × 34	W14 × 38	W14 × 30	W14 × 34	W14 × 34	W14 × 34
8	W14 × 22	W14 × 22	W14 × 22	W14 × 22	W14 × 22	W14 × 22
9	W14 × 90	W14 × 99	W14 × 90	W14 × 90	W14 × 90	W14 × 90
10	W14 × 99	W14 × 99	W14 × 99	W14 × 120	W14 × 99	W14 × 99
11	W14 × 99	W14 × 99	W14 × 99	W14 × 90	W14 × 90	W14 × 90
12	W14 × 90	W14 × 82	W14 × 74	W14 × 90	W14 × 90	W14 × 90
13	W14 × 74	W14 × 68	W14 × 68	W14 × 90	W14 × 68	W14 × 68
14	W14 × 53	W14 × 61	W14 × 61	W14 × 53	W14 × 61	W14 × 61
15	W14 × 34	W14 × 30	W14 × 34	W14 × 34	W14 × 34	W14 × 34
16	W14 × 22	W14 × 22	W14 × 22	W14 × 22	W14 × 22	W14 × 22
17	W30 × 90	W30 × 90	W30 × 90	W30 × 90	W30 × 90	W30 × 90
18	W6 × 15	W6 × 15	W8 × 18	W8 × 18	W8 × 18	W8 × 18
19	W24 × 55	W24 × 55	W24 × 55	W24 × 55	W24 × 55	W24 × 55

(continued)

Table 4.20 (continued)

Group number	CS [10]	ECBO [11]	AWEO [12]	This study		
				SSOA	PF-SSOA	STMP-SSOA
				$nEL = 24$	$nEL = 24$	$nEL = 24$
				$nS = 4$	$nS = 4$	$nS = 6, 4, 3$
20	W6 × 8.5	W6 × 8.5	W6 × 8.5	W6 × 8.5	W6 × 8.5	W6 × 8.5
Weight (lb)	201,451	201,618	202,194.02	202,338	201,186	201,186
Number of FE analyses	15,918	15,360	11,300	15,984	17,304	15,528
Average weight (lb)	205,096	209,644	203,412.88	205,551	202,151	201,776
Worst weight (lb)	N/A	N/A	N/A	212,568	203,244	202,620
Standard deviation (lb)	4533	N/A	N/A	2982	685	393
Maximum number of FE analyses	18,000	20,000	14,000	20,000	20,000	20,000
Number of runs	20	20	20	10	10	10

Table 4.21 Optimized weights for the 3-bay 24-story steel frame structure at six different stages of the optimization process (best runs)

Number of structural analyses	Optimized weight (lb)		
	SSOA	PF-SSOA	STMP-SSOA
	$nEL = 24$	$nEL = 24$	$nEL = 24$
3000	251,040	263,136	243,720
6000	230,400	219,636	213,240
9000	212,628	208,412	204,624
12,000	205,476	204,612	202,632
15,000	202,620	203,256	201,253
18,000	202,338	201,186	201,186

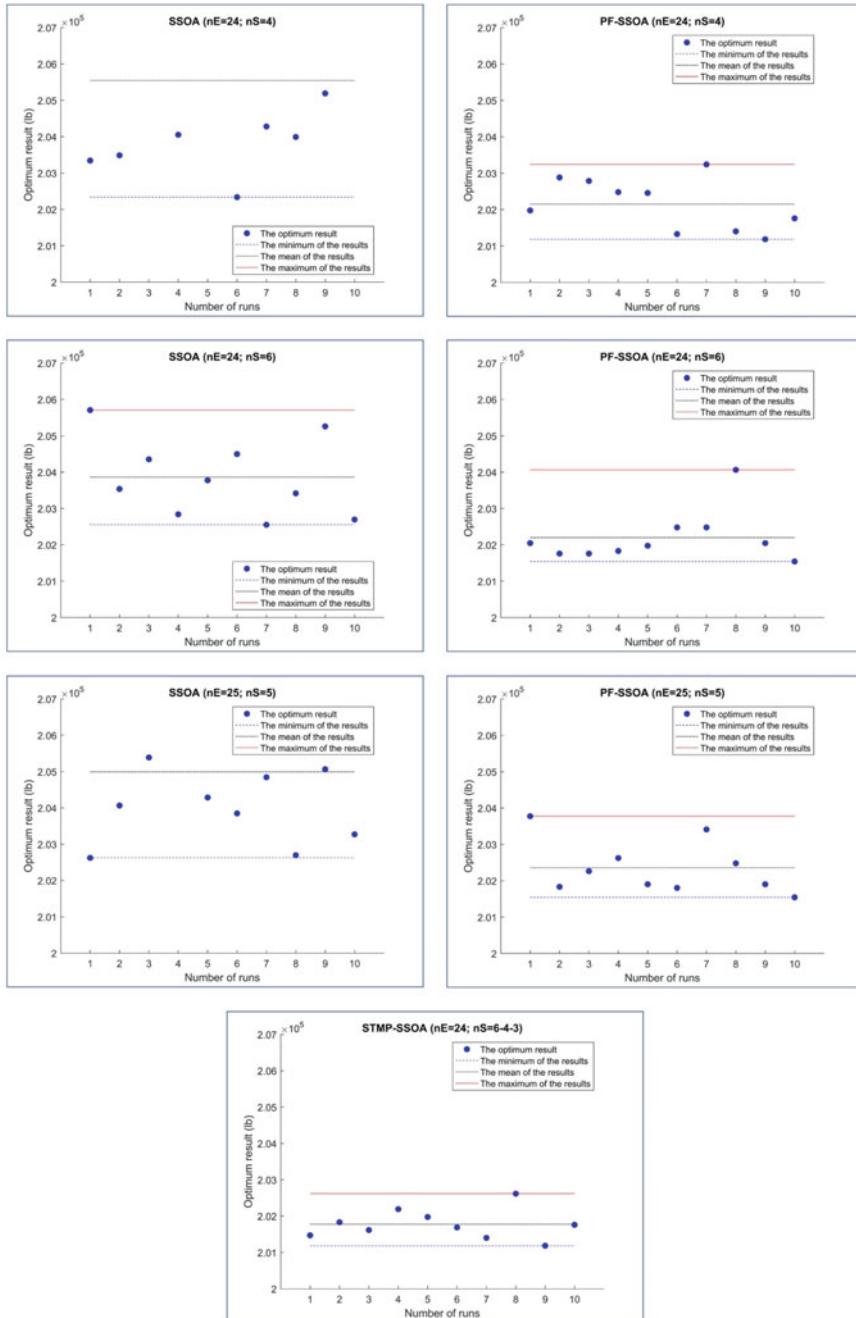


Fig. 4.18 Optimized weights obtained by the classical SSOA, PF-SSOA, and STMP-SSOA for the 3-bay 24-story steel frame structure over 10 independent runs

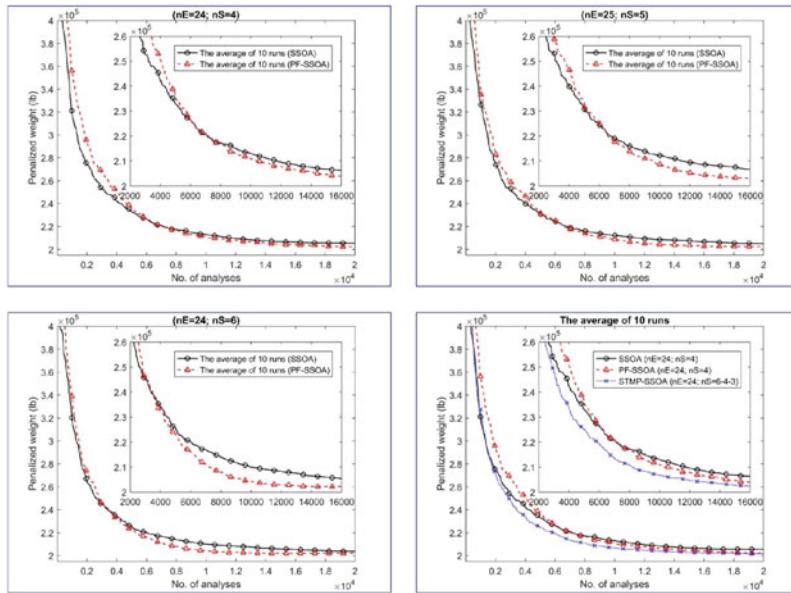


Fig. 4.19 Average weight convergence histories for the 3-bay 24-story steel frame structure obtained by the classical SSOA, PF-SSOA, and STMP-SSOA

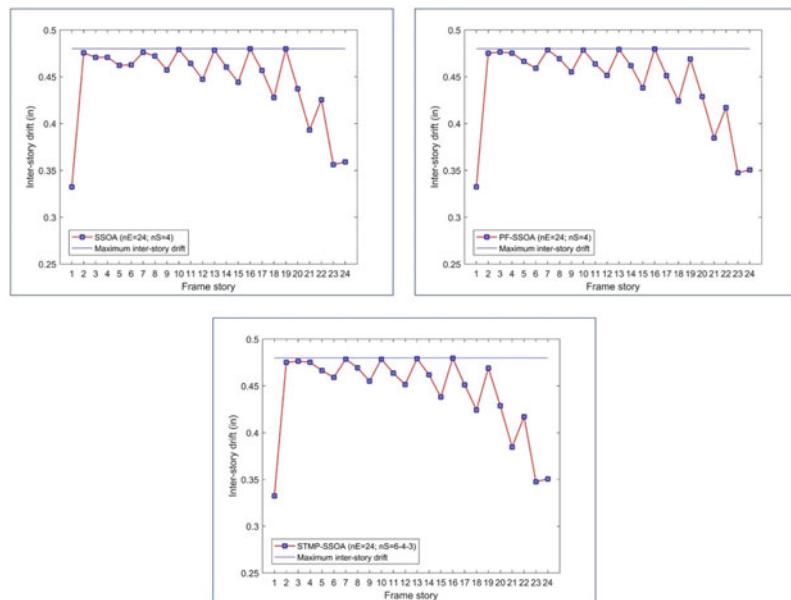


Fig. 4.20 Comparison of the inter-story drifts of the 3-bay 24-story steel frame structure (best runs)

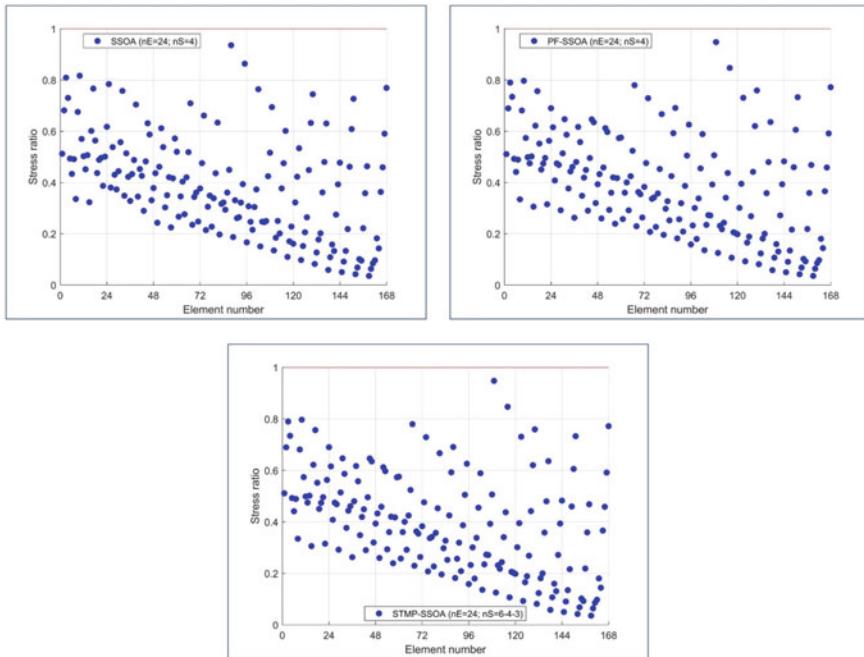


Fig. 4.21 Comparison of the stress ratios of the 3-bay 24-story steel frame structure (best runs)

4.7 Concluding Remarks

In this chapter, two enhanced versions of the shuffle shepherd optimization algorithm (SSOA) are proposed to solve sizing optimization of skeletal structures. The first proposed algorithm is a parameter-free version of SSOA called parameter-free SSOA (PF-SSOA), and the second one is a multi-phase version of PF-SSOA called set-theoretical multi-phase SSOA (STMP-SSOA). The proposed PF-SSOA requires less algorithm-specific parameters than the classical SSOA, which makes it easy to be applied. The main idea of STMP-SSOA is to divide the optimization process of PF-SSOA into several phases. Within each phase of STMP-SSOA, a self-contained PF-SSOA is executed with a specific number of subpopulations. As the optimization process goes on, the number of subpopulations decreases gradually, and consequently the subpopulations become larger. This multi-phase framework is applicable to almost all population-based metaheuristic algorithms. The framework aims to improve the balance of exploration and exploitation of the search process. In order to demonstrate the performance of the proposed algorithms, four well-known sizing optimization of skeletal structures were studied. The results indicate that both PF-SSOA and STMP-SSOA outperform the classical SSOA, especially in terms of robustness and convergence characteristics.

References

1. Kaveh A, Kamalinejad M, Biabani Hamedani K (2021) Enhanced versions of the shuffled shepherd optimization algorithm for optimal design of skeletal structures. *Structures* 29:1463–1495. <https://doi.org/10.1016/j.istruc.2020.12.032>
2. Kaveh A, Biabani Hamedani K, Kamalinejad M (2020) Set theoretical variants of the teaching–learning-based optimization algorithm for optimal design of truss structures with multiple frequency constraints. *Acta Mech* 231(9):3645–3672. <https://doi.org/10.1007/s00707-020-02718-3>
3. Cantor G (1915) Contributions to the founding of the theory of transfinite numbers. Open Court Publishing Company
4. Behravesh A, Kaveh A, Nani M, Sabet S (1988) A set theoretical approach for configuration processing. *Comput Struct* 30(6):1293–1302. [https://doi.org/10.1016/0045-7949\(88\)90194-0](https://doi.org/10.1016/0045-7949(88)90194-0)
5. Kaveh A, Biabani Hamedani K, Zaerreza A (2021) A set theoretical shuffled shepherd optimization algorithm for optimal design of cantilever retaining wall structures. *Eng Comput* 37:3265–3282. <https://doi.org/10.1007/s00366-020-00999-9>
6. Talbi EG (2009) Metaheuristics: from design to implementation, 1st edn. Wiley, USA
7. Kaveh A, Zaerreza A (2020) Shuffled shepherd optimization method: a new meta-heuristic algorithm. *Eng Comput* 37(7):2357–2389. <https://doi.org/10.1108/EC-10-2019-0481>
8. Camp CV, Bichon BJ, Stovall SP (2005) Design of steel frames using ant colony optimization. *J Struct Eng* 131(3):369–379. [https://doi.org/10.1061/\(ASCE\)0733-9445\(2005\)131:3\(369\)](https://doi.org/10.1061/(ASCE)0733-9445(2005)131:3(369))
9. Degerterkin SO (2008) Optimum design of steel frames using harmony search algorithm. *Struct Multidisc Optim* 36(4):393–401. https://doi.org/10.1007/978-3-642-03450-3_2
10. Kaveh A, Bakhshpoori T (2013) Optimum design of steel frames using Cuckoo search algorithm with Lévy flights. *Struct Des Tall Spec Build* 22(13):1023–1036. <https://doi.org/10.1002/tal.754>
11. Kaveh A, Ilchi Ghazaan M (2015) A comparative study of CBO and ECBO for optimal design of skeletal structures. *Comput Struct* 153:137–147. <https://doi.org/10.1016/j.compstruc.2015.02.028>
12. Kaveh A, Bakhshpoori T (2016) An accelerated water evaporation optimization formulation for discrete optimization of skeletal structures. *Comput Struct* 177:218–228. <https://doi.org/10.1016/j.compstruc.2016.08.006>
13. Bellagamba L, Yang TY (1981) Minimum-mass truss structures with constraints on fundamental natural frequency. *AIAA J* 19(11):1452–1458. <https://doi.org/10.2514/3.7875>
14. Grandhi R, Venkayyat VB (1988) Structural optimization with frequency constraints. *AIAA J* 26(7):858–866. <https://doi.org/10.2514/3.9979>
15. Gomez HM (2011) Truss optimization with dynamic constraints using a particle swarm algorithm. *Expert Syst Appl* 38:957–968. <https://doi.org/10.1016/j.eswa.2010.07.086>
16. Miguel LFF, Fadel Miguel LF (2012) Shape and size optimization of truss structures considering dynamic constraints through modern metaheuristic algorithms. *Expert Syst Appl* 39:9458–9467. <https://doi.org/10.1016/j.eswa.2012.02.113>
17. Kaveh A, Javadi SM (2014) Shape and size optimization of trusses with multiple frequency constraints using harmony search and ray optimizer for enhancing the particle swarm optimization algorithm. *Acta Mech* 225(6):1595–1605. <https://doi.org/10.1007/s00707-013-1006-z>
18. American Institute of Steel Construction (AISC) (2001) Manual of steel construction, load & resistance factor design, 3rd edn. IL, USA, Chicago
19. Kaveh A, Ilchi Ghazaan M (2015) Enhanced colliding bodies algorithm for truss optimization with frequency constraints. *J Comput Civ Eng* 29(6):04014104. [https://doi.org/10.1061/\(ASCE\)CP.1943-5487.0000445](https://doi.org/10.1061/(ASCE)CP.1943-5487.0000445)
20. Kaveh A, Ilchi Ghazaan M (2016) Truss optimization with dynamic constraints using UECBO. *Adv Comput Des* 1(2):119–138. <https://doi.org/10.12989/acd.2016.1.2.119>

21. Taheri SHS, Jalili S (2016) Enhanced biogeography-based optimization: a new method for size and shape optimization of truss structures with natural frequency constraints. *Lat Am J Solids Struct* 13(7):1406–1430. <https://doi.org/10.1590/1679-78252208>
22. Dumontel P (1992) Simple equations for effective length factors. *Eng J AISC* 29(3):111–115

Chapter 5

Set-Theoretical Metaheuristic Algorithms for Reliability-Based Design Optimization of Truss Structures



5.1 Introduction

In this chapter, set-theoretical variants of a number of population-based metaheuristic algorithms are used to solve the system reliability-based design optimization (SRBDO) problem of truss structures [1]. The algorithms include the ordered set-theoretical TLBO (OST-TLBO) and set-theoretical multi-phase TLBO (STMP-TLBO) algorithms, which were presented in Chap. 3, set-theoretical multi-phase shuffled shepherd optimization algorithm (STMP-SSOA), which was presented in Chap. 4, and set-theoretical Jaya algorithm (ST-JA), which will be proposed in this chapter. These set-theoretical algorithms are developed based on a simple framework in which the population of solutions is divided into several smaller well-arranged subpopulations of the same size. It has been known for many years that a fully satisfactory estimate of the reliability of a structure must be based on a structural system reliability analysis [2]. However, the literature review shows that most previous studies on reliability-based design optimization of truss structures have focused on the reliability of single structural members rather than the reliability of the structural system. This is because of the fact that reliability-based design optimization of truss structures with system-level reliability constraints usually involves computationally expensive calculations. This is especially the case of real redundant truss structures, where the number of all possible different failure modes is so large that it is impractical to take into account all of them in estimating the reliability of the structural system. However, it is not necessary to include all failure modes because the majority of them generally have a relatively small probability of occurrence. Therefore, only the most dominant failure modes should be included. In this study, the branch and bound method is employed to identify the dominant failure modes, and then to evaluate the system reliability using dominant failure modes. Additionally, the first-order second-moment (FOSM) method is used to estimate the failure probabilities. In order to demonstrate the efficiency and effectiveness

of the proposed SRBDO approach, three truss optimization problems with system reliability constraints are investigated.

Optimum design of truss structures is one of the active fields of research in structural optimization. Truss optimization problems can be classified into three categories of size optimization, shape optimization, and topology optimization. However, it should be noted that two or all three of the aforementioned categories can also be considered simultaneously. In a truss optimization problem, the objective is usually to minimize the weight of the truss structure while satisfying a set of predefined constraints. In the last three decades, several studies have adopted metaheuristic algorithms to solve truss optimization problems. For example, Dhingra and Bennage [3] applied simulated annealing (SA) algorithm for topological optimization of truss structures. Hasançebi and Kazemzadeh Azad used a refined version of the big bang-big crunch (BB-BC) algorithm [4] for discrete truss sizing optimization problems. Recently, Kaveh et al. [5, 6] applied set-theoretical variants of some population-based metaheuristic algorithms to solve truss optimization problems with frequency constraints. A review of the literature shows that in the past few years, research on the optimization of truss structures has mainly focused on deterministic design approaches where partial safety factors are used to design sufficiently safe structures. In deterministic design approaches, the uncertainties of the structural system are not explicitly taken into account in the structural design process. In these approaches, in order to account for uncertainties of the structural system, a number of partial safety factors are employed in the structural design process as factors on characteristic or nominal values of the major structural and loading variables, and in this way an adequate level of structural safety is achieved. It has been recognized for many decades that material properties, loads, and geometrical dimensions are in general associated with some uncertainties. As a result, deterministic design approaches are not able to directly incorporate the uncertainties in the structural design process, which may lead to an overdesign of the structure. On the other hand, reliability-based design approaches provide the possibility to directly take into account the aforementioned uncertainties in the structural design process. As a result, in recent years, several studies have focused on the reliability-based design optimization (RBDO) approaches in which the structural failure probability is directly considered as the optimization constraint [7]. For example, Murotsu and Shao [8] proposed a procedure for size and shape optimization of truss structures under the constraint on the structural failure probability. Nakib [9] reported analytical investigations into both deterministic and reliability-based optimization of truss bridges. Stocki et al. [10] investigated the efficiency of two discrete optimization methods when applied to reliability-based optimization problem of truss structures. In their study, constraints are imposed on the values of component reliability indices. Burton and Hajela [11] investigated the efficiency of a number of reliability-based optimization techniques in the shape optimization of truss structures.

It was mentioned earlier that in order to achieve a fully satisfactory estimate of the reliability of a structural system, a system approach needs to be followed [2]. In general, there are three main approaches for evaluating the system reliability of complex structures: (1) numerical integration methods; (2) failure path methods; and

(3) simulation methods [12]. In the numerical integration methods, the probability of failure is obtained directly by integrating the joint probability density function over the failure domain. However, since direct integration is difficult in multidimensional space, these methods are limited to two-dimensional spaces. The basic idea behind the simulation methods is to develop an analytical model to determine whether the structure fails or not. The model is simulated several times with different randomly generated loads and resistances. As a result, several predictions of the structure behavior are obtained. The probability of failure is obtained by dividing the number of failure events to the total number of simulations. Although simulation methods are relatively straightforward, these methods become computationally intensive when low probabilities of failure have to be estimated. In the failure path methods, two main steps are involved in the reliability analysis of a structural system: (1) identification of failure modes; and (2) calculation of failure probabilities of failure modes and the structural system. As aforementioned, a real structural system has many different failure modes and it is not practical to include all of them in estimating the reliability of the structural system. In order to overcome this difficulty, many studies have focused on the possibility of evaluation of the system reliability using only dominant failure modes [2, 13]. These studies are based on the assumption that the reliability of a structural system can be estimated using these dominant failure modes. Once the dominant failure modes are identified, the failure probability of the structural system is calculated using combination of the identified dominant failure modes and bounding techniques. Despite these advantages, the design optimization problem of a structural system under a system reliability constraint is still recognized as a computationally challenging problem. This is especially the case in real indeterminate structural systems that have many different failure modes. As a result, most of the previous reliability-based design optimization studies have focused on the reliability of single structural members [10, 11], and only some simple structures have been optimized under a system reliability constraint [2, 8].

In this study, in order to provide a practical tool for RBDO of truss structures, a system reliability-based design optimization approach is developed by integrating: (1) a set-theoretical framework for population-based optimization algorithms [5, 6]; and (2) the branch and bound method to identify the dominant failure modes of the structural systems. The allowable stresses of the members and the applied loads are assumed to be uncorrelated normally distributed random variables. Both perfectly brittle and perfectly ductile failure elements are considered. Buckling failure mode in compression members is also considered.

The rest of this chapter is organized as follows: In Sect. 5.2, set-theoretical variants of three population-based metaheuristic optimization algorithms are presented. In Sect. 5.3, system reliability analysis of truss structures is discussed. Section 5.4 formulates the design optimization problem of truss structures with system reliability constraints. In Sect. 5.5, three truss optimization problems with system reliability constraints are explored to provide evidence to support the applicability and validity of the proposed SRBDO approach. Finally, Sect. 5.6 concludes the chapter.

5.2 Set-Theoretical Variants of the Population-Based Optimization Algorithms

Set theory is a branch of mathematics concerned with the study of sets and their properties. In classical set theory, a set is defined as a well-defined collection of distinct objects called elements, or members, of the set. Almost all mathematical concepts can be defined in terms of set-theoretical concepts. Thus, set theory can be viewed as a foundation for mathematics. In recent years, set theory has found many applications in structural engineering. For example, Kaveh et al. [14] used some basic concepts of set theory to generalize the definition of the shuffled shepherd optimization algorithm, and Kaveh [15] employed a set of contours and its transversal in nodal ordering for bandwidth reduction.

Metaheuristic algorithms can be classified based on different criteria, the most common of which is population-based search versus single-solution-based search. Population-based metaheuristic algorithms work with a set of solutions called population during the search process and repeatedly modify them, whereas single-solution-based metaheuristic algorithms modify and improve a single candidate solution during the search process. The initial population of population-based metaheuristic algorithms can be viewed as a set of elements. Recently, Kaveh et al. [5] proposed a general set-theoretical framework for the population-based metaheuristic algorithms. The proposed algorithm is successfully applied it to the teaching–learning-based optimization (TLBO) algorithm [5] and the shuffled shepherd optimization algorithm (SSOA) [6]. The main idea behind the framework is to divide the population of solutions into smaller well-arranged subpopulations of the same size. This framework aims to improve the balance between exploration and exploitation of the search space [5]. In the next two subsections, set-theoretical variants of TLBO and SSOA are presented briefly. Next, set-theoretical variant of the Jaya algorithm (JA) is proposed.

5.2.1 *Set-Theoretical Variants of the Teaching–Learning-Based Optimization Algorithm*

The TLBO algorithm is one of the most well-known population-based metaheuristics developed by Rao et al. [16] in 2011. The algorithm simulates the traditional process of teaching and learning in an educational class. In this algorithm, a class of learners is considered as the population of solutions. The search process of the TLBO algorithm is divided to two different phases called teacher phase and learner phase. In the teacher phase, the teacher, which is considered as the best learner of the class, tries to transfer the knowledge to the learners, while in the learner phase, learners try to learn from each other. The main idea of set-theoretical variants of TLBO is to divide the main class (i.e., the population of learners) into smaller well-arranged

subclasses (i.e., subpopulations) of the same size [5]. According to the work of Kaveh et al. [5], two different set-theoretical variants of the TLBO algorithm named ordered set-theoretical TLBO (OST-TLBO) and set-theoretical multi-phase TLBO (STMP-TLBO) are presented as follows.

5.2.1.1 Ordered Set-Theoretical Teaching–Learning-Based Optimization (OST-TLBO) Algorithm

The OST-TLBO algorithm is explained in four steps as follows [5]:

Step one (initialization): The initial population of learners is generated randomly.

Step two (forming the subpopulations): Based on the procedure proposed by Kaveh et al. [14], the main population is divided into a number of subpopulations. For this purpose, let us consider an initial population of size nE . The goal is to generate a certain number of subpopulations (e.g., m) of the same size. First, the main population is sorted in ascending order of penalized objective function values. In the first step of generating subpopulations, the first m solutions of the sorted population are selected and each solution is placed in a different subpopulation randomly. In the second step, the next m solutions of the sorted populations are selected and each solution is placed in a different subpopulation randomly. The process continues until all solutions of the sorted population are placed in the subpopulations. At the end of the division process, all subpopulations contain the same number of solutions $nS = nE/m$.

Step three (main body of the TLBO algorithm): The main body of the TLBO algorithm is executed for each subpopulation separately. For this purpose, the teacher and learner phases are executed once for each subpopulation separately. It should be pointed out that once each phase is completed, a simple replacement strategy is applied to constitute the subpopulations of the next generation. The replacement is done if the newly generated solution has a better penalized objective function value than its corresponding current one. Next, all subpopulations are merged to form the population of the next generation.

Step four (termination criteria): If the termination criterion of the algorithm is met, the search process is terminated. Otherwise, the algorithm returns to step two.

Figure 5.1 shows the steps of the OST-TLBO algorithm.

5.2.1.2 Set-Theoretical Multi-phase Teaching–Learning-Based Optimization (STMP-TLBO) Algorithm

The STMP-TLBO algorithm is a multi-phase version of the OST-TLBO algorithm. In each phase of the STMP-TLBO algorithm, a self-contained OST-TLBO algorithm with a specific number of subpopulations is carried out. The initial population is considered as the input of the first phase of the STMP-TLBO algorithm. In the following phases, the input is the output of the previous phase. Each phase continues until the number of objective function evaluations ($NFEs$) reaches a predefined threshold. The only difference between the phases is the number of subpopulations.

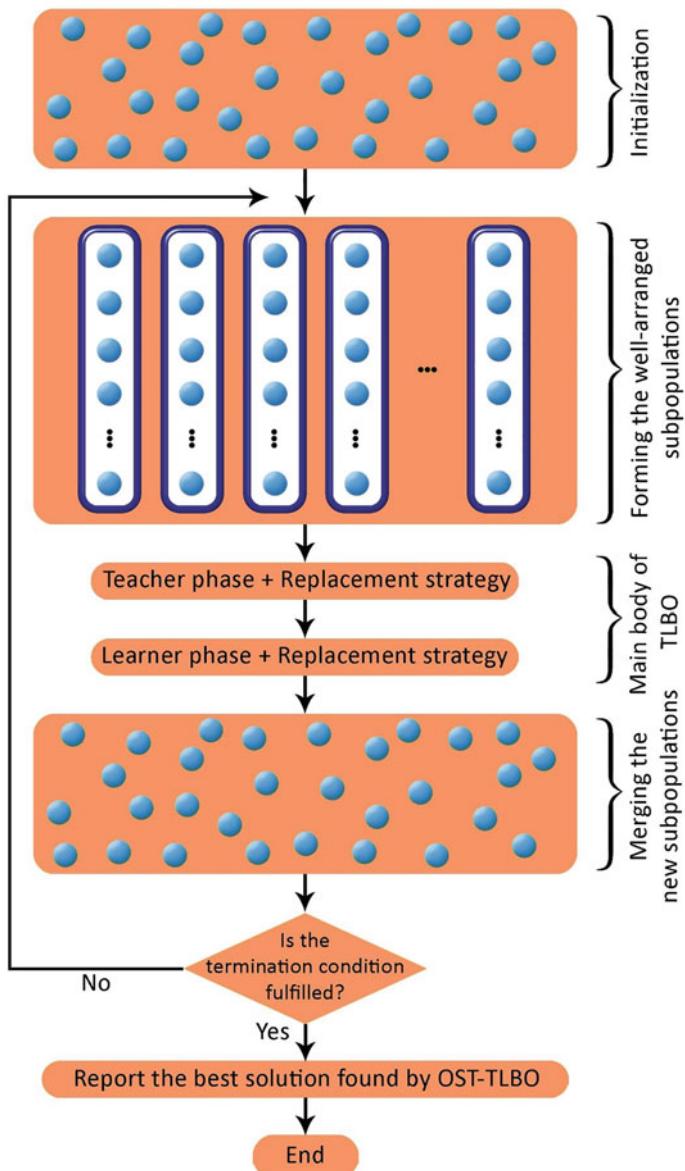


Fig. 5.1 Steps of the OST-TLBO algorithm

The number of subpopulations decreases from n_0 ($2 < n_0 \leq nE$) at the first phase to two at the last phase [5]. The steps of the STMP-TLBO algorithm are shown in Fig. 5.2. Please see Chap. 3 of the book for more details about OST-TLBO and STMP-TLBO algorithms.

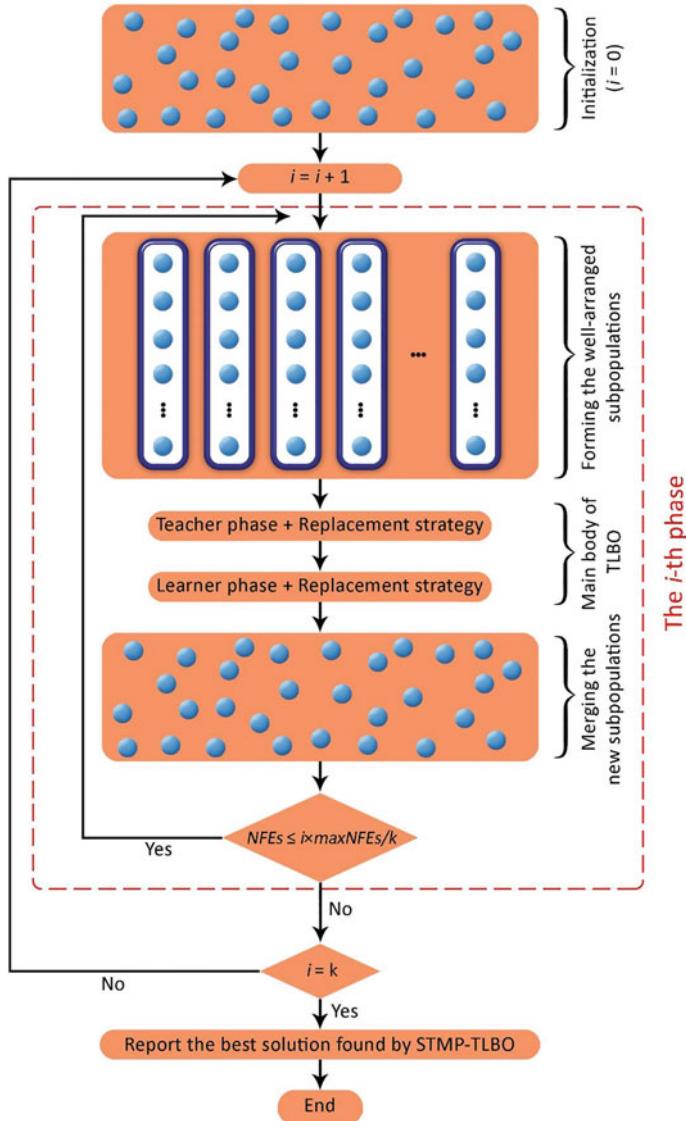


Fig. 5.2 Steps of the STMP-TLBO algorithm

5.2.2 Set-Theoretical Variant of the Shuffled Shepherd Optimization Algorithm

SSOA is a population-based metaheuristic optimization algorithm which simulates the herding behaviour of shepherds [17]. Recently, Kaveh et al. [14] generalized the definition of SSOA using the concepts of set theory. Furthermore, Kaveh et al. [6] proposed two enhanced versions of SSOA named parameter-free shuffled shepherd optimization algorithm (PF-SSOA) and set-theoretical multi-phase shuffled shepherd optimization algorithm (STMP-SSOA). As its name implies, PF-SSOA is a parameter-free version of the classical SSOA [6]. STMP-SSOA is a multi-phase version of PF-SSOA. In each phase of STMP-SSOA, a self-contained PF-SSOA with a specific number of subpopulations is executed. In other words, different phases of STMP-SSOA differ only in the number of subpopulations. As the optimization process goes on, the number of subpopulations decreases. Each phase continues until the number of objective function evaluations (*NFEs*) reaches a predefined threshold. Please see Chap. 4 of the book for more details about STMP-SSOA. The steps of STMP-SSOA are shown in Fig. 5.3. The steps of PF-SSOA is as follows [6]:

Step one (initialization): The initial population is generated randomly.

Step two (forming the subpopulations): According to the procedure described in the second step of the OST-TLBO algorithm, the main population is divided into a number of smaller well-arranged subpopulations.

Step three (position update rule): The new position of the j -the candidate solution of the i th subpopulation ($newEL_{i,j}$) is determined as follows:

$$newEL_{i,j} = EL_{i,j} + stepsize_{i,j}; i = 1, 2, \dots, nS \text{ and } j = 1, 2, \dots, m \quad (5.1)$$

$$\begin{aligned} stepsize_{i,j} &= \beta \cdot (EL_B_{i,j} - EL_{i,j}) + \alpha \cdot (EL_{i,j} - EL_W_{i,j}); \\ i &= 1, 2, \dots, nS \text{ and } j = 1, 2, \dots, m \end{aligned} \quad (5.2)$$

$$\beta = rand(2).rand(1, nV) \quad (5.3)$$

$$\alpha = ones(1, nV) - \beta \quad (5.4)$$

where $EL_{i,j}$ is the current position of the j -the candidate solution of the i th subpopulation; $EL_B_{i,j}$ is the position of a candidate solution of the i th subpopulation which is better than its j -the candidate solution; $EL_W_{i,j}$ is the position of a candidate solution of the i th subpopulation which is worse than its j -the candidate solution; $stepsize_{i,j}$ is the step size of movement of the j -the candidate solution of the i th subpopulation; nV is the number of design variables; α and β are parameters of the algorithm; nS is the number of subpopulations; m is the number of solutions in each subpopulation; $rand$ is a random number from the interval $[0, 1]$; and $rand(2)$ returns a pseudorandom scalar integer between 1 and 2.

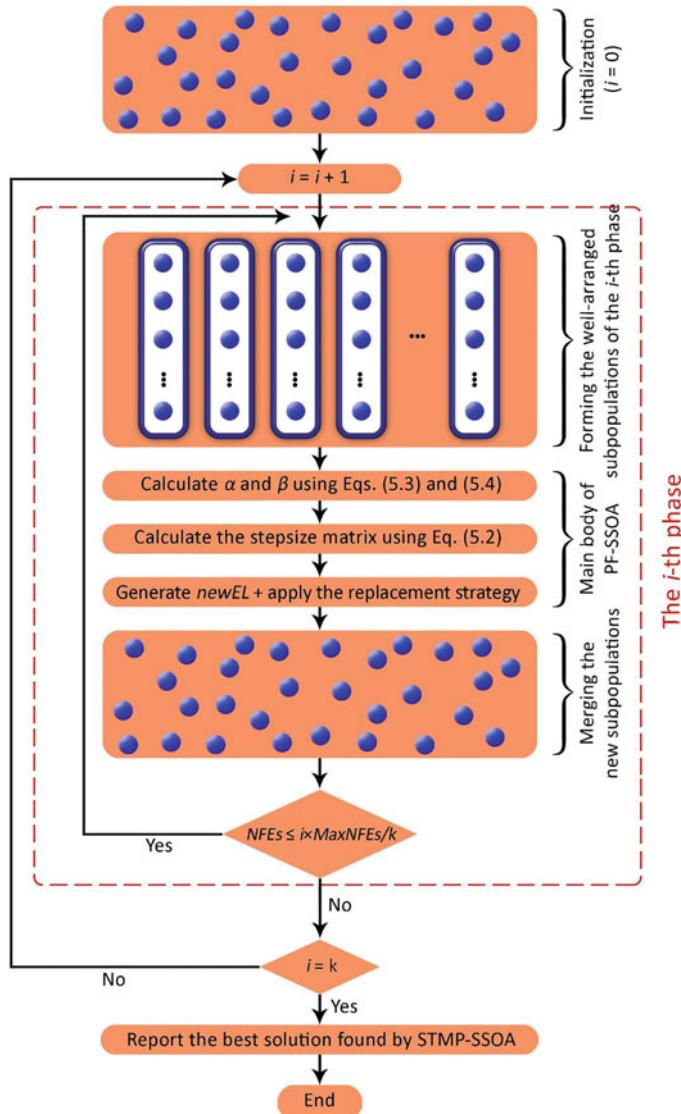


Fig. 5.3 Steps of the STMP-SSOA

It should be pointed out that a simple replacement strategy is applied to determine the subpopulations of the next generation. In the replacement strategy, each newly generated solution is compared with its corresponding current one and the solution with better penalized objective function value is preferred. Before the next iteration starts, all subpopulations are merged to form the main population of the next generation.

Step four (termination criteria): If the termination condition of the algorithm is satisfied, the optimization process is terminated. Otherwise, the algorithm returns to step two.

5.2.3 Set-Theoretical Variant of the Jaya Algorithm

The JA is a simple yet powerful population-based metaheuristic optimization algorithm proposed by Rao [18] in 2016. The Jaya algorithm is based on a simple concept: moving towards the best solution and avoiding the worst one. The candidate solutions in the Jaya algorithm are updated as follows:

$$\begin{aligned} A'(i, j, k) = & A(i, j, k) + r(i, j, 1)(A(i, j, b) - |A(i, j, k)|) \\ & - r(i, j, 2)(A(i, j, w) - |A(i, j, k)|) \end{aligned} \quad (5.5)$$

where i , j , and k are the indices of the iteration numbers, design variables, and candidate solutions, respectively; $A(i, j, k)$ and $A'(i, j, k)$ are the current and new positions of the j th design variable of the k th candidate solution of the i th iteration, respectively; $r(i, j, 1)$ and $r(i, j, 2)$ are uniformly distributed random numbers in the range $[0, 1]$; and $A(i, j, b)$ and $A(i, j, w)$ are the positions of the j th design variable of the best and worst candidate solutions of the i th iteration, respectively. Like almost all the other population-based metaheuristic algorithms, JA starts with a set of randomly generated candidate solutions, which constitute the initial population. In each iteration after the solutions are updated, a simple replacement strategy is applied to determine the population of the next iteration. For this purpose, the current and new positions of each solution are compared with each other and the one having better penalized objective function value is preferred. The search process continues until the termination condition is reached.

Based on the framework proposed for the OST-TLBO algorithm, the set-theoretical Jaya algorithm (ST-JA) is developed in five steps:

Step one (initialization): The initial population is generated randomly within the search space of the problem.

Step two (forming the subpopulations): Similar to the second step of the OST-TLBO algorithm, the main population is divided into a number of smaller well-arranged subpopulations of the same size.

Step three (position update rule): The solutions are updated based on Eq. (5.5). It should be noted that each subpopulation has its own unique best and worst solutions. In other words, different subpopulations move towards different promising regions and move away from different poor regions of the search space. This allows keeping the diversification of the search process.

Step four (replacement strategy): In this step, a simple replacement strategy is applied to determine the population of the next iteration. To this end, the new position of each candidate solution is compared with its current position and the position

having better penalized objective function value is preferred. Next, all subpopulations are merged to constitute the population of the next generation.

Step five (termination criteria): If the termination condition of the algorithm is fulfilled, the search process is terminated. Otherwise, the algorithm returns to step two.

The steps of ST-JA are shown in Fig. 5.4.

5.3 System Reliability Analysis of Truss Structures

In the case of a statically determinate truss structure, support reactions and member forces can be easily calculated by solving the static equilibrium equations [19]. The failure of any one of the members of this kind of truss structures leads to the failure of the entire structural system. However, this is not necessary true for statically indeterminate truss structures. In other words, for this kind of truss structures, failure of one member does not necessarily lead to complete structural failure, which is due to the fact that the remaining members may still be able to withstand the applied. In this case, the most frequently used definition of structural failure is formation of a collapse mechanism in the structure. If this definition is adopted, simultaneous failure of a set of structural members forming a collapse mechanism is called a failure mode. A failure mode is generated in the following way. When a member fails, the internal forces are redistributed among the remaining members of the structure and the member next to fail is determined. An important point should be taken into account in redistribution of the internal forces. When any one member fails, corresponding to the type of failure, its residual strength is applied as an artificial force to the end nodes of the member, and the member mechanical contribution to the stiffness matrix is set to zero. The resulting stiffness matrix is then called the reduced structural stiffness matrix. It is worth mentioning that the residual strength represents the post-failure load-carrying capacity of the member. The consecutive failure of structural members continues until a collapse mechanism is formed and the structural system fails [2]. Let us consider the case where a specified number of structural members (e.g., members m_1, m_2, \dots , and m_{p_q}) are lost. In this case, formation of a collapse mechanism is determined by investigating the singularity of the stiffness matrix of the structure composed of $n - p_q$ members in survival. This can be expressed in the following form:

$$\left| K^{(p_q)} \right| = 0 \quad (5.6)$$

where $\left| K^{(p_q)} \right|$ is the determinant of the matrix $K^{(p_q)}$; and $K^{(p_q)}$ is the structure stiffness matrix formed with the $n - p_q$ remaining members, where n is the total number of structural members.

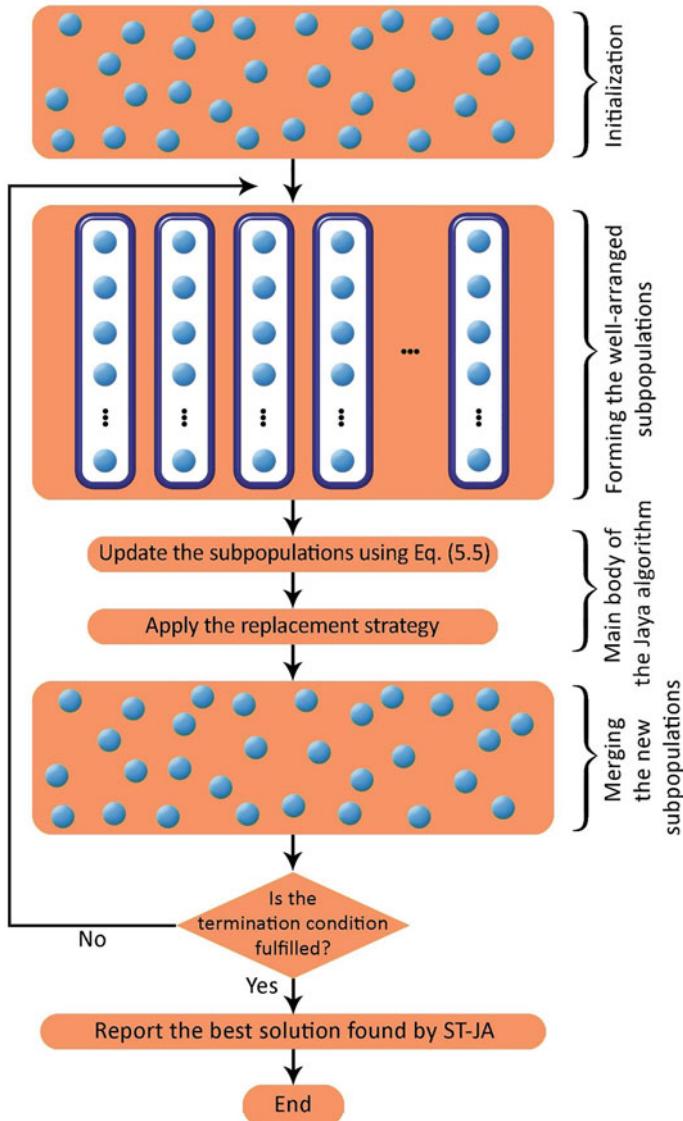


Fig. 5.4 Steps of ST-JA

In the following subsection, a systematic procedure for automatic generation of safety margins for truss structures is presented briefly.

5.3.1 Generation of Safety Margins for Truss Structures

Consider a space truss structure consisting of n members. Failure of a structural member occurs when the internal force of the member exceeds its strength. Therefore, the safety margin is defined as the difference between the strength and the internal force. The safety margin of the i th member can be expressed as follows:

$$\begin{aligned} Z_i &= R_i(A_i, \sigma_i) - S_i(A_1, \dots, A_n; P_1, \dots, P_{3l}; E_1, \dots, E_n; l_1, \dots, l_n) \\ &= A_i \sigma_i - \left| \sum_{j=1}^{3l} b_{ij} P_j \right| \end{aligned} \quad (5.7)$$

where Z_i , R_i , and S_i denote the safety margin, strength, and internal force of the i th truss member, respectively; A_i , Z_i , σ_i , and l_i are the cross-sectional area, allowable stress, modulus of elasticity, and length of the i th truss member, respectively; n is the number of truss members; P_j is the external load applied to the j th degree of freedom of the structure; l is the number of nodes of the structure; and b_{ij} is the internal force of the i th member caused by applying a unit load to the j th degree of freedom in the same direction of P_j . In case of tensile or compressive failure, the yield stress σ_y is taken as the allowable stress, while when buckling failure occurs under compression, the buckling stress σ_b is taken as the allowable stress. As noted above, in the case of a statically determinate truss structure, the failure of any one of the truss members will lead to the failure of the structure. Consequently, the structural failure criterion for a statically determinate truss structure is defined as follows:

$$Z_i \leq 0; \forall i \in \{1, 2, \dots, n\} \quad (5.8)$$

If R_i and S_i are normally distributed independent random variables, the probability of failure can be calculated in the following form [2]:

$$P_{f_i} = \Phi(-\beta_i) = \Phi\left(\frac{\mu_{S_i} - \mu_{R_i}}{\sqrt{\sigma_{S_i}^2 + \sigma_{R_i}^2}}\right) \quad (5.9)$$

where μ_{S_i} and σ_{S_i} are the mean value and standard deviation of the internal force of the i th truss member, respectively; μ_{R_i} and σ_{R_i} are the mean value and standard deviation of the strength of the i th truss member, respectively; P_{f_i} is the probability of failure of the i th truss member; β_i is the reliability index of the i th truss member; and Φ is the standard normal cumulative distribution function. The distribution function Φ for the random variable X is given by:

$$\Phi(X) = \int_{-\infty}^X \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}x^2\right) dx \quad (5.10)$$

By using the safety margins, a criterion for structural failure of a statically indeterminate truss structure can be defined as follows:

$$Z_{m_p(m_1, m_2, \dots, m_{p-1})}^{(p)} \leq 0; p = 1, 2, \dots, p_q \quad (5.11)$$

The safety margins of the members in survival can be calculated by the following equation:

$$Z_{i(m_1, m_2, \dots, m_{p-1})}^{(p)} = A_i \sigma_i - \left| \left(\sum_{j=1}^{3l} b_{ij}^{(p)} L_j - a_{im_1}^{(p)} R_{m_1} - a_{im_2}^{(p)} R_{m_2} - \dots - a_{im_{p-1}}^{(p)} R_{m_{p-1}} \right) \right| \quad (5.12)$$

In the above equations, $(m_1, m_2, \dots, m_{p-1})$ denotes the set of failed members arranged in the sequential order of failure; the set of failed members and the order of failure; $Z_{i(m_1, m_2, \dots, m_{p-1})}^{(p)}$ is the safety margin of the i th member at the p th stage of failure; $b_{ij}^{(p)}$ is the internal force of the i th member at the p th stage of failure, which is caused by applying a unit load to the j th degree of freedom in the same direction of P_j ; and $a_{ij}^{(p)}$ is the internal force of the i th member at the p th stage of failure, which is caused by applying a unit load to both end nodes of the failed member m_j in the opposite direction of the internal force of m_j .

As aforementioned, a structural system fails as soon as any of its failure modes occurs. The number of possible failure modes of a real redundant structure is so astronomically large that it is impossible in practice to take into account all of them in estimating the reliability of the structure. In such a case, only the stochastically dominant failure modes are considered in the structural system reliability analysis. The dominant failure modes of structural systems can be identified using the failure path methods, such as the branch and bound method, the β -unzipping method, and the incremental loading method. In the following subsection, the branch and bound method is presented briefly.

5.3.2 The Branch and Bound Method

In the failure path methods, the structural reliability analysis takes place in two main steps: (1) identification of the stochastically dominant failure modes; and (2) estimation of failure probabilities of the identified failure modes and the structural system. The branch and bound method is one of the well-known failure path methods developed by Murotsu et al. [20] in the late 1980s and early 1990s. The branch and bound method has three main steps as follows:

Step one (partitioning): All the potential failure members are added to the incomplete failure path under consideration. Unlike a complete failure path, an incomplete one does not lead to the formation of a collapse mechanism. Add the newly partitioned failure paths to the set of candidate failure paths for branching. Next, calculate the

upper bounds of the probabilities of failure of the partitioned failure paths by using the following upper bound:

$$P_{fp(m_1m_2\cdots m_p)(U)} = \min_{j \in \{2, 3, \dots, p\}} P \left[(Z_{m_1}^{(1)} \leq 0) \cap (Z_{m_j(m_1, m_2, \dots, m_{j-1})}^{(j)} \leq 0) \right] \quad (5.13)$$

Step two (branching): Select the failure path which gives the largest upper bound among the partitioned failure paths. Next, investigate whether a collapse mechanism is formed or not. If a collapse mechanism is not formed, go to step one and proceed the partitioning process by adding all the potential failure members to the selected failure path. Otherwise, go to step three for bounding operation.

Step three (bounding): Remove the selected failure path from the set of candidate failure paths for branching and add it to the set of the selected complete failure paths. Next, calculate the lower bound of the selected complete failure path probability by using the following equation:

$$\begin{aligned} P_{fp(m_1m_2\cdots m_p)(L)} = & \max \left[0, P \left[(Z_{m_1}^{(1)} \leq 0) \right] - P \left[(Z_{m_1}^{(1)} \leq 0) \cap (Z_{m_2(m_1)}^{(2)} > 0) \right] \right. \\ & \left. - \sum_{j=3}^p \min \left\{ P_{fp(m_1m_2\cdots m_p)(U)}, P \left[(Z_{m_1}^{(1)} \leq 0) \cap (Z_{m_j(m_1, m_2, \dots, m_{j-1})}^{(j)} > 0) \right] \right\} \right] \end{aligned} \quad (5.14)$$

Update the reference value for bounding. For this purpose, find the maximum lower bound of probability of the complete failure paths selected so far. The reference value for bounding is updated as follows:

$$\text{If } RF < P_{fp(m_1m_2\cdots m_p)(L)} \Rightarrow RF = P_{fp(m_1m_2\cdots m_p)(L)} \quad (5.15)$$

Next, identify the partitioned failure paths which have the probabilities of failure lower than $F \times 10^{-\delta}$ and remove them from the set of candidate failure paths for branching. It is noted that δ is the bounding constant which is determined based on the desired accuracy in estimation of failure probabilities. In general, the larger the bounding constant, the fewer failure paths are discarded, and thus the more accurate the failure probability. If the set of candidate failure paths for branching is empty, it means that there are no further failure paths for branching. In such a case, the branch and bound method is terminated. Otherwise, go to step two.

At the end of the branch and bound method, the set of the selected complete failure paths contains the stochastically dominant failure modes of the structure. Finally, the probability of failure of the structural system is estimated by employing the bounding techniques and based on the dominant failure modes.

5.3.3 Evaluation of the System Reliability

As mentioned earlier, a structural system will fail when the weakest failure mode occurs. Therefore, the structural system can be modelled based on a series system where the elements in the series system are failure modes. Generally, exact calculation of the probability of failure of such systems involves multi-dimensional integrals, which are complicated or even impossible to solve [21]. However, computationally simple bounding techniques exist to estimate the probability of failure of series systems, such as Cornell's bounds, Ditlevsen's bounds, and Vanmarcke's upper bound. The lower and upper Cornell's bounds for the probability of failure of series systems are given by:

$$\max_{i=\{1,2,\dots,nM\}} (P_{fp(U)_i}) \leq P_{fs} \leq 1 - \prod_{i=1}^{nM} (1 - P_{fp(U)_i}) \quad (5.16)$$

where P_{fs} is the probability of failure of the series system; nM is the number of stochastically dominant failure modes identified by the branch and bound method; and $P_{fp(U)_i}$ is the upper bound of the probability of failure of the i th dominant failure mode, which is determined by using Eq. (5.11).

5.4 System Reliability-Based Design Optimization of Truss Structures

The main goal of structural reliability theory is to make it possible to design structures based on the reliability concepts. It has been recognized that the actual load-carrying capacity of a structural system is usually much higher than what is obtained by the design of individual structural members, which may be attributed to the interaction between the structural members forming the structural system [22]. Therefore, the SRBDO approaches should be used. The SRBDO approaches mainly aim to minimize the structural weight under the constraint on the probability of failure of the structural system. The problem of system reliability-based design optimization of truss structures can be expressed mathematically as follows:

$$\text{Find: } \{X\} = \{x_1, x_2, \dots, x_{nDV}\}; x_i \in D_i \quad (5.17)$$

$$\text{to minimize: } W(\{X\}) = \sum_{i=1}^n \rho_i A_i l_i \quad (5.18)$$

so that:

1. The truss structure would be geometrically stable;
2. The following constraints would not be violated.

$$x_i^L \leq x_i \leq x_i^U; i = 1, 2, \dots, nDV \quad (5.19)$$

$$P_f(\{X\}) \leq P_f^t = \Phi(-\beta_t) \quad (5.20)$$

In the above equations, $\{X\}$ is the solution vector of the design variables; x_i is the i th design variable; nDV is the number of design variables; D_i is the allowable set of continuous values for the design variable x_i ; $W(\{X\})$ is the total weight of the structure; ρ_i is the material density of the i th truss member; x_i^L and x_i^U are the lower and upper bounds of the design variable x_i , respectively; $P_f(\{X\})$ is the upper bound of the probability of failure of the structure, which is obtained by the branch and bound method; β_t is the target system reliability index; and P_f^t is the target failure probability of the structure. The design variable x_i can vary continuously within a certain range as follows:

$$D_i = \{x_i | x_i \in [x_i^L, x_i^U]\} \quad (5.21)$$

5.5 Numerical Examples

In this section, in order to investigate the performance of the proposed SRBDO approach, three truss optimization problems with constraints on the target failure probability are considered as follows: size optimization of a statically indeterminate 16-member planar truss; size optimization of a statically indeterminate 65-member truss bridge; and size optimization of a statically indeterminate 67-member truss bridge. It should be noted that since there are no direct comparative SRBDO studies available in the literature for some of these truss structures, the results are compared with deterministic solutions, if possible. The objective of optimization is to minimize the structural weight while satisfying constraint on the probability of structural failure. The cross-sectional areas of the members and the nodal coordinates are taken as continuous design variables. The members are assumed to be circular solid sections made of a perfectly ductile material. The loads and the strengths of structural members are assumed to be normally distributed uncorrelated random variables. Buckling failure mode in compression members is also included. In the case of buckling failure mode, the mean and coefficient of variation of buckling stress of the member with initial deflection under compression are calculated by the following formulas [2]:

$$\mu_{\sigma_b} = \frac{1}{2} \left(\mu_{\sigma_y} + \sigma_E + \frac{\mu_{W_0}}{S} \sigma_E \right) \left(1 - \sqrt{1 - \frac{(4\mu_{\sigma_y}\sigma_E)}{\left(\mu_{\sigma_y} + \sigma_E + \frac{\mu_{W_0}}{S} \sigma_E\right)^2}} \right) \quad (5.22)$$

$$CV_{\sigma_b} = \frac{\sqrt{\left(CV_{\sigma_y}\mu_{\sigma_y}\right)^2 + \left(\sigma_E/S\right)^2 \left(CV_{W_0}\mu_{W_0}\right)^2}}{\left(\mu_{\sigma_y} + \sigma_E + \frac{\mu_{W_0}}{S} \sigma_E\right) \sqrt{1 - \frac{(4\mu_{\sigma_y}\sigma_E)}{\left(\mu_{\sigma_y} + \sigma_E + \frac{\mu_{W_0}}{S} \sigma_E\right)^2}}} \quad (5.23)$$

where μ_{σ_y} and CV_{σ_y} are the mean and coefficient of variation of the yield stress; μ_{W_0} and CV_{W_0} are the mean and coefficient of variation of the initial deflection of the members; S is the radius of gyration of the cross-sectional area of the members; and $\sigma_E = \pi^2 E / (l/S)^2$ is the Euler buckling stress. In the above equations, initial deflection and yield stress are treated as normally distributed uncorrelated random variables. The termination criterion of the algorithms is the maximum number of objective function evaluations (*MaxNFEs*). The optimization algorithms, as well as structural reliability analysis codes, are implemented in Matlab.

5.5.1 *Statically Indeterminate 16-Member Planar Truss*

Size optimization of a 16-member truss with three degrees of redundancy is considered as the first design example. The structure is shown in Fig. 5.5. Reliability analysis of this structure was previously done by many researchers such as Murotsu et al. [20]. The layout of the structure is kept unchanged during the optimization process. Truss members are divided into eight groups of design variables. The design optimization is carried out for both the cases of with and without considering the effect of buckling failure in compression members. Table 5.1 provides the numerical data of the structure, such as design variable bounds, material properties, statistical data of the random variables, etc. The optimum design solutions for the case of neglecting the effect of buckling failure are presented in Table 5.2 for various set-theoretical optimization algorithms. Table 5.3 presents the optimum design solutions for the case where buckling failure is included. As Tables 5.2 and 5.3 prove, constraint on the probability of failure is satisfied in all cases. The results show that, in the case where buckling failure is not considered, the best designs of OST-TLBO, STMP-TLBO, STMP-SSOA, and ST-JA are very close to each other, while, in the case where buckling failure is included, STMP-SSOA and ST-JA perform better than OST-TLBO and STMP-TLBO in terms of the best weight. A comparison between the two cases results indicates that considering buckling failure mode in compression members significantly affects the optimal values of design variables. Consequently, buckling failure mode should be taken into account in the optimum design of truss structures. The convergence histories of the best optimum designs are presented in Figs. 5.6 and 5.7. It can be observed from the figures that STMP-SSOA and ST-JA converge faster than OST-TLBO and STMP-TLBO.

Fig. 5.5 Statically indeterminate 16-member truss

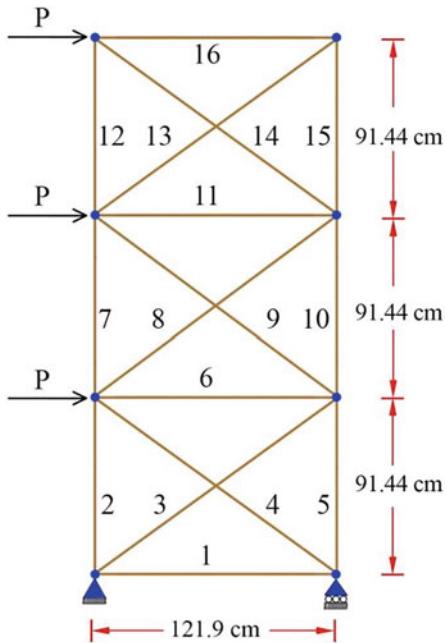


Table 5.1 Data of the statically indeterminate 16-member truss

Property/Unit	Value
E (Elasticity modulus)/GPA	206
ρ (Material density)/kg/m ³	2700
μ_{σ_y} (Mean value of yield stress)/MPA	276
CV_{σ_y} (Coefficient of variation of yield stress)	0.02
μ_P (Mean value of applied loads)/kN	44.45
CV_P (Coefficient of variation of applied loads)	0.02
μ_{W_0}/S	0.1
CV_{W_0} (Coefficient of variation of initial deflection)	0.1
Lower bound of design variables/cm ²	0.1
Upper bound of design variables/cm ²	15
P_f^t (Target failure probability)	10^{-5}

5.5.2 Statically Indeterminate 65-Member Truss Bridge

The second design example considers size optimization of a 65-member truss bridge as shown in Fig. 5.8. The layout and geometry of the structure are taken from Nakib [9]. The structure has only one degree of redundancy. The structural members are assumed to fail only in tension or compression. With respect to the symmetry of the

Table 5.2 Comparison of optimization results for the 16-member truss (buckling is neglected)

Design variable: A_i (cm^2)		OST-TLBO	STMP-TLBO	STMP-SSOA	ST-JA
Group number	Members of the group				
A_1	1	3.59	3.60	3.62	3.62
A_2	2, 5	7.58	7.56	7.58	7.58
A_3	3, 4, 14	3.26	3.26	3.26	3.26
A_4	6	1.54	1.51	1.54	1.55
A_5	7, 8, 10	3.47	3.48	3.47	3.47
A_6	9	2.59	2.63	2.59	2.59
A_7	11, 12, 15	1.81	1.80	1.81	1.81
A_8	13, 16	0.10	0.11	0.10	0.10
P_f		10^{-5}	9.96×10^{-6}	9.96×10^{-6}	9.92×10^{-6}
Weight (kg)		15.23	15.24	15.23	15.23
Number of structural analyses		5000	5000	5000	5000

Table 5.3 Comparison of optimization results for the 16-member truss (buckling is included)

Design variable: A_i (cm^2)		OST-TLBO	STMP-TLBO	STMP-SSOA	ST-JA
Group number	Members of the group				
A_1	1	2.41	3.20	3.13	3.13
A_2	2, 5	12.66	12.11	12.09	12.09
A_3	3, 4, 14	11.26	11.24	11.28	11.29
A_4	6	5.96	0.15	0.10	0.10
A_5	7, 8, 10	7.16	6.70	6.62	6.61
A_6	9	11.33	14.36	14.28	14.25
A_7	11, 12, 15	1.87	1.96	1.94	1.94
A_8	13, 16	0.11	0.17	0.10	0.10
P_f		9.94×10^{-6}	9.94×10^{-6}	9.94×10^{-6}	9.99×10^{-6}
Weight (kg)		35.67	34.66	34.49	34.49
Number of structural analyses		5000	5000	5000	5000

structure, the structural members are categorized into eight different groups of design variables. Table 5.4 summarizes the material properties, design variables bounds, and statistical data of random variables of the truss bridge. The target failure probability of the structure is set to be 10^{-5} . This problem was previously studied by Nakib [9] based on a deterministic design approach. Table 5.5 compares the optimum designs of the 65-member truss bridge obtained by various set-theoretical optimization algorithms. As can be seen from the table, the optimum designs of the 65-member truss bridge are very close to each other in terms of structural weight and cross-sectional areas

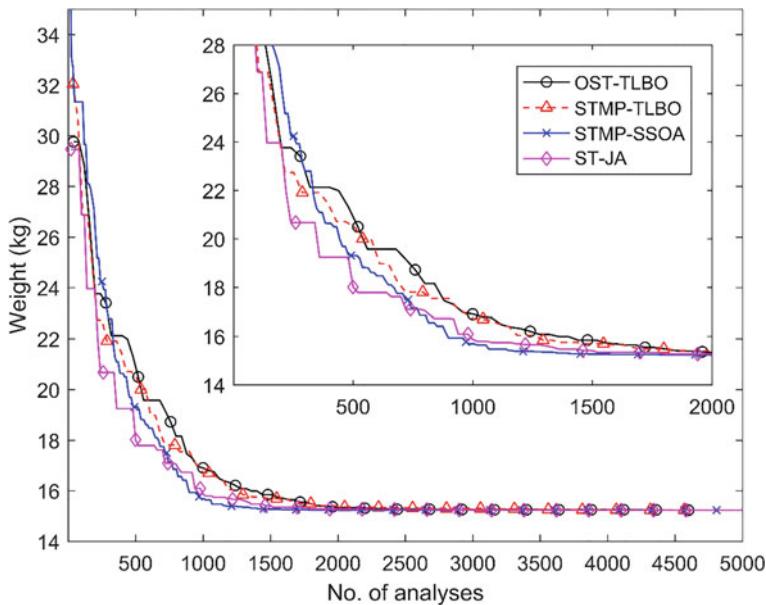


Fig. 5.6 Convergence histories for the 16-member truss (buckling is neglected)

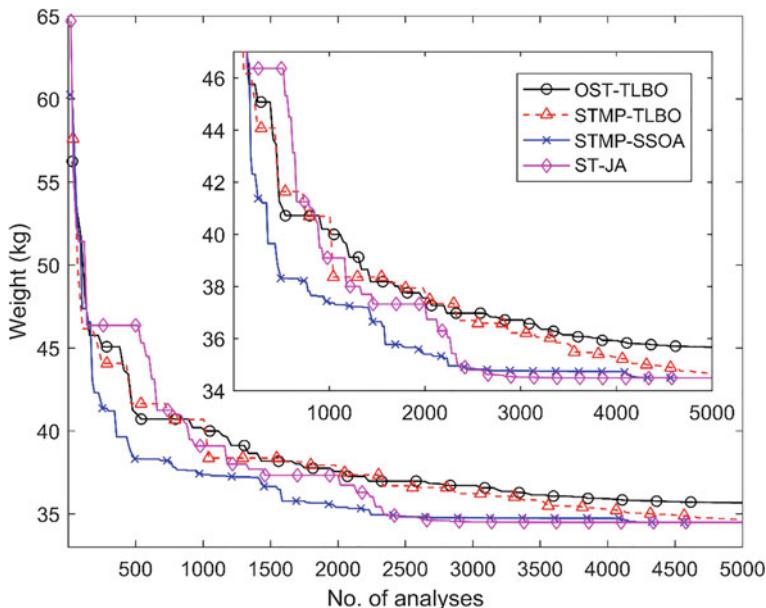


Fig. 5.7 Convergence histories for the 16-member truss (buckling is included)

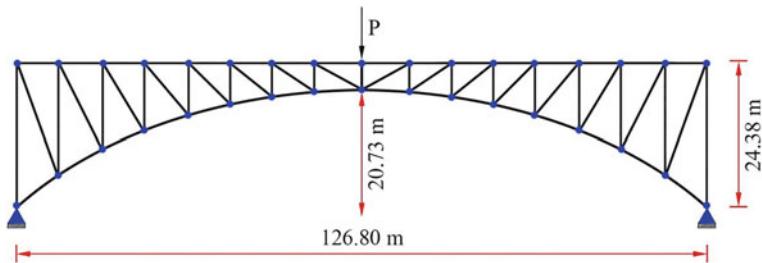


Fig. 5.8 Statically indeterminate 65-member truss bridge

Table 5.4 Data of the statically indeterminate 65-member truss bridge

Property/Unit	Value
E (Elasticity modulus)/GPa	199.95
ρ (Material density)/kg/m ³	7850.02
μ_{σ_y} (Mean value of yield stress)/MPA	248.21
CV_{σ_y} (Coefficient of variation of yield stress)	0.02
μ_P (Mean value of applied loads)/kN	133.36
CV_P (Coefficient of variation of applied loads)	0.05
CV_{w_0} (Coefficient of variation of initial deflection)	0.645
Lower bound of design variables/cm ²	32.258
Upper bound of design variables/cm ²	10^{-5}
P_f^t (Target failure probability)	199.95

of members. Figure 5.9 compares the convergence histories of the best solutions of the OST-TLBO, STMP-TLBO, STMP-SSOA, and ST-JA for the 65-member truss bridge. It is found that ST-JA converges significantly faster than the other considered set-theoretical metaheuristics in this problem.

5.5.3 *Statically Indeterminate 67-Member Truss Bridge*

Figure 5.10 shows a 67-member statically indeterminate truss bridge. As can be seen in the figure, this structure is formed by adding two redundant members to the 65-member truss bridge of the previous example. Thus, the resulting structure has three degrees of redundancy. This example investigates the effect of ductile redundant members added to the 65-member truss bridge. This is a size optimization problem with nine design variables. According to the symmetry of the structure, eight independent design variables are considered (please see Table 5.6). The numerical data are the same as those of the previous example (please see Table 5.4). Table 5.6 presents the optimization results of the 67-member statically indeterminate truss

Table 5.5 Comparison of optimization results for the 65-member truss bridge

Design variable: A_i (cm^2)		Reliability-based approach				Deterministic approach Nakib [9]
		OST-TLBO	STMP-TLBO	STMP-SSOA	ST-JA	
Group number	Members of the group					
A_1	2, 6, 10, 14, 50, 54, 58, 62	8.35	8.35	8.34	8.35	6.53
A_2	18, 22, 26, 30, 34, 38, 42, 46	7.24	7.26	7.24	7.24	5.57
A_3	4, 8, 12, 16, 52, 56, 60, 64	0.65	0.65	0.65	0.65	0.65
A_4	20, 24, 28, 32, 36, 40, 44, 48	19.07	19.04	19.09	19.07	16.21
A_5	1, 5, 9, 13, 53, 57, 61, 65	0.80	0.80	0.80	0.80	0.88
A_6	17, 21, 25, 29, 33, 37, 41, 45, 49	6.80	6.80	6.81	6.80	5.37
A_7	3, 7, 11, 15, 51, 55, 59, 63	0.89	0.89	0.89	0.89	0.95
A_8	19, 23, 27, 31, 35, 39, 43, 47	8.71	8.72	8.71	8.71	6.83
P_f		10^{-5}	10^{-5}	10^{-5}	10^{-5}	N/A
Weight (kg)		2794.94	2795.23	2794.98	2794.87	2286.16
Number of structural analyses		6000	6000	6000	6000	N/A

bridge. The optimum weights found by OST-TLBO, STMP-TLBO, STMP-SSOA, and ST-JA are 2368.29 kg, 2373.39 kg, 2378.60 kg, and 2368.03 kg, respectively, which are 15.27%, 15.09%, 14.90%, and 15.27% less than those of the 65-member truss bridge, respectively. The results prove that the addition of ductile redundant members to the 65-member truss bridge significantly decreases the optimized structural weight. This is due to the fact that the load carrying capacity of structures with a high degree of redundancy is often much larger than those with lower degree of redundancy. Figure 5.11 shows the convergence histories of the best solutions obtained for the 67-member truss bridge. As can be seen from the figure, ST-JA has

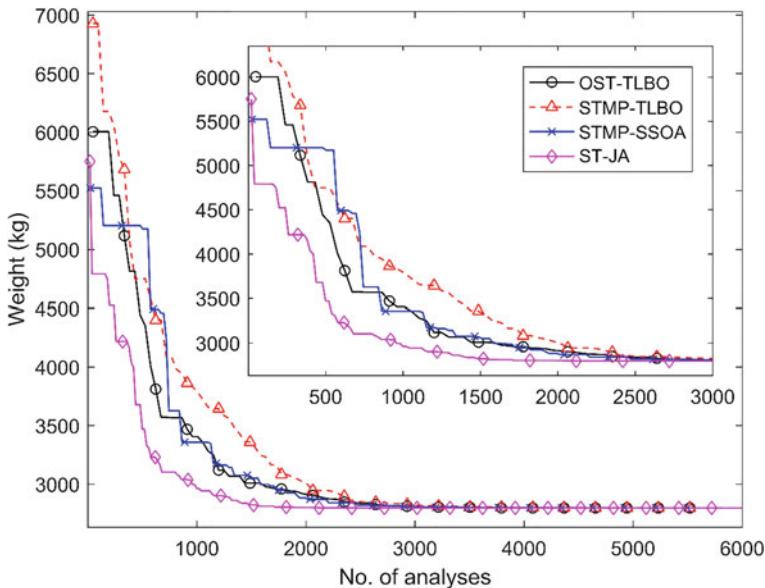


Fig. 5.9 Convergence histories for the 65-member truss bridge

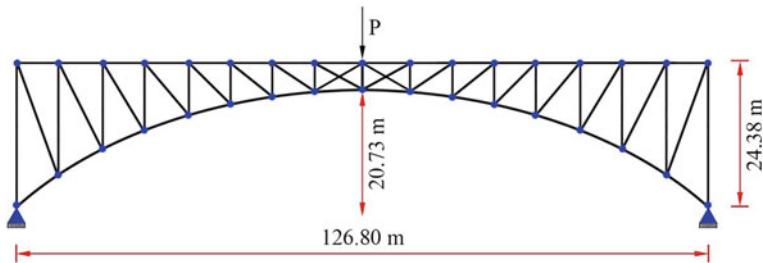


Fig. 5.10 Statically indeterminate 67-member truss bridge

better convergence performance compared to the other set-theoretical optimization algorithms in this problem.

5.6 Concluding Remarks

This study presents an approach to solve system reliability-based design optimization problem of truss structures. The proposed approach employs the branch and bound method to identify the dominant failure modes of truss structures. A general

Table 5.6 Comparison of optimization results for the 67-member truss bridge

Design variable: A_i (cm^2)		OST-TLBO	STMP-TLBO	STMP-SSOA	ST-JA
Group number	Members of the group				
A_1	2, 6, 10, 14, 50, 54, 58, 62	9.61	9.48	9.63	9.60
A_2	18, 22, 26, 30, 34, 38, 42, 46	9.11	8.93	9.08	9.13
A_3	4, 8, 12, 16, 52, 56, 60, 64	1.07	0.99	1.05	1.07
A_4	20, 24, 28, 32, 36, 40, 44, 48	9.17	9.82	9.16	9.16
A_5	1, 5, 9, 13, 53, 57, 61, 65	1.12	1.07	1.11	1.12
A_6	17, 21, 25, 29, 33, 37, 41, 45, 49	2.91	2.97	2.99	2.91
A_7	3, 7, 11, 15, 51, 55, 59, 63	1.27	1.22	1.30	1.27
A_8	19, 23, 27, 31, 35, 39, 43, 47	6.32	6.44	6.37	6.32
A_9	66, 67	7.96	7.61	8.28	7.97
P_f		9.96×10^{-6}	9.85×10^{-6}	9.68×10^{-6}	10^{-5}
Weight (kg)		2368.29	2373.39	2378.60	2368.03
Number of structural analyses		6000	6000	6000	6000

set-theoretical framework, which was recently proposed for population-based metaheuristic algorithms, is also employed for optimization. The main idea of this framework is to divide the population of solutions into smaller well-arranged subpopulations of the same size. Therefore, several subpopulations explore the search space. As a result, the diversity of the population is maintained during the search process, and thus the ability to escape from local optima is improved. Set-theoretical variants of three population-based metaheuristic algorithms including TLBO, SSOA, and JA are employed to solve the optimization problems considered in this study. The performance of the proposed SRBDO approach is validated through three truss optimization problems with system reliability constraints. The results show the effectiveness and applicability of set-theoretical framework for the optimum design of truss structures subject to system reliability constraints. A major advantage of the proposed SRBDO approach is that, due to the use of a systems approach for evaluation of structural reliability, a more accurate estimation of the load-carrying capacity of truss structures is made, and thus more realistic optimization results can be obtained. This would effectively help to strike a suitable balance between minimum weight and target safety.

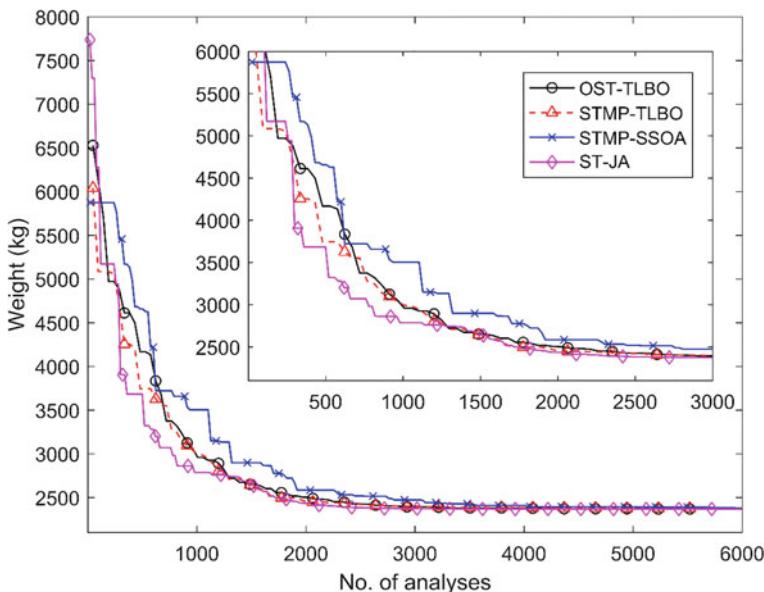


Fig. 5.11 Convergence histories for the 67-member truss bridge

References

1. Kaveh A, Biabani Hamedani K, Kamalinejad M (2021) Set theoretical variants of optimization algorithms for system reliability-based design of truss structures. Period Polytech Civ Eng 65(3):717–729. <https://doi.org/10.3311/PPci.17519>
2. Thoft-Christensen P, Murotsu Y (2012) Application of structural systems reliability theory. Springer Science & Business Media
3. Dhingra AK, Bennage WA (1995) Topological optimization of truss structures using simulated annealing. Eng Optim 24(4):239–259. <https://doi.org/10.1080/03052159508941192>
4. Hasançebi O, Kazemzadeh Azad S (2014) Discrete size optimization of steel trusses using a refined big bang–big crunch algorithm. Eng Optim 46(1):61–83. <https://doi.org/10.1080/0305215X.2012.748047>
5. Kaveh A, Biabani Hamedani K, Kamalinejad M (2020) Set theoretical variants of the teaching–learning-based optimization algorithm for optimal design of truss structures with multiple frequency constraints. Acta Mech 231(9):3645–3672. <https://doi.org/10.1007/s00707-020-02718-3>
6. Kaveh A, Kamalinejad M, Biabani Hamedani K (2021) Enhanced versions of the shuffled shepherd optimization algorithm for optimal design of skeletal structures. Structures 29:1463–1495. <https://doi.org/10.1016/j.istruc.2020.12.032>
7. Movahedi Rad M, Lógo J (2011) Plastic behavior and stability constraints in the reliability based shakedown analysis and optimal design of skeletal structures. Asian J Civ Eng 12(4):395–413. <https://doi.org/10.4203/ccp.93.203>
8. Murotsu Y, Shao S (1990) Optimum shape design of truss structures based on reliability. Struct Optim 2:65–76. <https://doi.org/10.1007/BF01745455>
9. Nakib R (1997) Deterministic and reliability-based optimization of truss bridges. Comput Struct 65(5):767–775. [https://doi.org/10.1016/S0045-7949\(94\)E0289-E](https://doi.org/10.1016/S0045-7949(94)E0289-E)

10. Stocki R, Kolanek K, Jendo S, Kleiber M (2001) Study on discrete optimization techniques in reliability-based optimization of truss structures. *Comput Struct* 79(22–25):2235–2247. [https://doi.org/10.1016/S0045-7949\(01\)00080-3](https://doi.org/10.1016/S0045-7949(01)00080-3)
11. Burton S, Hajela P (2001) Reliability-based shape optimization of truss structures. In: 19th AIAA applied aerodynamics conference June 11, p 1681. <https://doi.org/10.2514/6.2001-1681>
12. Park S, Choi S, Sikorsky C, Stubbs N (2004) Efficient method for calculation of system reliability of a complex structure. *Int J Solids Struct* 41(18–19):5035–5050. <https://doi.org/10.1016/j.ijsolstr.2004.04.028>
13. Tang LK, Melchers RE (1987) Dominant mechanisms in stochastic plastic frames. *Reliab Eng* 18(2):101–115. [https://doi.org/10.1016/0143-8174\(87\)90025-4](https://doi.org/10.1016/0143-8174(87)90025-4)
14. Kaveh A, Biabani Hamedani K, Zaerreza A (2021) A set theoretical shuffled shepherd optimization algorithm for optimal design of cantilever retaining wall structures. *Eng Comput* 37:3265–3282. <https://doi.org/10.1007/s00366-020-00999-9>
15. Kaveh A (1992) Structural mechanics: graph and matrix methods. Macmillan International Higher Education
16. Rao RV, Savsani VJ, Vakharia DP (2011) Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems. *Comput Aided Des* 43:303–315. <https://doi.org/10.1016/j.cad.2010.12.015>
17. Kaveh A, Zaerreza A (2020) Shuffled shepherd optimization method: a new meta-heuristic algorithm. *Eng Comput* 37(7):2357–2389. <https://doi.org/10.1108/EC-10-2019-0481>
18. Rao RV (2016) Jaya: a simple and new optimization algorithm for solving constrained and unconstrained optimization problems. *Int J Ind Eng Comput* 7(1):19–34. <https://doi.org/10.5267/j.ijiec.2015.8.004>
19. Wang H, Yu W, Chen G (2017) An approach of topology optimization of multi-rigid-body mechanism. *Comput Aided Des* 84:39–55. <https://doi.org/10.1016/j.cad.2016.12.002>
20. Murotsu Y, Okada H, Niwa K, Miwa S (1979) A new method for evaluating lower and upper bounds of failure probability in redundant truss structures. *Bull Univ Osaka Prefect, Ser A, Eng Nat Sci* 28(1):79–91. <https://doi.org/10.24729/00008657>
21. Kaveh A, Javadi SM, Mahdipour Moghanni R (2020) Reliability analysis via an optimal covariance matrix adaptation evolution strategy: emphasis on applications in civil engineering. *Period Polytech Civ Eng* 64(2):579–588. <https://doi.org/10.3311/PPci.15793>
22. Nowak AS (2004) System reliability models for bridge structures. *Bull Pol Ac Tech* 52(4):321–328

Chapter 6

Optimal Analysis in the Service of Frequency-Constrained Structural Optimization with Set-Theoretical Jaya Algorithm



6.1 Introduction

In this chapter, an efficient eigensolution method for free vibration analysis of rotationally repetitive structures is employed for optimal design of cyclic symmetric dome structures with multiple frequency constraints [1]. By means of the employed eigensolution method, the initial free-vibration eigenproblem is decomposed into some sub-eigenproblems with significantly smaller dimensions using an efficient block-diagonalization technique, and thus the computational time of structural analysis is reduced. The optimization process is carried out by a set-theoretical Jaya algorithm. In order to demonstrate the efficiency and effectiveness of the proposed method, two large-scale frequency-constrained dome optimization examples are studied.

The dynamic characteristics (i.e., natural vibration frequencies and mode shapes) are arguably the single most important property of a mechanical system affecting the dynamic behavior of the system. For example, in a mechanical system with low-frequency vibrations, the dynamic response of the system is mainly a function of its fundamental frequency. In such cases, the performance of the structure can be considerably improved by manipulating the selected frequency. Structural optimization considering frequency constraints provides a systematic design approach for engineering designers to manipulate the dynamic characteristics of the structural systems in various ways. For example, in designing most space vehicles, to avoid the resonance phenomenon that causes the vibration failure, it is necessary to impose constraints on the natural frequency ranges of the designed vehicles. Since the early 1980s, some researchers have applied gradient-based optimization methods to the optimal design of structures with frequency constraints [2, 3]. However, the structural optimization problems with frequency constraints are considered challenging optimization problems with highly nonlinear, non-convex, and multimodal search spaces [4]. Thus, gradient-based optimization methods may not be suitable

for this type of optimization problem. On the other hand, metaheuristic optimization algorithms could be considered appropriate alternatives [5–13].

Metaheuristic optimization algorithms have been one of the most popular research areas in computer science for more than three decades. These approximate optimization techniques have been extensively applied to a wide variety of engineering optimization problems. This is because metaheuristics: (1) are easy to design and implement as compared to other optimization methods, (2) require no gradient information during the search, and (3) are not problem-specific. The main challenge in designing a metaheuristic is to provide a proper balance between global diversification and local intensification during the search process. Aiming for this purpose, Kaveh et al. [4] have recently proposed a general set-theoretical framework for population-based metaheuristic optimization algorithms and applied it to some metaheuristics, including teaching–learning-based optimization [4] and Jaya algorithm [14]. The main idea of this framework is to divide the population of solutions into a predefined number of well-arranged subpopulations of the same size. This set-theoretical framework causes vast exploration of the search space (diversification) in the early stages of the search and exploitation of its promising regions (intensification) in the final stages of the search.

A significant drawback of metaheuristics is their relatively high computational costs, especially when applied to large-scale optimization problem. When applied to structural optimization problems, metaheuristic algorithms usually require a large number of function evaluations (i.e., numerous structural analyses) to be performed to find optimal or near-optimal designs. Thus, it may be very time-consuming or even impractical to solve large-scale structural optimization problems using metaheuristic algorithms. However, optimal structural analysis provides efficient and practical methods of structural analysis which can be employed as powerful tools to reduce significantly the computational time of structural analyses performed in the structural optimization process. For example, in structural optimization problems with frequency constraints, the structural analyses involve relatively large generalized eigenproblems to find the vibration characteristics of structures [6]. The dimensions of the involved matrices in the free-vibration eigenproblem of a structural system are proportional to the number of degrees of freedom. This indicates that the required computational time of the frequency-constrained optimization problems depends strongly on the size of the structure. In such circumstances, efficient eigensolution methods, which require less computational effort to solve a single free-vibration eigenproblem, could be very beneficial for frequency-constrained structural optimization problems, especially in the case of large-scale structures.

The models of many structures have specific characteristics which may provide significant advantages in the eigensolution of matrices involved in static, dynamic, and buckling analyses of such structures [15]. Symmetry is one of these characteristics which is widely used in a variety of structures. In the development of the model of a symmetric structure, due to the symmetry of the geometry and loading conditions, only a segment of the structure, such as a half or quarter of the structure, needs to be modeled. This may strongly reduce the computational time required for the analysis. It should be pointed out that appropriate boundary conditions must be imposed at the

interfaces between identical segments, to ensure continuity of the full structure. For many years, several researchers have focused on the analysis of structures by using concepts of symmetry and regularity. Renton [16], who was one of the first researchers to apply the concept of symmetry to structural analysis, investigated the stability analysis of symmetric frameworks. Hussey [17] studied the buckling under cyclic symmetric loading. Many researchers have also investigated the application of group theory to vibration eigenvalue analysis in engineering [18, 19]. The group-theoretic approach provides a systematic way to describe the full symmetry of any structure, and the extension to group representation theory allows for the maximum utilization of a structure's symmetry properties. In other words, the group-theoretic approach is of a general nature, meaning that it has the capability of taking into account all the symmetry properties of a configuration in a systematic manner. Through the group-theoretic approach, much of the behaviour of a symmetric physical system can be explained in terms of the intrinsic properties of the associated symmetry group and its representations. Summarizing, if a physical system possesses symmetry properties which can be described by a group, decomposition of the vector space of the system results in a block-diagonal form of the matrix of equations describing the behaviour of the system, allowing separation into independent sets of equations each corresponding to a particular subspace. Since the mid-2000s, Kaveh and his students have worked extensively on developing efficient graph-theoretic methods for eigenvalue problems of symmetric, repetitive, and regular structures [20, 21]. In the cases where engineering structures exhibit small deviations from perfect symmetry, both the group-theoretic and graph-theoretic approaches can still be employed to simplify the analysis. It should be noted that graph-theoretic methods cannot take into account the types of symmetry for which they are not suited, and in the case of structural configurations possessing complex or mixed symmetry properties, only a portion of the total symmetry can be exploited.

A large number of complex structural models conform to cyclic symmetry. Such structures, known as cyclic symmetric or cyclic repetitive structures, could be viewed as the cyclic repetition of a sub-structure around the cyclic symmetry axis of the structure [15]. Some examples of cyclic symmetric structures include domes, cooling towers, chimneys, etc. The structural matrices associated with cyclic symmetric structures can be arranged into a unique canonical form [15]. Several researchers have focused on the potential advantage of this canonical form in the eigenvalue problems of cyclic symmetric structures [20, 21]. Kaveh and Rahmani proved that all the canonical forms for bilaterally symmetric structures are a special form of cyclic symmetric structures.

The main objective of this study is to provide a fast and robust method for the optimal design of cyclic symmetric dome structures with multiple frequency constraints by integrating an efficient eigensolution method for free vibration analysis of rotationally repetitive structures and a set-theoretical Jaya algorithm (ST-JA) for optimization. Through the present eigensolution method, the initial free-vibration eigenproblem is decomposed into some sub-eigenproblems with significantly smaller dimensions using an efficient block-diagonalization technique. As a result, the computational time and memory required for the optimization process

are considerably reduced. The efficiency and effectiveness of the proposed method are examined through two large-scale frequency-constrained dome optimization examples.

The rest of this chapter is organized as follows: In Sect. 6.2, the free vibration analysis of structures is presented. Section 6.3 introduces an efficient eigensolution method for free vibration analysis of rotationally repetitive structures. Section 6.4 presents the mathematical formulation of the optimization problem. In Sect. 6.5, after reviewing the original JA, the set-theoretical JA is outlined. In Sect. 6.6, two large-scale cyclic symmetric dome structures are optimized to demonstrate the effectiveness and computational efficiency of the proposed method. Finally, Sect. 6.7 draws concluding remarks and provides possible extensions of this study.

6.2 Free Vibration Analysis of Structures

Free vibration means the motion of a structure without any externally applied vibration forcing. Vibration characteristics (i.e., natural vibration frequencies and mode shapes) play an essential role in the dynamic analysis of structures. Determining the vibration characteristics of an undamped structure requires the solution of the following algebraic equation, known as the matrix eigenvalue problem:

$$K\phi_i = \gamma_i M\phi_i; i = 1, 2, \dots, N \quad (6.1)$$

where K is the elastic stiffness matrix of the structure (hereafter called stiffness matrix of the structure), M is the mass matrix of the structure, which is a linear combination of structural and non-structural mass matrices, ϕ_i is the i th natural mode shape of vibration of the structure corresponding to the i th eigenvalue (γ_i), and N is the number of degrees of freedom of the structure. The i th natural frequency of vibration (ω_i) and its corresponding natural period of vibration (T_i) are related to the i th eigenvalue through the following equation:

$$\gamma_i = \omega_i^2 = (2\pi/T_i)^2; i = 1, 2, \dots, N \quad (6.2)$$

There are many different methods to solve eigenvalue problems. However, it should be noted that there is no single method that can always give a very efficient solution to every eigenvalue problem. Existing classical eigensolution methods do not take advantage of the potentially beneficial properties of the matrices involved in eigenvalue problems (i.e., the matrices K and M) and thus deal with large-dimension matrices. As a result, the required computational effort of these methods depends strongly on the size of the structure (i.e., the number of degrees of freedom) [6]. Hence, it is time-consuming and inefficient to solve the generalized eigenvalue problem given by Eq. (6.1) with existing classical eigensolution methods, especially for large-scale structures. However, in the case of general structures, it is inevitable

to use classical eigensolution methods. On the other hand, the design optimization process of a structure with frequency constraints usually requires many free vibration analyses to be carried out. As mentioned before, the mathematical formulation of free vibration analysis of structures leads to the generalized eigenvalue problem given by Eq. (6.1). The most time-consuming part of frequency constraint optimization problems is usually the solution of eigenvalue problems. As a result, design optimization of large-scale structures with frequency constraints could not be performed using the existing classical eigensolution methods in a reasonable time. Consequently, alternative efficient eigensolution methods, which take the maximum advantage of the properties of the matrices involved in eigenvalue problems to decrease the required computational time and memory, should be considered. In the next section, based on the works of Kaveh et al. [22–24], an efficient eigensolution method for free vibration analysis of cyclic symmetric structures is presented.

6.3 Efficient Free Vibration Analysis of Cyclic Symmetric Structures

In the following, the stiffness matrix and mass matrix of a three-dimensional truss element are expressed in the Cartesian coordinate system. Next, these matrices are transformed into the cylindrical coordinate system. In the cylindrical coordinate system, the global stiffness matrix and mass matrix of a cyclic symmetric structure exhibit a unique pattern known as block circulant [15]. Circulant matrices can be expressed as the sum of Kronecker products in which the first components satisfy the commutative property of multiplication [22]. This property facilitates the block diagonalization of circulant matrices. Therefore, using this property of block circulant matrices, the initial generalized eigenvalue problem, derived from the free vibration analysis, is decomposed physically into highly smaller sub-eigenproblems [24]. This approach leads to not only the high accuracy of the free vibration analysis results but also a significant decrease in computational time and memory usage as compared to the existing classical eigenvalue solutions [15]. However, this block-diagonalization technique is suitable only for cyclic symmetric configurations. In other words, in the case of structural configurations possessing complex or mixed symmetry properties, only the portion of the total symmetry exhibits cyclic symmetry can be exploited by the present block-diagonalization technique. On the other hand, the group-theoretic approach provides reduction in computational effort in a rather different manner than the present diagonalization technique. There is no physical sub-structuring. Through the group-theoretic approach, the vector space of the full problem is decomposed into a number of smaller group-invariant subspaces spanned by symmetry-adapted variables as basis vectors, thus allowing calculations of quantities of interest within the independent subspaces [19]. However, in the case of cyclic symmetric configurations, the graph-theoretic approach is simpler than the group-theoretic approach.

6.3.1 Structural Matrices in the Cartesian and Cylindrical Coordinate Systems

Figure 6.1 shows a three-dimensional truss element and the corresponding displacement components in the global Cartesian coordinate system. The element stiffness matrix in the Cartesian coordinate system can be written as [24]:

$$K_{ij}^{xyz} = \frac{E_{ij} A_{ij}}{L_{ij}} \begin{bmatrix} d_{ij} & -d_{ij} \\ -d_{ij} & d_{ij} \end{bmatrix}, d_{ij} = \begin{bmatrix} l_{ij}^2 & l_{ij}m_{ij} & l_{ij}n_{ij} \\ l_{ij}m_{ij} & m_{ij}^2 & m_{ij}n_{ij} \\ l_{ij}n_{ij} & m_{ij}n_{ij} & n_{ij}^2 \end{bmatrix} \quad (6.3)$$

where K_{ij}^{xyz} is the stiffness matrix of element ij in the Cartesian coordinate system, and E_{ij} , A_{ij} , and L_{ij} are the modulus of elasticity, cross-sectional area, and length of element ij , respectively. In the sub-matrix d_{ij} , l_{ij} , m_{ij} , and n_{ij} are the direction

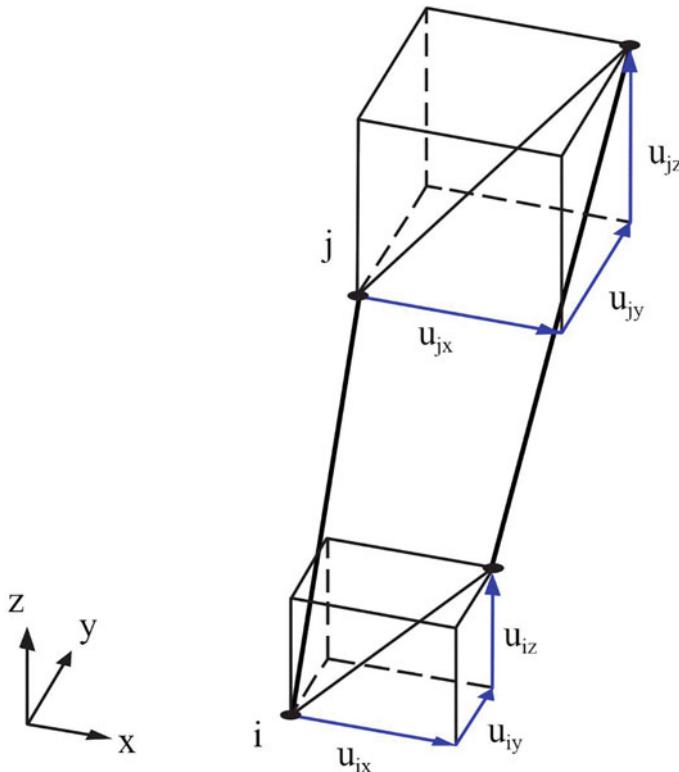


Fig. 6.1 A three-dimensional truss element in the global Cartesian coordinate system

cosines of the unit vector of the axis of element ij with respect to the global axes x , y , and z , respectively, and are defined as:

$$l_{ij} = \frac{x_j - x_i}{L_{ij}}, m_{ij} = \frac{y_j - y_i}{L_{ij}}, n_{ij} = \frac{z_j - z_i}{L_{ij}} \quad (6.4)$$

As can be seen from Eq. (6.3), the element stiffness matrix in Cartesian coordinates is not invariant against rotation about any axis; thus, the global stiffness matrix of a cyclic symmetric structure in the Cartesian coordinate system does not generally exhibit any unique pattern [24].

As mentioned before, the structural model includes both structural and non-structural masses. In other words, the mass matrix of element ij is a linear combination of structural and non-structural mass matrices. The structural mass matrix depends on whether a consistent mass or a lumped mass model is used. In the case of a truss element, a consistent mass model assumes a continuous distribution of the mass on the element, and a lumped mass model lumps the mass at the nodes of the element. If the density and cross-sectional area of element ij are constant, the structural consistent mass matrix of the element in the Cartesian coordinate system is given by:

$$M_{ij(c)}^{xyz} = \frac{\rho_{ij} A_{ij} L_{ij}}{6} \begin{bmatrix} 2 & 0 & 0 & 1 & 0 & 0 \\ 0 & 2 & 0 & 0 & 1 & 0 \\ 0 & 0 & 2 & 0 & 0 & 1 \\ 1 & 0 & 0 & 2 & 0 & 0 \\ 0 & 1 & 0 & 0 & 2 & 0 \\ 0 & 0 & 1 & 0 & 0 & 2 \end{bmatrix} \quad (6.5)$$

where $M_{ij(c)}^{xyz}$ is the structural consistent mass matrix of element ij in the global Cartesian coordinate system and ρ_{ij} is the material density of element ij .

In a lumped mass model, it is assumed that the total mass of the element is distributed equally between the nodes of the element, and the masses are associated with translational degrees of freedom along the three axes X , Y , and Z . The structural lumped mass matrix of a three-dimensional truss element is given by:

$$M_{ij(l)}^{xyz} = \frac{\rho_{ij} A_{ij} L_{ij}}{2} \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (6.6)$$

where $M_{ij(l)}^{xyz}$ is the structural lumped mass matrix of element ij in the global Cartesian coordinate system.

It is worth mentioning that if small but massive objects are lumped at the nodes of a lightweight structures, the lumped mass model will result in nearly exact solutions of free vibration analysis. On the other hand, provided that the displacement shape functions are constructed using the actual deformed shape (under dynamic conditions), the consistent mass model will lead to exact results. Since the static displacement distributions are often used for the displacement shape functions, the resulting mass distribution will only be approximate; however, the accuracy is still sufficient for most practical applications. Optionally, additional non-structural lumped masses can be attached to specific nodes of the truss structure. Such non-structural lumped masses can also be modeled by using the lumped mass model. It is worth mentioning that both the consistent mass and lumped mass matrices are invariant against rotation (see Eqs. (6.5) and (6.6), respectively). It should be noted that in this study, since the density and cross-sectional area of elements are constant, the consistent mass model is used to form the structural mass matrix. In addition, as suggested above, non-structural masses attached to specific nodes of the structure are modeled by using the lumped mass model.

Figure 6.2 shows a three-dimensional truss element and the corresponding displacement components in the cylindrical coordinate system. The element stiffness matrix in the cylindrical coordinate system ($K_{ij}^{r\theta z}$) is related to the element stiffness matrix in the Cartesian coordinate system (K_{ij}^{xyz}) by a simple transformation as follows [6]:

$$K_{ij}^{r\theta z} = R^t K_{ij}^{xyz} R \quad (6.7)$$

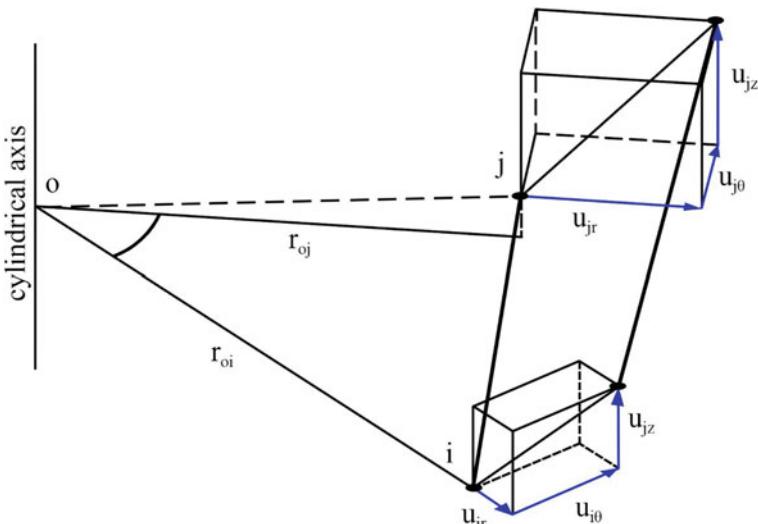


Fig. 6.2 A three-dimensional truss element in the global Cartesian coordinate system

where R is the transformation matrix between the Cartesian coordinates and cylindrical coordinates and superscript t denotes the transpose of a matrix. The matrix R is given by:

$$R = \begin{bmatrix} R_{oi} & 0 \\ 0 & R_{oj} \end{bmatrix} \quad (6.8)$$

where the sub-matrices R_{oi} and R_{oj} are:

$$R_{oi} = \begin{bmatrix} l_{oi} & -m_{oi} & 0 \\ m_{oi} & l_{oi} & 0 \\ 0 & 0 & 1 \end{bmatrix}, R_{oj} = \begin{bmatrix} l_{oj} & -m_{oj} & 0 \\ m_{oj} & l_{oj} & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (6.9)$$

If the Cartesian coordinates of point “o” are taken as $x_o = y_o = 0$, it can be shown that:

$$l_{oi} = \frac{x_i}{r_{oi}}, m_{oi} = \frac{y_i}{r_{oi}}, l_{oj} = \frac{x_j}{r_{oj}}, m_{oj} = \frac{y_j}{r_{oj}} \quad (6.10)$$

where

$$r_{oi} = \sqrt{x_i^2 + y_i^2}, r_{oj} = \sqrt{x_j^2 + y_j^2} \quad (6.11)$$

The expanded form of the element stiffness matrix in the cylindrical coordinate system is as follows [24]:

$$K_{ij}^{rz\theta} = \frac{E_{ij} A_{ij}}{L_{ij}} \begin{bmatrix} s_1^2 & -s_1s_2 & s_1n_{ij} & -s_1s_3 & s_1s_4 & -s_1n_{ij} \\ s_2^2 & -s_2s_1 & s_2n_{ij} & -s_2s_3 & s_2s_4 & -s_2n_{ij} \\ n_{ij}^2 & -s_3n_{ij} & s_3n_{ij} & -s_2s_4 & s_2n_{ij} & -n_{ij}^2 \\ s_3^2 & -s_3s_4 & s_3n_{ij} & s_3s_4 & -s_3n_{ij} & s_3n_{ij} \\ sym & & & s_4^2 & -s_4n_{ij} & n_{ij}^2 \\ & & & & n_{ij}^2 & \end{bmatrix} \quad (6.12)$$

where

$$s_1 = l_{ij}l_{oi} + m_{ij}m_{oi}s_2 = l_{ij}m_{oi} - m_{ij}l_{oi}s_3 = l_{ij}l_{oj} + m_{ij}m_{oj}s_4 = l_{ij}m_{oj} - m_{ij}l_{oj} \quad (6.13)$$

As can be seen from Eq. (6.12), the element stiffness matrix in the cylindrical coordinate system is invariant against rotation about the cylindrical axis. As a result, all similar members of a cyclic symmetric structure, regardless of their rotation about the cyclic symmetry axis of the structure, have the same stiffness matrix in the cylindrical coordinate system [24]. This is also true for all similar sub-structures

of a cyclic symmetric structure. However, it should be noted that the pattern of the stiffness matrix is dependent on the nodal numbering scheme employed. Indeed, if an identical nodal numbering scheme is applied to all similar sub-structures of a cyclic symmetric structure, regardless of their rotation about the cyclic symmetry axis of the structure, they have the same stiffness matrix. Therefore, the stiffness matrix of a cyclic symmetric structure conforms to a unique pattern, known as block circulant. Block circulant matrices have a high potential to be used as an efficient tool for solving eigenproblems. Through a similar transformation process, the element mass matrix in the cylindrical coordinate system can be derived.

6.3.2 Efficient Eigensolution Method for Free Vibration Analysis of Cyclic Symmetric Structures

As mentioned before, for a three-dimensional truss element in a cylindrical coordinate system, the stiffness and mass matrices are invariant against rotation about the cylindrical axis. Consequently, in the cylindrical coordinate system, all similar sub-structures of a cyclic symmetric structure, regardless of their rotation about the cyclic symmetry axis of the structure, have the same stiffness and mass matrices. Thus, the global stiffness and mass matrices of a cyclic symmetric structure conform to a canonical form, known as block tri-diagonal matrix with corner blocks (BTMCB) form, as shown in Eq. (6.14) [15]. Moreover, it should be pointed out that in order to form such a pattern, the following requirements must also be considered: (1) the boundary conditions must be cyclic symmetric, and (2) the nodes of all similar sub-structures must be numbered in a similar manner [24].

$$\begin{bmatrix} A & B & & B^t \\ B^t & A & B & \\ & B^t & A & B \\ & & \ddots & \\ & & & \ddots & \\ & & & & \ddots & \\ & & & B^t & A & B \\ & & & & B^t & A & B \\ & & & & & B^t & A \end{bmatrix} \quad (6.14)$$

Note again that a cyclic symmetric structure could be viewed as the cyclic repetition of a sub-structure around the cyclic symmetry axis of the structure. To determine the dimensions of the matrix above, let us consider a cyclic symmetric three-dimensional truss structure composed of n similar sub-structures, each of which has m nodes. The global stiffness matrix of such structure is a $3 \times 3mn$ square matrix with the canonical form of Eq. (6.14). The sub-matrices A , B , and B^t are also square matrices of order $3m$. Since the boundary conditions are also cyclic symmetric, the

canonical form of Eq. (6.14) will be preserved after applying the boundary conditions. All related matrices of a cyclic symmetric structure conform to the canonical form of Eq. (6.14); thus, the mass matrix M and the stiffness matrix K can be expressed as the sum of three Kronecker products, as follows [24]:

$$K_{(3mn \times 3mn)} = I_{n \times n} \otimes A_{K(3m \times 3m)} + H_{n \times n} \otimes B_{K(3m \times 3m)} + H_{n \times n}^t \otimes B_{K(3m \times 3m)}^t \quad (6.15)$$

$$M_{(3mn \times 3mn)} = I_{n \times n} \otimes A_{M(3m \times 3m)} + H_{n \times n} \otimes B_{M(3m \times 3m)} + H_{n \times n}^t \otimes B_{M(3m \times 3m)}^t \quad (6.16)$$

where \otimes denotes Kronecker product, subscripts K and M in A , B , and B^t denote the stiffness and mass matrices, respectively, I is an identity matrix of order n , and H is an $n \times n$ unsymmetric permutation matrix:

$$H = \begin{bmatrix} 0 & 1 & & & & 0 \\ 0 & 0 & 1 & & & \\ & 0 & 0 & 1 & & \\ & & & \ddots & & \\ & & & & \ddots & \\ & & & & & \ddots \\ & & & & 0 & 0 & 1 \\ & & & & & 0 & 0 & 1 \\ 1 & & & & & & & 0 & 0 \end{bmatrix} \quad (6.17)$$

As seen from Eqs. (6.14) and (6.17), the matrix H can be obtained by substituting $A = 0$, $B = 1$, and $B^t = 0$ in Eq. (6.14).

In the following, block diagonalization of matrices with the BTMCB form is summarized. This problem was previously studied by Kaveh and Kohestani [23] and Kohestani and Kaveh [24]. The formal solution of Eq. (6.1) can be written in the form of a set of N homogeneous algebraic equations as:

$$\Omega_i = [K - \omega_i^2 M] \phi_i = 0 \quad (6.18)$$

where N is the number of degrees of freedom. This set always has the trivial solution $\phi_i = 0$, which implies no motion. Also, the set has a non-trivial solution if and only if the determinant of Ω_i is equal to zero:

$$\det[\Omega_i] = \det[K - \omega_i^2 M] \phi_i = 0 \quad (6.19)$$

where \det stands for determinant.

The problem is to find the eigenvalues of Ω_i . Equation (6.19) is known as the frequency equation or characteristics equation. Since both the mass matrix M and stiffness matrix K conform to the BTMBCB form, the matrix $[K - \omega_i^2 M]$ also exhibits the BTMBCB form. The block diagonalization of Ω_i leads to an efficient eigensolution for free vibration analysis of cyclic symmetric structures. By block diagonalization of the matrix $[K - \omega_i^2 M]$, the initial eigenvalue problem is decomposed into some highly smaller sub-eigenproblems. Here, the eigenvalues of a $3mn \times 3mn$ matrix are obtained only by the solution of n different eigenvalue problems of order $3m$, and thus a substantial reduction in computational time and memory is achieved. Let us consider the following equations:

$$\begin{aligned} A &= A_K - \gamma_i A_M \\ B &= B_K - \gamma_i B_M \\ B^t &= B_K^t - \gamma_i B_M^t \end{aligned} \quad (6.20)$$

By considering Eqs. (6.15), (6.16), and (6.20), Ω_i can be expressed in the following form:

$$\Omega_i = I \otimes A + H \otimes B + H^t \otimes B^t \quad (6.21)$$

This form of Ω_i is block-diagonalizable. The j th diagonal block of Ω_i is obtained as follows:

$$\Omega_i^j = A + \lambda_j B + \overline{\lambda_j} B^t \quad (6.22)$$

where Ω_i^j is the j th diagonal block of Ω_i , λ_j is the j th root of the characteristic polynomial of the matrix H , and the bar sign in the third term denotes the conjugate of a complex number.

As mentioned before, H is an $n \times n$ permutation matrix with the characteristic polynomial $f(\lambda)$:

$$f(\lambda) = \lambda^n - 1 = 0 \quad (6.23)$$

In general, the characteristic polynomial $f(\lambda)$ has n real and complex roots. The $f(\lambda)$ has exactly two real roots (± 1) for any even n , and has only one real root (+1) whenever n is odd [23]. The complex roots of Eq. (6.23) can also easily be obtained. As can be seen from the equation, the absolute value of all the roots is unity. It can easily be shown that the roots of Eq. (6.23) are the roots of the following equation [23]:

$$\cos(n\theta) + i \sin(n\theta) = 1 \quad (6.24)$$

Table 6.1 Roots of the characteristic polynomial of the matrix H

	Real roots	Complex roots
n is even	$-1, 1$	$\cos(\theta) \pm i\sin(\theta); \theta = 2k\pi/n, k = 1, 2, \dots, (n-2)/2$
n is odd	1	$\cos(\theta) \pm i\sin(\theta); \theta = 2k\pi/n, k = 1, 2, \dots, (n-1)/2$

The complex and real roots of the characteristic polynomial $f(\lambda)$ are presented in Table 6.1.

The determinant of Ω_i could then be expressed as:

$$\det(\Omega_i) = \prod_{j=1}^n \Omega_i^j \quad (6.25)$$

This is due to the fact that the determinant of a block-diagonal matrix is the product of the determinants of the diagonal blocks. The determinant of the j -the block of Ω_i can then be transformed into a new generalized sub-eigenproblem. Therefore, the initial eigenproblem is decomposed into n highly smaller and smaller sub-eigenproblem, as follows:

$$K_j x_i = \gamma_i M_j x_i; j = 1, 2, \dots, n \quad (6.26)$$

where

$$K_j = A_K + \lambda_j B_K + \overline{\lambda_j} B_K^t \quad (6.27)$$

$$M_j = A_M + \lambda_j B_M + \overline{\lambda_j} B_M^t \quad (6.28)$$

where x_i can easily be converted to the required eigenvector corresponding to γ_i .

6.4 Mathematical Formulation of the Optimization Problem

In a truss sizing optimization problem with frequency constraints, the aim is to minimize the total weight of the structure while satisfying some constraints on natural vibration frequencies. The cross-sectional areas of structural members are considered as continuous design variables. The layout of the structure is pre-defined and kept unchanged during the optimization process. The mathematical formulation of the optimization problem is as follows:

$$\text{Find: } \{X\} = [x_1, x_2, \dots, x_{nDV}] \quad (6.29)$$

$$\text{to minimize: } P(\{X\}) = f(\{X\}) \times f_{penalty}(\{X\}) \quad (6.30)$$

$$\text{subject to: } \begin{cases} \omega_j \geq \omega_j^* \text{ for some natural vibration frequencies } j \\ \omega_k \leq \omega_k^* \text{ for some natural vibration frequencies } k \\ L_{b,i} \leq x_i \leq U_{b,i}, i = 1, 2, \dots, nDV \end{cases} \quad (6.31)$$

where $\{X\}$ denotes the vector of design variables, including sizing design variables, nDV is the number of design variables, which is selected considering the member-grouping configuration, x_i is the cross-sectional area of the structural members of the i th member group, $f(\{X\})$ is the objective function of the optimization problem to be minimized, which represents the total weight of the structure in a weight minimization problem, $f_{penalty}(\{X\})$ is the penalty function which is used to handle the problem constraints, and $P(\{X\})$ is the penalized objective function. $L_{b,i}$ and $U_{b,i}$ are the lower and upper bounds of the cross-sectional area of the structural members of the i th member group, respectively, ω_j and ω_k are the j th and the k th natural vibration frequencies of the structure, respectively, ω_j^* is the lower bound of the j th natural vibration frequency of the structure, and ω_k^* is the upper bound of the k th natural vibration frequency of the structure. The objective function is considered to be the total weight of the structure and can be defined as follows:

$$f(\{X\}) = W(\{X\}) = \sum_{i=1}^{nE} \rho_i \times A_i \times L_i \quad (6.32)$$

where ρ_i , A_i , and L_i are the material density, cross-sectional area, and length of the i -th structural member, respectively, nE is the number of structural members of the structure, and $W(\{X\})$ is the total weight of the structure.

Various strategies have been suggested to handle constraints in optimization problems, one of the most popular of which is penalizing strategies. The main idea of penalizing strategies is to transform a constrained optimization problem into an unconstrained one by penalizing the infeasible solution and extending an unconstrained objective function. Here, a dynamic penalty function is used to tackle the violated constraints:

$$f_{penalty}(X) = (1 + \varepsilon_1 \times v)^{\varepsilon_2}; v = \sum_{i=1}^{nC} v_i \quad (6.33)$$

where nC is the number of constraints of the problem, ε_1 and ε_2 are the penalty parameters that affect the severity of violated constraints, and v denotes the sum of the constraint violations. The value of v_i is set to zero if the i th constraint is satisfied, while in the case of a violated constraint, it is selected considering the severity of the violation. The mathematical expression of v_i is as follows:

$$v_i = \begin{cases} \left| 1 - \frac{\omega_i}{\omega_i^*} \right|, & \text{the } i\text{-th frequency constraint is violated} \\ 0, & \text{otherwise} \end{cases} \quad (6.34)$$

Dynamic penalty functions take into account the progress of the optimization process so that penalty is imposed at a dynamic or increasing rate. This means that a low degree of penalty is imposed at the beginning of the search process. However, as the search process progresses, the degree of the penalty also gradually increases. Such a dynamic strategy encourages the diversification in the search space (i.e., more exploration) in the early iterations of the optimization process, but more emphasis on the intensification of the best solutions found (i.e., more exploitation) in the last iterations [6].

The parameters ε_1 and ε_2 control how much an infeasible solution is penalized. The severity of penalizing is very sensitive to these parameters. Hence, setting the parameters ε_1 and ε_2 is a challenging task and requires many preliminary trials. Indeed, if they are chosen too small, feasible regions of search space may not be explored effectively, and even the algorithm may never converge toward a feasible solution. On the other hand, if they are too large, premature convergence may occur. In this study, a constant value for the parameter ε_1 is chosen, whereas the parameter ε_2 increases monotonically with the number of iterations.

6.5 Optimization Algorithms

This section consists of two subsections: firstly, a brief overview of the original Jaya algorithm (JA) is presented, and then the proposed set-theoretical Jaya algorithm (ST-JA) is outlined.

6.5.1 Jaya Algorithm

Jaya algorithm is a simple yet powerful population-based metaheuristic developed by Rao [25] in 2016. This algorithm is inspired by the concept of moving towards the best solution and getting away from the worst one. The solutions are updated as follows:

$$\begin{aligned} A'(i, j, k) = & A(i, j, k) + r(i, j, 1)(A(i, j, b) - |A(i, j, k)|) \\ & - r(i, j, 2)(A(i, j, w) - |A(i, j, k)|) \end{aligned} \quad (6.35)$$

where i , j , and k are the indices of iteration number, design variable, and candidate solution, respectively; $A(i, j, k)$ represents the j th design variable of k th candidate solution in i th iteration; $A(i, j, b)$ and $A(i, j, w)$ are the j th design variable of the

best solution and the worst solution among the current population (i.e., i th iteration), respectively; $|A(i, j, k)|$ is the absolute value of $A(i, j, k)$; $A'(i, j, k)$ is the j th design variable of k th updated solution of i th iteration; and $r(i, j, 1)$ and $r(i, j, 2)$ are two uniformly distributed random numbers between 0 and 1.

Once the solutions are updated, the boundary conditions are applied as follows:

$$\begin{cases} A'(i, j, k) = UB_j, & A'(i, j, k) > UB_j \\ A'(i, j, k) = LB_j, & A'(i, j, k) < LB_j \end{cases} \quad (6.36)$$

where LB_j and UB_j are the lower and upper bounds of the j th design variables, respectively.

Next, a simple replacement strategy is applied to determine the population of the next iteration. For this purpose, the updated solutions are compared with the corresponding old solutions and the better ones (i.e., solutions with better penalized objective function values) are considered as the population of the next iteration. For a minimization optimization problem, this can be expressed as follows:

$$A(i+1, ., k) = \begin{cases} A'(i, j, k), & P(\{A'(i, j, k)\}) < P(\{A(i, j, k)\}) \\ A(i, j, k), & P(\{A'(i, j, k)\}) > P(\{A(i, j, k)\}) \end{cases} \quad (6.37)$$

where P represents the penalized objective function.

Figure 6.3 shows the pseudocode of the original Jaya algorithm.

Pseudocode of Jaya algorithm

Initialization phase

Set the algorithm parameters: population size (nP) and maximum number of iterations ($MaxIt$). Generate initial population randomly and evaluate them

Initialize time: $i = 1$

Cyclic body of the algorithm

While $i \leq MaxIt$

 Identify the best and worst solutions among the current population

For $k = 1: nP$

 Update the k -th candidate solution based on Eq. (6.35)

 Apply boundary conditions based on Eq. (6.36)

 Evaluate the k -th updated solution

 Apply replacement strategy based on Eq. (6.37)

End for k

$i = i + 1$

End while

Output the best solution

Fig. 6.3 Pseudocode of the original Jaya algorithm

6.5.2 Set-Theoretical Jaya Algorithm

The set-theoretical Jaya algorithm (ST-JA) is based on the idea of division of the population of solutions into a predefined number of well-arranged subpopulations [14]. In addition to the common control parameters of population size (nP) and the maximum number of iterations ($MaxIt$), ST-JA requires an additional parameter, namely the number of subpopulations (nS). The ST-JA is stated in four steps as follows [14]:

Step one (initialization): The initial population is generated randomly considering the lower and upper bounds of the design variables of the problem.

Step two (formation of subpopulations): Based on a simple method proposed by Kaveh et al. [26], the initial population is divided into a predefined number of well-arranged subpopulations of the same size. To this end, let us consider an initial population nP candidate solutions. The population is sorted in ascending order of penalized objective function values. In the first step of formation of subpopulations, the first nS candidate solutions of the sorted population are chosen and one candidate solution is placed in each of the nS subpopulations randomly. In the second step of formation of subpopulations, the next nS candidate solutions of the sorted population are chosen and one candidate solution is placed in each of the nS subpopulations randomly. The process continues until all the candidate solutions are assigned to subpopulations. The number of subpopulations (nS) should be chosen from the set of divisors of population size (nP). It is worth mentioning that each subpopulation has its own best and worst solutions.

Step three (main loop of the Jaya algorithm): The main loop of the original JA (i.e., Eqs. (6.35), (6.36), and (6.37)) is performed for all the nS subpopulations separately. Therefore, nS new subpopulations, each of which has nP/nS new candidate solutions are formed. The new subpopulations collectively form the population of the next iteration.

Step four (termination condition): If the termination condition of the algorithm is satisfied, output the best solution; otherwise, go to step two.

The set-theoretical Jaya algorithm takes into account both the global exploration and local exploitation of the search. Indeed, since each subpopulation has its own best and worst solutions, ST-JA can explore the search space more globally compared to the original JA. Furthermore, ST-JA can exploit a large number of promising regions in the search space compared to the original JA. Figure 6.4 shows the pseudocode of set-theoretical JA.

6.6 Results and Discussion

Two large-scale cyclic symmetric dome optimization problems are provided in this section to demonstrate the efficiency of the present eigensolution method for free vibration analysis of rotationally repetitive structures and to examine the performance

Pseudocode of set-theoretical Jaya algorithm

Initialization phase

Set the algorithm parameters: population size (nP), the maximum number of iterations ($MaxIt$), and number of subpopulations (nS)

Generate initial population randomly and evaluate them

Initialize time: $i = 1$

Cyclic body of the algorithm

While $i \leq MaxIt$

 Sort the population of solutions in ascending order of penalized objective function values

 Form the subpopulations based on the method proposed by Kaveh et al. [26]

For $j = 1: nS$

 Identify the best and worst solutions among the solutions of the j -th subpopulation

For $k = 1: nP/nS$

 Update the k -th candidate solution of the j -th subpopulation based on Eq. (6.35)

 Apply boundary conditions based on Eq. (6.36)

 Evaluate the k -th updated solution of the j -th subpopulation

 Apply replacement strategy based on Eq. (6.37)

End for k

End for j

 Collect the subpopulations and form the population of solutions of the i -th iteration

$i = i + 1$

End while

Output the best solution

Fig. 6.4 Pseudocode of the set-theoretical Jaya algorithm (ST-JA) [14]

of the proposed ST-JA and compare it with the original JA. The first optimization problem is the sizing optimization of a 600-bar single-layer dome structure with 25 design variables and the second one is the sizing optimization of a 1410-bar double-layer dome structure with 47 design variables. Table 6.2 lists the material properties, cross-sectional area bounds, and frequency constraints of the problems. In order to make it possible to compare the results of the present study with those reported in the literature, the values of the parameters listed in the table are the same as those of the literature. The computational efficiency of the present eigensolution method is compared with the existing classical eigensolution method. Furthermore, the optimization results obtained by ST-JA are compared with those of the original JA and other optimization methods reported in the literature. To consider the stochastic nature of the optimization process, ten independent runs were performed for each problem and the results of the best run among ten runs are reported. The optimization results are reported in terms of best weight, average weight, worst weight, and standard deviation of ten runs. The population size nP is chosen to be 20 for both the JA and ST-JA in all problems. The number of subpopulations (nS) of the ST-JA is set to be four in all problems. The maximum number of iterations ($MaxIt$) is taken as the termination criterion of the optimization process and is set to 600 for the first problem and 1000 for the second one. The finite element models and the optimization codes are implemented in the Matlab environment. It is noted that the optimizations were performed on a PC with Windows 10, Intel(R) Core (TM) i5-7200U CPU 2.50 2.71 GHz, and 8.00 GB RAM.

Table 6.2 Material properties, cross-sectional area bounds, and frequency constraints of the problems

Problem	Elasticity modulus E (N/m ²)	Material density ρ (kg/m ³)	Cross-sectional area bounds (m ²)	Frequency constraints (Hz)
600-bar dome-like truss	2×10^{11}	7850	$0.0001 \leq A_i \leq 0.01$	$\omega_1 \geq 5, \omega_3 \geq 7$
1410-bar dome-like truss	2×10^{11}	7850	$0.0001 \leq A_i \leq 0.01$	$\omega_1 \geq 7, \omega_3 \geq 9$

6.6.1 A 600-Bar Single-Layer Dome-Like Truss

The 600-bar single-layer dome depicted in Fig. 6.5 is considered as the first example. The dome structure is composed of 216 nodes and 600 elements and could be generated by the cyclic repetition of a sub-structure with 9 nodes and 25 elements around the cyclic symmetry axis of the dome. Figure 6.5c shows the details of the nodal numbering of the first sub-structure. The angle of cyclic symmetry between similar sub-structures is equal to 15 degrees, which results in a total of 24 similar sub-structures. The nodal coordinates of the first sub-structure in the Cartesian coordinate system are provided in Table 6.3. The connectivity information of the first sub-structure is also given in Table 6.5. The cross-sectional area of each element of this sub-structure is considered as a design variable. However, the layout of the dome is kept unchanged during the optimization process. Therefore, this is a sizing optimization problem with 25 design variables. A non-structural mass of 100 kg is attached at each free node of the dome. As Table 6.2 shows, the frequency constraints are imposed on the first and third natural frequencies. This problem was previously studied by many researchers using different metaheuristic optimization algorithms [5, 7–11].

Table 6.4 compares the computational efficiency of the proposed efficient eigen-solution method with the classical eigensolution method for free vibration analysis of the 600-bar dome structure. To perform the free vibration analysis of the 600-bar dome structure, instead of solving a matrix eigenvalue problem of dimension 576 (i.e. the degrees of freedom of the whole structure), which is the case of the classical eigensolution methods, the proposed efficient eigensolution method requires only to calculate the eigenvalues of 24 matrices of dimension 24 (i.e., the degrees of freedom of the sub-structure shown in Fig. 6.5-c). This highly affects the computational time and memory required to perform the analysis. Indeed, the average computational time for a free vibration analysis of the 600-bar dome structure using the classical eigensolution method is 0.0363 s, which is more than six times that of the proposed efficient eigensolution method (0.0054 s). This leads to a considerable saving in the computational time of the optimization process. We have estimated that 4352.36 s (about 73 min) would be required to perform ten independent runs of the proposed

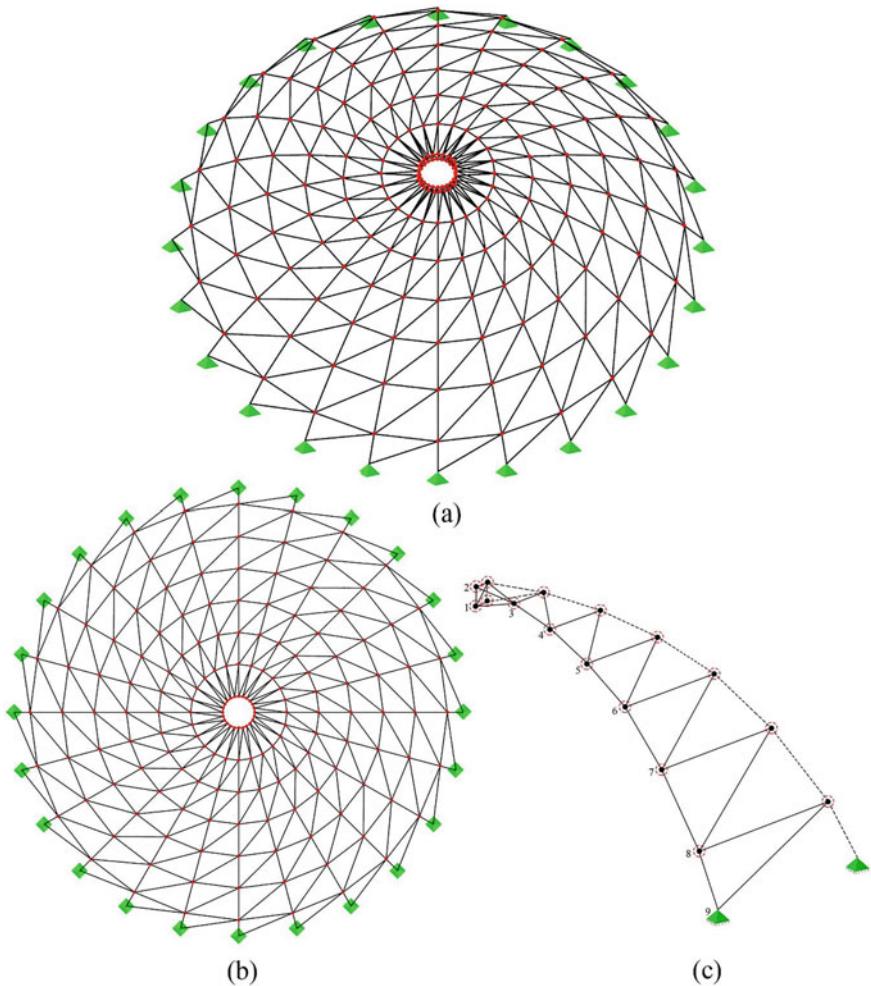


Fig. 6.5 Schematic of the 600-bar single-layer dome: **a** perspective view; **b** top view; **c** substructure

ST-JA using the classical eigensolution method. However, using the efficient eigen-solution method, the same algorithm requires only 645.57 s (about 11 min) to perform these runs.

Table 6.5 shows a comparison of the optimal design results obtained by JA, ST-JA, and other methods in the literature [5, 7–11]. From Table 6.5, it can be seen that the best and worst weights, average weight, and standard deviation achieved by the ST-JA over ten independent runs are significantly better than those of the JA and other methods in the literature. The average weight of the ST-JA is even better than the best weight obtained by all the other methods, including JA. The best weight of the ST-JA is 6065.811 kg, which is about 16.078 kg lighter than that of the original

Table 6.3 Nodal coordinates (m) of the first sub-structure of the 600-bar dome-like truss

Node number	Coordinates (x, y, z)
1	(1.0, 0.0, 7.0)
2	(1.0, 0.0, 7.5)
3	(3.0, 0.0, 7.25)
4	(5.0, 0.0, 6.75)
5	(7.0, 0.0, 6.0)
6	(9.0, 0.0, 5.0)
7	(11.0, 0.0, 3.5)
8	(13.0, 0.0, 1.5)
9	(14.0, 0.0, 0.0)

Table 6.4 Comparison of the computational efficiency of the methods for the 600-bar dome structure

Method	Eigenvalue problem	CPU time (s)
Classical eigensolution method	1 matrix of 576×576	0.0363
Efficient eigensolution method	24 matrices of 24×24	0.0054
Time ratio	6.742	

JA. In terms of computational cost, the ST-JA requires only 6880 analyses to find a feasible design corresponding to a structural weight of 6081.133 kg, which is lighter than the best designs reported in the literature. Furthermore, the maximum number of structural analyses of the ST-JA is set to be 12,000, which is much less than those of the other methods. Figure 6.6 compares the average-weight convergence histories for the original JA and ST-JA. The curves are magnified to better display the convergence characteristics. As observed, the convergence rate of the ST-JA is much faster than the original JA. Table 6.6 lists the first five natural frequencies of the optimal designs obtained by JA, ST-JA, and other methods in the literature. It can be observed that none of the frequency constraints are violated.

6.6.2 A 1410-Bar Double-Layer Dome-Like Truss

The second example is the sizing optimization of the 1410-bar double-layer dome structure depicted in Fig. 6.7. The dome structure is comprised of a total of 390 nodes and 1410 elements and. As can be seen from the figure, the structure exhibits cyclic symmetry so that it could be generated by the cyclic repetition of a sub-structure with 13 nodes and 47 elements around the cyclic symmetry axis of the dome. The nodal numbering of the first sub-structure is illustrated in Fig. 6.7c. The angle of cyclic symmetry between similar sub-structures is 12 degrees leading to a total of 30 similar sub-structures. The Cartesian coordinates of the nodes of the first sub-structure are

Table 6.5 Comparison of optimal results obtained for the 600-bar dome-like truss (cm^2)

Element number (element nodes)	DPSO [7]	VPS [8]	ECBO-Cascade [9]	CBO [5]	MDVC-UVPS [10]	PEJA [11]	This study
	JA	JA	JA	JA	JA	JA	ST-JA
1 (1–2)	1.365	1.3030	1.0299	1.2404	1.2575	1.1867	1.0703
2 (1–3)	1.391	1.3998	1.3664	1.3797	1.3466	1.2967	1.2699
3 (1–10)	5.686	5.1072	5.1095	5.2597	4.9738	4.5771	4.0174
4 (1–11)	1.511	1.3882	1.3011	1.2658	1.4025	1.3356	1.0036
5 (2–3)	17.711	16.9217	17.0572	17.2255	17.3802	18.3157	16.7759
6 (2–11)	36.266	38.1432	34.0764	38.2991	37.9742	38.5097	39.2560
7 (3–4)	13.263	11.8319	13.0985	12.2234	13.0306	13.5917	13.2920
8 (3–11)	16.919	16.6149	15.5882	15.4712	15.9209	16.8824	15.1664
9 (3–12)	13.333	11.3403	12.6889	11.1577	11.9419	13.8766	10.8041
10 (4–5)	9.534	9.3865	10.3314	9.4636	9.1643	9.5286	8.9660
11 (4–12)	9.884	8.7692	8.5313	8.8250	8.4332	9.4218	8.7332
12 (4–13)	9.547	9.6682	9.8308	9.1021	9.2375	9.7643	8.8557
13 (5–6)	7.866	6.9826	7.0101	6.8417	7.2213	7.2431	7.7360
14 (5–13)	5.529	5.4445	5.2917	5.2882	5.2142	5.3913	5.1991
15 (5–14)	7.007	6.3247	6.2750	6.7702	6.7961	6.7468	6.5265
16 (6–7)	5.462	5.1349	5.4305	5.1402	5.2078	5.1493	5.1082
17 (6–14)	3.853	3.3991	3.6414	5.1827	3.4586	3.8342	3.7784
18 (6–15)	7.432	7.7911	7.2827	7.4781	7.6407	8.0665	7.8962
19 (7–8)	4.261	4.4147	4.4912	4.5646	4.3690	4.2800	4.0664
20 (7–15)	2.253	2.2755	1.9275	1.8617	2.1237	2.2509	2.3832

(continued)

Table 6.5 (continued)

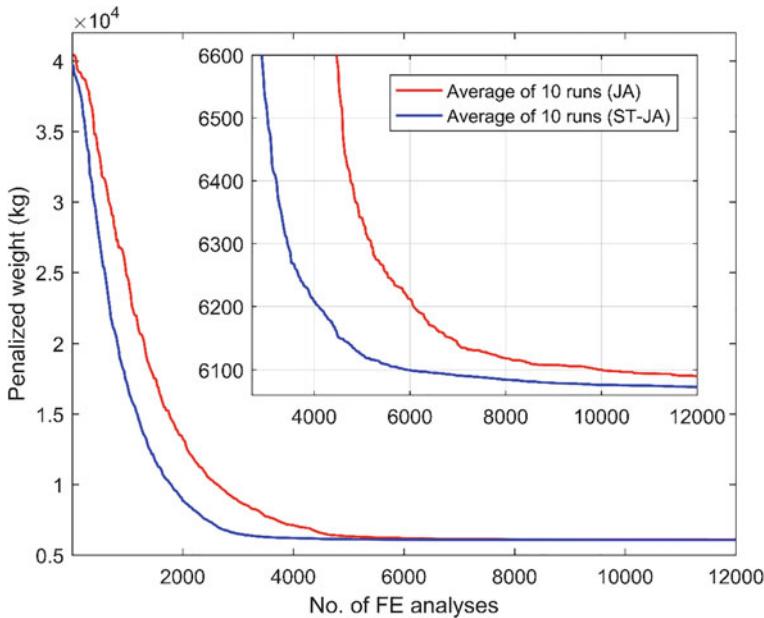


Fig. 6.6 Comparison of the convergence histories of JA and ST-JA for the 600-bar dome-like truss

Table 6.6 First five natural frequencies (Hz) of the 600-bar dome-like truss evaluated at optimal designs

Frequency number	DPSO [7]	VPS [8]	ECBO-Cascade [9]	CBO [5]	MDVC-UVPS [10]	PFJA [11]	This study	
							JA	ST-JA
1	5.000	5.0000	5.001	5.000	5.000	5.0011	5.0052	5.0002
2	5.000	5.0003	5.001	5.000	5.000	5.0011	5.0052	5.0002
3	7.000	7.0000	7.001	7.000	7.000	7.0000	7.0002	7.0002
4	7.000	7.0001	7.001	7.000	7.000	7.0000	7.0009	7.0006
5	7.000	7.0002	7.002	7.001	7.000	7.0000	7.0009	7.0006

given in Table 6.7. The connectivity information of the first sub-structure is given in Table 6.9. The cross-sectional area of each element of this sub-structure represents a design variable of the problem. However, the layout of the structure is predefined and remains unchanged during the optimization process. Hence, this is a sizing optimization problem with 47 design variables. A non-structural mass of 100 kg is attached to each free node of the dome. As Table 6.2 shows, the frequency constraints are imposed on the first and third natural vibration frequencies of the dome. This structure was previously optimized with different metaheuristic optimization algorithms [6, 7, 9–13].

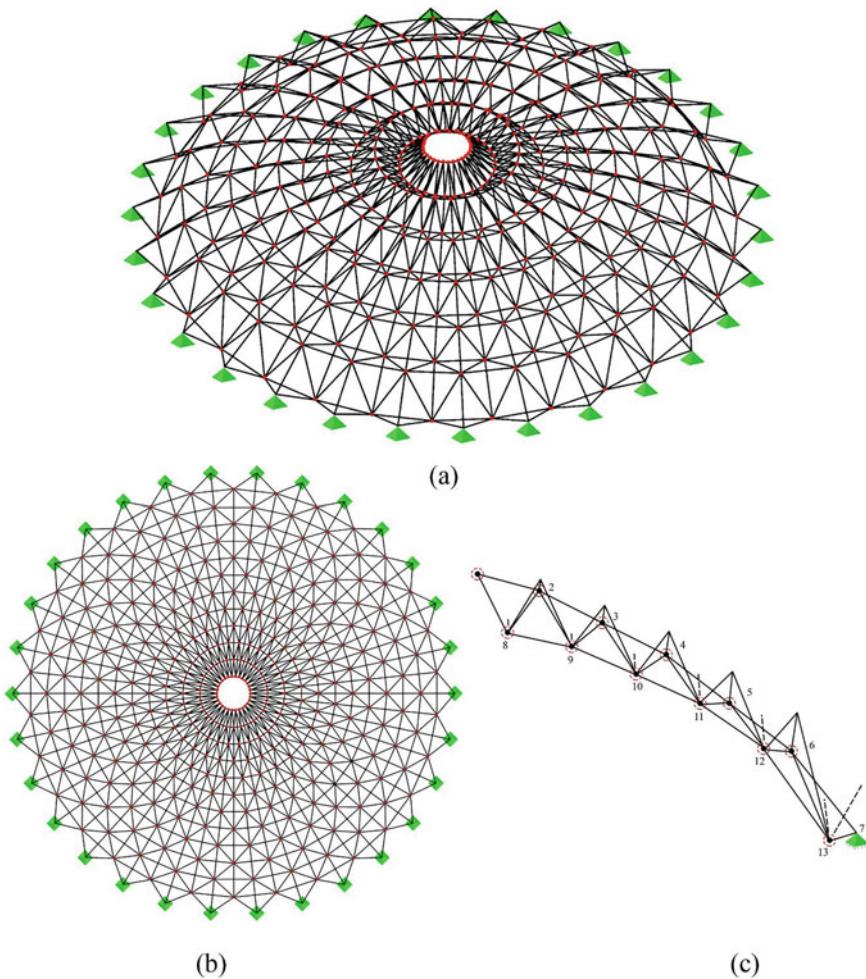


Fig. 6.7 Schematic of the 1410-bar single-layer dome: **a** perspective view; **b** top view; **c** substructure

Table 6.8 provides a comparison of the computational efficiency of the proposed efficient eigensolution method and the classical eigensolution method for free vibration analysis of the 1410-bar dome structure. In order to perform the free vibration analysis of the 1410-bar dome structure, the proposed efficient eigensolution method needs only to calculate the eigenvalues of 30 matrices of dimension 36 (i.e., the degrees of freedom of the sub-structure shown in Fig. 6.7c), while the classical eigensolution method requires the solution of a matrix eigenvalue problem of dimension 1080 (i.e., the degrees of freedom of the whole structure). Consequently, the computational time and memory required for the proposed efficient eigensolution method are far less than that of the classical eigensolution method. The average

Table 6.7 Nodal coordinates (m) of the first sub-structure of the 1410-bar dome-like truss

Node number	Coordinates (x, y, z)
1	(1.0, 0.0, 4.0)
2	(3.0, 0.0, 3.75)
3	(5.0, 0.0, 3.25)
4	(7.0, 0.0, 2.75)
5	(9.0, 0.0, 2.0)
6	(11.0, 0.0, 1.25)
7	(13.0, 0.0, 0.0)
8	(1.989, 0.209, 3.0)
9	(3.978, 0.418, 2.75)
10	(5.967, 0.627, 2.25)
11	(7.956, 0.836, 1.75)
12	(9.945, 1.0453, 1.0)
13	(11.934, 1.2543, -0.5)

computational time for a free vibration analysis of the 1410-bar dome structure using the classical eigensolution method is 0.1797 s, which is more than twelve times that of the proposed efficient eigensolution method (0.0140 s). This causes a significant decrease in the computational time of the optimization process. We have estimated that 35,941.3224 s (about 599 min) would be required to perform ten independent runs of the proposed ST-JA using the classical eigensolution method. However, using the efficient eigensolution method, the same algorithm requires only 2806.7635 s (about 47 min) to perform these runs.

Table 6.9 compares the optimization results obtained by JA, ST-JA, and other optimization methods in the literature [6, 7, 9–13]. As it can be seen, the best, average, and the worst results obtained by the ST are better than those of those previously reported in the literature. The best weight of the ST-JA is 10283.094 kg, which is far better than that of the original JA (i.e., 10,400.079 kg). The average weight achieved by the ST-JA is even better than the best weights obtained by CPA [6], DPSO [7], ECBO-Cascade [9], and CRPSO [12]. In terms of computational efficiency, the maximum number of structural analyses of the ST-JA is 20000, which is less than or equal to those of the other methods except MJA [13]. Furthermore,

Table 6.8 Comparison of the computational efficiency of the methods for the 1410-bar dome structure

Method	Eigenvalue problem	CPU time (s)
Classical eigensolution method	1 matrix of 1080×1080	0.1797
Efficient eigensolution method	30 matrices of 36×36	0.0140
Time ratio	12.805	

Table 6.9 Comparison of optimal results obtained for the 1410-bar dome-like truss (cm^2)

Element number (element nodes)	CPA [6]	DPSO [7]	ECBO-Cascade [9]	MDYC-UPVS [10]	PFA [11]	CRPSO [12]	MIA [13]	This study JA	This study ST-JA
1 (1–2)	7.416	7.209	7.9969	5.8499	6.1902	2.5	7.3465	5.1377	5.3860
2 (1–8)	4.768	5.006	6.1723	4.5115	4.4036	6.0	4.2998	3.7009	4.4361
3 (1–14)	38.993	38.446	35.5011	19.4823	31.2253	18.0	31.8485	33.9653	27.7395
4 (2–3)	8.966	9.438	10.2510	8.8480	8.4715	9.5	8.8767	9.5992	8.1780
5 (2–8)	4.511	4.313	5.3727	5.0084	4.8590	6.0	4.9778	5.5482	6.1189
6 (2–9)	1.544	1.494	1.3488	1.3568	1.5759	1.0	1.7469	3.6969	1.5996
7 (2–15)	8.371	8.455	11.4427	17.4331	12.9451	29.5	11.6099	21.8390	18.7495
8 (3–4)	9.276	9.488	9.7157	9.1098	9.3263	8.0	9.2972	8.6255	9.2413
9 (3–9)	3.583	3.480	1.3005	2.8712	3.2716	2.0	3.3406	3.5731	2.6310
10 (3–10)	3.476	3.495	2.5046	3.5473	3.2878	1.5	3.2006	3.3162	2.2419
11 (3–16)	15.531	16.037	10.7849	12.3768	12.6719	1.0	12.1131	9.0269	7.9773
12 (4–5)	10.285	9.796	10.1954	10.1099	10.0979	7.5	9.7121	9.5465	10.1147
13 (4–10)	2.497	2.413	2.2300	2.5797	2.5803	1.0	2.5294	2.4441	2.1849
14 (4–11)	5.397	5.681	5.1186	5.8381	5.3769	6.0	5.8102	4.0831	6.4570
15 (4–17)	16.503	15.806	14.0053	13.6402	16.0581	14.5	16.5566	12.3422	18.2837
16 (5–6)	8.193	8.078	8.9713	9.9096	8.6789	9.0	8.3162	8.6568	8.2311
17 (5–11)	3.829	3.931	4.0756	3.6543	3.3199	1.0	3.2415	4.0585	3.1540
18 (5–12)	6.151	6.099	5.9211	6.1529	6.4966	8.0	6.4539	5.2401	5.8960
19 (5–18)	10.465	10.771	10.6915	11.2448	10.8804	19.5	10.7040	13.4226	19.8678
20 (6–7)	13.925	13.775	10.6220	13.1071	14.0056	16.5	13.8031	13.3777	13.5511

(continued)

Table 6.9 (continued)

Element number (element nodes)	CPA [6]	DPSO [7]	ECBO-Cascade [9]	MDYC-UPVS [10]	PFA [11]	CRPSO [12]	MIA [13]	This study JA	This study ST-JA
21 (6–12)	4.415	4.231	4.5064	5.2361	5.0843	5.0	5.0161	6.0785	5.5768
22 (6–13)	6.863	6.995	8.4086	7.0691	6.9952	9.0	7.6509	8.0438	6.7898
23 (6–19)	1.769	1.837	5.8405	2.0015	1.0270	1.0	1.0762	1.0226	1.0175
24 (7–13)	4.339	4.397	5.0342	4.7178	4.3788	5.0	4.3282	4.2592	4.1502
25 (8–9)	2.115	2.115	3.8932	2.6101	2.1951	6.5	2.2062	2.1809	2.7295
26 (8–14)	4.951	4.923	6.1647	4.5434	4.2562	5.5	4.8730	3.9382	4.1142
27 (8–15)	4.147	4.047	6.8990	4.6174	4.6605	7.0	4.8202	5.2629	5.8905
28 (8–21)	6.044	5.906	11.6387	9.6758	8.8694	15.5	9.0166	12.1908	11.5388
29 (9–10)	3.222	3.392	3.8343	3.6296	3.2333	4.5	3.4591	4.1286	4.3093
30 (9–15)	1.970	1.902	1.4772	1.4891	1.7611	2.5	1.9876	3.6731	1.8975
31 (9–16)	4.290	4.381	1.3075	3.4020	3.2831	2.5	3.4317	2.6622	2.5412
32 (9–22)	8.020	8.442	4.4876	6.2153	7.1936	1.0	7.7208	4.8959	4.6417
33 (10–11)	4.857	5.011	6.0196	5.9308	4.9840	6.0	4.8261	5.4788	5.4994
34 (10–16)	3.689	3.577	2.6729	3.2334	3.6672	1.0	2.9942	3.4673	2.4481
35 (10–17)	2.831	2.805	1.6342	2.7173	2.4062	1.0	2.5166	2.4088	2.0894
36 (10–23)	1.985	2.024	1.8410	1.3932	2.1576	1.0	1.8493	1.1880	1.7796
37 (11–12)	6.373	6.709	6.8841	6.5660	7.1043	10.0	7.1007	7.1520	7.9676
38 (11–17)	4.865	5.054	4.1393	4.8170	5.2070	5.5	5.1141	4.3348	4.9463
39 (11–18)	3.412	3.259	3.3264	3.2626	3.6853	3.5	4.0067	2.5376	3.3697

(continued)

Table 6.9 (continued)

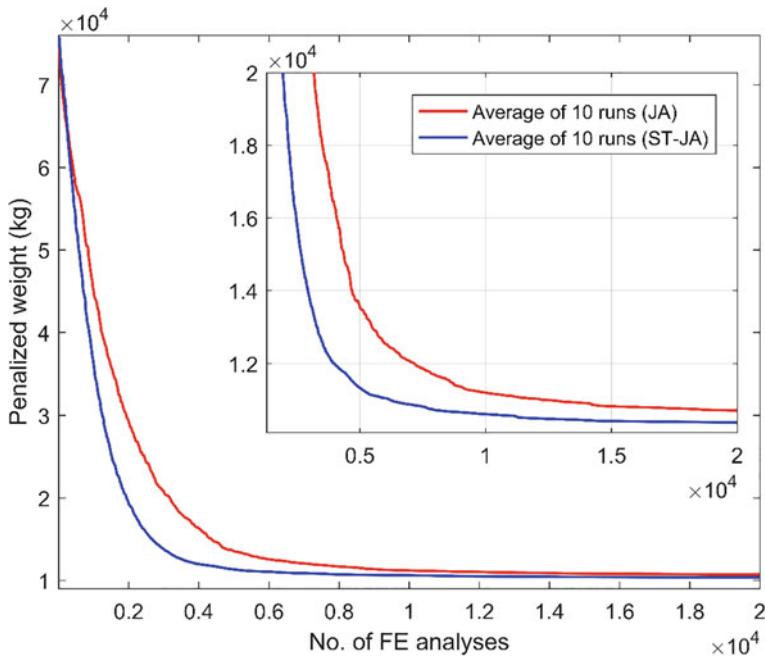


Fig. 6.8 Comparison of the convergence histories of JA and ST-JA for the 1410-bar dome-like truss

the ST-JA needs only 14,440 analyses to achieve a feasible design with structural weight of 10,325.697 kg, which is lighter than the best weights of the other methods in the literature. Figure 6.8 compares the average-weight convergence histories of the ST-JA and the original JA. In order to better display the convergence characteristics, the curves are magnified. As can be seen, the ST-JA converges much faster than the original JA. Table 6.10 summarizes the first five natural frequencies of the best designs found by JA, ST-JA and other methods in the literature. It can be seen that all the frequency constraints are satisfied.

6.7 Concluding Remarks

Design optimization of structural systems usually requires a large number of structural analyses, which can be computationally challenging in the case of large-scale structures. However, optimal structural analysis provides efficient methods of structural analysis which can substantially reduce the computational time of the process of structural optimization. In this study, an efficient eigensolution method for free vibration analysis of cyclic symmetric structures is developed for the optimal design of rotationally repetitive structures subject to frequency constraints. Through the

Table 6.10 First five natural frequencies (Hz) of the 14/10-bar dome-like truss evaluated at optimal designs

Frequency number	CPA [6]	DPSO [7]	ECBO-Cascade [9]	MDVC-UPVS [10]	PFJA [11]	CRPSO [12]	MJA [13]	This study	
	JA	ST-JA						JA	ST-JA
1	7.000	7.001	7.0020	7.000	7.0009	7.0008	7.0003	7.0017	7.0002
2	7.000	7.001	7.0030	7.001	7.0009	–	7.0003	7.0017	7.0002
3	9.000	9.003	9.0010	9.000	9.0001	9.0068	9.0000	9.0027	9.0006
4	9.002	9.005	9.0010	9.000	9.0002	–	9.0002	9.0035	9.0021
5	9.002	9.005	9.0030	9.000	9.0002	–	9.0002	9.0035	9.0021

present eigensolution method, the initial free-vibration eigenproblem is decomposed into highly smaller sub-eigenproblems, which results not only in a substantial reduction in both computational time and memory requirements compared to the existing classical method, but also exact free vibration solutions. Two large-scale cyclic symmetric dome structures are given to illustrate the efficiency and accuracy of the present eigensolution method and the results are compared with those of the existing classical method. As the results show, the present eigensolution method requires significantly less computational time compared to the existing classical method. In the case of using classical eigensolution method, more than 11 h would be required to execute all the optimization runs required in this work. However, via the present efficient eigensolution method, all the same runs are executed in less than 1 h.

A set-theoretical Jaya algorithm (ST-JA) is proposed and applied to the optimal design of large-scale dome structures with frequency constraints. The performance of the proposed ST-JA is compared with those of the original JA and some other methods from the literature. The main idea of the ST-JA is the division of the initial population of the original JA into a predefined number of well-arranged subpopulations. The purpose of the proposed ST-JA is to improve the global exploration and local exploitation capabilities of the original JA and make a fine balance between them. Optimization results confirm that the best and average weights found by the ST-JA are better than those acquired by other methods in the literature. Furthermore, thanks to wide exploration of the search space at the beginning of the search and deep exploitation of promising regions of the search space in the final stages of the search, the ST-JA outperforms the original JA in both terms of effectiveness and robustness. In both the design examples, the best weight, average weight, worst weight, and standard deviation obtained by the proposed ST-JA are much better than those of the original JA. This shows that the proposed ST-JA can offer a robust and competitive optimization algorithm for the optimal design of truss structures with multiple frequency constraints.

For future work, we plan to apply the efficient free vibration analysis method to other types of cyclic symmetric structures so that we can solve optimization problems related to such structures with less computational effort.

References

1. Kaveh A, Biabani Hamedani K, Joudaki A, Kamalinejad M (2021) Optimal analysis for optimal design of cyclic symmetric structures subject to frequency constraints. *Structures* 33:3122–3136. <https://doi.org/10.1016/j.istruc.2021.06.054>
2. Rao SS, Reddy ES (1981) Optimum design of stiffened conical shells with natural frequency constraints. *Comput Struct* 14(1–2):103–110. [https://doi.org/10.1016/0045-7949\(81\)90089-4](https://doi.org/10.1016/0045-7949(81)90089-4)
3. Bellagamba L, Yang TY (1981) Minimum-mass truss structures with constraints on fundamental natural frequency. *AIAA J* 19(11):1452–1458. <https://doi.org/10.2514/3.7875>
4. Kaveh A, Biabani Hamedani K, Kamalinejad M (2020) Set theoretical variants of the teaching–learning-based optimization algorithm for optimal design of truss structures with multiple frequency constraints. *Acta Mech* 231(9):3645–3672. <https://doi.org/10.1007/s00707-020-02718-3>

5. Kaveh A, Ilchi Ghazaan M (2016) Optimal design of dome truss structures with dynamic frequency constraints. *Struct Multidisc Optim* 53(3):605–621. <https://doi.org/10.1007/s00158-015-1357-2>
6. Kaveh A, Zolghadr A (2018) Optimal design of cyclically symmetric trusses with frequency constraints using cyclical parthenogenesis algorithm. *Adv Struct Eng* 21(5):739–755. <https://doi.org/10.1177/1369433217732492>
7. Kaveh A (2017) Optimal analysis and design of large-scale domes with frequency constraints. In: Applications of metaheuristic optimization algorithms in civil engineering. Springer, Cham, pp 257–279. https://doi.org/10.1007/978-3-319-48012-1_14
8. Kaveh A, Ilchi Ghazaan M (2017) Vibrating particles system algorithm for truss optimization with multiple natural frequency constraints. *Acta Mech* 228(1):307–322. <https://doi.org/10.1007/s00070-016-1725-z>
9. Kaveh A, Ilchi Ghazaan M (2018) Meta-heuristic algorithms for optimal design of real-size structures. Springer International Publishing, Switzerland
10. Kaveh A, Ilchi Ghazaan M (2018) A new hybrid meta-heuristic algorithm for optimal design of large-scale dome structures. *Eng Optim* 50(2):235–252. <https://doi.org/10.1080/0305215X.2017.1313250>
11. Degertekin SO, Bayar GY, Lamberti L (2021) Parameter free Jaya algorithm for truss sizing-layout optimization under natural frequency constraints. *Comput Struct* 245:106461. <https://doi.org/10.1016/j.compstruc.2020.106461>
12. Carvalho JP, Lemonge AC, Carvalho ÉC, Hallak PH, Bernardino HS (2018) Truss optimization with multiple frequency constraints and automatic member grouping. *Struct Multidisc Optim* 57(2):547–577. <https://doi.org/10.1007/s00158-017-1761-x>
13. Degertekin SO, Yalcin Bayar G, Lamberti L (2019) Jaya algorithm for sizing and layout optimization of truss structures with natural frequency constraints. In: Topping BHV, Ivany P (eds), Proceedings of the sixteenth international conference on civil, structural & environmental engineering computing. Civil-Comp Press, Stirlingshire (UK)
14. Kaveh A, Biabani Hamedani K, Kamalinejad M (2021) Set theoretical variants of optimization algorithms for system reliability-based design of truss structures. *Period Polytech Civ Eng* 65(3):717–729. <https://doi.org/10.3311/PPci.17519>
15. Kaveh A (2013) Optimal analysis of structures by concepts of symmetry and regularity. Springer, New York
16. Renton JD (1964) On the stability analysis of symmetrical frameworks. *Q J Mech Appl Math* 17(2):175–195. <https://doi.org/10.1093/qjmam/17.2.175>
17. Hussey MJ (1967) General theory of cyclically symmetric frames. *J Struct Div* 93(2):163–176. <https://doi.org/10.1061/JSDEAG.0001638>
18. Zloković Đ (1989) Group theory and G-vector spaces in structural analysis: vibration, stability, and statics. Ellis Horwood
19. Zingoni A (2009) Group-theoretic exploitations of symmetry in computational solid and structural mechanics. *Int J Numer Methods Eng* 79(3):253–289. <https://doi.org/10.1002/nme.2576>
20. Kaveh A, Nemati F (2010) Eigensolution of rotationally repetitive space structures using a canonical form. *Int J Numer Methods Biomed Eng* 26(12):1781–1796. <https://doi.org/10.1002/cnm.1265>
21. Kaveh A, Rahami H (2010) An efficient analysis of repetitive structures generated by graph products. *Int J Numer Methods Eng* 84(1):108–126. <https://doi.org/10.1002/nme.2893>
22. Kaveh A, Rahami H (2011) Block circulant matrices and applications in free vibration analysis of cyclically repetitive structures. *Acta Mech* 217(1):51–62. <https://doi.org/10.1007/s00070-010-0382-x>
23. Kaveh A, Kohestani K (2009) Combinatorial optimization of special graphs for nodal ordering and graph partitioning. *Acta Mech* 207(1):95–108. <https://doi.org/10.1007/s00070-008-0107-6>
24. Kohestani K, Kaveh A (2010) Efficient buckling and free vibration analysis of cyclically repeated space truss structures. *Finite Elem Anal Des* 46(10):943–948. <https://doi.org/10.1016/j.finel.2010.06.009>

25. Rao R (2016) Jaya: a simple and new optimization algorithm for solving constrained and unconstrained optimization problems. *Int J Ind Eng Comput* 7(1):19–34. <https://doi.org/10.5267/j.ijiec.2015.8.004>
26. Kaveh A, Biabani Hamedani K, Zaerreza A (2021) A set theoretical shuffled shepherd optimization algorithm for optimal design of cantilever retaining wall structures. *Eng Comput* 37(4):3265–3282. <https://doi.org/10.1007/s00366-020-00999-9>

Chapter 7

Discrete Structural Optimization with Set-Theoretical Jaya Algorithm



7.1 Introduction

In this chapter, a discrete version of the set-theoretical Jaya algorithm (ST-JA) is employed to solve discrete structural optimization problems [1]. As discussed in Chapters 5 and 6, ST-JA aims to improve both the exploration and exploitation capabilities of the classical Jaya algorithm. The main idea of ST-JA, which is derived from the concepts of set theory, is to divide the population of solutions into smaller well-arranged subpopulations of the same size. The performance of ST-JA is demonstrated through four benchmark truss optimization problems with discrete design variables. The results achieved by the classical JA and ST-JA are compared with each other and with those of some other state-of-the-art optimization methods existing in the literature.

Over the past decades, structural optimization has attracted intensive efforts both in theoretical and practical aspects, due to its potential to reduce the structural weight and cost while satisfying the design requirements. Structural optimization not only provides an effective way to reduce the structural weight and cost, but also allows sensitivity analysis to improve the structural integrity. Accordingly, a great number of optimization methods have been proposed and successfully utilized in a large variety of structural optimization problems. Examples of classical optimization techniques applied to structural optimization include the integer linear programming (ILP) [2] and sequential quadratic programming (SQP) [3]. It should be noted that most of the classical optimization techniques are more suitable for continuous search spaces. However, in real-world structural optimization problems, the structural elements must usually be selected from a list of commercially available cross-sections. Moreover, real-world structural optimization problems often involve multiple highly nonlinear constraints. Because of these reasons, classical optimization approaches were found not suitable or easy to deal with real-world structural optimization problems. However, some classical optimization methods integrated with the round-off techniques have been proposed for solving discrete structural

optimization problems. Nevertheless, the rounded-off techniques may result in solutions that are far from the optimal point, or even infeasible solutions, especially when dealing with a large number of design variables.

In order to overcome the shortcomings of classical optimization methods, a large number of stochastic optimization techniques, known as metaheuristic optimization algorithms, have been proposed in recent decades. In contrast to most of the classical optimization methods, metaheuristic algorithms do not need the derivatives of the objective and constraint functions, which make them easy to be implemented. Most of the metaheuristic algorithms are inspired by nature and are hence called nature-inspired metaheuristic algorithms. Based on the source of inspiration, nature-inspired metaheuristic algorithms are classified in four major categories: evolutionary-based, swarm-based, human-based, and physics-based. In recent years, many classical population-based metaheuristic algorithms and their variants have been widely applied to solve a large variety of discrete structural optimization problems, such as water cycle algorithm (WCA) and improved mine blast algorithm (IMBA) [4], differential evolution (DE) and adaptive elitist differential evolution (aeDE) [5], biogeography-based optimization and differential evolution (BBO-DE) [6], firefly algorithm (FA), electromagnetism-like algorithm (EM) and Electromagnetism-like Firefly Algorithm (EFA) [7], arithmetic optimization algorithm (AOA) and improved arithmetic optimization algorithm (IAOA) [8], etc.

Although the aforementioned works have overcome many of the restrictions of classical optimization techniques in solving discrete structural optimization problems, structural designers are increasingly faced with the need to adopt more robust and efficient methods for optimizing structural systems with discrete design variables. The Jaya algorithm (JA) is a powerful population-based metaheuristic optimization algorithm proposed based on the simple concept of moving towards the best search agent and getting away from the worst one [9]. The main advantage of JA is that the algorithm does not require any algorithm-specific parameters to be tuned and it requires only the common control parameters of population size and number of iterations, making it easy to be understood and applied. Since its development in 2016, the classical Jaya algorithm and its variants have been successfully applied to a wide variety of optimization problems. However, it has been found that the standard JA suffers from premature convergence and may easily get trapped in local optima [10]. In order to address the drawbacks of the classical JA, different variants of the classical JA have been developed in the literature such as self-adaptive Jaya algorithm [11], chaotic Jaya algorithm (C-Jaya) [12], etc. More recently, a multi-population Jaya algorithm, called set-theoretical Jaya algorithm (ST-JA), has been proposed by Kaveh et al. [13]. The ST-JA, which is inspired by the concepts of set theory, has proven its effectiveness and robustness in solving structural optimization problems with continuous search spaces [13, 14]. However, the ST-JA has not been yet applied to discrete structural optimization. In this chapter, a discrete version of ST-JA is applied to truss optimization problems with discrete design variables.

The rest of this chapter is organized as follows: In Sect. 7.2, the mathematical model of discrete structural optimization is presented. Section 7.3 provides a discussion of the classical JA. In Sect. 7.4, the set-theoretical Jaya algorithm (ST-JA) is

presented. In Sect. 7.5, four benchmark discrete structural optimization problems are investigated and optimization results are discussed. Finally, Sect. 7.6 draws the concluding remarks.

7.2 Structural Optimization with Discrete Design Variables

In general, the objective of structural optimization is to achieve a design that minimizes the structural weight while satisfying all design requirements, such as nodal displacements, member stresses, member buckling, etc., as well as limitations of design variables. In most practical sizing optimization problems of skeletal structures, the cross-sectional areas of structural members are considered as discrete design variable and all of them are selected from a discrete set of available standard cross-sections. Due to simplicity and engineering applicability, the structural members are divided into groups of elements that have the same design variables. In other words, the cross-sectional areas of the members of each group are the same as each other, and consequently the number of design variables is considerably reduced. The truss sizing optimization problem with discrete design variables can be formulated as follows:

$$\text{Find: } \{\boldsymbol{X}\} = \{X_1, X_2, \dots, X_d\} \quad (7.1)$$

$$\text{to minimize: } W(\{\boldsymbol{X}\}) = \sum_{i=1}^m \rho_i A_i L_i \quad (7.2)$$

$$\text{subject to: } \begin{cases} g_n(\{\boldsymbol{X}\}) \leq 0, n = 1, 2, \dots, c \\ X_j^{\min} \leq X_j \leq X_j^{\max}, n = 1, 2, \dots, d \end{cases} \quad (7.3)$$

where $\{\boldsymbol{X}\}$ is the vector of design variables containing the cross-sectional areas of truss members; d is the number of design variables (i.e., the number of element groups); X_j is the cross-sectional area of the j -th element group ($j = 1, 2, \dots, d$); ρ_i , A_i , and L_i are the material density, cross-sectional area, and length of the i -th truss member, respectively; m is the total number of members of the truss; $W(\{\boldsymbol{X}\})$ is the objective function corresponding to the total weight of the truss structure; X_j^{\min} and X_j^{\max} are the lower and upper bounds of the j -th design variable, respectively; $g_n(\{\boldsymbol{X}\})$ represents the n -th design constraint of the problem and c is the number of design constraints of the problem.

As mentioned above, in truss sizing optimization problems with discrete design variables, design variables can only be selected from a discrete set of cross-sections. This can be stated mathematically as follows:

$$X_i \in D_i = \{d_{i,j}; j = 1, 2, \dots, k_i\}; i = 1, 2, \dots, d \quad (7.4)$$

where D_i is the set of discrete sections available for the i -th design variable; k_i is the number of discrete sections available for the i -th design variable and $d_{i,j}$ is the j -th cross-sectional area available for the i -th design variable.

The above minimization problem is a constrained optimization problem. In order to handle the constraints effectively, a popular constraint-handling approach known as the penalty function method is utilized in this study. Through the penalty function method, the design constraints are incorporated in the objective function, and consequently the original constrained optimization problem is converted into an unconstrained one. In this study, a dynamic multiplicative penalty function is employed to handle the design constraints:

$$f_{\text{penalty}}(\mathbf{X}) = (1 + \varepsilon_1 \times v)^{\varepsilon_2}, v = \sum_{n=1}^c v_i \quad (7.5)$$

where $f_{\text{penalty}}(\mathbf{X})$ is the penalty function; v is the sum of constraint violations; c is the number of design constraints and v_i is the value of the i -th design constraint. If the i -th design constraint is violated, then the value of v_i is calculated based on the severity of the violation, otherwise it is set to zero. This can be expressed mathematically as follows:

$$v_i = \begin{cases} g_n(\{\mathbf{X}\}), & \text{if the } n\text{-th design constraint is violated} \\ 0, & \text{otherwise} \end{cases} \quad (7.6)$$

Then the optimization problem presented in Eqs. (7.1), (7.2), and (7.3) can be is rewritten as follows:

$$\text{Find: } \{\mathbf{X}\} = \{X_1, X_2, \dots, X_d\} \quad (7.7)$$

$$\text{to minimize: } P(\{\mathbf{X}\}) = W(\{\mathbf{X}\}) \times f_{\text{penalty}}(\mathbf{X}) \quad (7.8)$$

$$\text{subject to: } X_j^{\min} \leq X_j \leq X_j^{\max}, n = 1, 2, \dots, d \quad (7.9)$$

where $P(\{\mathbf{X}\})$ is the penalized objective function.

It is noted that ε_1 and ε_2 are the parameters of the penalty function and control the severity of penalty imposed on infeasible solutions. These parameters should be adjusted to maintain a proper balance between exploration and exploitation tendencies throughout the search process. For all design optimization problems presented in this study, the parameter ε_1 is set to a constant value while the parameter ε_2 increases linearly with the number of iterations. As a result, as the search process progresses, a relatively larger penalty is imposed on infeasible solutions. This implies that in the early stages of the search process, highly infeasible solutions are also

permitted into the population, which means that search agents have the opportunity to more freely explore the search space. However, as the search process progresses, infeasible solutions with low constraint violations are preferred over those with high violations.

7.3 Classical Jaya Algorithm (JA)

As mentioned before, the Jaya algorithm is a parameter-free metaheuristic optimization algorithm inspired by the simple concept of moving towards the best search agent and getting away from the worst one. In the classical Jaya algorithm, the position of search agents (i.e., candidate solutions) are updated based on the following equation:

$$\begin{aligned} A'(i, j, k) = & A(i, j, k) + r(i, j, 1)(A(i, j, b) - |A(i, j, k)|) \\ & - r(i, j, 2)(A(i, j, w) - |A(i, j, k)|) \end{aligned} \quad (7.10)$$

where i , j , and k denote the indices of iteration numbers, design variables, and solutions, respectively; $A(i, j, k)$ and $A'(i, j, k)$ are the current and new positions of the j -th design variable of the k -th solution of the i -th iteration, respectively; $A(i, j, b)$ and $A(i, j, w)$ are the j -th design variable of the best and worst solutions among the population of solutions of the i -th iteration, respectively; $|A(i, j, k)|$ represents the absolute value of $A(i, j, k)$, and $r(i, j, 1)$ and $r(i, j, 2)$ are uniformly distributed pseudorandom numbers between 0 and 1.

After updating the position of solutions, boundary constraints are imposed as follows:

$$\begin{cases} A'(i, j, k) = UB_j \text{ if } A'(i, j, k) > UB_j \\ A'(i, j, k) = LB_j \text{ if } A'(i, j, k) < LB_j \end{cases} \quad (7.11)$$

where LB_j and UB_j are the lower and upper bounds of the j -th design variable, respectively.

After imposing the boundary constraints, a simple replacement strategy is applied to obtain the population of the next iteration. For this purpose, the penalized objective function value of each new solution is compared with that of the corresponding current solution. If the penalized objective function value of the new solution is better than that of the corresponding current solution, then the current solution is replaced with the new one. Otherwise, the current solution is preserved. Therefore, for a minimization optimization problem, the replacement strategy can be formulated as follows:

$$A(i+1, ., k) = \begin{cases} A'(i, j, k) \text{ if } P(\{A'(i, j, k)\}) < P(\{A(i, j, k)\}) \\ A(i, j, k) \text{ if } P(\{A'(i, j, k)\}) \geq P(\{A(i, j, k)\}) \end{cases} \quad (7.12)$$

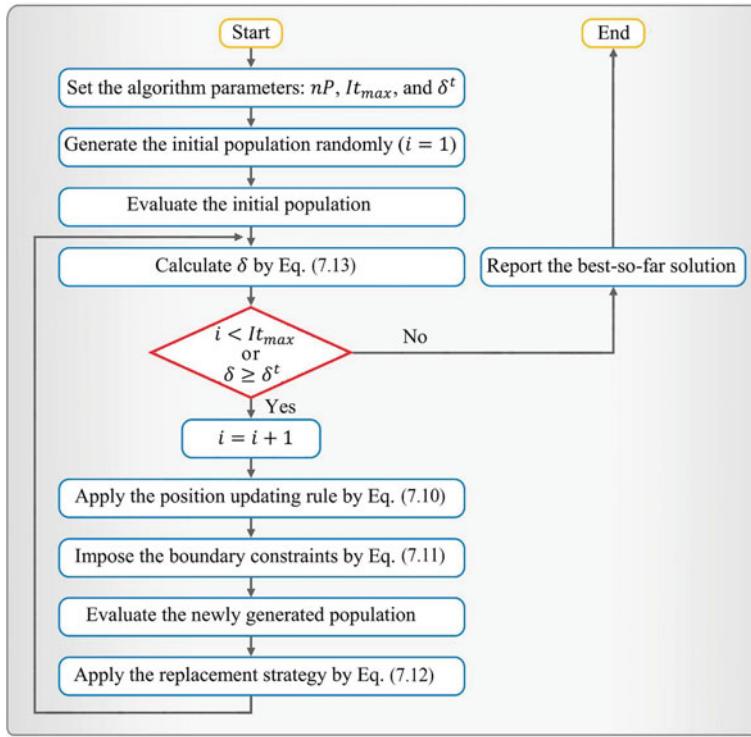


Fig. 7.1 Flowchart of the classical Jaya algorithm

The flowchart of the classical JA is shown in Fig. 7.1. For more detail about the classical JA, readers can refer to [9].

7.4 Set-Theoretical Jaya Algorithm (ST-JA)

As mentioned in the previous section, the classical JA is a simple yet powerful metaheuristic optimization algorithm, which has been widely used to solve various types of optimization problems. However, it has been found that the classical JA suffers from the problem of premature convergence and may easily get trapped in local optima [10]. This drawback is due to the fact that the original JA employs a single learning strategy, which does not allow full use of the population information, and consequently the search space is not explored effectively [10]. To understand this drawback, it is necessary to understand the position updating rule of the classical JA. As it follows from the formulation of the classical JA (please see Eq. (7.10)), the best solution found so far has the key role in the search process of the classical JA, because it draws all other solutions to its own position. In other words, the

first term in Eq. (7.10) (i.e., $+r(i, j, 1)(A(i, j, b) - |A(i, j, k)|)$) implies focusing the search only around the best solution found so far, and the second term (i.e., $-r(i, j, 2)(A(i, j, w) - |A(i, j, k)|)$) implies getting away from the worst solution of the population of the current iteration. This may cause the loss of population diversity and raise the possibility of premature convergence to non-optimal points. Recently, in order to address the drawbacks of the classical JA, a multi-population Jaya algorithm, called set-theoretical Jaya algorithm (ST-JA), has been proposed by Kaveh et al. [13]. The main idea of ST-JA, which is derived from the concepts of set theory, is based on the division of the population of solutions into smaller well-arranged subpopulations. The subpopulations are generated in such a way that the diversity among the solutions of each subpopulation is preserved [13]. In this way, each subpopulation of ST-JA has its own unique best and worst solutions, which guide the search process of all solutions in the subpopulation. This means that the search process of ST-JA is diversified but is still concentrated on different promising regions of the search space, which is expected to be more effective than focusing the search only around the best solution found so far. The ST-JA aims to improve both the exploration and exploitation capabilities of the classical JA. Indeed, in the early stages of the search process, different regions of the search space are explored, which means that diversification is preserved at the initial iterations. Consequently, the possibility of being trapped in local optima is reduced, and in this way, premature convergence is avoided. On the contrary, in the later stages of the search process, the search process is concentrated on the most promising solutions found so far, allowing the intensification of the search. The formulation of the set-theoretical JA for discrete structural optimization is presented in four steps as follows:

Step one (initialization): In the first step, the algorithm parameters including population size (n_P), number of subpopulations (n_S), maximum number of iterations (It_{max}) are adjusted. The convergence threshold (δ^t), which is introduced in step six, is also adjusted. It should be noted that the number of subpopulations must be selected from the set of divisors of the population size. Next, the initial population of solutions is generated randomly within the feasible search space.

Step two (forming the subpopulations): According to a simple procedure proposed by Kaveh et al. [15], the population of solutions is divided into smaller well-arranged subpopulations of the same size. For this purpose, the population is sorted in ascending order of penalized objective function values (for a minimization problem). In the first step of formation of subpopulations, the first n_S solutions of the sorted population are chosen and one solution is allocated randomly to each subpopulation. Then, the next n_S solutions of the sorted population are chosen and one solution is allocated randomly to each subpopulation. This procedure continues until all solutions of the sorted population are allocated to the subpopulations.

Step three (position updating rule): As mentioned earlier, each subpopulation has its own best and worst solutions. Therefore, the position updating rule of Eq. (7.10) is applied to each subpopulation separately.

Step four (boundary constraints): After applying the position updating rule to all subpopulations, boundary constraints are imposed by Eq. (7.11).

Step five (replacement strategy): The replacement strategy is applied to each subpopulation separately to determine its new solutions. It is noted that the new solution of each subpopulation should be compared with its corresponding current solution. The new subpopulations collectively form the population of the next iteration.

Step six (termination criteria): If the termination criteria are satisfied, return the best solution found so far; otherwise go to step two. In this study, the optimization process is terminated when either the value of δ becomes less than the convergence threshold (i.e., when $\delta < \delta^t$), or the number of iterations reaches the maximum number of iterations (i.e., when $It = It_{max}$). The convergence threshold is defined as follows:

$$\delta = \left| \frac{P_{mean}}{P_{best}} - 1 \right| < \delta^t \quad (7.13)$$

where P_{mean} is the average penalized objective function value of the population of the current iteration and P_{best} is the penalized objective function value of the best solution found so far. It is worth mentioning that such a termination criterion allows fewer evaluations of the objective function and can help the algorithm to avoid unnecessary evaluations.

The pseudocode and flowchart of the ST-JA algorithm are shown in Figs. 7.2 and 7.3, respectively.

Pseudocode of set-theoretical Jaya algorithm

Initialization phase

Set the algorithm parameters: population size (nP), the maximum number of iterations (It_{max}), number of subpopulations (nS), and convergence threshold (δ^t)

Generate initial population randomly and evaluate them

Initialize time: $i = 1$

Cyclic body of the algorithm

While $i < It_{max}$ or $\delta \geq \delta^t$

 Sort the population of solutions in ascending order of penalized objective function values

 Form the subpopulations based on the method proposed by Kaveh et al. [15]

For $j = 1: nS$

 Identify the best and worst solutions among the solutions of the j -th subpopulation

For $k = 1: nP/nS$

 Update the k -th candidate solution of the j -th subpopulation based on Eq. (7.10)

 Apply boundary conditions based on Eq. (7.11)

 Evaluate the k -th updated solution of the j -th subpopulation

 Apply replacement strategy based on Eq. (7.12)

End for k

End for j

 Collect the subpopulations and form the population of solutions of the i -th iteration

$i = i + 1$

End while

Output the best solution

Fig. 7.2 Pseudocode of the set-theoretical Jaya algorithm

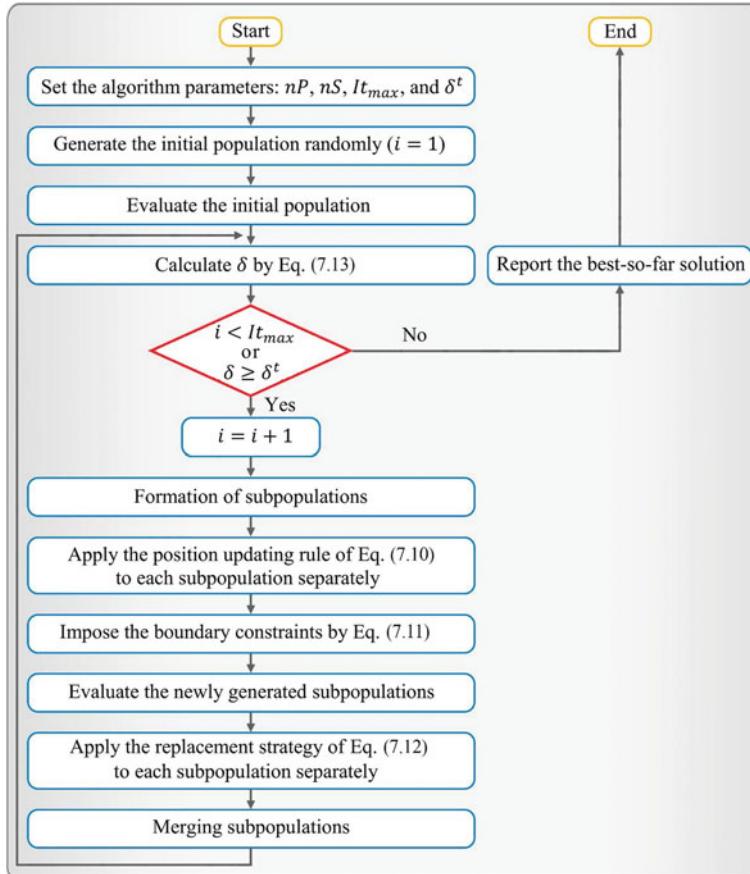


Fig. 7.3 Flowchart of the set-theoretical Jaya algorithm

7.5 Numerical Examples

In this section, to demonstrate the efficiency of the proposed ST-JA in discrete structural optimization, four benchmark truss optimization problems with discrete design variables are studied. The design examples include a 72-bar spatial truss structure with 16 sizing design variables, a 47-bar planar power line tower with 27 sizing design variables, a 52-bar planar truss structure with 12 sizing design variables, and a 160-bar spatial truss structure with 38 sizing design variables. The optimization results obtained by ST-JA are compared with those of the classical JA and some other metaheuristic algorithms in the literature. It should be noted that in order to make a fair comparison, only improved and hybrid metaheuristics are considered for comparison. However, the results of classical single metaheuristics are also discussed in brief. In order to account for the stochastic nature of the optimization process, each

problem was solved 20 times (i.e., 20 independent runs) with both the classical JA and ST-JA. Based on the results of the sensitivity analysis on the size of the population, which will be presented in Sect. 7.5.1.1, the population size is set to $nP = 30$ for both the classical JA and ST-JA in all design examples. Based on the results of the sensitivity analysis on the number of subpopulations on the performance of ST-JA, which will also be presented in Sect. 7.5.1.1, the number of subpopulations is set to $nS = 2$ in all design examples. For all design examples, the value of the convergence threshold is set to $\delta^t = 10^{-5}$ in both the classical JA and ST-JA. However, the maximum number of iterations for both the classical Jaya and ST-JA is set to 400, 160, 133, and 500 for the first, second, third, and fourth design examples, respectively. To evaluate the performances of the classical JA and ST-JA, various statistical results like best, mean, worst, and standard deviation of optimized weights over 20 independent runs are considered. It is noted that only the optimal designs corresponding to the best runs are reported. The number of finite element (FE) analyses to obtain the best optimized weight and the average number of FE analyses are also reported. The Friedman rank test is also performed to rank all considered optimizers on the basis of best, mean, and standard deviation of optimized weights. All the problems and optimization algorithms were implemented in the Matlab environment and structures are analyzed using the direct stiffness method.

7.5.1 A 72-Bar Spatial Truss Structure

The first design example considers the sizing optimization problem of a 72-bar spatial truss structure as shown in Fig. 7.4. The modulus of elasticity is $E = 10000$ ksi and the material density is $\rho = 0.1$ lb/in³. The 72 bars of the truss are categorized into 16 groups corresponding to 16 sizing design variables (see Table 7.5). The design variables must be selected from the discrete set of available cross-sectional areas presented in Table 7.1. The structure is subjected to two loading conditions as shown in Table 7.2. All members of the truss are subjected to stress limitations of ± 25 ksi. The allowable displacement for each node in all coordinate directions is ± 0.25 in.

7.5.1.1 Sensitivity Analysis of the Population Size and the Number of Subpopulations

Before implementing the optimization process, sensitivity analyses need to be carried out to investigate the influence of population size on the performance of both the classical JA and ST-JA. For this purpose, the population size was changed from 10 to 80 for the 72-bar spatial truss problem. In all experiments, the convergence threshold was set to $\delta^t = 10^{-5}$ and the maximum number of FE analyses was set to 6000 except for $nP = 70$, where it is 5950. Table 7.3 provides the results of sensitivity analysis for 20 independent runs. It is noted that all optimized weights reported in Table 7.3 correspond to fully feasible designs. As can be seen, with

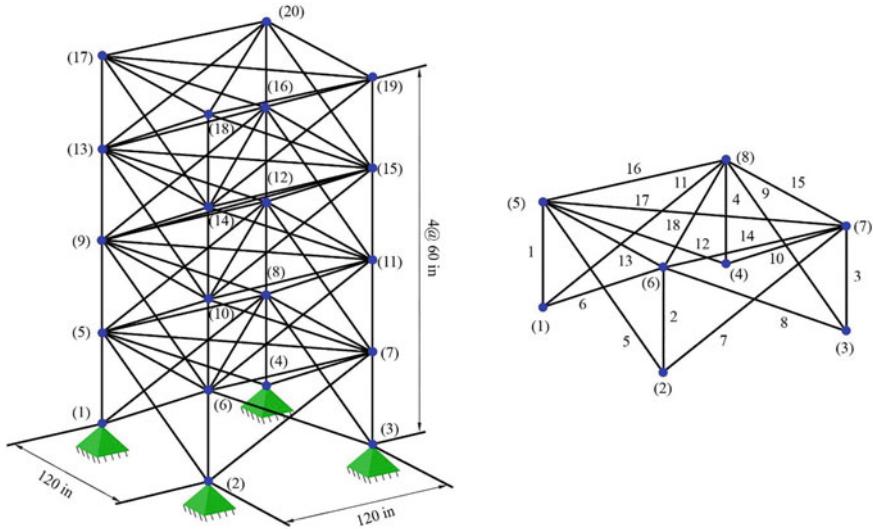


Fig. 7.4 Schematic of the 72-bar spatial truss structure

Table 7.1 Cross-sectional areas provided in the AISC code

Section number	Area (in ²)						
1	0.111	17	1.563	33	3.840	49	11.500
2	0.141	18	1.620	34	3.870	50	13.500
3	0.196	19	1.800	35	3.880	51	13.900
4	0.250	20	1.990	36	4.180	52	14.200
5	0.307	21	2.130	37	4.220	53	15.500
6	0.391	22	2.380	38	4.490	54	16.000
7	0.442	23	2.620	39	4.590	55	16.900
8	0.563	24	2.630	40	4.800	56	18.800
9	0.602	25	2.880	41	4.970	57	19.900
10	0.766	26	2.930	42	5.120	58	22.000
11	0.785	27	3.090	43	5.740	59	22.900
12	0.994	28	3.130	44	7.220	60	24.500
13	1.000	29	3.380	45	7.970	61	26.500
14	1.228	30	3.470	46	8.530	62	28.000
15	1.266	31	3.550	47	9.300	63	30.000
16	1.457	32	3.630	48	10.850	64	33.500

Table 7.2 Load cases for the 72-bar spatial truss problem

Load case	Node	Direction of loading		
		P_x (lb)	P_y (lb)	P_z (lb)
1	17	5000	5000	-5000
2	17	0	0	-5000
	18	0	0	-5000
	19	0	0	-5000
	20	0	0	-5000

a population size of $nP = 30$, the classical JA could obtain the best results in terms of best weight, mean weight, worst weight, and standard deviation. Therefore, in all optimization problems, the population size of both the classical JA and ST-JA is set to $nP = 30$. The same procedure was followed to select the number of subpopulations of the ST-JA. The number of subpopulations was changed from 2 to 15 (i.e., $nS \in \{2, 3, 5, 6, 10, 15\}$) for the 72-bar spatial truss problem. In all experiments, the convergence threshold was set to $\delta^t = 10^{-5}$ and the maximum number of FE analyses was set to 3990. The results of sensitivity analysis for 20 independent runs are provided in Table 7.4. From this table it is observed that best results were obtained for $nS = 2$. Hence, in all optimization problems, the number of subpopulations of the ST-JA is set to $nS = 2$.

7.5.1.2 Comparison with Other Metaheuristic Optimization Algorithms

This problem was previously investigated in the literature by several researchers such as Sadollah et al. [4] WCA and IMBA, Ho-Huu et al. [5] using DE and aeDE, Jalili and Hosseinzadeh [6] using BBO-DE, Le et al. [7] using FA, EM, and EFA, Kaveh and Biabani Hamedani [8] using AOA and IAOA, etc. As expected, experimental results clearly reveal that the proposed ST-JA performs better than classical single metaheuristics including WCA [4], DE [5], FA [7], EM [7], and AOA [8]. Table 7.5 provides a comparison between the optimal design results obtained by the classical JA and ST-JA and those reported in the literature. It is seen that the best weight obtained by both the classical JA and ST-JA (i.e., 389.3342 lb) is the same with those in previous studies. Furthermore, the mean weight given by the ST-JA is better than that of the others. The standard deviation obtained by the ST-JA is 0.3713 lb, which is only slightly higher than that of the IAOA (i.e., 0.3570 lb), but much lower than those of all other methods. In terms of computational cost, the average number of structural analyses of both the classical JA and ST-JA is 3390, which is more than that of the EFA (i.e., 3123), but less than those of all other methods. It is however clear from the results that the ST-JA performs much better than the classical JA and EFA in terms of mean weight, worst weight, and standard deviation. It should be noted that the results reported in [4] do not seem reliable, because the worst weight of the

Table 7.3 Sensitivity analysis of the population size of the classical JA (72-bar spatial truss)

Statistical results	Population size							
	$nP = 10$	$nP = 20$	$nP = 30$	$nP = 40$	$nP = 50$	$nP = 60$	$nP = 70$	$nP = 80$
Best weight (lb)	392.8348	389.3342	389.3342	389.3342	389.4579	390.2702	392.7318	397.0981
Number of FE analyses	2070	2640	3510	4800	5950	5880	4970	5600
Mean weight (lb)	451.6468	396.8250	389.4475	390.1700	392.7082	397.1100	404.5375	413.9212
Worst weight (lb)	698.0100	424.9835	389.8899	391.7342	396.9073	406.3310	418.5384	428.1925
Standard deviation (lb)	72.0566	12.2560	0.1734	0.8641	2.0990	4.1576	6.4468	9.0019
Average number of FE analyses	1842	4553	6000	6000	6000	6000	5950	6000
Maximum number of FE analyses	6000	6000	6000	6000	6000	6000	5950	6000

Table 7.4 Sensitivity analysis of the number of subpopulations of the ST-JA (72-bar spatial truss)

Statistical results	Number of subpopulations					
	$nS = 2$	$nS = 3$	$nS = 5$	$nS = 6$	$nS = 10$	$nS = 15$
Best weight (lb)	389.3342	389.3342	389.3342	389.3342	389.3342	390.2702
Number of FE analyses	3270	3060	2790	2610	3810	3960
Mean weight (lb)	389.6884	390.1812	389.8604	389.9075	390.4513	400.1902
Worst weight (lb)	390.4001	393.3287	393.4450	393.8256	393.1607	424.7292
Standard deviation (lb)	0.3713	1.1561	0.9833	1.0864	1.0455	7.7873
Average number of FE analyses	3990	3990	3972	3969	3990	3990

IMBA (i.e., 389.457 lb) is better than its mean weight (i.e., 389.823 lb), which is impossible. The results of the Friedman rank test for the 72-bar spatial truss problem are provided in Table 7.6. As aforementioned, the results support that ST-JA ranked first in terms of best and mean weights. However, in terms of standard deviation, IAOA ranked first followed by ST-JA. Figure 7.5 compares the optimized weights obtained by the classical JA and ST-JA over 20 runs. The figure shows that the

Table 7.5 Optimal designs of the 72-bar spatial truss obtained by different algorithms

Design variable (area (in^2)), (elements)	Sadollah et al. [4]	Ho-Huu et al. [5]	Jalili and Hosseinzadeh [6]	Le et al. [7]	Kaveh and Biabani Hamedani [8]	This study	
	IMBA	aeDE	BBO-DE	EFA	IAOA	JA	ST-JA
1 (A_1-A_4)	1.990	1.990	1.990	1.990	1.990	1.990	1.990
2 (A_5-A_{12})	0.442	0.563	0.563	0.563	0.563	0.563	0.563
3 ($A_{13}-A_{16}$)	0.111	0.111	0.111	0.111	0.111	0.111	0.111
4 ($A_{17}-A_{18}$)	0.111	0.111	0.111	0.111	0.111	0.111	0.111
5 ($A_{19}-A_{22}$)	1.228	1.228	1.228	1.228	1.228	1.228	1.228
6 ($A_{23}-A_{30}$)	0.563	0.442	0.442	0.442	0.563	0.563	0.442
7 ($A_{31}-A_{34}$)	0.111	0.111	0.111	0.111	0.111	0.111	0.111
8 ($A_{35}-A_{36}$)	0.111	0.111	0.111	0.111	0.111	0.111	0.111
9 ($A_{37}-A_{40}$)	0.563	0.563	0.563	0.563	0.563	0.563	0.563
10 ($A_{41}-A_{48}$)	0.563	0.563	0.563	0.563	0.442	0.442	0.563
11 ($A_{49}-A_{52}$)	0.111	0.111	0.111	0.111	0.111	0.111	0.111
12 ($A_{53}-A_{54}$)	0.111	0.111	0.111	0.111	0.111	0.111	0.111
13 ($A_{55}-A_{58}$)	0.196	0.196	0.196	0.196	0.196	0.196	0.196
14 ($A_{59}-A_{66}$)	0.563	0.563	0.563	0.563	0.563	0.563	0.563
15 ($A_{67}-A_{70}$)	0.391	0.391	0.391	0.391	0.391	0.391	0.391
16 ($A_{71}-A_{72}$)	0.563	0.563	0.563	0.563	0.563	0.563	0.563
Weight (lb)	389.334	389.334	389.33	389.334	389.3342	389.3342	389.3342
Number of FE analyses	6250	4160	5840	2700	3440	3240	3270
Mean weight (lb)	389.823	390.913	390.62	391.376	389.8321	391.5082	389.6884

(continued)

Table 7.5 (continued)

Design variable (area (in^2)), (elements)	Sadollah et al. [4]	Ho-Huu et al. [5]	Jalili and Hosseinzadeh [6]	Le et al. [7]	Kaveh and Biabani Hamedani [8]	This study	
	IMBA	aeDE	BBO-DE	EFA	IAOA	JA	ST-JA
Worst weight (lb)	389.457	393.325	394.98	393.826	390.3524	395.9751	390.4001
Standard deviation (lb)	0.84	1.161	1.43	1.376	0.3570	2.0025	0.3713
Average number of FE analyses	N/A ^a	4101	8000	3123	4000	3990	3990
Number of runs	50	20	30	20	10	20	20

^aNot available

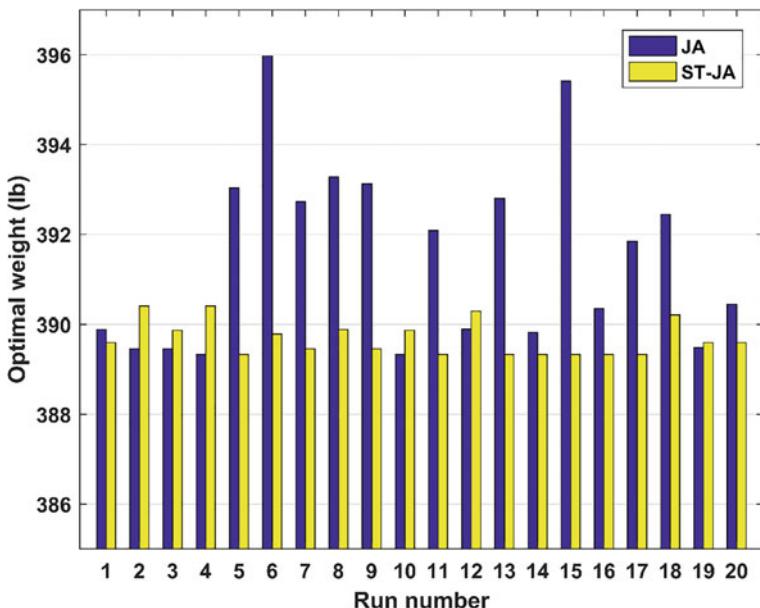
ST-JA consistently converged to optimal or near-optimal solutions. However, it can be seen that the classical JA converged to non-optimal points in some of the runs. Figure 7.6 provides the convergence histories of the average results obtained by the classical JA and ST-JA over 20 runs. It is observed that the ST-JA converges faster than the classical JA. Figures 7.7 and 7.8 show the existing nodal displacements and stress ratios of the problem evaluated at the optimum design obtained by the ST-JA. As expected, none of the design constraints of the problem are violated. It can be concluded from the figures that in the case of the 72-bar spatial truss optimization problem, the displacement constraints are more critical than the stress constraints.

7.5.2 A 47-Bar Planar Power Line Tower

The second design optimization problem investigated in this study is the 47-bar planar power line tower structure illustrated in Fig. 7.9. The material density is $\rho = 0.3 \text{ lb/in}^3$ and the modulus of elasticity is $E = 30000 \text{ ksi}$. The structure includes 47 truss elements categorized into 27 groups of elements (see Table 7.7). The layout of the structure remains fixed during the design procedure. Therefore, this is a sizing optimization problem with 27 design variables corresponding to the cross-sectional areas of the 27 groups of elements. The design variables (i.e., cross-sectional areas of the members) must be selected from the AISC set of 64 discrete values listed in Table 7.1. The structure is subjected to three independent loading conditions as follows: (1) 6 kips acting in the positive x-direction and 14 kips acting in the negative y-direction at nodes 17 and 22; (2) 6 kips acting in the positive x-direction and 14 kips

Table 7.6 The Friedman rank test results for the 72-bar spatial truss problem

	Sadollah et al. [4]	Ho-Huu et al. [5]	Jalili and Hosseinzadeh [6]	Le et al. [7]	Kaveh and Biabani Hamedani [8]	This study	
	IMBA	aeDE	BBO-DE	EFA	IAOA	JA	ST-JA
Friedman rank of best weight	1	1	1	1	1	1	1
Friedman rank of mean weight	2	5	4	6	3	7	1
Friedman rank of standard deviation	3	4	6	5	1	7	2

**Fig. 7.5** Diversity of the optimal weights obtained by the JA and ST-JA for the 72-bar spatial truss

acting in the negative y-direction at node 17; and (3) 6 kips acting in the positive x-direction and 14 kips acting in the negative y-direction at node 22. The structure is optimized for minimum weight subject to stress and buckling constraints. The allowable maximum tensile and compressive stresses of the truss members are set to 20 and 15 ksi, respectively. For compression members, the Euler buckling stress

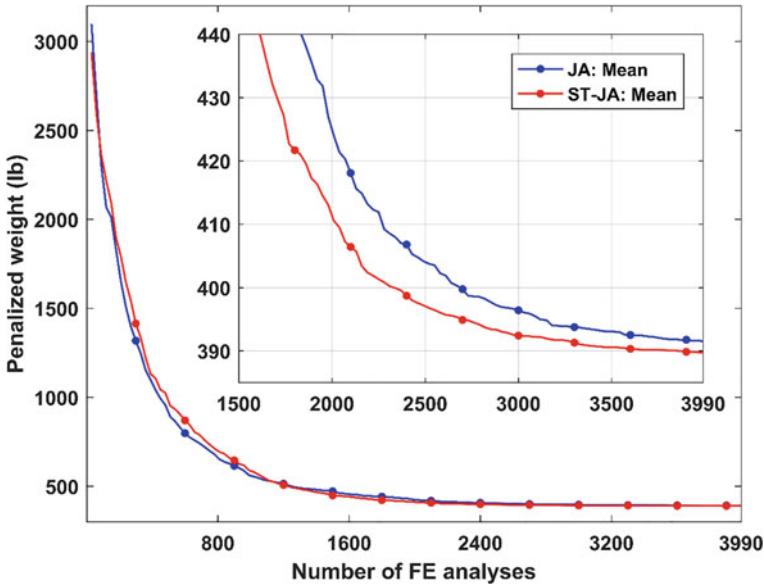


Fig. 7.6 Convergence histories of the JA and ST-JA for the 72-bar spatial truss

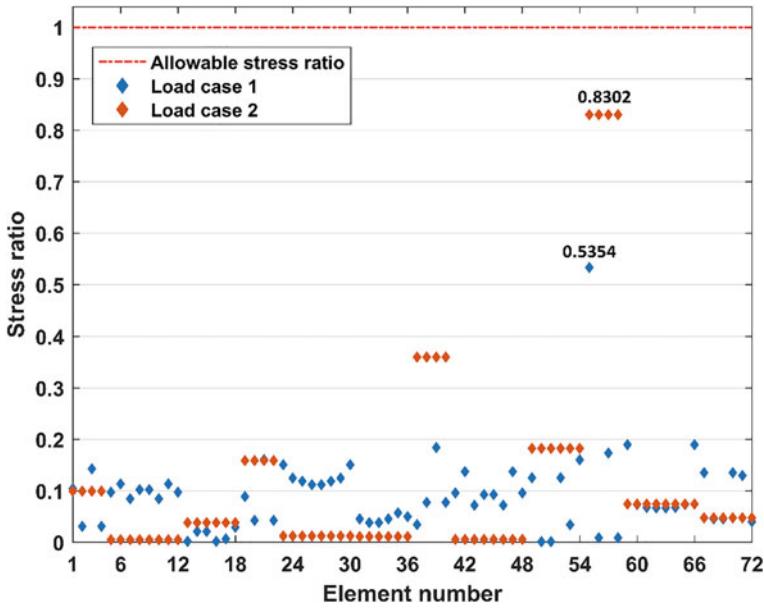


Fig. 7.7 Stress ratios of the 72-bar truss problem evaluated at the optimal design found by the ST-JA

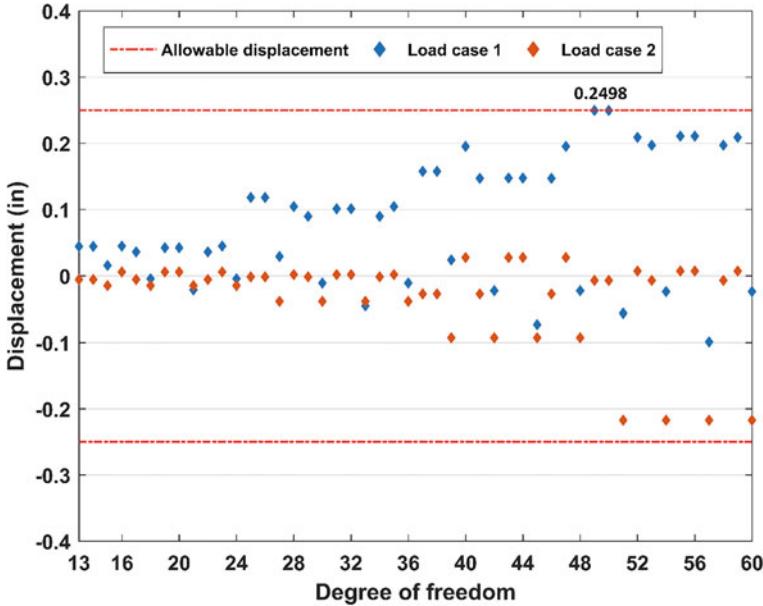


Fig. 7.8 Nodal displacements of the 72-bar truss problem evaluated at the optimal design found by the ST-JA

constraints is also considered. For a member under compression, the Euler buckling stress is calculated as:

$$\sigma_i^b = \frac{-K E A_i}{L_i^2}; i = 1, 2, \dots, 47 \quad (7.14)$$

where E is the modulus of elasticity; A_i and L_i are the cross-sectional area and length of the i -th truss member; σ_i^b is the Euler buckling stress of the i -th truss member and K is a constant value which depends on the cross-sectional geometry (here, $K = 3.96$).

This problem was previously solved by Lee et al. [16] using harmony search (HS), Kaveh and Mahdavi [17] using colliding bodies optimization (CBO), Jalili and Hosseinzadeh [6] using BBO-DE, Degertekin et al. [18] using discrete advanced Jaya algorithm (DAJA), etc. As expected, experimental results show that the proposed ST-JA clearly performs better than classical single metaheuristics including HS [16] and CBO [17]. The results obtained by the classical JA and ST-JA are compared with those reported in the literature in Table 7.7. It can be seen that the ST-JA shows the best performance in terms of best weight, mean weight, worst weight, and standard deviation. In particular, the best weight obtained by the ST-JA is 2372.1499 lb, which is equal to that of the BBO-DE (i.e., 2372.15 lb) and is much lighter than 2376.019

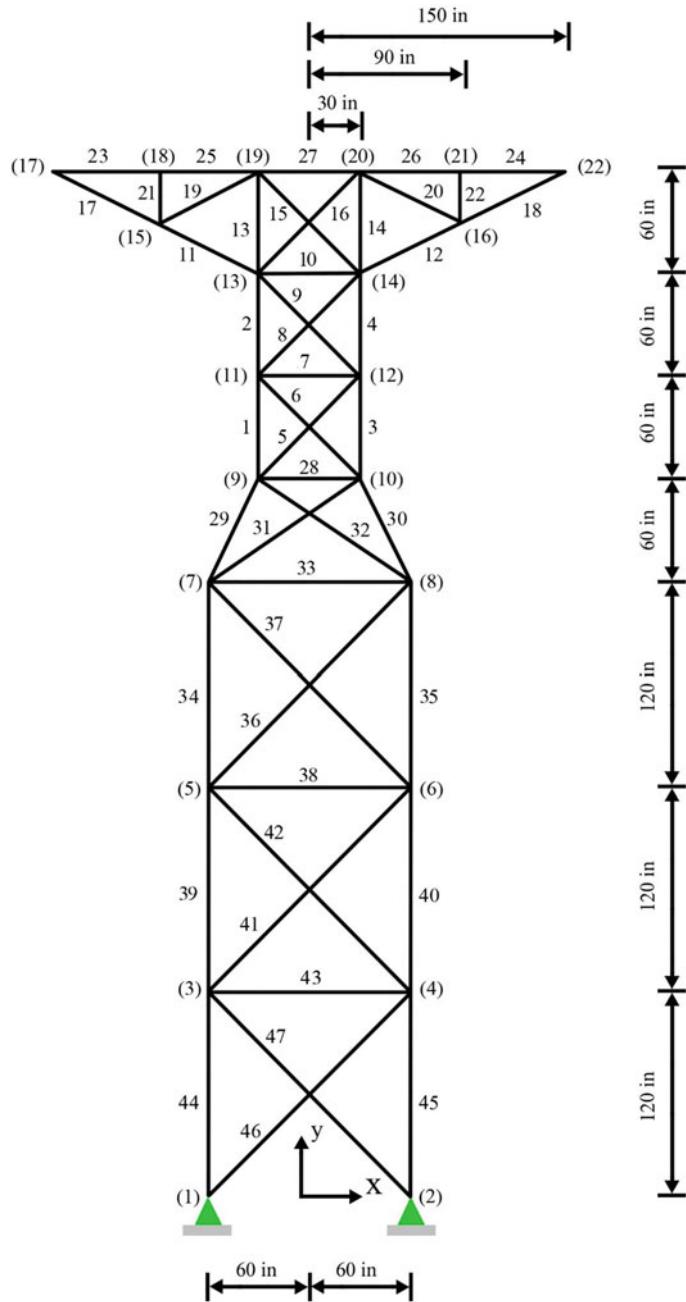


Fig. 7.9 Schematic of the 47-bar planar power line tower

Table 7.7 Optimal designs of the 47-bar planar power line tower obtained by different algorithms

Design variable (area (in^2)), (elements)	Jalili and Hosseinzadeh [6]	Degertekin et al. [18]	This study	
	BBO-DE	DAJA	JA	ST-JA
1 ($A_1 = A_3$)	3.840	3.840	3.840	3.840
2 ($A_2 = A_4$)	3.380	3.380	3.380	3.380
3 ($A_5 = A_6$)	0.766	0.766	0.785	0.766
4 (A_7)	0.111	0.111	0.111	0.111
5 ($A_8 = A_9$)	0.785	0.785	0.785	0.785
6 (A_{10})	1.990	1.990	2.130	1.990
7 ($A_{11} = A_{12}$)	2.130	2.130	2.130	2.130
8 ($A_{13} = A_{14}$)	1.228	1.228	1.228	1.228
9 ($A_{15} = A_{16}$)	1.563	1.563	1.563	1.563
10 ($A_{17} = A_{18}$)	2.130	2.130	2.130	2.130
11 ($A_{19} = A_{20}$)	0.111	0.111	0.141	0.111
12 ($A_{21} = A_{22}$)	0.111	0.111	0.111	0.111
13 ($A_{23} = A_{24}$)	1.800	1.800	1.800	1.800
14 ($A_{25} = A_{26}$)	1.800	1.800	1.800	1.800
15 (A_{27})	1.457	1.457	1.457	1.457
16 (A_{28})	0.563	0.563	0.602	0.563
17 ($A_{29} = A_{30}$)	3.630	3.630	3.630	3.630
18 ($A_{31} = A_{32}$)	1.457	1.457	1.457	1.457
19 (A_{33})	0.250	0.250	0.250	0.250
20 ($A_{34} = A_{35}$)	3.090	3.090	3.090	3.090
21 ($A_{36} = A_{37}$)	1.228	1.266	1.266	1.228
22 (A_{38})	0.307	0.307	0.307	0.307
23 ($A_{39} = A_{40}$)	3.840	3.840	3.840	3.840
24 ($A_{41} = A_{42}$)	1.563	1.563	1.563	1.563
25 (A_{43})	0.141	0.141	0.111	0.141
26 ($A_{44} = A_{45}$)	4.590	4.590	4.590	4.590
27 ($A_{46} = A_{47}$)	1.457	1.457	1.457	1.457
Weight (lb)	2372.15	2376.019	2380.3360	2372.1499
Number of FE analyses	14,400	8046	12,000	9240
Mean weight (lb)	2405.32	2399.92	2404.6938	2386.4209
Worst weight (lb)	2514.91	2418.19	2466.8554	2412.8940
Standard deviation (lb)	28.33	13.15	19.9751	12.1498
Average number of FE analyses	30,000	N/A	12,000	10,717.5
Number of runs	30	20	20	20

Table 7.8 The Friedman rank test results for the 47-bar planar power line tower problem

	Jalili and Hosseinzadeh [6]	Degertekin et al. [18]	This study	
	BBO-DE	DAJA	JA	ST-JA
Friedman rank of best weight	1	3	4	1
Friedman rank of mean weight	4	2	3	1
Friedman rank of standard deviation	4	2	3	1

by DAJA and 2380.3360 by JA. Furthermore, the mean weight obtained by the ST-JA is much lighter than those of the referenced algorithm. In terms of computational cost, the ST-JA requires 9240 structural analyses to obtain its best weight, which is much less than those of the compared methods except for DAJA, which is 8046. However, the best weight obtained by the ST-JA is 3.8691 lb lower than that of the DAJA. The Friedman rank test results for the 47-bar power line tower problem are presented in Table 7.8. The results indicate that ST-JA ranked first in all terms of best weight, mean weight, and standard deviation. It can also be seen that the ST-JA achieves significant improvements in all aspects as compared to the classical JA. The optimized weights obtained by the classical JA and ST-JA over 20 runs are illustrated in Fig. 7.10. The figure indicates higher stability of the ST-JA compared to the classical JA. Figure 7.11 compares the convergence histories obtained for the average results of the classical JA and ST-JA over 20 runs. It demonstrates clearly that the ST-JA converges faster than the classical JA. Figure 7.12 shows the existing stress ratios of the problem evaluated at the optimum design obtained by the ST-JA. As expected, none of the stress constraints of the problem are violated.

7.5.3 A 52-Bar Planar Truss Structure

The third design example considers the sizing optimization problem of the 52-bar planar truss structure shown in Fig. 7.13. The material density and the modulus of elasticity are $\rho = 7860\text{kg/m}^3$ and $E = 207\text{GPa}$, respectively. The 52 members of the truss are divided into 12 groups of elements as shown in Table 7.9. The elements of each group have a same value of the cross-sectional area. The layout of the structure, however, remains fixed during the design process. Hence, this is a sizing optimization problem with 12 design variables. The cross-sectional areas are selected from the set of discrete values listed in Table 7.1. The structure is subjected to a single loading condition consisting of both horizontal and vertical loads, as follows: 100 kN acting in the positive x-direction and 200 kN acting in the positive y-direction at nodes 17 to 20. The maximum allowable stress for all members of the truss is 180 MPa in both tension and compression.

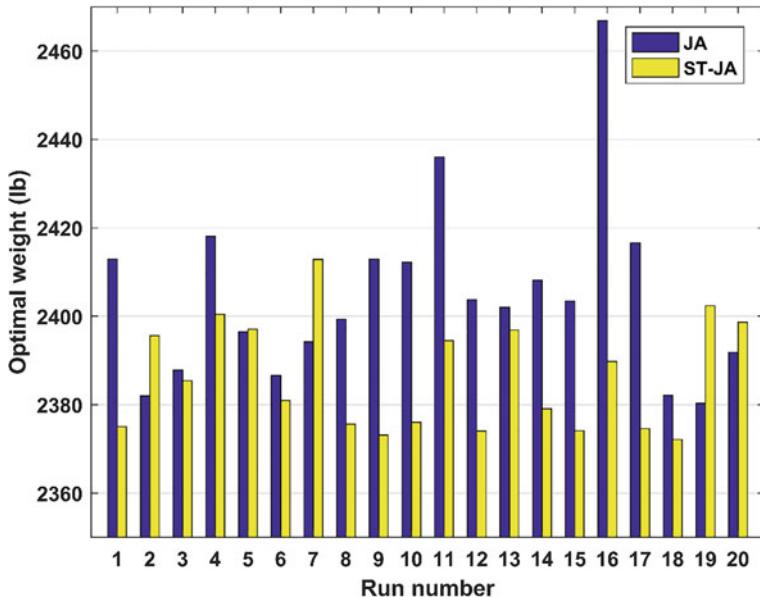


Fig. 7.10 Diversity of the optimal weights obtained by the JA and ST-JA for the 47-bar power line tower

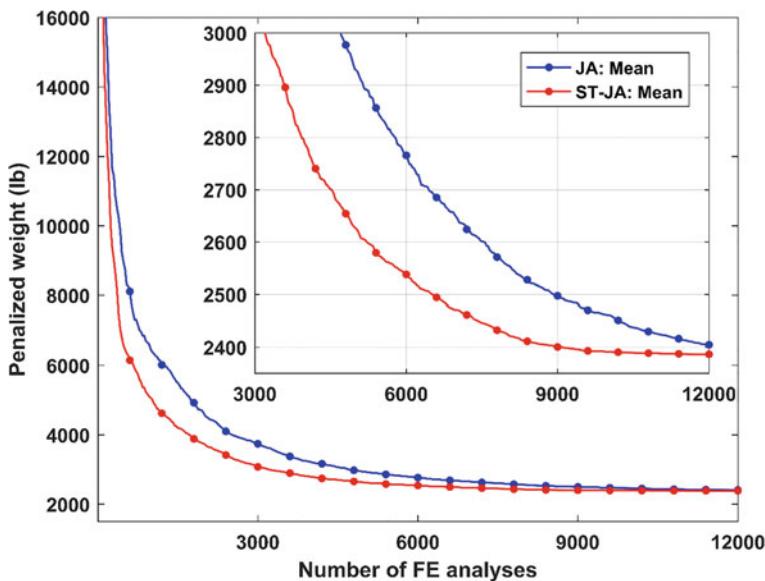


Fig. 7.11 Convergence histories of the JA and ST-JA for the 47-bar planar power line tower

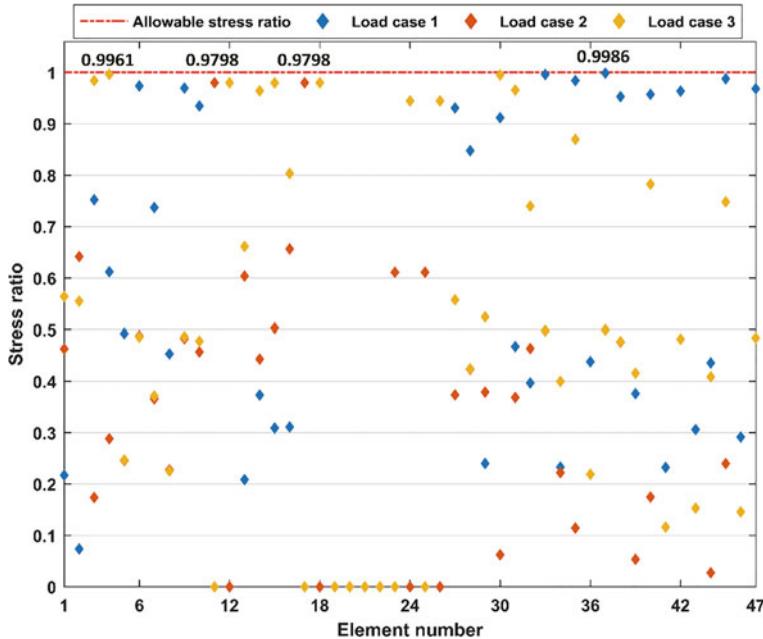


Fig. 7.12 Stress ratios of the 47-bar truss problem evaluated at the optimal design found by the ST-JA

This problem has been previously investigated by many researchers such as Ho-Huu et al. [5] using DE and aeDE, Sadollah et al. [4] using WCA and IMBA, Cheng et al. [19] using hybrid harmony search (HHS), Li et al. [20] using heuristic particle swarm optimization (HPSO), Baghlan et al. [21] using improved firefly algorithm (AFA), etc. Similar to previous examples, the proposed ST-JA outperforms classical single metaheuristics including DE [5] and WCA [4] in both terms of efficiency and accuracy. Table 7.9 provides a comparison between the results obtained by the classical JA and ST-JA and those of other methods in the literature. It should be mentioned that there are studies in the literature which violate the stress constraint, and thus were excluded from the comparison. It can be seen from the table that the best weight obtained by the ST-JA is 1902.6055 kg, which is the same with those of other methods such as aeDE, IMBA, and HHS. The mean weight obtained by the ST-JA is 1904.5320 kg, which is the second-best after that of the IMBA (i.e., 1903.076 kg). The standard deviation of the optimized weight achieved by the ST-JA is 2.5349 kg, which is slightly larger than those of the IMBA and HHS (i.e., 1.13 and 1.309 kg, respectively). In terms of computational cost, the ST-JA requires 4650 structural analyses to reach the optimum design, which is slightly more than those of the HHS and aeDE (i.e., 4523 and 3720 structural analyses, respectively). Table 7.10 presents the Friedman rank test results for the 52-bar planar truss problem. It can be concluded from the table that the ST-JA performs much better than the classical JA in all compared measures. Figure 7.14 compares the diversity of the optimized

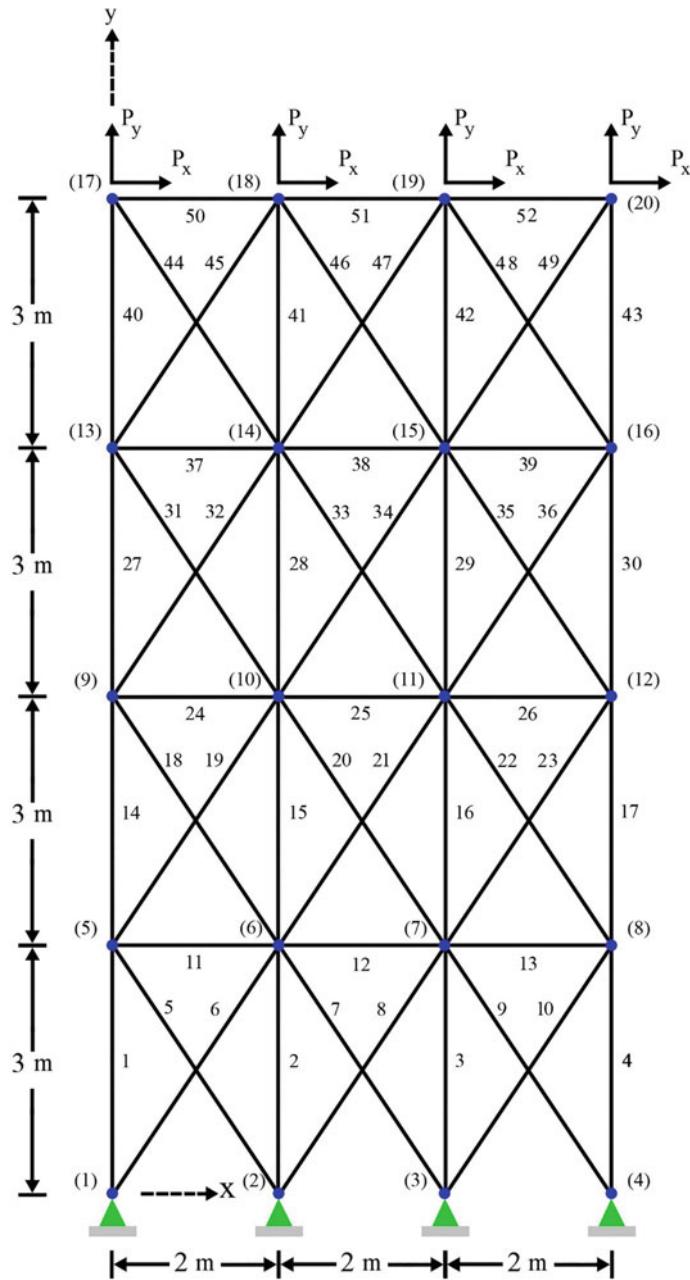


Fig. 7.13 Schematic of the 52-bar planar truss structure

Table 7.9 Optimal designs of the 52-bar planar truss obtained by different algorithms

Design variable (area (in^2)), (elements)	Li et al. [20]	Ho-Huu et al. [5]	Sadollah et al. [4]	Baghlanl et al. [21]	Cheng et al. [19]	This study	
	HPSO	aeDE	IMBA	AFA	HHS	JA	ST-JA
1 (A_1-A_4)	7.220	7.220	7.220	7.220	7.220	7.220	7.220
2 (A_5-A_{10})	1.800	1.800	1.800	1.800	1.800	1.800	1.800
3 ($A_{11}-A_{13}$)	0.563	0.766	0.766	0.563	0.766	0.785	0.766
4 ($A_{14}-A_{17}$)	5.120	5.120	5.120	5.120	5.120	5.120	5.120
5 ($A_{18}-A_{23}$)	1.457	1.457	1.457	1.457	1.457	1.457	1.457
6 ($A_{24}-A_{26}$)	0.766	0.766	0.766	0.766	0.766	0.766	0.766
7 ($A_{27}-A_{30}$)	3.470	3.470	3.470	3.470	3.470	3.550	3.470
8 ($A_{31}-A_{36}$)	1.563	1.563	1.563	1.563	1.563	1.620	1.563
9 ($A_{37}-A_{39}$)	0.602	0.766	0.766	0.994	0.766	0.602	0.766
10 ($A_{40}-A_{43}$)	1.990	1.990	1.990	1.990	1.990	2.380	1.990
11 ($A_{44}-A_{49}$)	1.800	1.800	1.800	1.800	1.800	1.620	1.800
12 ($A_{50}-A_{52}$)	1.228	0.766	0.766	0.766	0.766	0.766	0.766
Weight (kg)	1905.495	1902.605	1902.605	1903.37	1902.605	1913.3007	1902.6055
Number of FE analyses	150,000	3720	4750	52,600	4523	4740	4650
Mean weight (kg)	N/A	1906.735	1903.076	N/A	1904.587	1943.7637	1904.5320
Worst weight (kg)	N/A	1925.714	1904.83	N/A	N/A	2006.7388	1913.3820
Standard deviation (kg)	N/A	6.679	1.13	N/A	1.309	24.9810	2.5349
Average number of FE analyses	N/A	3402	N/A	N/A	5000	4800	4800

(continued)

Table 7.9 (continued)

Design variable (area (in^2)), (elements)	Li et al. [20]	Ho-Huu et al. [5]	Sadollah et al. [4]	Baghlanı et al. [21]	Cheng et al. [19]	This study	
Number of runs	HPSO	aeDE	IMBA	AFA	HHS	JA	ST-JA
N/A	20	50	N/A	30	20	20	

Table 7.10 The Friedman rank test results for the 52-bar planar truss structure

	Li et al. [20]	Ho-Huu et al. [5]	Sadollah et al. [4]	Baghlanı et al. [21]	Cheng et al. [19]	This study	
	HPSO	aeDE	IMBA	AFA	HHS	JA	ST-JA
Friedman rank of best weight	6	1	1	5	1	7	1
Friedman rank of mean weight	N/A	4	1	N/A	3	5	2
Friedman rank of standard deviation	N/A	4	1	N/A	2	5	3

weights gained by the classical JA and ST-JA over 20 independent runs. It is clear from the figure that the ST-JA is much more stable than the classical JA. Figure 7.15 shows the convergence histories of the average results of the classical JA and ST-JA. It can be seen that the convergence rate of the ST-JA is better than that of the classical JA. Figure 7.16 displays the existing member stresses of the 52-bar planar truss evaluated at the optimum design obtained by the ST-JA. As expected, none of the stress constraints are violated.

7.5.4 A 160-Bar Spatial Truss Structure

The 160-bar spatial truss structure shown in Fig. 7.17 is considered as the fourth design example. The material density is $\rho = 0.00785\text{kg/cm}^3$ and the modulus of elasticity is $E = 2.047 \times 10^6\text{kN/mm}^2$, respectively. The nodal coordinates and the connectivity information of the members of the truss are given in Tables 7.11 and 7.12, respectively. Due to the symmetry of the structure, the 160 members of the truss are categorized into 38 element groups as shown in Table 7.12. Each element group contains structural elements with the same cross-sectional area. Thus, this is a sizing

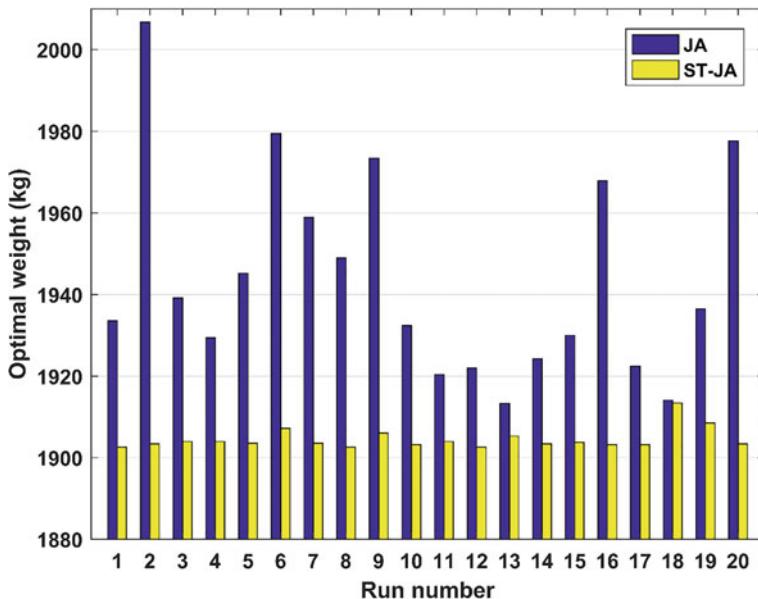


Fig. 7.14 Diversity of the optimal weights obtained by the JA and ST-JA for the 52-bar planar truss structure

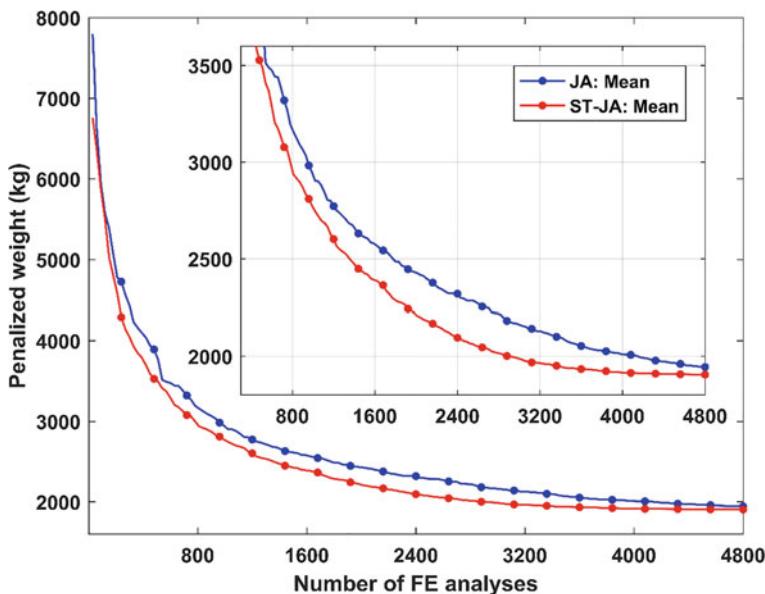


Fig. 7.15 Convergence histories of the JA and ST-JA for the 52-bar planar truss structure

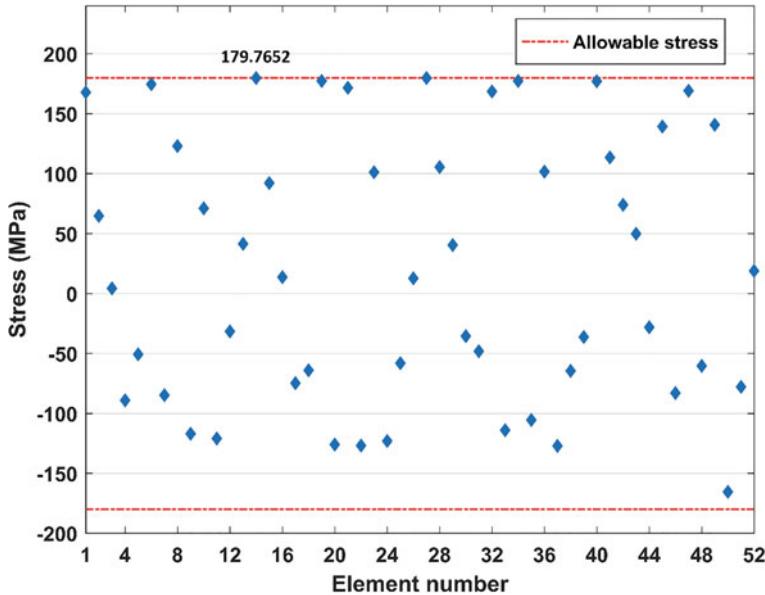


Fig. 7.16 Member stresses of the 52-bar truss problem evaluated at the optimal design found by the ST-JA

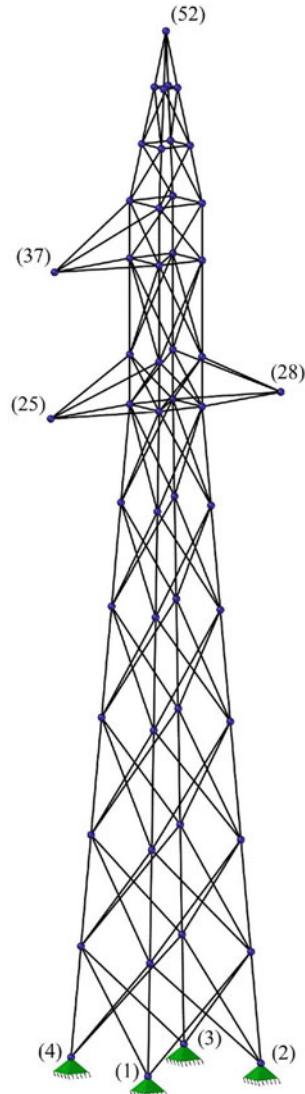
optimization problem with 38 design variables. The available cross-sectional areas together with their corresponding radius of gyration are listed in Table 7.13. The structure is subjected to eight different loading conditions as shown in Table 7.14. A maximum allowable stress of $1500\text{kg}/\text{cm}^2$ is allowed in either tension or compression. In addition, for compression members, the buckling constraint is also taken into account as follows:

$$\sigma_i^b = \begin{cases} 1300 - \frac{(\lambda_i)^2}{24}, & \text{if } \lambda_i \leq 120 \\ \frac{10^7}{(\lambda_i)^2}, & \text{otherwise} \end{cases} \quad (7.15)$$

where λ_i is the slenderness ratio of the i -th member ($\lambda_i = \frac{KL_i}{r_i}$); L_i and r_i are the length and radius of gyration of the i -th member, respectively, and σ_i^b is the buckling stress of the i -th member of the truss.

This problem has been previously studied in the literature by several researchers such as Kaveh et al. [22] using improved shuffled based Jaya (IS-Jaya) algorithm, Groenwold and Stander [23] using selective dynamic rounding (SDR) algorithm, Groenwold et al. [24] using regional genetic algorithm (R-GA), Ho-Huu et al. [5] using DE and aeDE, Capriles et al. [25] using rank-based ant system (RBAS), Le et al. [7] using FA, EM, and EFA, etc. Once again, the results obtained by the proposed ST-JA are better than those obtained by classical single metaheuristics including DE

Fig. 7.17 Schematic of the 160-bar spatial truss structure



[5], FA [7], and EM [7] in both terms of computational cost and accuracy. Table 7.15 compares the optimal design results obtained by the classical JA and ST-JA and those reported in the literature. It can be seen from the table that the best weight obtained by the ST-JA is the same with those of the IS-Jaya [23] and aeDE [5]. However, it is observed that the ST-JA outperform all other considered algorithms in terms of mean weight, worst weight, and standard deviation. In particular, the mean weight obtained by the ST-JA over 20 independent runs is 1337.4593 kg, which is much better than those of the classical JA (i.e., 1390.2129 kg) and other methods in the literature (i.e.,

Table 7.11 Nodal coordinates of the 160-bar spatial truss structure

Node no	X (cm)	Y (cm)	Z (cm)	Node no	X (cm)	Y (cm)	Z (cm)
1	-105	-105	0	27	40	-40	1027.5
2	105	-105	0	28	214	0	1027.5
3	105	105	0	29	40	40	1027.5
4	-105	105	0	30	-40	40	1027.5
5	-93.929	-93.929	175	31	-40	-40	1105.5
6	93.929	-93.929	175	32	40	-40	1105.5
7	93.929	93.929	175	33	40	40	1105.5
8	-93.929	93.929	175	34	-40	40	1105.5
9	-82.859	-82.859	350	35	-40	-40	1256.5
10	82.859	-82.859	350	36	40	-40	1256.5
11	82.859	82.859	350	37	-207	0	1256.5
12	-82.859	82.859	350	38	40	40	1256.5
13	-71.156	-71.156	535	39	-40	40	1256.5
14	71.156	-71.156	535	40	-40	-40	1346.5
15	71.156	71.156	535	41	40	-40	1346.5
16	-71.156	71.156	535	42	40	40	1346.5
17	-60.085	-60.085	710	43	-40	40	1346.5
18	60.085	-60.085	710	44	-26.592	-26.592	1436.5
19	60.085	60.085	710	45	26.592	-26.592	1436.5
20	-60.085	60.085	710	46	26.592	26.592	1436.5
21	-49.805	-49.805	872.5	47	-26.592	26.592	1436.5
22	49.805	-49.805	872.5	48	-12.737	-12.737	1526.5
23	49.805	49.805	872.5	49	12.737	-12.737	1526.5
24	-49.805	49.805	872.5	50	12.737	12.737	1526.5
25	-214	0	1027.5	51	-12.737	12.737	1526.5
26	-40	-40	1027.5	52	0	0	1615

1355.875 for aeDE, 1342.807 for IS-Jaya, 1342.6083 for RBAS, and 1372.551 for EFA). Furthermore, the standard deviation obtained by the ST-JA is 2.1544 kg, which is also much better than those of the classical Jaya and other considered methods. In terms of computational cost, the ST-JA requires only 11,220 structural analyses to achieve the optimal design, while the number of structural analyses obtained by using the aeDE, IS-Jaya, EFA, and classical JA are 23,925, 11,740, 16,870, and 14,970, respectively. Moreover, the average number of structural analyses of the ST-JA is 13432.5, which is much less than those of all other methods. Table 7.16 provides the results of the Friedman rank test for the 160-bar spatial truss problem. As aforementioned, the results support that the ST-JA ranked first in all terms of best weight, mean weight, and standard deviation. Figure 7.18 illustrates the optimized

Table 7.12 The connectivity information and element groups of the 160-bar spatial truss structure

Element no	Node		Element group	Element no		Node	Element group		Element no	Node		Element group	Element no		Node	Element group		
	1	2		1	2		1	2		1	2		1	2		1	2	
1	1	5	1	41	13	18	8	81	25	31	17	121	36	40	29			
2	2	6	1	42	14	17	8	82	28	32	17	122	38	41	29			
3	3	7	1	43	14	19	8	83	28	33	17	123	39	42	29			
4	4	8	1	44	15	18	8	84	25	34	17	124	35	43	29			
5	5	1	6	2	45	15	20	8	85	26	31	18	125	40	41	30		
6	6	2	5	2	46	16	19	8	86	27	32	18	126	41	42	30		
7	7	2	7	2	47	16	17	8	87	29	33	18	127	42	43	30		
8	8	3	6	2	48	13	20	8	88	30	34	18	128	43	40	30		
9	9	3	8	2	49	17	21	9	89	26	32	19	129	35	36	31		
10	10	4	7	2	50	18	22	9	90	27	31	19	130	36	38	31		
11	11	4	5	2	51	19	23	9	91	29	34	19	131	38	39	31		
12	12	1	8	2	52	20	24	9	92	30	33	19	132	39	35	31		
13	13	5	9	3	53	17	22	10	93	27	33	20	133	40	44	32		
14	14	6	10	3	54	18	21	10	94	29	32	20	134	41	45	32		
15	15	7	11	3	55	19	24	10	95	30	31	20	135	42	46	32		
16	16	8	12	3	56	20	23	10	96	26	34	20	136	43	47	32		
17	17	5	10	4	57	18	23	11	97	26	29	21	137	40	45	33		
18	18	6	9	4	58	19	22	11	98	27	30	21	138	41	46	33		
19	19	6	11	4	59	20	21	11	99	31	35	22	139	42	47	33		
20	20	7	10	4	60	17	24	11	100	32	36	22	140	43	44	33		

(continued)

Table 7.12 (continued)

Element no	Node		Element group	Element no		Node	Element group	Element no	Node	Element no		Node	Element group
	1	2		1	2					1	2		
21	7	12	4	61	21	26	12	101	33	38	22	141	44
22	8	11	4	62	22	27	12	102	34	39	22	142	45
23	8	9	4	63	23	29	12	103	33	39	23	143	46
24	5	12	4	64	24	30	12	104	32	35	23	144	47
25	9	13	5	65	21	27	13	105	31	36	23	145	44
26	10	14	5	66	22	26	13	106	34	38	23	146	48
27	11	15	5	67	23	30	13	107	32	38	24	147	45
28	12	16	5	68	24	29	13	108	33	36	24	148	49
29	9	14	6	69	22	29	14	109	34	35	24	149	50
30	10	13	6	70	23	27	14	110	31	39	24	150	46
31	10	15	6	71	24	26	14	111	37	35	25	151	47
32	11	14	6	72	21	30	14	112	37	39	25	152	44
33	11	16	6	73	26	27	15	113	37	40	26	153	48
34	12	15	6	74	27	29	15	114	37	43	26	154	49
35	12	13	6	75	29	30	15	115	35	40	27	155	50
36	9	16	6	76	30	26	15	116	36	41	27	156	51
37	13	17	7	77	25	26	16	117	38	42	27	157	48
38	14	18	7	78	27	28	16	118	39	43	27	158	52
39	15	19	7	79	25	30	16	119	35	38	28	159	50
40	16	20	7	80	29	28	16	120	36	39	28	160	51

Table 7.13 The list of available cross-sections for the 160-bar spatial truss problem

Section number	Area (cm ²)	R (cm) ^a	Section number	Area (cm ²)	R (cm)	Section number	Area (cm ²)	R (cm)
1	1.84	0.47	15	9.40	1.35	29	33.90	2.33
2	2.26	0.57	16	10.47	1.36	30	34.77	2.97
3	2.66	0.67	17	11.38	1.45	31	39.16	2.54
4	3.07	0.77	18	12.21	1.55	32	43.00	2.93
5	3.47	0.87	19	13.79	1.75	33	45.65	2.94
6	3.88	0.97	20	15.39	1.95	34	46.94	2.94
7	4.79	0.97	21	17.03	1.74	35	51.00	2.92
8	5.27	1.06	22	19.03	1.94	36	52.10	3.54
9	5.75	1.16	23	21.12	2.16	37	61.82	3.96
10	6.25	1.26	24	23.20	2.36	38	61.90	3.52
11	6.84	1.15	25	25.12	2.57	39	68.30	3.51
12	7.44	1.26	26	27.50	2.35	40	76.38	3.93
13	8.06	1.36	27	29.88	2.56	41	90.60	3.92
14	8.66	1.46	28	32.76	2.14	42	94.13	3.92

^aRadius of gyration

Table 7.14 Load cases for the 160-bar spatial truss structure

Load case	Node	P_X (N)	P_Y (N)	P_Z (N)	Load case	Node	P_X (N)	P_Y (N)	P_{Zs} (N)
1	52	-868	0	-491	5	52	-917	0	-491
	37	-996	0	-546		37	-951	0	-546
	25	-1091	0	-546		25	-1015	0	-546
	28	-1091	0	-546		28	-636	1259	-428
2	52	-493	1245	-363	6	52	-917	0	-491
	37	-996	0	-546		37	-572	1303	-428
	25	-1091	0	-546		25	-1015	0	-546
	28	-1091	0	-546		28	-1015	0	-546
3	52	-917	0	-491	7	52	-917	0	-491
	37	-951	0	-546		37	-951	0	-546
	25	-1015	0	-546		25	-1015	0	-546
	28	-1015	0	-546		28	-636	1303	-428
4	52	-917	0	-546	8	52	-498	1460	-363
	37	-572	1259	-428		37	-951	0	-546
	25	-1015	0	-546		25	-1015	0	-546
	28	-1015	0	-546		28	-1015	0	-546

weights obtained by the classical JA and ST-JA over 20 runs. The figure indicates that the ST-JA consistently converged to optimal or near-optimal solutions. However, it is seen that the classical JA converged to non-optimal points in most of the runs. Figure 7.19 compares the convergence histories of the average results obtained by the classical JA and ST-JA over 20 runs. It can be observed that the ST-JA converges much faster than the classical JA. Figure 7.20 shows the existing stress ratios of the problem evaluated at the optimum design obtained by the ST-JA. As expected, none of the stress constraints are violated.

7.6 Concluding Remarks

The Jaya algorithm (JA) is a powerful population-based metaheuristic optimization algorithm proposed based on the simple concept of moving towards the best search agent and getting away from the worst one. So far, the classical JA has been widely used to solve various types of optimization problems. However, it has been found that the classical JA suffers from the problem of premature convergence and may easily get trapped in local optima. Recently, in order to address the drawbacks of the classical JA, a multi-population variant of the Jaya algorithm, called set-theoretical Jaya algorithm (ST-JA), has been proposed. The main idea of the ST-JA, which is derived from the concepts of set theory, is based on the division of the population of solutions into smaller well-arranged subpopulations. It follows that different subpopulations have different best and worst solutions. In this way, the ST-JA aims to strengthen both the exploration and exploitation capabilities of the classical JA and strike a balance between them. The ST-JA has proven its effectiveness and robustness in solving structural optimization problems with continuous search spaces. Nevertheless, it has not been yet applied to discrete structural optimization. In this study, the ST-JA is employed to solve truss optimization problems with discrete design variables. The performance of the ST-JA is demonstrated through four well-known truss optimization problems with discrete design variables and its results are compared with those of the classical JA as well as other metaheuristic algorithms in the literature. Numerical results reveal that ST-JA significantly outperforms the classical JA, especially in terms of convergence speed and accuracy, and provides results superior to other state-of-the-art metaheuristics.

Table 7.15 Optimal designs of the 160-bar spatial truss obtained by different algorithms

Design variable (area (cm^2))	Groenwold and Stander [23]	Groenwold et al. [24]	Ho-Huu et al. [5]	Kaveh et al. [22]	Capriles et al. [25]	Le et al. [7]	This study
SDR	R-GA	aeDE	IS-Jaya	RBAS	EFA	JA	ST-JA
1	19.03	19.03	19.03	19.03	19.03	19.03	19.03
2	5.27	5.27	5.27	5.27	5.27	5.27	5.27
3	19.03	19.03	19.03	19.03	19.03	19.03	19.03
4	5.27	5.27	5.27	5.27	5.27	5.27	5.27
5	19.03	19.03	19.03	19.03	19.03	19.03	19.03
6	5.75	5.75	5.75	5.75	5.75	5.75	5.75
7	17.03	15.39	15.39	15.39	15.39	15.39	15.39
8	6.25	5.75	5.75	5.75	5.75	5.75	5.75
9	13.79	13.79	13.79	13.79	13.79	13.79	13.79
10	6.25	5.75	5.75	5.75	5.75	5.75	5.75
11	5.75	5.75	5.75	5.75	5.75	5.75	5.75
12	12.21	13.79	12.21	12.21	12.21	12.21	12.21
13	6.84	6.25	6.25	6.25	6.25	6.25	6.25
14	5.75	5.75	5.75	5.75	5.75	5.75	5.75
15	2.66	2.66	3.88	3.88	3.47	3.88	3.88
16	7.44	7.44	7.44	7.44	7.44	7.44	7.44
17	1.84	1.84	1.84	1.84	1.84	1.84	1.84
18	8.66	8.66	8.66	8.66	9.40	8.66	8.66
19	2.66	2.66	2.66	2.66	2.66	2.66	2.66
20	3.07	3.07	3.07	3.07	3.07	3.07	3.07

(continued)

Table 7.15 (continued)

Design variable (area (cm^2))	Groenwold and Stander [23]	Groenwold and [24]	Ho-Huu et al. [5]	Kaveh et al. [22]	Capriles et al. [25]	Le et al. [7]	This study
SDR	R-GA	aeDE	IS-Jaya	RBAS	EFA	JA	ST-JA
21	2.66	2.66	2.66	3.07	2.66	3.07	2.66
22	8.06	8.06	8.06	8.66	8.06	8.06	8.06
23	5.27	5.27	5.75	5.75	5.75	5.75	5.75
24	7.44	6.25	6.25	6.25	6.25	6.25	6.25
25	6.25	5.75	5.75	5.75	5.75	5.75	5.75
26	1.84	1.84	2.26	2.26	1.84	1.84	2.26
27	4.79	4.79	4.79	4.79	4.79	5.27	4.79
28	2.66	2.66	2.66	2.66	2.66	2.66	2.66
29	3.47	3.47	3.47	3.47	3.47	3.47	3.47
30	1.84	1.84	1.84	1.84	1.84	1.84	1.84
31	2.26	2.26	2.26	2.26	2.26	2.26	2.26
32	3.88	3.88	3.88	3.88	3.88	3.88	3.88
33	1.84	1.84	1.84	1.84	1.84	1.84	1.84
34	1.84	1.84	1.84	1.84	2.26	1.84	2.26
35	3.88	3.88	3.88	3.88	3.88	3.88	3.88
36	1.84	1.84	1.84	1.84	1.84	1.84	1.84
37	1.84	1.84	1.84	1.84	1.84	1.84	1.84
38	3.88	3.88	3.88	3.88	3.88	3.88	3.88
Weight (kg)	1359.781	1337.442	1336.634	1336.7439	1336.704	1340.8854	1336.6341

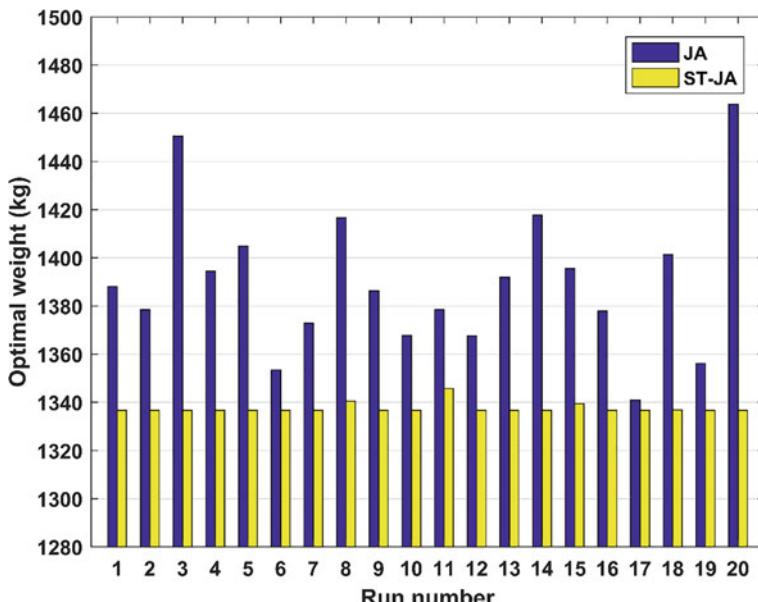
(continued)

Table 7.15 (continued)

Design variable (area (cm^2))	Groenwold and Standler [23]	Groenwold and [24]	Ho-Huu et al. [5]	Kaveh et al. [22]	Capriles et al. [25]	Le et al. [7]	This study
SDR	R-GA	aeDE	IS-Jaya	RBAS	EFA	JA	ST-JA
Number of FE analyses	N/A	N/A	23,925	11,740	N/A	16,870	14,970
Mean weight (kg)	N/A	N/A	1355.875	1342.807	1342.6083	1372.551	1390.2129
Worst weight (kg)	N/A	N/A	1410.611	1366.933	1353.4686	1429.253	1463.6945
Standard deviation (kg)	N/A	N/A	18.805	8.649	N/A	34.706	29.6535
Average number of FE analyses	N/A	N/A	21,265	20,000	90,000	15,183.5	15,000
Number of runs	N/A	N/A	20	20	30	20	20

Table 7.16 The Friedman rank test results for the 160-bar spatial truss structure

	Groenwold and Stander [23]	Groenwold et al. [24]	Ho-Huu et al. [5]	Kaveh et al. [22]	Capriles et al. [25]	Le et al. [7]	This study	
	SDR	R-GA	aeDE	IS-Jaya	RBAS	EFA	JA	ST-JA
Friedman rank of best weight	8	6	1	1	5	4	7	1
Friedman rank of mean weight	N/A	N/A	4	3	2	5	6	1
Friedman rank of standard deviation	N/A	N/A	3	2	N/A	5	4	1

**Fig. 7.18** Diversity of the optimal weights obtained by the JA and ST-JA for the 160-bar spatial truss

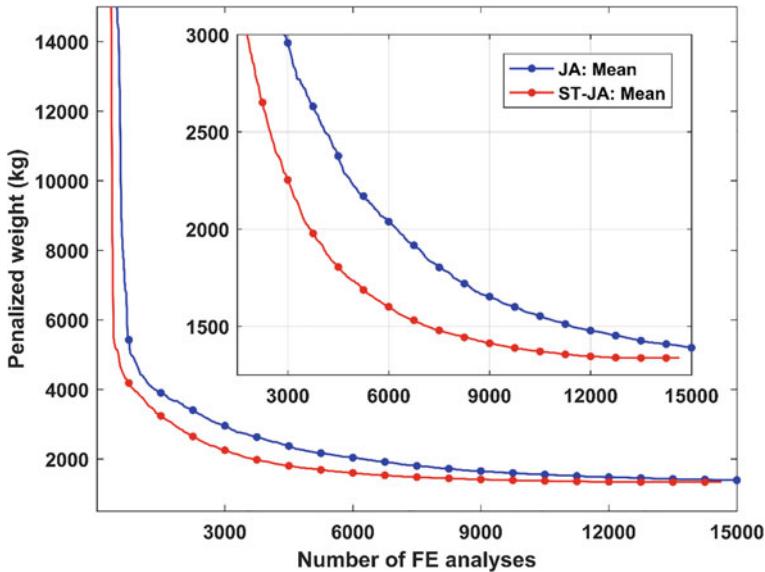


Fig. 7.19 Convergence histories of the JA and ST-JA for the 160-bar spatial truss

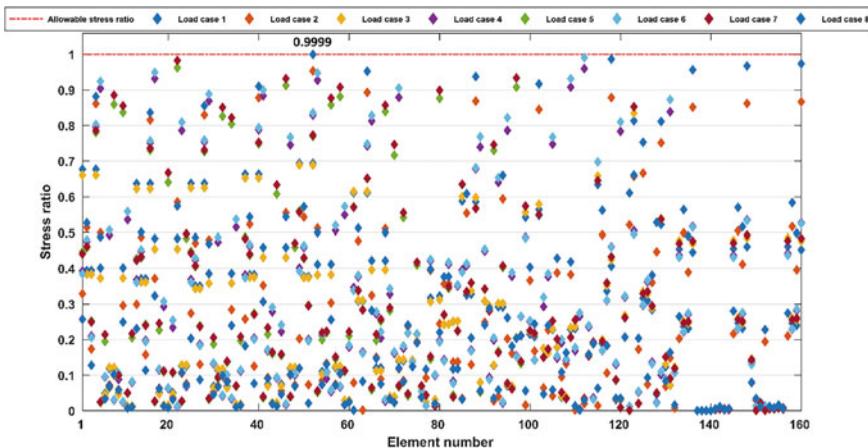


Fig. 7.20 Stress ratios of the 160-bar truss problem evaluated at the optimal design found by the ST-JA

References

1. Kaveh A, Biabani Hamedani K (2022) Discrete structural optimization with set-theoretical Jaya algorithm. *Iran J Sci Technol Trans Civ Eng*. <https://doi.org/10.1007/s40996-022-00868-z>
2. Sivapuram R, Picelli R (2018) Topology optimization of binary structures using integer linear programming. *Finite Elem Anal Des* 139:49–61. <https://doi.org/10.1016/j.finel.2017.10.006>
3. Zhang W, Wang X, Wang Z, Yuan S (2014) Structural optimization of cylinder-crown integrated hydraulic press with hemispherical hydraulic cylinder. *Procedia Eng* 81:1663–1668. <https://doi.org/10.1016/j.proeng.2014.10.209>
4. Sadollah A, Eskandar H, Bahreininejad A, Kim JH (2015) Water cycle, mine blast and improved mine blast algorithms for discrete sizing optimization of truss structures. *Comput Struct* 149:1–16. <https://doi.org/10.1016/j.compstruc.2014.12.003>
5. Ho-Huu V, Nguyen-Thoi T, Vo-Duy T, Nguyen-Trang T (2016) An adaptive elitist differential evolution for optimization of truss structures with discrete design variables. *Comput Struct* 165:59–75. <https://doi.org/10.1016/j.compstruc.2015.11.014>
6. Jalili S, Hosseinzadeh Y (2018) Design optimization of truss structures with continuous and discrete variables by hybrid of biogeography-based optimization and differential evolution methods. *Struct Des Tall Spec Build* 27(14):e1495. <https://doi.org/10.1002/tal.1495>
7. Le DT, Bui DK, Ngo TD, Nguyen QH, Nguyen-Xuan H (2019) A novel hybrid method combining electromagnetism-like mechanism and firefly algorithms for constrained design optimization of discrete truss structures. *Comput Struct* 212:20–42. <https://doi.org/10.1016/j.compstruc.2018.10.017>
8. Kaveh A, Hamedani KB (2022) Improved arithmetic optimization algorithm and its application to discrete structural optimization. *Structures* 35:748–764. <https://doi.org/10.1016/j.istruc.2021.11.012>
9. Jaya R (2016) A simple and new optimization algorithm for solving constrained and unconstrained optimization problems. *Int J Ind Eng Comput* 7(1):19–34. <https://doi.org/10.5267/j.ijiec.2015.8.004>
10. Zhang Y, Chi A, Mirjalili S (2021) Enhanced Jaya algorithm: A simple but efficient optimization method for constrained engineering design problems. *Knowl Based Syst* 233:107555. <https://doi.org/10.1016/j.knosys.2021.107555>
11. Rao RV, More KC (2017) Design optimization and analysis of selected thermal devices using self-adaptive Jaya algorithm. *Energy Convers Manag* 140:24–35. <https://doi.org/10.1016/j.enercon.2017.02.068>
12. Farah A, Belazi A (2018) A novel chaotic Jaya algorithm for unconstrained numerical optimization. *Nonlinear Dyn* 93(3):1451–1480. <https://doi.org/10.1007/s11071-018-4271-5>
13. Kaveh A, Biabani Hamedani K, Kamalinejad M (2021) Set theoretical variants of optimization algorithms for system reliability-based design of truss structures. *Period Polytech Civ Eng* 65(3):717–729. <https://doi.org/10.3311/PPci.17519>
14. Kaveh A, Biabani Hamedani K, Joudaki A, Kamalinejad M (2021) Optimal analysis for optimal design of cyclic symmetric structures subject to frequency constraints. *Structures* 33:3122–3136. <https://doi.org/10.1016/j.istruc.2021.06.054>
15. Kaveh A, Biabani Hamedani K, Zaerreza A (2021) A set theoretical shuffled shepherd optimization algorithm for optimal design of cantilever retaining wall structures. *Eng Comput* 37:3265–3282. <https://doi.org/10.1007/s00366-020-00999-9>
16. Lee KS, Geem ZW, Lee SH, Bae KW (2005) The harmony search heuristic algorithm for discrete structural optimization. *Eng Optim* 37(7):663–684. <https://doi.org/10.1080/03052150500211895>
17. Kaveh A, Mahdavi VR (2014) Colliding bodies optimization method for optimum discrete design of truss structures. *Comput Struct* 139:43–53. <https://doi.org/10.1016/j.compstruc.2014.04.006>
18. Degertekin SO, Lamberti L, Ugur IB (2019) Discrete sizing/layout/topology optimization of truss structures with an advanced Jaya algorithm. *Appl Soft Comput* 79:363–390. <https://doi.org/10.1016/j.asoc.2019.03.058>

19. Cheng MY, Prayogo D, Wu YW, Lukito MM (2016) A Hybrid Harmony Search algorithm for discrete sizing optimization of truss structure. *Autom Constr* 69:21–33. <https://doi.org/10.1016/j.autcon.2016.05.023>
20. Li LJ, Huang ZB, Liu F (2009) A heuristic particle swarm optimization method for truss structures with discrete variables. *Comput Struct* 87(7–8):435–443. <https://doi.org/10.1016/j.compstruc.2009.01.004>
21. Baghlanl A, Makiabadi MH, Sarcheshmehpour M (2014) Discrete optimum design of truss structures by an improved firefly algorithm. *Adv Struct Eng* 17(10):1517–1530. <https://doi.org/10.1260/1369-4332.17.10.1517>
22. Kaveh A, Hosseini SM, Zaerreza A (2021) Improved Shuffled Jaya algorithm for sizing optimization of skeletal structures with discrete variables. *Structures* 29:107–128. <https://doi.org/10.1016/j.istruc.2020.11.008>
23. Groenwold AA, Stander N (1997) Optimal discrete sizing of truss structures subject to buckling constraints. *Struct Optim* 14(2):71–80. <https://doi.org/10.1007/BF01812508>
24. Groenwold AA, Stander N, Snyman JA (1999) A regional genetic algorithm for the discrete optimal design of truss structures. *Int J Numer Methods Eng* 44(6):749–766. [https://doi.org/10.1002/\(SICI\)1097-0207\(19990228\)44:6%3C749::AID-NME523%3E3.0.CO;2-F](https://doi.org/10.1002/(SICI)1097-0207(19990228)44:6%3C749::AID-NME523%3E3.0.CO;2-F)
25. Capriles PV, Fonseca LG, Barbosa HJ, Lemonge AC (2007) Rank-based ant colony algorithms for truss weight minimization with discrete variables. *Commun Numer Methods Eng* 23(6):553–575. <https://doi.org/10.1002/cnm.912>

Chapter 8

Enhanced Forensic-Based Investigation Algorithm



8.1 Introduction

In this chapter, an enhanced version of the forensic-based investigation (FBI) algorithm called enhanced forensic-based investigation (EFBI) is proposed and then applied to the optimal design of dome-like truss structures with frequency constraints [1]. The proposed EFBI aims to reinforce the relationship between exploration and exploitation strategies of the standard FBI. In order to demonstrate the performance of the proposed EFBI algorithm, three numerical examples are investigated and the results obtained by EFBI are compared to those of the standard FBI and other optimization methods in the literature. The results show superior performance of the EFBI algorithm compared to the standard FBI algorithm and other state-of-the-art metaheuristics.

In free vibration of a structure, natural frequencies are the fundamental characteristics affecting the dynamic behavior of the structure [2]. As an example, the dynamic response of a low frequency vibration system is mainly a function of the fundamental natural frequency of the system [3]. In such cases, the dynamic characteristics of the structure can be significantly improved by manipulating the selected frequency. This aim can be achieved through the optimal design of structures with frequency constraints. As a result, structural optimization with frequency constraints makes it possible to manipulate the dynamic characteristics in a variety of ways. For example, in designing a spaceship, it is necessary to impose constraints on several of the lowest natural frequencies of the vehicle to prescribed ranges natural in order to avoid resonance phenomenon.

Optimum design of structures considering frequency constraints has attracted the attention of many researchers since the 1980s. Bellagamba and Yang [4] applied a nonlinear programming procedure to the minimum mass design of truss structures under static thermal and mechanical loads considering various constraints, including fundamental natural frequency and local buckling. Grandhi and Venkayya [3] employed an optimality criterion method based on uniform Lagrangian density for the

minimum weight design of structures with multiple frequency constraints. Tong and Liu [5] proposed a dynamically constrained optimization procedure for the optimal design of truss structures with discrete design variables under dynamic constraints. Gomes [6] employed particle swarm optimization (PSO) algorithm for size and shape optimization of truss structures taking into account frequency constraints. Kaveh and Zolghadr [7] proposed a hybridization of the charged system search (CSS) and the Big Bang-Big Crunch (BB-BC) algorithms with trap recognition capability and applied it to the optimization of truss structures with frequency constraints. Khatibnia and Naseralavi [2] introduced the orthogonal multi-gravitational search algorithm (OMGSA) to solve size and shape truss optimization problems with frequency constraints. Kaveh and Ilchi Ghazaan [8] performed the optimal design of large-scale dome structures with multiple natural frequency constraints. Kaveh and Zolghadr [9] utilized the cyclical parthenogenesis algorithm (CPA) for optimal design of cyclically symmetric trusses with frequency constraints.

Frequency-constrained optimization problems are well-known as highly nonlinear, non-convex, and multimodal optimization problems with respect to the design variables. The most common difficulty with the frequency-constrained optimization problems is the switching of vibration modes due to structural size and shape modifications, which may cause convergence difficulties. As a result, classical methods of optimization based on gradients may not be effective to solve this type of optimization problems because these methods require the gradient information of the frequency with respect to the design variables. Thus, metaheuristic optimization algorithms can be regarded as appropriate alternatives.

Metaheuristic optimization algorithms represent a branch of approximate optimization techniques that have been one of the most active fields of research in computer science over the last years. These techniques have the capability to find optimal or near-optimal solutions to tough optimization problems and even NP-hard problems within a reasonable computational time. Metaheuristics have found a wide range of engineering applications because they: (1) are easy to design and implement; (2) use no gradient information during the optimization process; and (3) are applicable to a large variety of optimization problems. Over the past three decades, numerous metaheuristics have been developed, most of which are nature-inspired. Nature-inspired metaheuristic algorithms can be categorized into four main groups (see Fig. 8.1): evolution-based, physics-based, swarm-based, and human-based [10]. Recently, the forensic-based investigation algorithm is developed based on the suspect investigation-location-pursuit process [11]. There are two teams in the criminal investigation process, one is an investigation team, and the other is a pursuit team. Accordingly, FBI has two phases, i.e., investigation and pursuit phases, each of which has its own distinct population of solutions. In other words, each team continues independently the search process. The teams interact with each other only through the best solution found so far. Poor communication between the investigation and pursuit teams causes the convergence rate of FBI to be decreased.

In this chapter, the enhanced forensic-based investigation (EFBI) is proposed for frequency-constrained truss optimization problems. The proposed EFBI algorithm is designed in order to eliminate the drawback of the standard FBI as mentioned

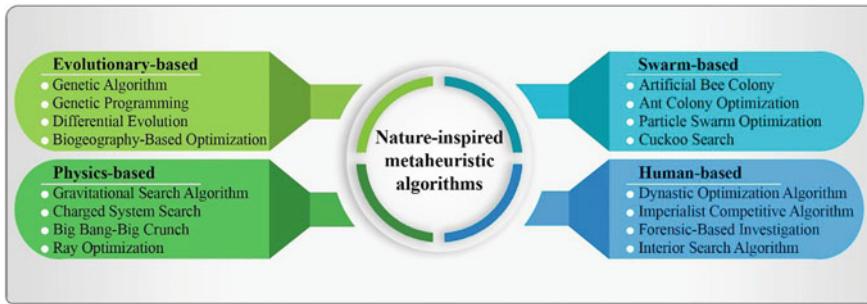


Fig. 8.1 Classification of nature-inspired metaheuristic algorithms

above, and improve the balance between the intensification and the diversification tasks (i.e., more effective interaction between the investigation and pursuit teams). To achieve this purpose, the investigation and pursuit teams are assumed to work in close cooperation with each other, so that each team starts its own search process with the output of the previous phase. In order to demonstrate the efficiency and capability of EFBI, three size and shape dome-like truss optimization problems with multiple frequency constraints are considered from literature. The optimization results are compared with those in the literature. The results reveal that the proposed EFBI significantly performs better than the standard FBI in terms of accuracy, reliability, and convergence rate.

The rest of this chapter is organized as follows: The standard FBI is reviewed in Sect. 8.2. In Sect. 8.3, after pointing out the drawbacks of the standard FBI, an enhanced version of the FBI algorithm is proposed. In Sect. 8.4, truss optimization problem with natural frequency constraints is stated. Three numerical examples are studied in Sect. 8.5. Finally, some concluding remarks are provided in Sect. 8.6.

8.2 Forensic-Based Investigation (FBI) Algorithm

The forensic-based investigation (FBI) is a parameter-free human-based metaheuristic optimization algorithm developed by Chou and Nguyen in 2020 [11]. The algorithm is inspired by the forensic investigation process in criminal cases. The forensic investigation process is described below.

8.2.1 *Forensic Investigation Process*

The process of large-scale criminal investigations can be seen as a process composed of five steps: start of the investigation, interpretation of the findings, directions of inquiry, actions, and prosecution of a suspect. The middle three steps of the criminal



Fig. 8.2 Steps of the criminal investigation process

investigation process can be seen as a cycle. Figure 8.2 shows steps of the criminal investigation process. This cyclic process starts as soon as a crime is reported and ends when the truth is discovered. Immediately after a crime is reported, a criminal investigation team is formed to start the investigation process. The team investigates the crime scene and the victim with a number of standard procedures. In the first step of the investigation process, initial information about the crime is provided, which is a starting point for the investigators. In the second step, members of the investigation team interpret the findings and share their points of view with each other. Next, based on the interpretation of the findings, the team members suggest several directions of inquiry. The team reviews the findings and sees whether they match with existing directions of inquiry. As a result, new directions of inquiry may be suggested, and existing ones may be changed, ignored, or confirmed. In the next step of a criminal investigation, the team decides about further actions that need to be taken. The investigation team usually investigates the most promising direction of research first. The actions taken result in new clues and information. As the new findings are collected, the team members interpret them, which may result in new several directions of inquiry and actions. This process continues until the truth is discovered (i.e., when a serious suspect is identified and prosecuted).

8.2.2 Mathematical Model

The standard FBI is designed in two main search phases: the investigation phase and the pursuit phase. The investigation phase deals with the diversification of the search, while the pursuit phase ensures the intensification task. Search space of the algorithm is defined as anywhere the suspect might be hiding, and the probability of locating the suspect serves as a metaphor for the objective function. In the following steps, the optimization problem is defined in the maximization form of the objective

function. Thus, the more the value of the objective function means the better chance of locating the suspect. In order to start the criminal investigation process, a number of initial random suspected locations are considered as follows:

$$X_{A_{ij}} = LB_j + rand_{ij} \times (UB_j - LB_j); i = 1, 2, \dots, NP \text{ and } j = 1, 2, \dots, D \quad (8.1)$$

where NP represents the number of suspected locations (i.e., population size); D is the number of design variables of the problem; LB and UB denote the lower and upper bound vectors of design variables; $rand$ is a random number between 0 and 1; and X_{A_i} determines the i -th possible location of the suspect, which hereafter is called i -th suspected location.

8.2.2.1 Investigation Phase (A)

Investigation phase includes the steps of: (1) interpretation of the findings; and (2) directions of inquiry. Immediately after a crime is detected, a team of investigators is formed to realize what might have happened. The criminal investigation team starts the investigation by use of the information relating to the crime scene. The investigation phase is carried out in two steps as follows:

Interpretation of the Findings (A_1): Members of the investigation team interpret their findings and discuss different points of view about the case. Interaction between members of the investigation team influences the interpretation of the findings. The team identifies new suspected locations in light of others. This step is modeled as follows [11]:

$$\begin{aligned} X'_{A_{ij}} &= X_{A_{ij}} + rand_{ij} \times (X_{A_{kj}} - (X_{A_{kj}} + X_{A_{hj}})/2); \\ i &= 1, 2, \dots, NP \text{ and } j = 1, 2, \dots, D \end{aligned} \quad (8.2)$$

where i , k , and h are three possible locations of the suspect, where k and h are chosen randomly; $rand$ is a random number in the range [-1, 1]; and X'_{A_i} -s are newly identified suspected locations ($i = 1, 2, \dots, NP$).

The team investigates newly identified suspected locations and compares them with the current ones. The current suspected locations are replaced with the corresponding new ones via a greedy manner so that the location where the suspect is more likely to be hiding is preferred. Furthermore, the investigation team determines the best-so-far suspected location (X_{Best}) for further investigations. This can be written as:

$$X_{A_i} = X'_{A_i}, P'_{A_i} > P_{A_i}; i = 1, 2, \dots, NP \quad (8.3)$$

$$X_{Best} = X'_{A_i}, P'_{A_i} > P_{Best}; i = 1, 2, \dots, NP \quad (8.4)$$

where X_{Best} represents the best-so-far suspected location; P_{Best} is the probability of locating the suspect at X_{Best} ; and P_{A_i} is the probability of locating the suspect at X_{A_i} (representing the objective function of solution X_{A_i}). In a similar way, P'_{A_i} is defined as the probability of locating the suspect at X'_{A_i} .

The outputs of this step are X_{Best} and X_A . To decide which step to take next, the most and least probabilities of locating the suspect (P_{max} and P_{min}) are determined and compared. If $P_{min} \neq P_{max}$, the algorithm goes to step A₂; otherwise, the algorithm goes to step B₁.

Directions of Inquiry (A₂): In this step of the investigation process, the investigation team suggests several directions of inquiry based on the outcome of the previous step. The team discusses the motives for the crime and considers different scenarios. Next, the team members confront the findings with the directions of inquiry they believe, which may lead to constructing new directions of inquiry or a change or confirmation of the existing directions of inquiry. This phase can be expressed as follows [11]:

$$X'_{A_{ij}} = \begin{cases} X_{Best_j} + X_{Adj} + rand_{ij} \times (X_{A_{kj}} - X_{A_{hj}}), & rand > Pr(X_{A_i}) \\ X_{A_{ij}}, & \text{otherwise} \end{cases} \quad (8.5)$$

where d , k , h , and i are four possible locations of the suspect, where d , k , and h are chosen randomly; $rand$ is a random number between 0 and 1; and $Pr(X_{A_i})$ is the relative probability of locating the suspect at X_{A_i} . The variables i and j are defined same as those in Eq. (8.1).

It can be seen from Eq. (8.5) that a suspected location is updated only if it has a relative probability greater than $rand$; otherwise, it remains unchanged. The relative probability of locating the suspect at X_{A_i} can be calculated as [11]:

$$r(X_{A_i}) = \frac{P_{A_i} - P_{min}}{P_{max} - P_{min}}; i = 1, 2, \dots, NP \quad (8.6)$$

where P_{max} is the highest probability of locating the suspect (corresponding to the best objective function value); and P_{min} is the lowest probability of locating the suspect (corresponding to the worst objective function value).

Again, the current suspected locations ($X_{A_i}; i = 1, 2, \dots, NP$) are compared and replaced with the corresponding new ones ($X'_{A_i}; i = 1, 2, \dots, NP$) by using Eq. (8.3). The best-so-far suspected location is also updated by using Eq. (8.4).

8.2.2.2 Pursuit Phase (B)

The pursuit phase also includes two steps, both of which model the last step of the investigation process, i.e. actions. The members of the pursuit team obey the commands of their headquarters and report back to them. Throughout the investigation process, the investigation and pursuit teams need to work closely together to resolve complicated cases. For this purpose, the investigation team directs the pursuit team towards the best-so-far suspected location. Reciprocally, members of the pursuit team frequently report their activities to the investigation team. This interactive collaboration allows the teams to constantly update their information and maximize their efficiency. The pursuit phase is carried out in two steps as follows:

Actions (B₁): The investigation team monitors the best-so-far suspected location and reports it to the pursuit team. All members of the pursuit team surround the designated location to arrest the suspect. This step can be mathematically expressed as follows [11]:

$$\begin{aligned} X'_{B_{ij}} &= \text{rand1}_{ij} \times X_{B_{ij}} + \text{rand2}_{ij} \times (X_{Best_j} - X_{B_{ij}}); \\ i &= 1, 2, \dots, NP \text{ and } j = 1, 2, \dots, D \end{aligned} \quad (8.7)$$

where rand1 and rand2 are two random numbers in the range $[0, 1]$; X_{Best} denotes the best-so-far suspected location (reported by the investigation team); X_{B_i} is the current position of i -th member of the pursuit team; and X'_{B_i} is the new position of i -th member of the pursuit team.

Similar to steps A₁ and A₂ in investigation phase, the current position of members of the pursuit team ($X_{B_i}; i = 1, 2, \dots, NP$) are compared and replaced with their corresponding new ones ($X'_{B_i}; i = 1, 2, \dots, NP$), and X_{Best} is updated as follows:

$$X_{B_i} = X'_{B_i}, P'_{B_i} > P_{B_i}; i = 1, 2, \dots, NP \quad (8.8)$$

$$X_{Best} = X'_{B_i}, P'_{B_i} > P_{Best}; i = 1, 2, \dots, NP \quad (8.9)$$

where P_{B_i} and P'_{B_i} denote the probability of locating the suspect at X_{B_i} and X'_{B_i} , respectively.

Extending the Process of Actions (B₂): This step completes the investigation process. The headquarters constantly update the best-so-far suspected location. Reciprocally, members of the pursuit team constantly report back their activities to the headquarters. Interaction between members of the pursuit team might be helpful to decide about the directions of inquiry and actions. The new position of members of the pursuit team will be calculated by:

$$X'_{B_{ij}} = \begin{cases} X_{B_{rj}} + rand3_{ij} \times (X_{B_{rj}} - X_{B_{ij}}) + rand4_{ij} \times (X_{Best_j} - X_{B_{rj}}), & P_{B_r} > P_{B_i} \\ X_{B_{ij}} + rand3_{ij} \times (X_{B_{ij}} - X_{B_{rj}}) + rand4_{ij} \times (X_{Best_j} - X_{B_{ij}}), & P_{B_i} > P_{B_r} \end{cases} \quad (8.10)$$

where i and r are two members of the pursuit team, where r is chosen randomly; $rand3$ and $rand4$ are two random numbers in the range $[0, 1]$; X_{B_i} is the current position of i -th member of the pursuit team; X_{B_i} and X_{B_r} are the current position of i -th and r -th members of the pursuit team, respectively; and X'_{B_i} is the new position of i -th member of the pursuit team.

Once again, the current position of members of the pursuit team are compared and replaced with the corresponding new ones via a greedy manner, and X_{Best} is updated (see Eqs. (8.8) and (8.9)).

8.2.2.3 Termination of FBI

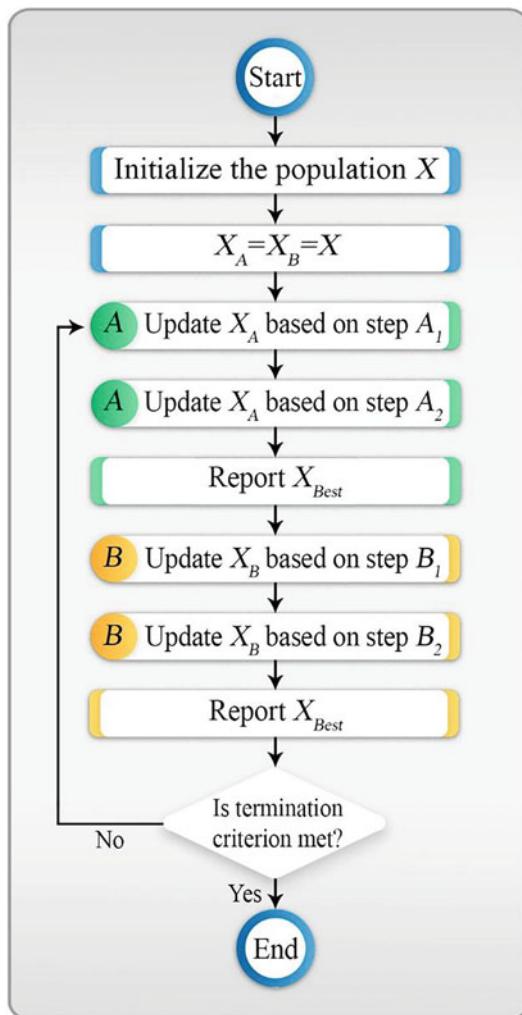
In the last step of the algorithm, if the termination condition is satisfied, the algorithm terminates and reports the best best-so-far suspected location; otherwise, it returns to step A₁ (i.e., interpretation of the findings) to identify more new suspected locations. Here, the maximum number of iterations ($MaxNITs$) is considered as the termination condition of the algorithm. However, other criteria such as the maximum number of function evaluations ($MaxNFEs$) could also be considered.

The flowchart and pseudocode of the standard FBI algorithm are provided in Figs. 8.3 and 8.4, respectively.

8.3 Enhanced Forensic-Based Investigation (EFBI)

As seen from Fig. 8.4, in the initialization phase of the standard FBI, both the initial suspected locations (X_A) and the initial position of members of the pursuit team (X_B) are set to be same as the initial population (X). As a result, before the main body of the standard FBI algorithm starts to work, the populations X_A and X_B are similar to each other. Throughout the first phase of the standard FBI (i.e., steps A₁ and A₂), the suspected locations ($X_{A_i}; i = 1, 2, \dots, NP$) are updated, while the position of members of the pursuit team ($X_{B_i}; i = 1, 2, \dots, NP$) remain unchanged. On the contrary, during the second phase (i.e., steps B₁ and B₂), the population X_B is updated, while the population X_A remains unchanged. The two phases of the standard FBI algorithm interact with each other only through the best-so-far suspected location (X_{Best}). At the end of the investigation phase, the investigation team reports X_{Best} to the pursuit team for further actions. Reciprocally, at the end of the pursuit phase, the pursuit team reports X_{Best} to the investigation team for further investigations. Nevertheless, members of the investigation team do not share their personal experiences with the pursuit team members and vice versa, and there seems

Fig. 8.3 Flowchart of the standard FBI algorithm



to be a gap between the investigation and pursuit teams. The miscommunication between the teams may lead to various difficulties such as low convergence rate, trapping in local optima, or even lack of convergence. Furthermore, as mentioned before, the investigation and pursuit phases encourage the diversification and the intensification tasks of the standard FBI algorithm, respectively. Hence, the standard FBI algorithm would not be expected to be able to make a good balance between the diversification and the intensification of the search.

In this following, an enhanced version of the forensic-based investigation algorithm, named as enhanced forensic-based investigation (EFBI), is proposed with the aim to reinforce communication between the investigation and pursuit teams. The

Pseudocode of Forensic-Based Investigation

Initialization phase

Set the algorithm parameters: NP and $MaxNITs$

Initialize the random population (X) by Eq. (8.1)

Evaluate the initial population

Determine X_{Best}

$X_A = X_B = X$

$NITs = 0$

Cyclic body of the algorithm

while $NITs < MaxNITs$

Investigation phase (A₁)

Identify new suspected locations (X'_A) using Eq. (8.2)

Evaluate new suspected locations

Update X_A and X_{Best} based on Eqs. (8.3) and (8.4)

Identify the best and worst suspected locations (P_{max} and P_{min})

Investigation phase (A₂)

if $P_{min} \neq P_{max}$

 Identify new suspected locations (X'_A) by Eqs. (8.5) and (8.6)

 Evaluate new suspected locations

 Update X_A and X_{Best} using Eqs. (8.3) and (8.4)

end if

Pursuing phase (B₁)

Generate new positions (X'_B) based on Eq. (8.7)

Evaluate new positions

Update X_B and X_{Best} by Eqs. (8.8) and (8.9)

Pursuing phase (B₂)

Generate new positions (X'_B) using Eq. (8.10)

Evaluate new positions

Update X_B and X_{Best} based on Eqs. (8.8) and (8.9)

$NITs = NITs + 1$

end while

return X_{Best}

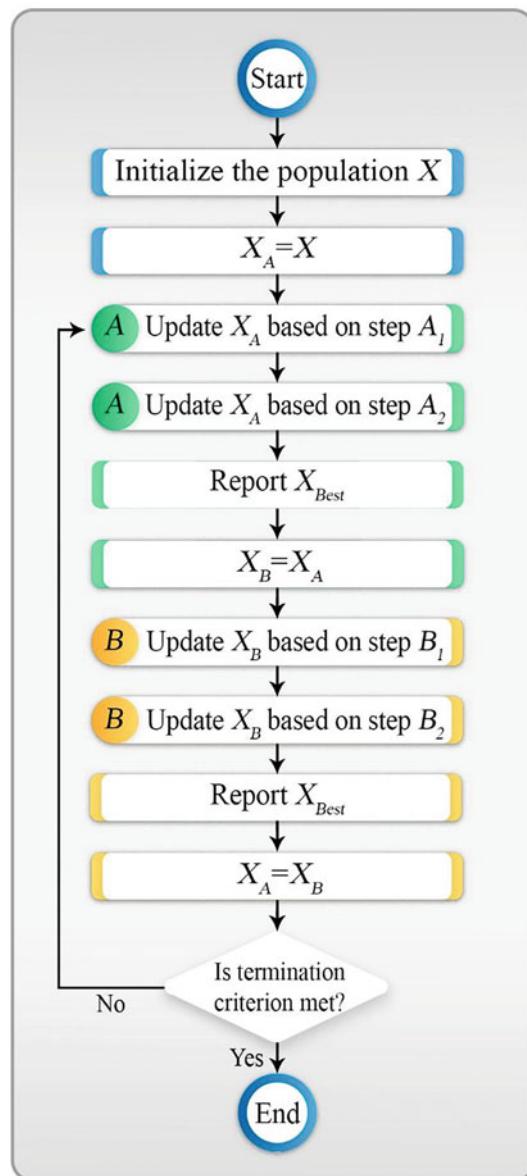
Fig. 8.4 Pseudocode of the standard FBI algorithm

steps of the proposed EFBI are same as those of the standard FBI. The main differences between the standard FBI and EFBI are in: (1) the formulation for updating the position of members of the investigation team in step A₂; and (2) the way the algorithms assign population to X_A and X_B . The proposed EFBI can be described as follows:

In the initialization phase, the initial population (X) is generated and the initial investigation team (X_A) is set to be same as the initial population (X). During the investigation phase (i.e., steps A₁ and A₂), investigation team (X_A) is updated. At the end of the investigation phase, the initial pursuit team (X_B) is set to be same as the updated investigation team (X_A). During the pursuit phase (i.e., steps B₁

and B_2), pursuit team (X_B) is updated. Finally, at the end of the pursuit phase, investigation team (X_A) is set to be same as updated pursuit team (X_B). The flowchart and pseudocode of the proposed EFBI algorithm are provided in Figs. 8.5 and 8.6, respectively.

Fig. 8.5 Flowchart of the proposed EFBI algorithm



Pseudocode of Enhanced Forensic-Based Investigation

Initialization phase

Set the algorithm parameters: NP and $MaxNITs$

Initialize the random population (X) by Eq. (8.1)

Evaluate the initial population

Determine X_{Best}

$X_A = X$

$NITs = 0$

Cyclic body of the algorithm

while $NITs < MaxNITs$

Investigation phase (A₁)

Identify new suspected locations (X'_A) using Eq. (8.2)

Evaluate new suspected locations

Update X_A and X_{Best} based on Eqs. (8.3) and (8.4)

Identify the best and worst suspected locations (P_{max} and P_{min})

Investigation phase (A₂)

if $P_{min} \neq P_{max}$

 Identify new suspected locations (X'_A) by Eqs. (8.6) and (8.11)

 Evaluate new suspected locations

 Update X_A and X_{Best} using Eqs. (8.3) and (8.4)

end if

$X_B = X_A$

Pursuing phase (B₁)

Generate new positions (X'_B) based on Eq. (8.7)

Evaluate new positions

Update X_B and X_{Best} by Eqs. (8.8) and (8.9)

Pursuing phase (B₂)

Generate new positions (X'_B) using Eq. (8.10)

Evaluate new positions

Update X_B and X_{Best} based on Eqs. (8.8) and (8.9)

$X_A = X_B$

$NITs = NITs + 1$

end while

return X_{Best}

Fig. 8.6 Pseudocode of the proposed EFBI algorithm

In step A2 of the standard FBI, the position of members is updated based on Eq. (8.5). As seen from the equation, the positions are updated by using a linear summation of three components: (1) the best-so-far suspected location (X_{Best}); (2) a randomly chosen suspected location (X_{A_d}); and (3) the difference between the positions of two other randomly chosen suspected locations (X_{A_k} and X_{A_h}). It should be mentioned that there is no coefficient before the first and second components (X_{Best} and X_{A_d}), which means the coefficient is equivalent to one. As a result, new positions may become too large. So, they may exceed search space boundaries. In order to address this issue, a new position update rule is proposed for step A2 of the

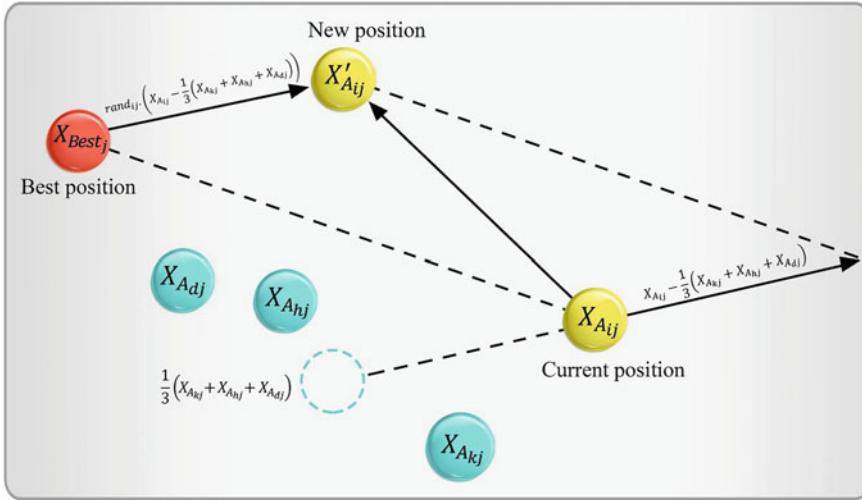


Fig. 8.7 Proposed position update rule for step A₂ of the EFBI algorithm

EFBI algorithm, as follows:

$$X'_{Aij} = \begin{cases} X_{Best_j} + rand_{ij} \times (X_{Aij} - (X_{Akj} + X_{Ahj} + X_{Adj})/3), & rand > Pr(X_{A_i}) \\ X_{Aij}, & \text{otherwise} \end{cases} \quad (8.11)$$

where d , k , h , and i are four possible locations of the suspect, where d , k , and h are chosen randomly; $rand$ is a random number between 0 and 1; X_{A_i} and X'_{A_i} are the current and new positions of the i -th suspected location, respectively; X_{Best} is the best-so-far suspected location; and $Pr(X_{A_i})$ is the relative probability of locating the suspect at X_{A_i} . The variables i and j are defined same as those in Eq. (8.1). Figure 8.7 illustrates mechanism of position update in step A₂ of the proposed EFBI algorithm.

8.4 Formulation of the Optimization Problem

In this section, both the standard FBI and EFBI algorithms are employed for size and shape optimization of frequency-constrained truss structures. The objective is to minimize the weight of the truss structure while satisfying some constraints on natural frequencies. The mathematical formulation of the problem can be defined mathematically as:

$$\text{Find: } \{X\} = [x_1, x_2, \dots, x_{nDV}] \quad (8.12)$$

$$\text{to minimize: } P(\{X\}) = f(\{X\}) \times f_{penalty}(\{X\}) \quad (8.13)$$

$$\text{subject to: } \begin{cases} \omega_j \geq \omega_j^* \text{ for some natural frequencies} \\ \omega_k \leq \omega_k^* \text{ for some natural frequencies} \\ x_i^L \leq x_i \leq x_i^U \quad i = 1, 2, \dots, nDV \end{cases} \quad (8.14)$$

where $\{X\}$ represents a solution vector containing the set of design variables; nDV is the number of design variables; x_i is the i -th design variable; $f(\{X\})$ is the objective function, which is considered as the weight of the structure in a weight minimization problem; $f_{penalty}(\{X\})$ is the penalty function, which is used to handle the problem constraints; $P(\{X\})$ is the penalized objective function to be minimized; x_i^L and x_i^U denote the lower and upper bounds of the i -th design variable, respectively; ω_j and ω_j^* are the j -th natural frequency of the structure and its corresponding lower bound, respectively; and ω_k and ω_k^* are the k -th natural frequency of the structure and its corresponding upper bound, respectively.

The objective function is the weight of the truss structure, which can be calculated as follows:

$$W(\{X\}) = \sum_{i=1}^{nE} \rho_i \times A_i \times L_i \quad (8.15)$$

where $W(\{X\})$ represent the weight of the truss structure; nE is the number of structural members; ρ_i is the material density; A_i is the cross-sectional area; and L_i is the length of the i -th structural member.

Through the penalty function method, the design constraints are incorporated in the objective function, and consequently the original constrained optimization problem is converted into an unconstrained one. Here, a dynamic penalty function is employed as follows:

$$f_{penalty}(X) = (1 + \varepsilon_1 \times v)^{\varepsilon_2}; v = \sum_{i=1}^{nC} v_i \quad (8.16)$$

where nC is the number of frequency constraints; v denotes the sum of the violations of the problem constraints; and ε_1 and ε_2 are parameters that determine the behavior of penalty function. The values of v_i ($i = 1, 2, \dots, nC$) are set to zero for satisfied constraints, while in case of violated constraints, they are selected considering the severity of violation. This can be expressed as follows:

$$v_i = \begin{cases} \left| 1 - \frac{\omega_i}{\omega_i^*} \right|, & \text{the } i\text{-th frequency constraint is violated} \\ 0, & \text{otherwise} \end{cases} \quad (8.17)$$

The values for parameters ε_1 and ε_2 determine how much a violated solution is penalized. Therefore, these parameters can affect the exploration and the exploitation of the search and can play a leading role in controlling the convergence rate. So, the parameters ε_1 and ε_2 should be selected in such a way that a low degree of penalty is imposed in the early stages of the optimization process, but as the number of iterations increases, so does the degree of penalty. In other words, the severity of penalizing infeasible solutions increases as the optimization process proceeds. Such a dynamic strategy allows a better exploration of the whole search space (i.e., more diversification) in the early stages of optimization, but more emphasis on exploitation of the best solutions found (i.e., more intensification) in the final stages [9]. This helps to balance the exploration and exploitation of the search process. In this study, the parameter ε_1 is set equal to unity, while the parameter ε_2 is set to monotonously increase from 2 to 4.

To determine the natural frequencies and modes of vibration of a system without damping, we need to solve the following algebraic equation, which is known as the matrix eigenvalue problem:

$$k\phi_n = \omega_n^2 m\phi_n \quad (8.18)$$

where k and m denote the stiffness and mass matrices of the system, respectively; ω_n is the n -th natural frequency of vibration of the system ($n = 1, 2, \dots, N$); ϕ_n is the n -th natural mode of vibration of the system; and N is the number of degrees of freedom of the system.

8.5 Numerical Examples

In this section, three dome-like truss optimization examples are investigated to demonstrate the efficiency and effectiveness of EFBI for solving frequency-constrained truss optimization problems. The considered examples are well-known benchmark problems. The first example is size and shape optimization of a small-scale dome-like truss, while the last two ones are mid-scale and large scale dome-like truss, respectively, which are employed to show robustness of the proposed algorithm. Table 8.1 lists the material properties, cross-sectional area bounds, and frequency constraints of all examples. Statistical results are presented in terms of the best weight, the worst weight, average weight, standard deviation, and the maximum number of finite element analyses. It should be noted that for each example, only optimal design of the best run is reported to compare with those of other optimization algorithms. Based on the suggestion of Chou and Nguyen [11], a population size of at least 50 individuals is required to explore the search space effectively. However, our experiment results indicate that smaller population sizes result in much better performance for the investigated examples. The population size (NP) is set to 10 for all problems. The maximum number of structural analyses ($MaxNFEs$) is set to

Table 8.1 Material properties, cross-sectional area bounds, and frequency constraints of the problems

Problem	Elasticity modulus E (N/m ²)	Material density ρ (kg/m ³)	Cross-sectional area bounds (m ²)	Frequency constraints (Hz)
52-bar dome-like truss	2.1×10^{11}	7800	$0.0001 \leq A_i \leq 0.001$	$\omega_1 \leq 50/\pi$, $\omega_2 \geq 90/\pi$
120-bar dome-like truss	2.1×10^{11}	7971.81	$0.0001 \leq A_i \leq 0.01293$	$\omega_1 \geq 9$, $\omega_2 \geq 11$
600-bar dome-like truss	2×10^{11}	7850	$0.0001 \leq A_i \leq 0.01$	$\omega_1 \geq 5$, $\omega_3 \geq 7$

10,000 for the 52-bar dome, 5000 for the 120-bar dome, and 12,000 for the 600-bar dome problems. Due to the stochastic nature of the optimization process, 10 independent runs are conducted for each algorithm on each problem and the obtained results of the best run are reported. The proposed EFBI algorithm, as well as finite element analysis code, is implemented by MATLAB software.

8.5.1 A 52-Bar Dome-Like Truss

Simultaneous size and shape optimization of a 52-bar dome-like truss is considered as the first example. Thus, both cross sectional area of the members and nodal coordinates are considered as design variables. The initial layout of the structure is depicted in Fig. 8.8. Each free node of the dome is allowed to move ± 2 m from its initial position in a symmetrical manner. Thus, this is a configuration optimization problem with 13 design variables (eight sizing design variables + five shape variables). As shown in Table 8.2, the structural members are categorized into eight different groups regarding the symmetry of the structure. Material properties, frequency constraints, and cross-sectional area bounds are provided in Table 8.1. A non-structural mass of 50 kg is attached to all free nodes of the structure. This problem has been previously studied by several researchers by using various optimization methods: Miguel and Fadel Miguel [12] using harmony search (HS) and firefly algorithm (FA), Kaveh and Zolghadr [7] utilizing a hybridization of the CSS and BB-BC algorithms, Kaveh and Zolghadr [13] employing democratic particle swarm optimization (DPSO), Ho-Huu et al. [14] by using improved differential evolution (IDE), and Kaveh and Ilchi Ghazaan utilizing [15] hybridized optimization algorithms.

A comparison of optimal results of the 52-bar dome-like truss obtained by the proposed algorithm with other methods in the literature is presented in Table 8.2. It can be seen that EFBI performs considerably better than the standard FBI in terms of best weight, average weight, worst weight, and standard deviation. Moreover, the average weight, worst weight, and standard deviation given by the EFBI are better than those of other algorithms, which demonstrates the effectiveness and robustness

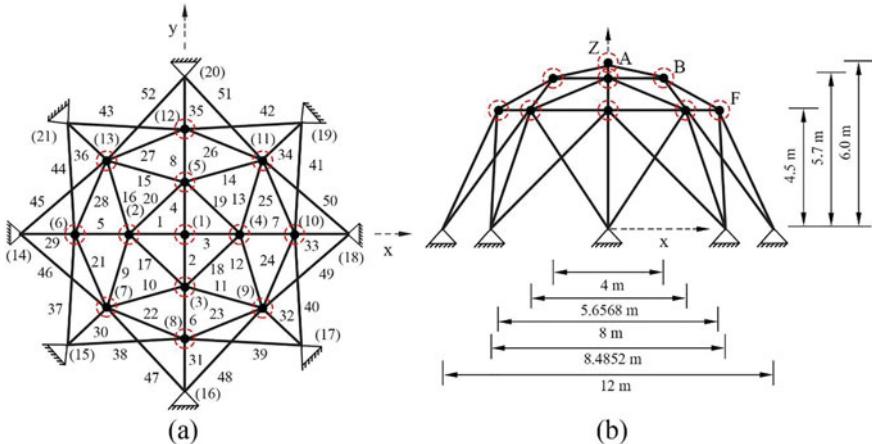


Fig. 8.8 Initial layout of the 52-bar dome-like truss: (a) top view, (b) side view

of the EFBI. Note that although the best weights obtained by the DE and IDE are slightly lighter than that of the EFBI, these methods demand more numbers of analyses than the EFBI to converge a global optimal design, i.e. DE with 20,002 FE analyses, and IDE with 12,191 FE analyses. Table 8.3 provides the first five optimal natural frequencies obtained by different methods for the 52-bar dome-like truss. It is clear that all the natural frequencies satisfy the frequency constraints. As Table 8.3 suggests, constraint on the second natural frequency of the structure controls the design process. Figure 8.9 presents a comparison between the average weight convergence histories of FBI and EFBI for the 52-bar dome-like truss. As can be seen from Fig. 8.9, EFBI has much better convergence performance compared to FBI.

8.5.2 A 120-Bar Dome-Like Truss

The 120-bar dome-like truss shown in Fig. 8.10 is considered as the second example. As shown in Fig. 8.10, the structural members are categorized into seven different groups considering the symmetry of the structure. The layout of the structure is kept unchanged during the optimization process. Thus, this is a size optimization problem with seven design variables. Material properties, frequency constraints, and cross-sectional area bounds are provided in Table 8.1. Non-structural masses are attached to all free nodes of the dome as follows: 3000 kg at node one, 500 kg at nodes 2 to 13, and 100 kg at the rest of the nodes. This problem has been solved with various methods by different researchers: Khatibinia and Naseralavi [2] using OMGSAs, Tejani et al. [16] employing improved symbiotic organisms search (ISOS), Kaveh and Zolghadr [13] by using DPSO, Kaveh and Zolghadr [19] utilizing particle

Table 8.2 Comparison of optimal results obtained for the 52-bar dome-like truss structure

Design variables: Z , X (m); A (cm^2)	HS [12]	FA [12]	CSS-BBBC [7]	DPSO [13]	DE [14]	IDE [14]	ALC-PSO [15]	HALC-PSO [15]	This study
	FBI	EFBI							
Z_A	4.7374	6.4332	5.331	6.1123	6.0136	6.0052	5.6972	5.9362	6.2948 6.0445
X_B	1.5643	2.2208	2.134	2.2343	2.2802	2.3004	2.0008	2.2416	2.2992 2.2002
Z_B	3.7413	3.9202	3.719	3.8321	3.7488	3.7332	3.7000	3.7309	4.0891 4.0032
X_F	3.4882	4.0296	3.935	4.0316	3.9980	4.0000	3.8052	3.9630	3.9059 3.8088
Z_F	2.6274	2.5200	2.500	2.5036	2.5000	2.5000	2.5000	2.5000	2.5126 2.5000
A_1-A_4	1.0085	1.0050	1.0000	1.0001	1.0000	1.0000	1.0000	1.0000	1.0064 1.0002
A_5-A_8	1.4999	1.3823	1.3056	1.1397	1.0981	1.0875	1.395	1.1654	1.1176 1.1620
A_9-A_{16}	1.3948	1.2295	1.4230	1.2263	1.2132	1.2135	1.3184	1.2323	1.1801 1.1992
$A_{17}-A_{20}$	1.3462	1.2662	1.3851	1.3335	1.4227	1.4460	1.5027	1.4323	1.3197 1.4108
$A_{21}-A_{28}$	1.6776	1.4478	1.4226	1.4161	1.4217	1.4315	1.3888	1.3901	1.4560 1.3945
$A_{29}-A_{36}$	1.3704	1.0000	1.0000	1.0001	1.0000	1.0000	1.0000	1.0001	1.0073 1.0000
$A_{37}-A_{44}$	1.4137	1.5728	1.5562	1.5750	1.5770	1.5623	1.724	1.6024	1.5528 1.4876
$A_{45}-A_{52}$	1.9378	1.4153	1.4485	1.4357	1.3722	1.3724	1.3187	1.4131	1.4348 1.4899
Weight (kg)	214.94	197.53	197.309	195.351	193.2481	193.2085	196.27	194.85	195.98 193.60
Number of FE analyses	N/A	N/A	N/A	N/A	20,020	11,040	9000	7500	N/A N/A
Average weight (kg)	229.88	212.80	N/A	198.71	196.0585	196.0478	207.13	196.85	209.69 194.17
Worst weight (kg)	N/A	N/A	N/A	N/A	202.4296	202.4215	N/A	N/A	229.72 195.34

(continued)

Table 8.2 (continued)

	HS [12]	FA [12]	CSS-BBBC [7]	DPSO [13]	DE [14]	IDE [14]	ALC-PSO [15]	HALC-PSO [15]	This study
							FBI	EFBI	
Design variables: Z , X (m); A (cm^2)									
Standard deviation (kg)	12.44	8.44	N/A	13.85	4.1708	4.1823	6.36	2.38	9.56
Average number of FE analyses	20,000	1000	4000	6000	20,002	12,191	20,000	20,000	10,000
Number of runs	5	5	20	30	30	30	30	10	10

Table 8.3 First five natural frequencies (Hz) of the 52-bar dome-like truss evaluated at optimal designs

Frequency number	HS [12]	FA [12]	CSS-BBBC [7]	DPSO [13]	DE [14]	IDE [14]	ALC-PSO [15]	HALC-PSO [15]		This study	
								FBI	EFBI	FBI	EFBI
1	12.2222	11.3119	12.987	11.315	11.5319	11.6033	10.4619	11.4339	9.6775	11.2011	
2	28.6577	28.6529	28.648	28.648	28.6494	28.6481	28.6479	28.6480	28.6479	28.6479	
3	28.6577	28.6529	28.679	28.648	28.6509	28.6481	28.6480	28.6480	28.6483	28.6479	
4	28.6618	28.8030	28.713	28.650	28.6509	28.6490	28.7129	28.6482	28.6508	28.6503	
5	30.0997	28.8030	30.262	28.688	28.6661	28.6530	28.8922	28.6848	29.3780	28.6578	

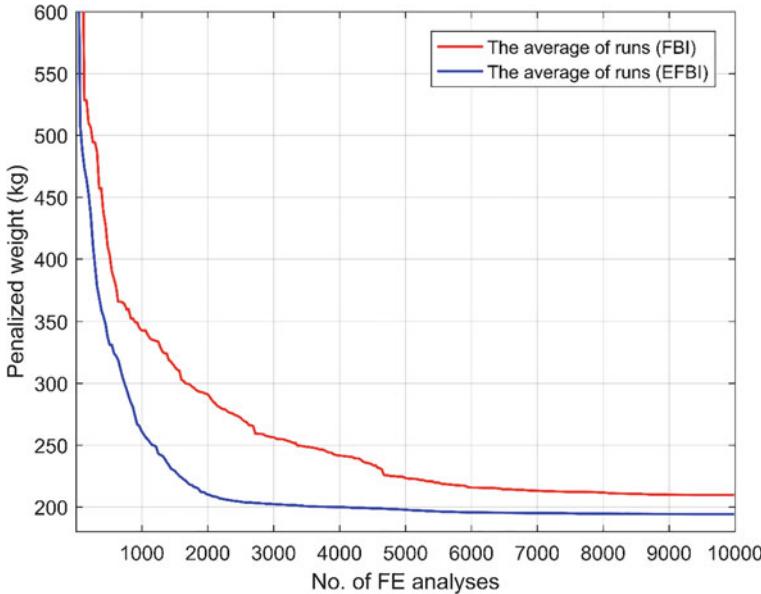


Fig. 8.9 Average weight convergence histories for the 52-bar dome-like truss

swarm ray optimization (PSRO), Taheri and Jalili [17] using enhanced biogeography-based optimization (EBBO), Dede et al. [18] using Jaya algorithm (JA), and Tejani et al. [20] using a modified sub-population teaching–learning-based optimization (MS-TLBO).

Table 8.4 represents the optimal results of the 120-bar dome truss obtained by the FBI and EFBI and their results are compared to those by other studies. It is found that the EFBI gives the best results compared to the FBI and other considered methods. Similar to the previous example, the best weight, average weight, worst weight, and standard deviation obtained by the EFBI are much better than those by FBI. Table 8.5 provides the first five optimized natural frequencies of the best designs, which shows that all frequency constraints are fulfilled. Figure 8.11 shows the convergence histories of the FIB and EFBI for this example. As expected, the EFBI converges much faster than the FBI.

8.5.3 A 600-Bar Dome-Like Truss

Figure 8.12 shows the schematic of a 600-bar single-layer dome structure, which is considered as the last example. The entire structure is composed of 216 nodes and 600 elements. The structure has a cyclically repeated pattern and could be generated by the cyclic repetition of a sub-structure composed of 9 nodes and 25 elements. The angle of rotation of the sub-structure about the axis of revolution is $\pi/12$. The sub-structure

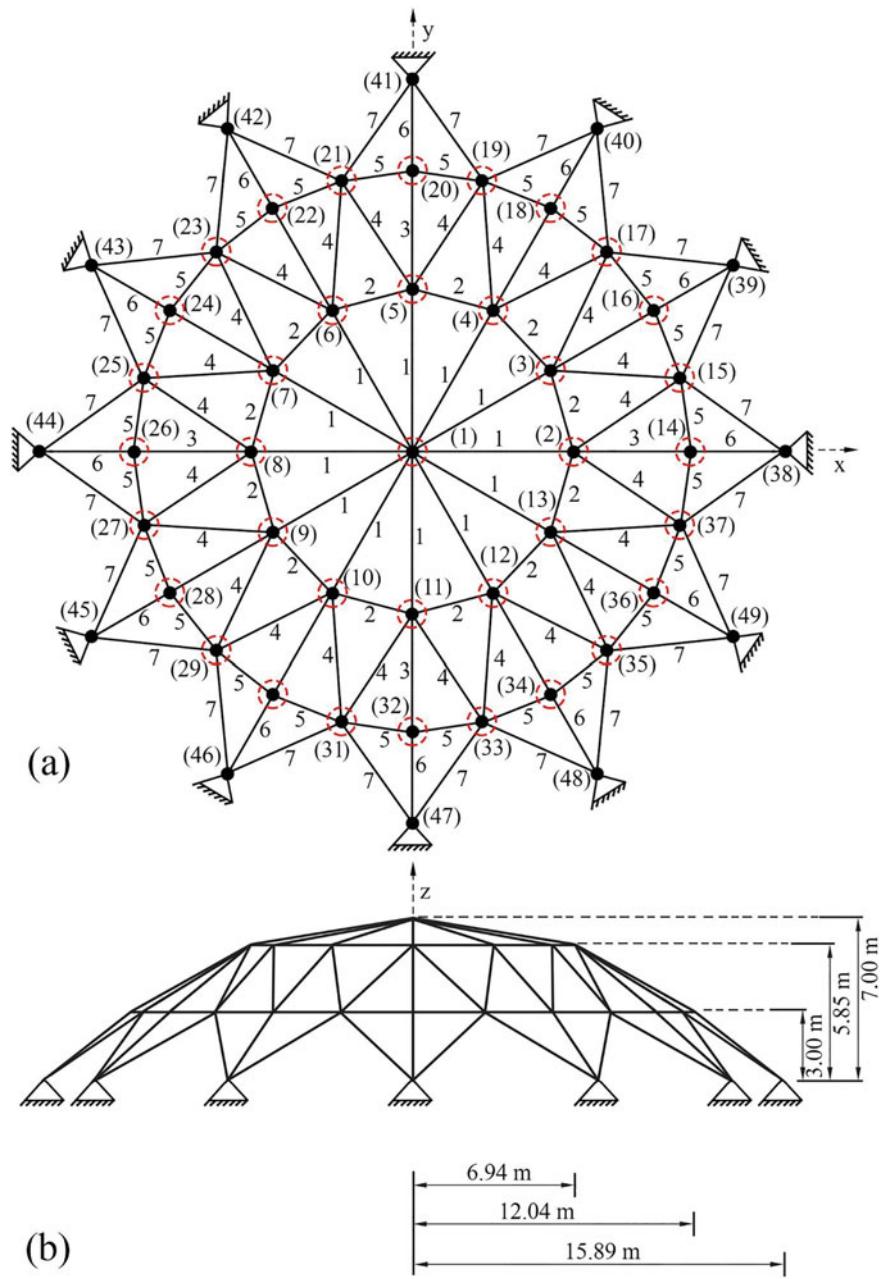


Fig. 8.10 Schematic of the 120-bar dome-like truss: **a** top view, **b** side view

Table 8.4 Comparison of optimal results obtained for the 120-bar dome-like truss structure

Design variable:	IGSA [2]	OMGSA [2]	ISOS [16]	DPSO [13]	PSRO [19]	EBBO [17]	JA [18]	MS-TLBO [20]	This study
A (cm^2)								FBI	EFBI
1	19.043	20.263	19.6662	19.607	19.972	19.8878	19.309	19.4486	19.3805
2	41.418	39.294	39.8539	41.290	39.701	39.8248	40.763	40.3949	40.7790
3	10.218	9.989	10.6127	11.136	11.323	10.5496	10.791	10.6921	10.9254
4	20.664	20.563	21.2901	21.025	21.808	21.0929	21.272	21.3139	20.9444
5	10.795	9.603	9.7911	10.060	10.179	9.4245	9.943	9.8943	9.7849
6	12.190	11.738	11.7899	12.758	12.739	11.6648	11.695	11.7810	11.3525
7	14.960	15.877	14.7437	15.414	14.731	15.1282	14.579	14.5979	15.0387
Weight (kg)	8727.28	8724.97	8710.0620	8890.48	8892.33	8711.95	8709.3539	8708.729	8710.39
Number of FE analyses	less than 11,000	7889	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Average weight (kg)	8798.55	8745.58	8728.5951	8895.99	8921.3	8718.5	8713.21	8734.7450	8723.74
Worst weight (kg)	8800.45	8760.75	8770.8110	N/A	N/A	N/A	N/A	N/A	8710.51
Standard deviation (kg)	7.195	1.183	14.2296	4.26	18.54	7.15	2.97	27.0503	11.44
Average number of FE analyses	15,000	14,700	4000	6000	4000	30,000	18,000	4000	5000
Number of runs	20	20	N/A	30	20	100	20	100	10

Table 8.5 First five natural frequencies (Hz) of the 120-bar dome-like truss evaluated at optimal designs

Frequency number	IGSA [2]	OMGSA [2]	ISOS [16]	DPSO [13]	PSRO [19]	EBBO [17]	JA [18]	MS-TLBO [20]	This study
								FBI	EFBI
1	9.001	9.002	9.0001	9.0001	9.000	9.0000	9.0000	9.0002	9.0001
2	11.003	11.003	10.9998	11.0007	11.000	11.0000	11.0002	11.0000	11.0000
3	11.003	11.003	N/A	11.0053	11.005	11.0002	11.0002	11.0000	11.0001
4	11.017	11.007	N/A	11.0129	11.012	11.0008	11.0008	11.0006	11.0001
5	11.089	11.076	N/A	11.0471	11.045	11.0657	11.0674	11.0672	11.0665

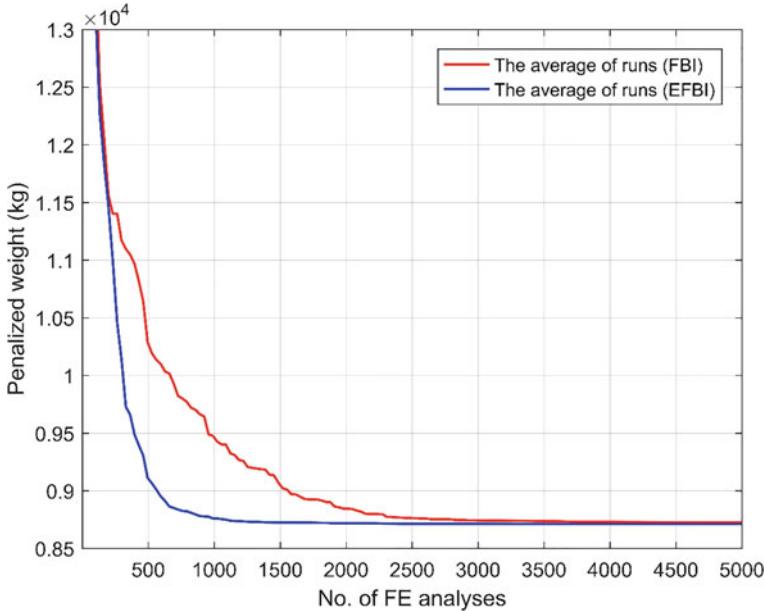
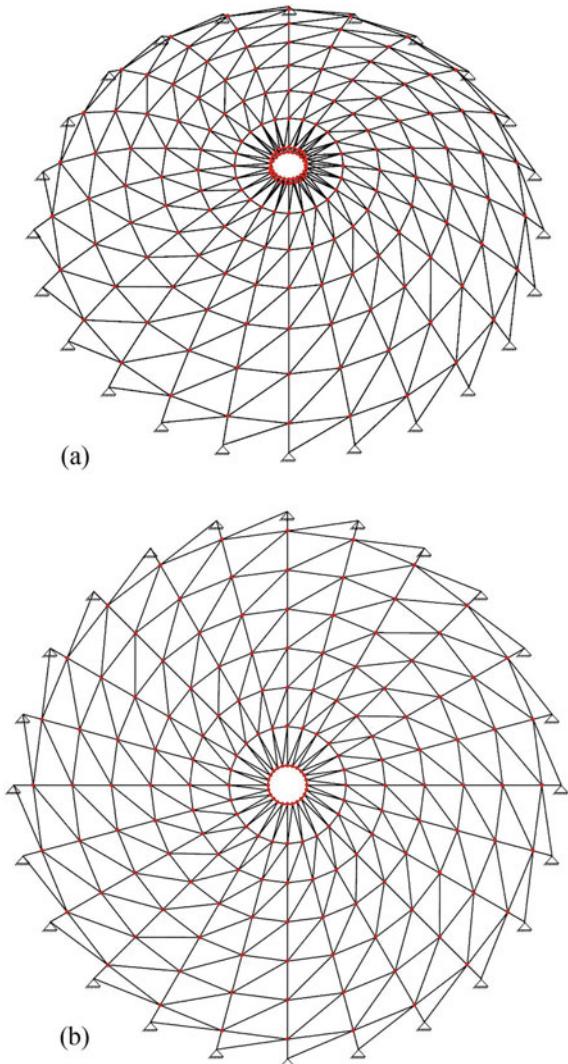


Fig. 8.11 Average weight convergence histories for the 120-bar dome-like truss

is shown in Fig. 8.13 in more detail for nodal numbering and coordinates. Table 8.6 lists the nodal coordinates of the typical sub-structure in the Cartesian coordinate system. Each element of the sub-structure is considered as a design variable. The layout of the structure is kept unchanged during the optimization process. Hence, this is a size optimization problem with 25 design variables. Material density, elasticity modulus, cross-sectional area bounds, and frequency constraints of the structure are listed in Table 8.1. A non-structural mass of 100 kg is attached to all free nodes of the dome. This problem has been investigated by several researchers via different optimization methods: Kaveh [21] using DPSO, Kaveh and Ilchi Ghazaan [22] utilizing vibrating particles system (VPS), Kaveh and Ilchi Ghazaan [8] employing enhanced colliding bodies optimization (ECBO) and its cascade version, Kaveh and Ilchi Ghazaan [23] using VPS and its hybrid version, and Kaveh and Ilchi Ghazaan [24] utilizing colliding bodies optimization (CBO).

Table 8.7 provides a comparison of optimal results of the 600-bar single-layer dome obtained by the proposed algorithm with other approaches. As shown in Table 8.7, the best and worst weights, average weight, and standard deviation obtained by the EFBI are better than those of other algorithms. The average weight gained by EFBI is better than the best weight obtained by all the other methods, including the standard FBI. Even the worst weight obtained by EFBI is better than the best weight obtained by all the other methods. Furthermore, it can be seen that the EFBI requires far fewer the number of FE analysis to achieve optimal designs. A comparison of optimal results obtained by the EFBI with those of FBI reveals that the EFBI significantly

Fig. 8.12 Schematic of the 600-bar dome-like truss: **a** perspective view, **b** top view



performs better than the standard FBI. In fact, the best weight found by EFBI is 6076.35 kg, which is 3.12% lighter than the best weight found by FBI. Figure 8.14 compares the average weight convergence histories of FBI and EFBI for the 600-bar single-layer dome. As it can be observed from Fig. 8.14, the convergence rate of EFBI is much higher than that of FBI. The first five optimal natural frequencies obtained by various methods for the 600-bar dome-like truss are presented in Table 8.8. The results confirm that the frequency constraints of the optimum designs obtained by FBI and EFBI are satisfied.

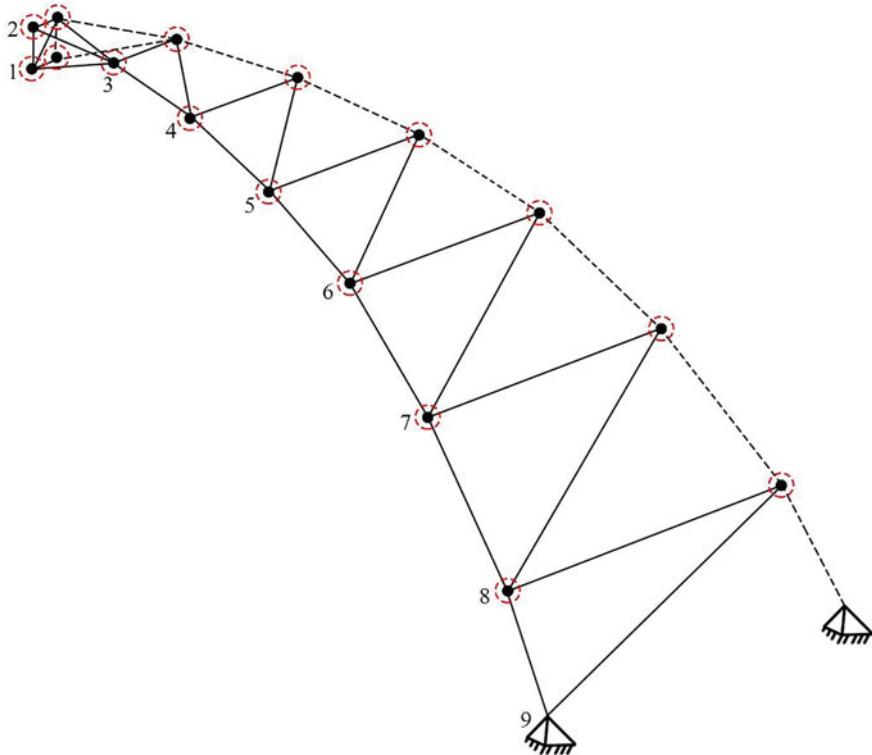


Fig. 8.13 Details of a sub-structure of the 600-bar dome-like truss

Table 8.6 Nodal coordinates (m) of the first sub-structure of the 600-bar dome-like truss

Node number	Coordinates (x, y, z)
1	(1.0, 0.0, 7.0)
2	(1.0, 0.0, 7.5)
3	(3.0, 0.0, 7.25)
4	(5.0, 0.0, 6.75)
5	(7.0, 0.0, 6.0)
6	(9.0, 0.0, 5.0)
7	(11.0, 0.0, 3.5)
8	(13.0, 0.0, 1.5)
9	(14.0, 0.0, 0.0)

8.6 Concluding Remarks

Forensic-based investigation (FBI) algorithm is a human-based metaheuristic algorithm which has been developed recently inspired by the forensic investigation process. The major drawback of the standard FBI algorithm is the lack of an effective

Table 8.7 Comparison of optimal results obtained for the 600-bar dome-like truss structure (cm^2)

Element number (element nodes)	DPSO [21]	VPS [22]	ECBO [8]	ECBO-Cascade [8]	CBO [24]	VPS [23]	MDVC-UPVVS [23]	This study FBI	This study EFBI
1 (1–2)	1.365	1.3030	1.4305	1.0299	1.2404	1.3155	1.2575	1.1565	1.0999
2 (1–3)	1.391	1.3998	1.3941	1.3664	1.3797	1.2299	1.3466	1.5495	1.4922
3 (1–10)	5.686	5.1072	5.5293	5.1095	5.2597	5.5506	4.9738	4.2881	6.0744
4 (1–11)	1.511	1.3882	1.0469	1.3011	1.2658	1.3867	1.4025	1.2744	1.6234
5 (2–3)	17.711	16.9217	16.9642	17.0572	17.2255	17.4275	17.3802	18.6386	17.4918
6 (2–11)	36.266	38.1432	35.1892	34.0764	38.2991	40.1430	37.9742	41.5551	37.2118
7 (3–4)	13.263	11.8319	12.2171	13.0985	12.2234	12.8848	13.0306	13.8552	12.7873
8 (3–11)	16.919	16.6149	16.7152	15.5882	15.4712	15.5413	15.9209	14.1519	14.8239
9 (3–12)	13.333	11.3403	12.5999	12.6889	11.1577	12.2428	11.9419	14.6455	12.1764
10 (4–5)	9.534	9.3865	9.5118	10.3314	9.4636	9.3776	9.1643	9.4627	9.0163
11 (4–12)	9.884	8.7692	8.9977	8.5313	8.8250	8.6684	8.4332	7.6139	8.5044
12 (4–13)	9.547	9.6682	9.4397	9.8308	9.1021	9.1659	9.2375	7.3169	8.9951
13 (5–6)	7.866	6.9826	6.8864	7.0101	6.8417	7.1664	7.2213	11.1706	7.0357
14 (5–13)	5.529	5.4445	4.2057	5.2917	5.2882	5.2170	5.2142	5.7036	5.0993
15 (5–14)	7.007	6.3247	7.2651	6.2750	6.7702	6.5346	6.7961	7.3517	6.1918
16 (6–7)	5.462	5.1349	6.1693	5.4305	5.1402	5.4741	5.2078	5.8131	4.9514
17 (6–14)	3.853	3.3991	3.9768	3.6414	5.1827	3.6345	3.4586	3.0348	3.9186
18 (6–15)	7.432	7.7911	8.3127	7.2827	7.4781	7.6034	7.6407	6.6133	7.6312
19 (7–8)	4.261	4.4147	4.1451	4.4912	4.5646	4.2251	4.3690	3.8692	4.4271
20 (7–15)	2.253	2.2755	2.4042	1.9275	1.8617	1.9717	2.1237	3.0077	2.3280

(continued)

Table 8.7 (continued)

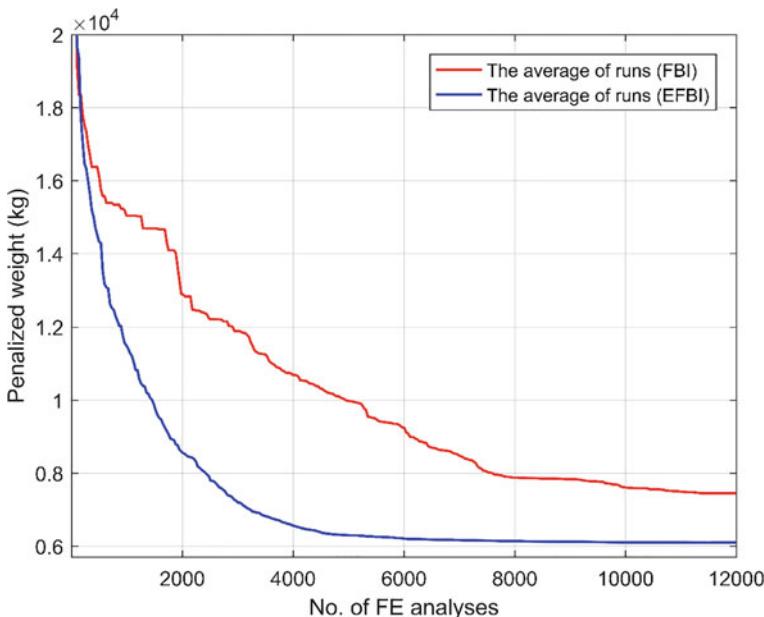


Fig. 8.14 Average weight convergence histories for the 600-bar dome-like truss

interaction between the diversification and the intensification phases of the search. In order to overcome this drawback, the enhanced FBI (EFBI) is proposed in this chapter. The proposed EFBI aims to reinforce communication between the investigation and pursuit teams (i.e., between the diversification and the intensification tasks). The other improvement is carried out on the position update rule of the investigation step. In the investigation phase of FBI (i.e., step A₂), the position update rule (Eq. 8.5) yields too large step sizes, which may result in divergence and strange oscillations and instabilities. To address this issue, a new position update rule is proposed based on the best-so-far suspected location and the current suspected locations. Both the FBI and the EFBI are then applied to solve size and shape optimization problems of dome truss structures with frequency constraints. This is the first time that FBI is applied in the field of structural optimization. Numerical results reveal that in most of the case studies, the average weight and standard deviation on average weight obtained by the EFBI are better than those gained by some other methods in the literature. Furthermore, thanks to the good balance between the diversification and the intensification tasks, the EFBI performs significantly better than the FBI in terms of efficiency and effectiveness. Especially, in all case studies, the best weight, average weight, worst weight, and standard deviation on average weight obtained by the EFBI are considerably better than those of the FBI. This shows that the EFBI can

Table 8.8 First five natural frequencies (Hz) of the 600-bar dome-like truss evaluated at optimal designs

Frequency number	DPSO [21]	VPS [22]	ECBO [8]	ECBO-Cascade [8]	CBO [24]	VPS [23]	MDVC-UPVS [23]	This study	
	FBI	EFBI	FBI	EFBI	FBI	EFBI	FBI	FBI	EFBI
1	5.000	5.0000	5.002	5.001	5.000	5.000	5.000	5.0001	5.0001
2	5.000	5.0003	5.003	5.001	5.000	5.000	5.000	5.0001	5.0001
3	7.000	7.0000	7.001	7.001	7.000	7.000	7.000	7.0001	7.0000
4	7.000	7.0001	7.001	7.001	7.000	7.000	7.000	7.0001	7.0000
5	7.000	7.0002	7.002	7.002	7.001	7.000	7.000	7.0003	7.0024

offer a high-performance optimization algorithm for solving truss optimization problems with multiple frequency constraints. Finally, the proposed algorithm could be extended to other structural optimization problems, such as frames, retaining walls, etc.

References

1. Kaveh A, Biabani Hamedani K, Kamalinejad M (2021) An enhanced Forensic-Based Investigation algorithm and its application to optimal design of frequency-constrained dome structures. *Comput Struct* 256:106643. <https://doi.org/10.1016/j.compstruc.2021.106643>
2. Khatibinia M, Naseralavi SS (2014) Truss optimization on shape and sizing with frequency constraints based on orthogonal multi-gravitational search algorithm. *J Sound Vib* 333(24):6349–6369. <https://doi.org/10.1016/j.jsv.2014.07.027>
3. Grandhi R, Venkayya VB (1988) Structural optimization with frequency constraints. *AIAA J* 26(7):858–866. <https://doi.org/10.2514/3.9979>
4. Bellagamba L, Yang TY (1981) Minimum-mass truss structures with constraints on fundamental natural frequency. *AIAA J* 19(11):1452–1458. <https://doi.org/10.2514/3.7875>
5. Tong WH, Liu GR (2001) An optimization procedure for truss structures with discrete design variables and dynamic constraints. *Comput Struct* 79(2):155–162. [https://doi.org/10.1016/S0045-7949\(00\)00124-3](https://doi.org/10.1016/S0045-7949(00)00124-3)
6. Gomez HM (2011) Truss optimization with dynamic constraints using a particle swarm algorithm. *Expert Syst Appl* 38:957–968. <https://doi.org/10.1016/j.eswa.2010.07.086>
7. Kaveh A, Zolghadr A (2012) Truss optimization with natural frequency constraints using a hybridized CSS-BBBC algorithm with trap recognition capability. *Comput Struct* 102:14–27. <https://doi.org/10.1016/j.compstruc.2012.03.016>
8. Kaveh A, Ilchi Ghazaan M (2016) Optimal design of dome truss structures with dynamic frequency constraints. *Struct Multidisc Optim* 53(3):605–621. <https://doi.org/10.1007/s00158-015-1357-2>
9. Kaveh A, Zolghadr A (2018) Optimal design of cyclically symmetric trusses with frequency constraints using cyclical parthenogenesis algorithm. *Adv Struct Eng* 21(5):739–755. <https://doi.org/10.1177/1369433217732492>
10. Mirjalili S, Lewis A (2016) The whale optimization algorithm. *Adv Eng Softw* 95:51–67. <https://doi.org/10.1016/j.advengsoft.2016.01.008>
11. Chou JS, Nguyen NM (2020) FBI inspired meta-optimization. *Appl. Soft Comput* 93:106339. <https://doi.org/10.1016/j.asoc.2020.106339>
12. Miguel LF, Miguel LF (2012) Shape and size optimization of truss structures considering dynamic constraints through modern metaheuristic algorithms. *Expert Syst Appl* 39(10):9458–9467. <https://doi.org/10.1016/j.eswa.2012.02.113>
13. Kaveh A, Zolghadr A (2014) Democratic PSO for truss layout and size optimization with frequency constraints. *Comput Struct* 130:10–21. <https://doi.org/10.1016/j.compstruc.2013.09.002>
14. Ho-Huu V, Vo-Duy T, Luu-Van T, Le-Anh L, Nguyen-Thoi T (2016) Optimal design of truss structures with frequency constraints using improved differential evolution algorithm based on an adaptive mutation scheme. *Autom Construc* 68:81–94. <https://doi.org/10.1016/j.autcon.2016.05.004>
15. Kaveh A, Ilchi Ghazaan M (2015) Hybridized optimization algorithms for design of trusses with multiple natural frequency constraints. *Adv Eng Softw* 79:137–147. <https://doi.org/10.1016/j.advengsoft.2014.10.001>
16. Tejani GG, Savsani VJ, Patel VK, Mirjalili S (2018) Truss optimization with natural frequency bounds using improved symbiotic organisms search. *Knowl Based Syst* 143:162–178. <https://doi.org/10.1016/j.knosys.2017.12.012>

17. Taheri SHS, Jalili S (2016) Enhanced biogeography-based optimization: a new method for size and shape optimization of truss structures with natural frequency constraints. *Lat Am J Solids Struct* 13(7):1406–1430. <https://doi.org/10.1590/1679-78252208>
18. Dede T, Grzywiński M, Rao RV (2020) Jaya: a new meta-heuristic algorithm for the optimization of braced dome structures. In: Advanced engineering optimization through intelligent techniques. Springer, Singapore, pp 13–20. https://doi.org/10.1007/978-981-13-8196-6_2
19. Kaveh A, Zolghadr A (2014) A new PSRO algorithm for frequency constraint truss shape and size optimization. *Struct Eng Mech* 52(3):445–468. <https://doi.org/10.12989/sem.2014.52.3.445>
20. Tejani GG, Savsani VJ, Patel VK (2016) Modified sub-population teaching-learning-based optimization for design of truss structures with natural frequency constraints. *Mech Based Des Struct Mach* 44(4):495–513. <https://doi.org/10.1080/15397734.2015.1124023>
21. Kaveh A (2017) Optimal analysis and design of large-scale domes with frequency constraints. In Applications of metaheuristic optimization algorithms in civil engineering. Springer, Cham, pp 257–279. https://doi.org/10.1007/978-3-319-48012-1_14
22. Kaveh A, Ilchi Ghazaan M (2017) Vibrating particles system algorithm for truss optimization with multiple natural frequency constraints. *Acta Mech* 228(1):307–322. <https://doi.org/10.1007/s00707-016-1725-z>
23. Kaveh A, Ilchi Ghazaan M (2018) A new hybrid meta-heuristic algorithm for optimal design of large-scale dome structures. *Eng Optim* 50(2):235–252. <https://doi.org/10.1080/0305215X.2017.1313250>
24. Kaveh A, Ilchi Ghazaan M (2018) Optimal design of dome-shaped trusses. In: Meta-heuristic algorithms for optimal design of real-size structures. Springer, Cham, pp 101–122. https://doi.org/10.1007/978-3-319-78780-0_7

Chapter 9

Improved Slime Mould Algorithm



9.1 Introduction

In this chapter, an improved version of the slime mould algorithm (SMA) called improved SMA (ISMA) is proposed for size optimization of truss structures with frequency constraints [1]. In order to overcome the drawbacks of the classical SMA, the proposed ISMA employs a simple elitist strategy in the replacement phase. Moreover, in the proposed ISMA, the exploration phase is suitably modified. In order to demonstrate the performance of the proposed ISMA, the size optimization of three large-scale dome-like truss structures with frequency constraints is investigated, and the results obtained by ISMA are compared to those of the classical SMA and other optimization methods in the literature.

Generally, optimization methods can be classified into two main categories: gradient-based methods and metaheuristic algorithms. Gradient-based methods require gradient information of the objective function and constraints with respect to the design variables to direct the search. Although gradient-based methods are interesting from a mathematical point of view, they often suffer from three significant drawbacks: the risk of being trapped in local optima, slow convergence rate, and strong dependency on the starting point. Moreover, in many optimization problems, analytical gradient information is not available or computationally expensive to calculate. In contrast to gradient-based methods, metaheuristic algorithms do not require gradient information in the search process, and thus can be applied to a wide range of optimization problems. The methods can efficiently find optimal or near-optimal solutions to discrete, non-convex, and discontinuous optimization problems within a reasonable time. As a result, metaheuristic algorithms are highly desirable for a wide range of engineering applications.

Numerous metaheuristic algorithms have been developed in recent decades. These algorithms are commonly operate based on the principles of different natural phenomena. Based on the source of inspiration, nature-inspired metaheuristic algorithms can be classified into four main categories: evolutionary-based, swarm-based,

physics-based, and human-based. The first group of nature-inspired metaheuristics includes evolutionary-based methods, which imitate various mechanisms of biological evolution such as reproduction, mutation, recombination, and selection. Some of the well-known evolutionary-based optimization algorithms are genetic algorithms (GA), which are based on the principle of genetics and evolution, genetic programming (GP), differential evolution (DE), and biogeography-based optimization (BBO), that is based on the mathematics of biogeography. The second category is swarm-based algorithms, which are inspired by the collective behavior of social animals. The most popular swarm-based algorithm is particle swarm optimization (PSO). Other swarm intelligence algorithms in the literature are ant colony optimization (ACO), that is based on the foraging behavior of some ant species, artificial bee colony (ABC), inspired by the intelligent behaviour of honey bees, and cuckoo search (CS), which simulates the obligate brood parasitic behaviour of some cuckoo species. The third category is physics-based metaheuristic algorithms, which are based on the laws governing physical phenomena in nature. The physics-based metaheuristic algorithms include gravitational search algorithm (GSA), that takes its inspiration from the law of gravity and mass interactions, charged system search (CSS), inspired by the governing Coulomb's law from electrostatics and the Newtonian's laws from mechanics, big bang-big crunch (BB-BC), which is based on the big bang and big crunch theories of the universe evolution, and ray optimization (RO), that is based on Snell's law of refraction. The last group of nature-inspired metaheuristic algorithms is human-based ones, which simulate human activities and behaviors. Some of these algorithms are dynastic optimization algorithm (DOA), based on human dynasties, imperialist competitive algorithm (ICA), which simulates the imperialist competition, forensic-based investigation (FBI), that takes its inspiration from the criminal investigation process, and interior search algorithm (ISA), inspired by interior design and decoration.

Structural optimization with frequency constraints provides an effective way to manipulate and control the dynamic characteristics of structures in the desired way. In most of the vibration problems where the structure is subjected to low-frequency vibration, the response of the structure is mainly a function of its fundamental frequencies and mode shapes [2]. In such cases, the performance of the structure can be significantly improved by imposing some limitations on the ranges of the natural frequencies. For example, in designing a space vehicle, to avoid resonance phenomenon, some constraints should be imposed on the lowest natural frequencies of the vehicle to prescribed ranges. This can be achieved via structural optimization with frequency constraints.

Since the early 1980s, structural optimization with natural frequency constraints has drawn the attention of many researchers. Bellagamba and Yang [3] were among the first to focus on structural optimization with frequency constraints. They developed a technique for constrained parameter optimization and applied it to the mass minimization of truss structures subject to static mechanical and thermal loads while satisfying constraints on design variables, displacements, stresses, local buckling, and fundamental natural frequency. Grandhi and Venkayya [2] investigated the problem

using an optimality criterion method based on a simple resizing scheme in conjunction with a scaling procedure. Gomes [4] investigated the use of particle swarm optimization (PSO) algorithm for size and shape optimization of truss structures with multiple frequency constraints.

The structural optimization with frequency constraints is recognized as a highly nonlinear, non-convex, and multimodal problem with respect to the design variables [5]. The most common difficulty with these problems seems to be the high sensitivity of natural frequencies to size and layout modifications, which may cause convergence difficulties. In fact, during the optimization process, size and layout modifications may cause the switching of vibration modes, which can lead to significant changes in natural frequencies. As noted by Sergeyev and Mroz [6], natural frequencies are much more sensitive to layout modifications than to size modifications. Hence, gradient-based methods seem to have difficulties when dealing with frequency-constrained optimization problems, because these methods require the gradient information of the frequency with respect to the design variables. On the other hand, local search algorithms are not suitable for global optimization and may converge to local optima. Under such considerations, global search algorithms should be used. Modern metaheuristic algorithms can be regarded as appropriate alternatives. Over the past decade, metaheuristic algorithms have been applied widely to optimal design of structures with frequency constraints. For example, Khatibnia and Naseralavi [5] introduced orthogonal multi-gravitational search algorithm (OMGSA) to solve truss size and shape optimization problems considering frequency constraints. Kaveh [7] employed democratic particle swarm optimization (DPSO) for solving large-scale dome optimization problems with frequency constraints. Kaveh and Ilchi Ghazaan [8] utilized a cascade version of enhanced colliding bodies optimization (ECBO-Cascade) for optimal design of dome truss structures with dynamic frequency constraints. Kaveh and Ilchi Ghazaan [9] developed vibrating particles system (VPS) for truss optimization with multiple natural frequency constraints. Kaveh and Zolghadr [10] applied cyclical parthenogenesis algorithm (CPA) to optimal design of cyclically symmetric trusses with frequency constraints. Kaveh et al. [11] developed an enhanced forensic-based investigation (EFBI) algorithm and applied it to solve frequency-constrained optimization of dome structures. Degertekin et al. [12] presented parameter free Jaya algorithm (PFJA) for truss size and shape optimization under natural frequency constraints.

The slime mould algorithm is a swarm-based metaheuristic recently developed by Li et al. [13]. The algorithm mimics the foraging behavior of the acellular slime mould *Physarum polycephalum*. The SMA has been successfully applied to some real-world optimization problems. For example, Abdel-Basset et al. [14] applied SMA and its hybrid variant to the image segmentation problem of chest X-ray images. Zubaidi et al. [15] developed a hybridized artificial neural network model with SMA for urban water demand prediction. Kumar et al. [16] applied SMA for the estimation of solar photovoltaic cell parameters. Nevertheless, it is still not applied to structural design optimization problems. Furthermore, like many other swarm-based metaheuristic algorithms, the SMA also suffers from a slow convergence rate

and prematurely converges to non-optimal solutions, especially when dealing with large-scale optimization problems [17].

To address these drawbacks, in this study, an effectively improved slime mould algorithm called ISMA is proposed and then applied to the optimal design of truss structures with natural frequency constraints. The key contributions of the proposed ISMA can be summarized as follows: (1) an elitist strategy is adopted in the replacement phase to promote the convergence rate, and (2) a slight modification is made to the exploration phase to avoid premature convergence. The efficiency of the proposed ISMA is investigated through three large-scale structural optimization problems with natural frequency constraints. The results obtained by the proposed ISMA are compared with those of the classical SMA and some other state-of-the-art metaheuristic algorithms. It is shown that the proposed ISMA not only overcomes the shortcoming of the premature convergence of the classical SMA but also promotes the convergence rate. The results obtained by the proposed ISMA are significantly superior to those of the classical SMA and are very competitive to those of other state-of-the-art metaheuristics.

The rest of this chapter is structured as follows: The classical slime mould algorithm is overviewed in Sect. 9.2. In Sect. 9.3, the improved slime mould algorithm (ISMA) is proposed. In Sect. 9.4, the formulation of the optimization problem is presented. In Sect. 9.5, three design examples are presented and discussed. Finally, Sect. 9.6 draws concluding remarks.

9.2 Overview of the Slime Mould Algorithm (SMA)

The slime mould algorithm (SMA) is a population-based nature-inspired metaheuristic algorithm recently proposed by Li et al. [13]. The SMA mimics the foraging behavior of the acellular slime mould *Physarum polycephalum*. The term “slime mould” hereafter refers to *Physarum polycephalum*. The plasmodium of slime mould is a large amoeboid organism consisting of a network of interlaced tube-like structures, allowing cytoplasm to flow throughout the venous network. Because of its unique morphology, slime mould can forage simultaneously on multiple food sources by forming intricate networks of veins connecting them. When a slime mould finds a food source, the biochemical oscillator of the slime mould propagates contraction waves throughout the venous network, leading to cytoplasmic streaming within the tubular veins. The thickness of the vein is directly proportional to the rate of cytoplasmic streaming. Thus, an increased rate of cytoplasmic streaming results in an increment in the vein thickness, whereas the vein contracts in response to a decrease in the rate of cytoplasmic streaming. This combined positive and negative feedback makes it possible for the slime mould to reach food sources via the optimal path.

The mathematical representation of the classical SMA is as follows: The first step of the SMA is to generate randomly the initial population of solutions (i.e., the initial slime moulds) with respect to the specified lower and upper limits of the search space, as shown in Eq. (9.1):

$$X^1(:, j) = lb(j) + r_1(popsize, 1) \cdot (ub(j) - lb(j)), \forall j \in \{1, 2, \dots, D\} \quad (9.1)$$

where.* denotes element by element multiplication, $popsize$ is the population size, D is the number of design variables; $lb(j)$ and $ub(j)$ represent the lower and upper bounds of the j -th design variable, respectively; $r_1(popsize, 1)$ is a column vector of $popsize$ uniformly distributed pseudorandom numbers between 0 and 1; and $X^1(i, :)$ represents the position of the i -th slime mould of the initial population, $i = 1, 2, \dots, popsize$.

Slime moulds are sensitive to odors in the environment and can detect food sources. Based on this property, the position update rule of the classical SMA algorithm is defined as follows:

$$X^{t+1}(i, j) = \begin{cases} lb(j) + (ub(j) - lb(j)) * r_3(i), r_2(i) < z \\ X_b^t(j) + vb(i, j) * (W(i, j) * X_A^t(i, j) - X_B^t(i, j)), r_2(i) \geq z \text{ and } r_4(i, j) < p(i) \\ vc(i, j) * X^t(i, j), r_2(i) \geq z \text{ and } r_4(i, j) \geq p(i) \end{cases} \quad (9.2)$$

where $vb(i, :)$ is a row vector of random numbers chosen from the continuous uniform distribution on the interval from $-a$ to a , where the value of a gradually approaches 0 as the iteration number increases; X_b^t represents the position of the best slime mould found so far; $vc(i, :)$ is a row vector of random numbers chosen from the continuous uniform distribution on the interval from $-b$ to b , where the value of b linearly varies from 1 to 0 as the iteration number increases; X_A^t and X_B^t represent the positions of two slime moulds randomly selected from the current (i.e., t -th) population of slime moulds; $W(i, :)$ is the weight of the i -th slime mould; $X^t(i, :)$ and $X^{t+1}(i, :)$ represent the current and updated positions of the i -th slime mould, respectively; r_2 , r_3 and r_4 are uniformly distributed pseudorandom numbers between 0 and 1; z is a parameter to control the balance between exploration and exploitation of the search. Li et al. [13] recommended a value of 0.03 for the parameter z . The value of $p(i)$ is determined based on the fitness value of the i -th slime mould and the fitness value of the best slime mould found so far. The formula of $p(i)$ is as follows:

$$p(i) = \tanh|f(X^t(i, :)) - f(X_b^t)| \quad (9.3)$$

where $f(X^t(i, :))$ is the fitness of the i -th slime mould, $i = 1, 2, \dots, popsize$; $f(X_b^t)$ is the fitness of the best slime mould found so far; $|.|$ represents the absolute value function; and $\tanh(.)$ is the tangent hyperbolic function.

The row vectors $vb(i, :)$ and $vc(i, :)$ are defined as follows:

$$vb(i, :) = unifrnd(-a, a, 1, D) \quad (9.4)$$

$$vc(i, :) = unifrnd(-b, b, 1, D) \quad (9.5)$$

where the parameters a and b are defined by

$$a = \tanh^{-1} \left(1 - \frac{t}{t_{max}} \right) \quad (9.6)$$

$$b = 1 - \frac{t}{t_{max}} \quad (9.7)$$

where t is the current iteration number; t_{max} is the maximum iteration number; and $\tanh^{-1}(.)$ is the inverse tangent hyperbolic function.

The SMA uses adaptive weights to simulate the positive and negative feedback between vein thickness and cytoplasmic streaming rate. The W is determined by the following equation:

$$W(Index(i), j) = \begin{cases} 1 + r_5(i, j) * \log_{10} \left(\frac{f_{Best} - f(X^t(Index(i), :))}{f_{Best} - f_{Worst} + eps} + 1 \right), & i \leq \frac{popsize}{2} \\ 1 - r_5(i, j) * \log_{10} \left(\frac{f_{Best} - f(X^t(Index(i), :))}{f_{Best} - f_{Worst} + eps} + 1 \right), & i > \frac{popsize}{2} \end{cases} \quad (9.8)$$

where r_5 is a uniformly distributed pseudorandom number between 0 and 1; f_{Best} and f_{Worst} denote the best and worst fitness values of the current population of slime moulds, respectively; eps is a very small floating-point number (i.e., 2^{-52}) to avoid singularity; $Index$ denotes the sequence of slime moulds sorted in ascending order of fitness values for a minimization problem, or in descending order for a maximization problem. The $Index$ is given by

$$[\sim, Index] = sort(F) \quad (9.9)$$

where

$$F = (f(X^t(1, :)), f(X^t(2, :)), \dots, f(X^t(popsize, :))) \quad (9.10)$$

By using the notations used in Eqs. (9.9) and (9.10), f_{Best} and f_{Worst} are determined as follows:

$$f_{Best} = f(X^t(Index(1), :)) \quad (9.11)$$

$$f_{Worst} = f(X^t(Index(popsize), :)) \quad (9.12)$$

If the boundary constraints of the j -th design variable of the i -th candidate solution are violated, it will return to the violated bound as follows:

$$X^t(i, j) = \begin{cases} lb(j), & X^t(i, j) < lb(j) \\ ub(j), & X^t(i, j) > ub(j) \end{cases} \quad (9.13)$$

A step-by-step explanation of the classical SMA is provided below:

Step one (initialization): The algorithm parameters, i.e., $popsize$, t_{max} , and z , are fixed. The iteration number is set to one. Next, the initial population is generated randomly within the given bounds of the design variables.

Step two (main body of the classical SMA): The boundary constraints are imposed by Eq. (9.13). Next, the population of slime moulds is evaluated and the best slime mould obtained so far is updated. The weight coefficients of the slime moulds are determined by Eq. (9.8). Afterwards, the parameters a and b are calculated by Eqs. (9.6) and (9.7), respectively. The positions of the slime moulds are updated by Eq. (9.2). The iteration number is updated.

Step three (termination criterion): If the termination criterion is met, the algorithm is stopped and the best slime mould obtained so far is reported, otherwise it iterates with step two.

The pseudocode of the classical SMA is depicted in Fig. 9.1.

Pseudocode of slime mould algorithm

Begin

Initialization phase:

Set the algorithm parameters: $popsize$, t_{max} , and z

Initialize the iteration number: $t = 1$

Initialize the population of slime moulds by Eq. (9.1): $X^t(i, :)$, $i = 1, 2, \dots, popsize$

Cyclic body of the classical SMA:

While $t \leq t_{max}$

 Control the boundary constraints by Eq. (9.13)

 Evaluate the fitness of all the slime moulds in the current population

 Update the best slime mould obtained so far, X_b^t

 Calculate the W for all the slime moulds by Eq. (9.8)

 Calculate the a by Eq. (9.6) and b by Eq. (9.7)

For $i = 1: popsize$

 Update the position of the i -th slime mould by Eq. (9.2)

End For i

$t = t + 1$

End While

Return the best slime mould obtained so far, X_b^t

End

Fig. 9.1 Pseudocode of the classical slime mould algorithm

9.3 Proposed Improved Slime Mould Algorithm (ISMA)

The classical version of SMA adopts a full generational replacement, where the current population is completely replaced by the updated one at each iteration. Such a replacement strategy allows for a greater exploration of the search space. The main disadvantage of the generational replacement strategy is the high probability of losing promising solutions. This may lead to poor exploration capability and a slow convergence rate. To address the problem of losing promising solutions during the optimization process, De Jong [18] introduced the elitist strategy, where the best individuals of each iteration are preserved in the next ones. This approach helps to impose higher selection pressure, which results in a faster convergence rate but also a higher possibility of premature convergence. Nguyen et al. [17] noted that SMA suffers from a slow convergence rate and is prone to trap into local optima, especially when dealing with large-scale optimization problems. Thus, in the replacement phase of the proposed ISMA, an elitist strategy that helps to preserve the best individuals of each iteration to the next iterations is adopted to replace the generational replacement strategy of the classical SMA. The elitist strategy is adopted to improve the convergence rate and enhance the exploitation capability of the ISMA. Figure 9.2 illustrates the mechanism of the elitist strategy. Therefore, in the proposed ISMA, the new position of each slime mould will replace its corresponding current position if the new position is more fitted than the current one. This can be expressed mathematically as follows:

$$X^{t+1}(i, :) = \begin{cases} X^t(i, :), f(X^t(i, :)) < f(X_{new}^t(i, :)) \\ X_{new}^t(i, :), f(X_{new}^t(i, :)) < f(X^t(i, :)) \end{cases} \quad (9.14)$$

where $X_{new}^t(i, :)$ is the new position of the i -th slime mould (see Eq. (9.15)).

As can be seen from Eq. (9.2), in the classical SMA, when the value of the pseudo-random number $r_2(i)$ is smaller than the parameter z , the position of the slime mould i is updated based on the equation $X^{t+1}(i, j) = lb(j) + (ub(j) - lb(j)) * r_3(i)$, $j = 1, 2, \dots, D$. It follows from Eq. (9.2) that when $r_2(i) < z$, all the components

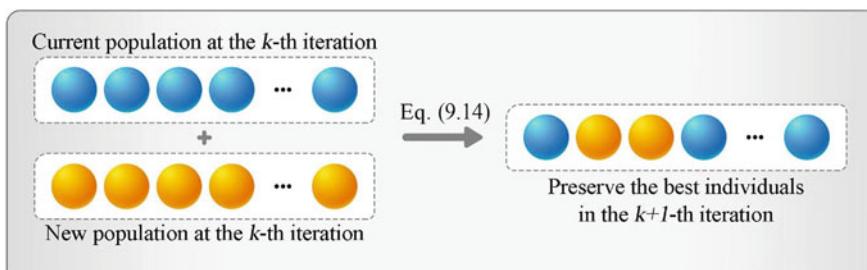


Fig. 9.2 Mechanism of the elitist strategy

$ub(j) - lb(j)$, $j = 1, 2, \dots, D$ are multiplied by the same pseudorandom number $r_3(i)$. Keeping this in mind, let us consider the case where all the design variables of the optimization problem at hand are constrained by the same bounds, i.e., when $lb(1) = lb(2) = \dots = lb(D)$ and $ub(1) = ub(2) = \dots = ub(D)$. In such a case, for each individual, all the design variables have the same value. This means that the individuals are generated randomly in only a small region of the search space. Consequently, such a position update rule may cause difficulties in the exploration of the search and may lead to premature convergence to non-optimal solutions. To overcome this problem and explore new regions of the search space, we slightly modify the formulation of the position update rule of the classical SMA as follows:

$$X_{new}^t(i, j) = \begin{cases} lb(j) + (ub(j) - lb(j)) * r_6(i, j), & r_2(i) < z \\ X_B^t(j) + vb(i, j) * (W(i, j) * X_A^t(i, j) - X_B^t(i, j)), & r_2(i) \geq z \text{ and } r_4(i, j) < p(i) \\ vc(i, j) * X^t(i, j), & r_2(i) \geq z \text{ and } r_4(i, j) \geq p(i) \end{cases} \quad (9.15)$$

where $r_6(1, D)$ is a row vector of D uniformly distributed pseudorandom numbers between 0 and 1. The rest of the parameters are defined similarly to those in Eq. (9.2).

In the proposed position update rule, when $r_2(i) < z$, the position of the slime mould i is updated based on the equation $X_{new}^t(i, j) = lb(j) + (ub(j) - lb(j)) * r_6(i, j)$, $j = 1, 2, \dots, D$. A careful examination of the equation suggests that there exist a distinct pseudorandom number $r_6(i, j)$ for each component $ub(j) - lb(j)$, $j = 1, 2, \dots, D$. Therefore, such a definition of position update rule makes it possible for slime moulds to search in all possible directions, and thus can effectively maintain the diversity of the population during the search process and prevent premature convergence. Figure 9.3 compares the exploration phase of the classical SMA and the proposed ISMA. As the figure shows, in the case of a two-dimensional search space, all possible positions of slime moulds in the exploration phase of the classical SMA are located on a line. In contrast, in the exploration phase of the proposed ISMA, the slime moulds can explore the whole search space.

A step-by-step explanation of the proposed ISMA is provided below:

Step one (initialization): The algorithm parameters, i.e., $popsize$, t_{max} , and z , are fixed. The iteration number is set to unity. The initial population is generated randomly within the given bounds of the design variables. The boundary constraints are imposed by Eq. (9.13). Next, the initial population of slime moulds is evaluated.

Step two (main body of the proposed ISMA): The best slime mould obtained so far is updated. The weight coefficients of the slime moulds are determined by Eq. (9.8). Afterwards, the parameters a and b are calculated by Eqs. (9.6) and (9.7), respectively. Next, the new positions of the slime moulds are updated by Eq. (9.15). The boundary constraints are imposed by Eq. (9.13) and the new population of slime moulds is evaluated. Next, the elitist selection strategy is applied by Eq. (9.14). The iteration number is updated.

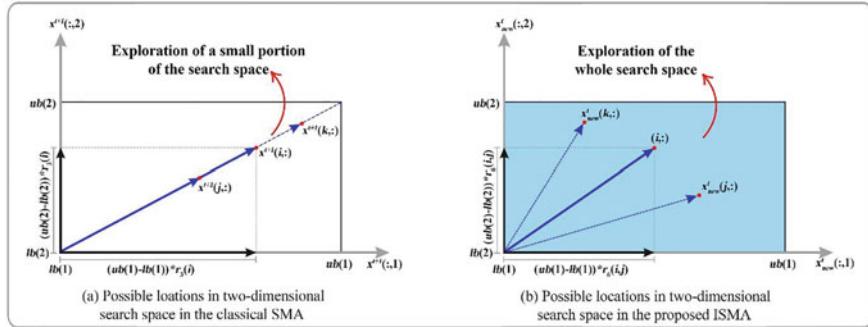


Fig. 9.3 Comparison of the exploration phase of the classical SMA and the proposed ISMA

Step three (termination criterion): If the termination criterion is met, the algorithm is stopped and the best slime mould obtained so far is reported, otherwise it iterates with step two.

Figures 9.4 and 9.5 depict the flowchart and pseudocode of the proposed ISMA, respectively. It should be noted that the proposed ISMA is designed to be applied to the optimal design of truss structures with frequency constraints. However, based on the no-free-lunch theorem, it is impossible to find a general-purpose universal metaheuristic that can efficiently solve all types of optimization problems. So, similar to other population-based metaheuristic algorithms, the proposed ISMA may still be expensive in computational terms to seek the global optimal solution of some kinds of optimization problems. In addition, similar to the classical SMA, the proposed ISMA is not applicable to optimization problems with discrete design variables.

9.4 Formulation of the Optimization Problem

In a truss optimization problem with natural frequency constraints, the objective is to minimize the total weight of the truss structure while satisfying multiple constraints on natural frequencies. The sizing optimization problem of a truss structure is mathematically formulated as follows [5]:

$$\text{Find: } \{A\} = [A_1, A_2, \dots, A_d] \quad (9.16)$$

$$\text{to minimize: } f(\{A\}) = \sum_{i=1}^{nm} \rho_i \times A_i \times L_i \quad (9.17)$$

$$\text{subject to: } \begin{cases} \omega_j \geq \omega_j^* \text{ for some natural vibration frequencies } j \\ \omega_k \leq \omega_k^* \text{ for some natural vibration frequencies } k \\ A_i^{\min} \leq A_i \leq A_i^{\max} i = 1, 2, \dots, d \end{cases} \quad (9.18)$$

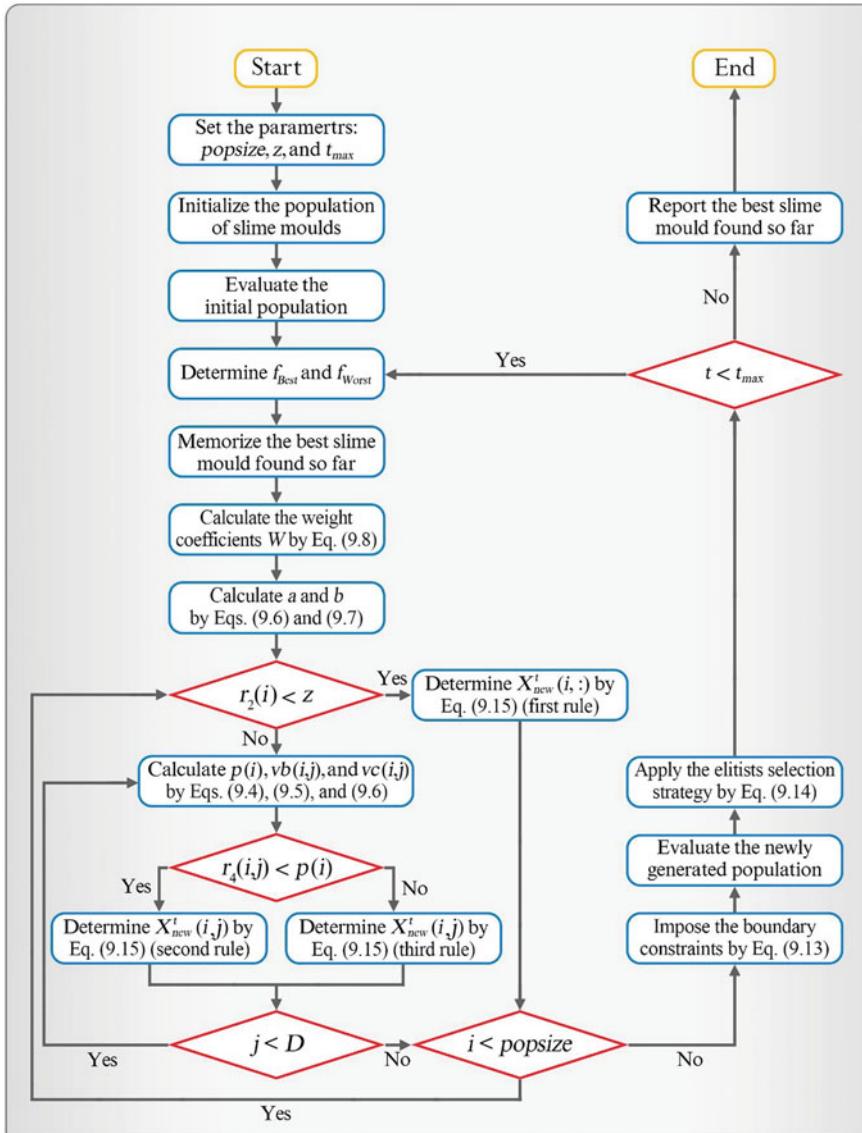


Fig. 9.4 Flowchart of the proposed improved slime mould algorithm

where $f(\{A\})$ is the objective function, which is taken as the weight of the structure; $\{A\}$ is the vector of design variables, i.e., the cross-sectional areas of structural members; d is the number of sizing design variables; nm is the number of structural members; A_i , ρ_i and L_i are the cross-sectional area, material density, and length of the i -th structural member, respectively; A_i^{min} and A_i^{max} are the lower and upper

Pseudocode of improved slime mould algorithm

Begin

Initialization phase:

Set the algorithm parameters: $popsize$, t_{max} , and z

Initialize the iteration number: $t = 1$

Initialize the population of slime moulds by Eq. (9.1): $X^t(i, :)$, $i = 1, 2, \dots, popsize$

Control the boundary constraints by Eq. (9.13)

Evaluate the fitness of all the slime moulds in the initial population

Cyclic body of the ISMA:

While $t \leq t_{max}$

 Update the best slime mould obtained so far, X_b^t

 Calculate the W for all the slime moulds by Eq. (9.8)

 Calculate the a by Eq. (9.6) and b by Eq. (9.7)

For $i = 1: popsize$

 Determine the new position of the i -th slime mould by Eq. (9.15)

 Control the boundary constraints by Eq. (9.13)

 Evaluate the fitness corresponding to the new position of the i -th slime mould

 Apply the elitist selection strategy by Eq. (9.14)

End For i

$t = t + 1$

End While

Return the best slime mould obtained so far, X_b^t

End

Fig. 9.5 Pseudocode of the proposed improved slime mould algorithm

bounds of the i -th design variable, respectively; ω_j and ω_j^* are the j -th natural frequency of the structure and its corresponding lower bound, respectively; and ω_k and ω_k^* are the k -th natural frequency of the structure and its corresponding upper bound, respectively.

There are various constraint handling approaches to deal with constraints in optimization problems, one of the most common of which is penalizing strategies. The general idea of penalizing strategies is to transform the constrained optimization problem into an unconstrained one. Therefore, by incorporating a penalty term in the objective function, the above problem can be rewritten as follows:

$$\text{Minimize: } f_{cost}(\{A\}) = f(\{A\}) \times f_{penalty}(\{A\}) \quad (9.19)$$

where $f_{penalty}(\{A\})$ is the penalty function; and $f_{cost}(\{A\})$ is the penalized objective function. In this study, the following well-known multiplicative penalty function, known as the Kaveh-Zolghadr technique, is utilized to handle the frequency constraints [19]:

$$f_{penalty}(\{A\}) = (1 + \varepsilon_1 \times c)^{\varepsilon_2}; c = \sum_{i=1}^{nc} c_i \quad (9.20)$$

where nc is the number of frequency constraints; c denotes the sum of the constraint violations; and ε_1 and ε_2 are parameters of the objective function. If the i -th constraint is satisfied, then the value of c_i is set to zero; otherwise, it is determined by the severity of the violation. It can be expressed as follows:

$$c_i = \begin{cases} \left| 1 - \frac{\omega_i}{\omega_i^*} \right|, & \text{the } i\text{-th frequency constraint is violated} \\ 0, & \text{otherwise} \end{cases} \quad (9.21)$$

The parameters ε_1 and ε_2 determine the degree to which an infeasible solution is penalized. These parameters have a significant influence on the convergence characteristics since they affect the exploration and exploitation of the search space. In this study, the parameter ε_1 is set equal to unity, while the parameter ε_2 is set to increase linearly with the number of iterations. This means that the degree of penalizing the infeasible solutions increases with time. In other words, as the search progresses, the infeasible solutions are penalized more severely. As a result, the search space is explored freely in the early stages of the search process (i.e., more diversification), but more emphasis on the exploitation of the promising regions of the search space (i.e., more intensification) in the last iterations [19].

The natural frequencies and natural modes of vibration of a structure without damping are determined by the solution of the following algebraic equation, known as the matrix eigenvalue problem:

$$k\phi_n = \omega_n^2 m\phi_n \quad (9.22)$$

where k and m are the stiffness and mass matrices of the structure, respectively; ω_n is the n -th natural frequency of vibration of the structure; ϕ_n is the n -th natural mode of vibration of the structure; and N is the number of degrees of freedom of the structure ($n = 1, 2, \dots, N$).

9.5 Numerical Examples

In this section, to demonstrate the efficiency and applicability of the proposed method, three large-scale dome-like truss optimization problems with natural frequency constraints are studied. The performance of the proposed ISMA is compared with the classical SMA and some other state-of-the-art metaheuristics. Furthermore, all numerical examples are also solved using the success-history based adaptive differential evolution (SHADE) algorithm [20], a highly competitive state-of-the-art metaheuristic from the CEC competitions, and the obtained results are compared with

Table 9.1 Material properties, cross-sectional area bounds, and frequency constraints of the problems

Property	600-bar truss	1180-bar truss	1410-bar truss
Elasticity modulus (N/m ²)	2×10^{11}	2×10^{11}	2×10^{11}
Material density (kg/m ³)	7850	7850	7850
Cross-sectional area bounds (m ²)	$0.0001 \leq A_i \leq 0.01$	$0.0001 \leq A_i \leq 0.01$	$0.0001 \leq A_i \leq 0.01$
Frequency constraints (Hz)	$\omega_1 \geq 5, \omega_3 \geq 7$	$\omega_1 \geq 7, \omega_3 \geq 9$	$\omega_1 \geq 7, \omega_3 \geq 9$

those of the SMA and ISMA. The SHADE algorithm is an enhancement to adaptive differential evolution with optional external archive (JADE) which employs a history-based parameter adaptation scheme [21]. The SHADE algorithm was ranked the fourth best out of 21 algorithms in the CEC 2013 competition. The parameter values suggested by Tanabe and Fukunaga [20] were used for the SHADE algorithm. Due to the stochastic nature of the optimization process, 25 independent runs are conducted for each algorithm on each example. The optimization results are reported in terms of the best, average, and worst optimized weights, the standard deviation on the optimized weights, possible constraint violations, the number of finite element (FE) analyses to obtain the best weight, the maximum number of FE analyses, and the number of runs required to achieve 25 successful runs. In order to make a fair comparison of all the algorithms, the Friedman rank test is performed to rank all the considered optimizers on the basis of the best weight, average weight, and standard deviation of weights. It is noted that only the optimal design corresponding to the best run is reported. The maximum number of iterations is considered as the termination condition of the algorithms. In all the design examples, the population size is set to be 20 (*popsize* = 20), and the maximum number of iterations is set to be 1000 (*t_{max}* = 1000) for both the SMA and ISMA. These values are selected based on the results of preliminary trials and the general recommendations given in the literature [22]. The parameter *z* is fixed at 0.03 for both the SMA and ISMA [13]. The design examples include a 600-bar dome-like truss with 25 design variables, an 1180-bar dome-like truss with 59 design variables, and a 1410-bar dome-like truss with 47 design variables. Table 9.1 summarizes the material properties, cross-sectional area bounds, and frequency constraints for all the investigated problems. The FE models and the optimization algorithms are implemented in the Matlab environment. The FEM code for free vibration analysis of truss structures was validated with existing results in literature.

9.5.1 A 600-Bar Dome-Like Truss

The first example considered is the 600-bar single-layer dome structure shown in Fig. 9.6. The structure has 216 nodes and 600 elements and consists of 24 identical

substructures forming a cyclic symmetric pattern with the angle of cyclic symmetry equal to 15 degrees (i.e., the structure could be formed by the cyclic repetition of the substructure around the axis of rotational symmetry of the structure). Each substructure has 9 nodes and 25 elements (see Fig. 9.6). Table 9.2 provides the nodal coordinates of the first substructure in the Cartesian coordinate system. The connectivity information of the first substructure is also given in Table 9.3. The cross-sectional areas of the elements of the sub-structure represent the design variables of the problem. The layout of the structure is fixed by the designer. Therefore, this is a truss sizing optimization problem with 25 design variables. A non-structural mass of 100 kg is attached to all free nodes of the dome. As Table 9.1 shows, the natural frequency constraints are imposed on the first and third natural frequencies. This problem has been previously studied by Kaveh [7], Kaveh and Ghazaan [8, 9], Degertekin et al. [12], etc.

In this example, the effectiveness of the elitist strategy, as well as the modification in the exploration phase of the classical SMA is investigated in detail. The optimal results obtained by the present algorithms with those of the other methods are summarized in Table 9.3 for comparison. From Table 9.3, it is found that the best weight, average weight, and worst weight of the ISMA are better than those obtained by the SMA and other competitors. It should be noted that the average weight of the SHADE algorithm is 6084.15 kg, which is slightly heavier than that of the ISMA (i.e., 6083.93 kg). Table 9.4 presents the Friedman rank test results for the 600-bar dome-like truss problem. The results support that the proposed ISMA ranked first followed by the SHADE algorithm in terms of the best weight and average weight. However, in terms of standard deviation, the SHADE algorithm ranked first followed by the ISMA. It is also observed from the results that the improvements made in the proposed ISMA have improved its performance significantly compared with the classical SMA. After only 10,520 FE analyses, ISMA has obtained a feasible optimum design with a structural weight of 6072.97 kg, which is better than those achieved by other compared methods. The worst weight of the ISMA is 6095.41 kg, which is even better than the averages obtained by other techniques, except for the SHADE. It can be seen that the best weight achieved by the SMA is 6097.16 kg, which is better than those of the other methods in the literature, except for the EFBI [11] and the SHADE algorithm. Nevertheless, the SMA showed the worst result in terms of average weight with 6965.94 kg, which is much heavier than that of the ISMA (i.e., about 882.01 kg (14.50%)), as well as other competitors. This is due to the fact that the classical SMA has prematurely converged to non-optimal solutions in many runs. Table 9.5 provides the optimized weights obtained by the SMA and ISMA over 25 independent runs. From Table 9.5, it can be seen that, in 56% of the runs (i.e., in 14 out of 25 independent runs), the classical SMA has prematurely converged to non-optimal solutions so far away from other near-optimal solutions. As pointed out earlier in Sect. 9.3, this occurs because of the poor exploration ability of the classical SMA, which can be attributed to the loss of population diversity in the early stages of the search process in the classical SMA. On the contrary, as can be seen from Table 9.5, thanks to the modification of the position update rule of the exploration phase of the ISMA, it can effectively escape from non-optimal solutions and reach near-optimal solutions.

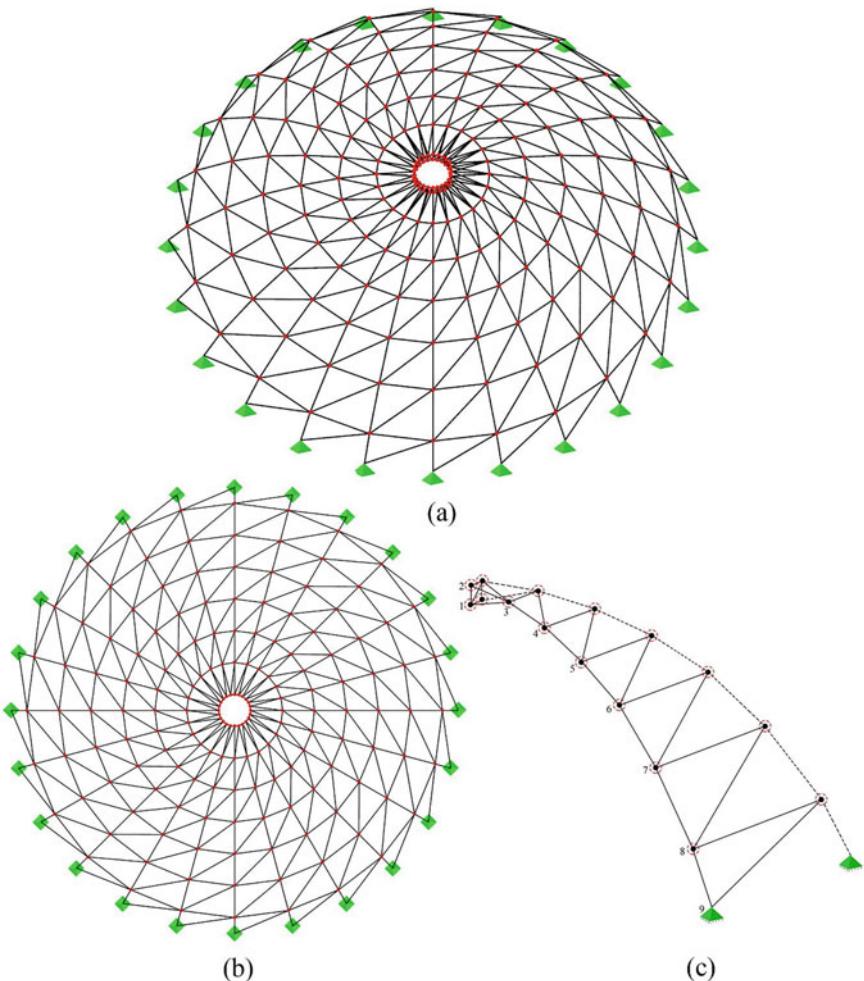


Fig. 9.6 Schematic of the 600-bar single-layer dome: **a** perspective view; **b** top view; **c** substructure

Moreover, in cases where the classical SMA has successfully escaped from non-optimal solutions, the optimized weights are still heavier than those of the ISMA, which can be attributed to the elitist strategy adopted in the selection phase of the proposed ISMA. Figure 9.7 depicts the optimized weights obtained by the SMA and ISMA over 25 independent runs. It is obvious that the classical SMA has converged to non-optimal solutions in many of the runs. On the other hand, it is seen that the proposed ISMA has consistently converged to near-optimal solution. Table 9.6 lists the first five natural frequencies of the best designs of the 600-bar dome-like truss structure obtained by the SMA, ISMA, and other compared methods. In addition, the first and third natural frequencies derived from our finite element method (FEM) code for free vibration analysis of truss structures are also provided (see rows 1* and

Table 9.2 Nodal coordinates (m) of the first sub-structure of the 600-bar dome-like truss

Node number	Coordinates (x, y, z)
1	(1.0, 0.0, 7.0)
2	(1.0, 0.0, 7.5)
3	(3.0, 0.0, 7.25)
4	(5.0, 0.0, 6.75)
5	(7.0, 0.0, 6.0)
6	(9.0, 0.0, 5.0)
7	(11.0, 0.0, 3.5)
8	(13.0, 0.0, 1.5)
9	(14.0, 0.0, 0.0)

3* in Table 9.6). It is seen that none of the methods violate the frequency constraints. Accordingly, as can be seen in Table 9.3, the constraint violations for all methods are zero. A careful examination of the table shows that the natural frequencies derived from our FEM code are quite close to those reported in the literature and very small differences (by a maximum of 0.304%) are due to the rounding of the values of the design variables reported in the literature. In terms of computational cost, the maximum number FE analyses of the proposed ISMA is 20000, which is less than or equal to those of other competitors, except for EFBI [11] with 12,000 and for DPSO [7] with 9000. However, ISMA has found a feasible optimum design with a structural weight of 6321.62 kg after 4660 FE analyses, which is better than those of the DPSO [7] and PFJA [12] after 9000 and 8580 FE analyses, respectively. The ISMA requires only 10,520 FE analyses to find a feasible optimum design with a structural weight of 6072.97 kg, which is better than those of the ECBO-Cascade [8], VPS [9], EFBI [11], MDVC-UVPS [23], and SHADE after 17,300, 19,740, 12,000, 17,513, and 19,900 FE analyses, respectively. Figure 9.8 provides the convergence histories of the average results obtained by the SMA and ISMA over 25 independent runs. As Fig. 9.8 suggests, the classical SMA exhibits a much faster convergence rate up to about 50-th iteration, but thereafter the rate of convergence decreases strictly, which often results in premature convergence to non-optimal solutions. This results from the local-search behavior of the exploration phase of the classical SMA, which causes loss of population diversity in the early stages of the search process, as illustrated in Fig. 9.3. From Fig. 9.8, it can be observed that the convergence behavior of the proposed ISMA is remarkably improved in comparison with the classical SMA. Indeed, in the first few iterations (i.e., before about 50-th iteration), the ISMA converges slower than the classical SMA, which can be regarded as the result of the exploration of the whole search space. However, in the subsequent iterations, due to the elitist strategy of the proposed ISMA, it converges much faster than the classical SMA. Therefore, the ISMA can strike a much better balance between exploration and exploitation as compared to the classical SMA.

Table 9.3 Comparison of optimal results obtained for the 600-bar dome-like truss (cm^2)

Element number (element nodes)	DPSO [7]	VPS [9]	ECBO-Cascade [8]	PFAA [12]	CPA [10]	MDVCAUVPS [23]	EFBI [11]	This study SHADE	SMA	ISMA
1 (1–2)	1.365	1.3030	1.0299	1.1867	1.155	1.2575	1.0999	1.7304	1.2210	1.1035
2 (1–3)	1.391	1.3998	1.3664	1.2967	1.304	1.3466	1.4922	1.3402	1.3619	1.5801
3 (1–10)	5.686	5.1072	5.1095	4.5771	4.178	4.9738	6.0744	5.6627	4.7700	6.2180
4 (1–11)	1.511	1.3882	1.3011	1.3356	1.335	1.4025	1.6234	1.3072	1.2947	1.0522
5 (2–3)	17.711	16.9217	17.0572	18.3157	18.375	17.3802	17.4918	16.8402	18.0803	17.1566
6 (2–11)	36.266	38.1432	34.0764	38.5097	39.914	37.9742	37.2118	37.9838	37.5861	36.5568
7 (3–4)	13.263	11.8319	13.0985	13.5917	13.609	13.0306	12.7873	13.2350	12.0041	12.8425
8 (3–11)	16.919	16.6149	15.5882	16.8824	16.470	15.9209	14.8239	15.1755	14.8771	15.3463
9 (3–12)	13.333	11.3403	12.6889	13.8766	14.108	11.9419	12.1764	10.9455	11.0374	11.9044
10 (4–5)	9.534	9.3865	10.3314	9.5286	10.038	9.1643	9.0163	9.7881	10.2072	9.4559
11 (4–12)	9.884	8.7692	8.5313	9.4218	9.514	8.4332	8.5044	8.2877	8.5748	8.1976
12 (4–13)	9.547	9.6682	9.8308	9.7643	9.329	9.2375	8.9951	8.8861	9.0565	9.0644
13 (5–6)	7.866	6.9826	7.0101	7.2431	6.938	7.2213	7.0357	7.0087	6.4887	7.6937
14 (5–13)	5.529	5.4445	5.2917	5.3913	5.545	5.2142	5.0993	5.4440	5.0230	5.1748
15 (5–14)	7.007	6.3247	6.2750	6.7468	6.763	6.7961	6.1918	6.2256	6.4099	6.7264
16 (6–7)	5.462	5.1349	5.4305	5.1493	5.209	5.2078	4.9514	5.3580	5.2841	4.8059
17 (6–14)	3.853	3.3991	3.6414	3.8342	3.842	3.4586	3.9186	3.7330	3.8829	3.6390
18 (6–15)	7.432	7.7911	7.2827	8.0665	8.112	7.6407	7.6312	7.4317	7.4162	7.7180
19 (7–8)	4.261	4.4147	4.4912	4.2800	4.252	4.3690	4.4271	4.3099	5.0024	4.0911
20 (7–15)	2.253	2.2755	1.9275	2.2509	2.227	2.1237	2.3280	2.2720	2.4209	2.1339

(continued)

Table 9.3 (continued)

	DPSO [7]	VPS [9] [8]	ECBO-Cascade [8]	PFJA [12]	CPA [10]	MDVC-UVPS [23]	EFBI [11]	This study SHADE	SMA	ISMA
Element number (element nodes)										
21 (7–16)	4.337	4.9974	4.6958	4.5372	4.582	4.5774	4.8534	4.6150	4.8491	4.4482
22 (8–9)	4.028	4.0145	3.3595	3.5615	3.336	3.4564	3.9632	3.2466	2.9875	3.4785
23 (8–16)	1.954	1.8388	1.7067	1.7744	1.725	1.7920	1.8527	1.8219	1.5063	1.8191
24 (8–17)	4.709	4.7965	4.8372	4.6445	4.675	4.8264	4.7818	4.9856	4.8696	4.8903
25 (9–17)	1.410	1.5551	2.0253	1.6141	1.673	1.7601	1.4354	1.6023	1.9253	1.7001
Weight (kg)	6344.55	6133.02	6140.51	6333.251	6336.85	6115.10	6076.35	6074.65	6097.16	6068.34
Number of FE analyses	N/A	19,740	17,300	8580	N/A	17,513	N/A	19,900	19,620	20,000
Average weight (kg)	6674.71	6142.03	6175.33	6380.31	6376.01	6119.95	6098.52	6084.15	6965.94	6083.93
Worst weight (kg)	N/A	N/A	N/A	N/A	N/A	N/A	6113.56	6098.01	7655.18	6095.41
Standard deviation (kg)	473.21	12.54	34.08	47.396	90.39	16.23	11.95	5.16	756.46	7.36
Maximum number of FE analyses	9000	30,000	20,000	25,000	40,000	30,000	12,000	20,000	20,000	20,000
Constraint violation (%)	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Number of runs	10	5	5	20	10	30	10	25 out of 31	25 out of 30	25 out of 28

Table 9.4 The Friedman rank test results for the 600-bar dome-like truss

	DPSO [7]	VPS [9]	ECBO-Cascade [8]	PFJA [12]	CPA [10]	MDVC-UVPS [23]	EFBI [11]	This study
Friedman rank of minimum weight	10	6	7	8	9	5	3	SHADE
Friedman rank of average weight	9	5	6	8	7	4	3	SMA
Friedman rank of standard deviation	9	4	6	7	8	5	3	ISMA

Table 9.5 Comparison of optimized weights obtained by SMA and ISMA over 25 independent runs

Run number	SMA (kg)	ISMA (kg)
Run #1	7646.11	6074.00
Run #2	6108.55	6084.21
Run #3	7640.79	6089.28
Run #4	7643.56	6077.88
Run #5	7626.31	6076.97
Run #6	6119.68	6087.94
Run #7	6097.16	6084.42
Run #8	6106.94	6068.34
Run #9	7624.78	6091.83
Run #10	7647.24	6076.57
Run #11	6128.72	6083.36
Run #12	7636.35	6095.41
Run #13	6114.76	6092.49
Run #14	7623.05	6094.55
Run #15	6115.06	6081.07
Run #16	6119.44	6074.58
Run #17	7624.90	6094.70
Run #18	6115.12	6077.01
Run #19	7623.73	6085.81
Run #20	7652.91	6083.43
Run #21	7633.02	6079.74
Run #22	6112.20	6095.07
Run #23	7631.82	6087.37
Run #24	6101.08	6083.87
Run #25	7655.18	6078.44

9.5.2 A 1180-Bar Dome-Like Truss

The 1180-bar single-layer dome structure shown in Fig. 9.9 is considered as the second example. The structure is composed of 20 identical substructures and exhibits cyclic symmetry. The angle of cyclic symmetry is equal to 18 degrees. Each substructure consists of 20 nodes and 59 elements (see Fig. 9.9). The nodal coordinates of the first substructure in the Cartesian coordinate system are provided in Table 9.7. The connectivity information of the first substructure is also given in Table 9.8. The cross-sectional areas of the elements of the substructure are considered as design variables of the problem. It is assumed that the layout of the structure is predefined and kept fixed throughout the design process. Therefore, this is a truss optimization problem with 59 sizing design variables. A non-structural mass of 100 kg is attached to all free nodes of the dome. Similar to the previous example, the natural frequency

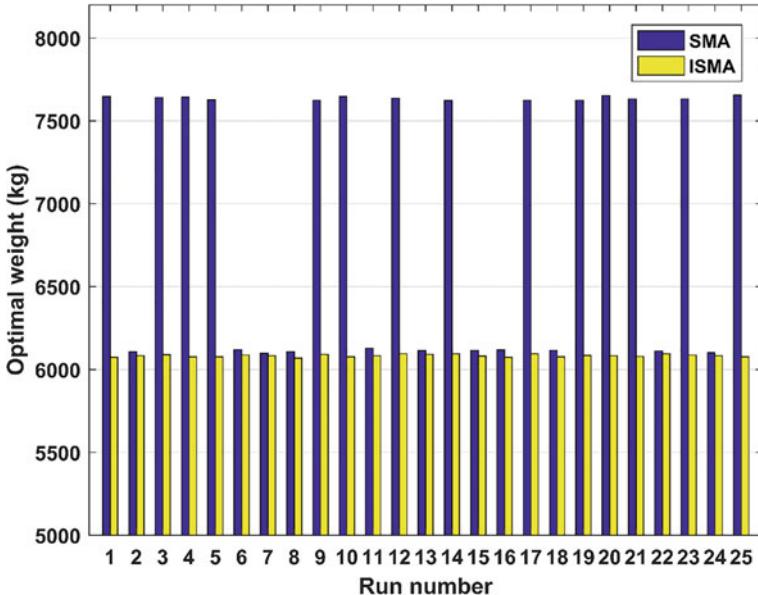


Fig. 9.7 Comparison of the optimized weights of the 600-bar dome-like truss obtained by SMA and ISMA

constraints are imposed on the first and third natural frequencies (see Table 9.1). This problem has been previously addressed in the literature considering different metaheuristic algorithms [7, 8, 10, 12, 23, 24].

Table 9.8 provides a comparison of the results obtained by the SMA and ISMA with those of the other methods. From the results shown in Table 9.8, it is observed that the ISMA is the most effective method with the smallest best weight, average weight, worst weight, and standard deviation. The average weight found by the proposed ISMA over 25 independent runs is even better than the best weights of all the compared methods, confirming the robustness of the proposed ISMA. After 12,420 FE analyses, the ISMA has found a feasible optimum design with a structural weight of 37,414.40 kg, which is better than the best weights obtained by other competitors. The results show that the performance of the proposed ISMA has been improved significantly compared to the classical SMA, confirming the effectiveness of the improvements made in the proposed ISMA. The worst weight of the ISMA is even better than the average weight of the SMA. Table 9.9 provides the Friedman rank test results for the 1180-bar dome-like truss problem. The results indicate that the proposed ISMA ranked first in all metrics. It is also concluded from the results that in both terms of the best and average weights, the MDVC-UVPS [23], SHADE, and SMA ranked second, third, and fourth, respectively. The average weight obtained by the ISMA is 37432.92 kg, which is better than the best weight of the SHADE algorithm (i.e., 37,534.17 kg). The worst weight of the ISMA (i.e., 37,555.37 kg) is also better than the average weight of the SHADE algorithm (i.e., 37,637.42 kg). In

Table 9.6 First five natural frequencies (Hz) of the 600-bar dome-like truss evaluated at optimal designs

Frequency number	DFSO [7]	VPS [9]	ECBO-Cascade [8]	PFJA [12]	CPA [10]	MDVC-UVPS [23]	EFB1 [11]	SHADE	SMA	ISMA
1	5.000	5.0000	5.001	5.0011	5.000	5.000	5.0001	5.0128	5.0001	5.0003
1*	5.0089	5.0022	5.0031	5.0100	5.0087	5.0023	5.0001	5.0128	5.0001	5.0003
2	5.000	5.0003	5.001	5.0011	5.000	5.000	5.0001	5.0128	5.0001	5.0003
3	7.000	7.0000	7.001	7.0000	7.000	7.000	7.0000	7.0002	7.0000	7.0002
3*	7.0213	7.0047	7.0070	7.0210	7.0213	7.0047	7.0000	7.0002	7.0000	7.0002
4	7.000	7.0001	7.001	7.0000	7.000	7.000	7.0000	7.0020	7.0000	7.0002
5	7.000	7.0002	7.002	7.0000	7.000	7.000	7.0024	7.0020	7.0000	7.0003

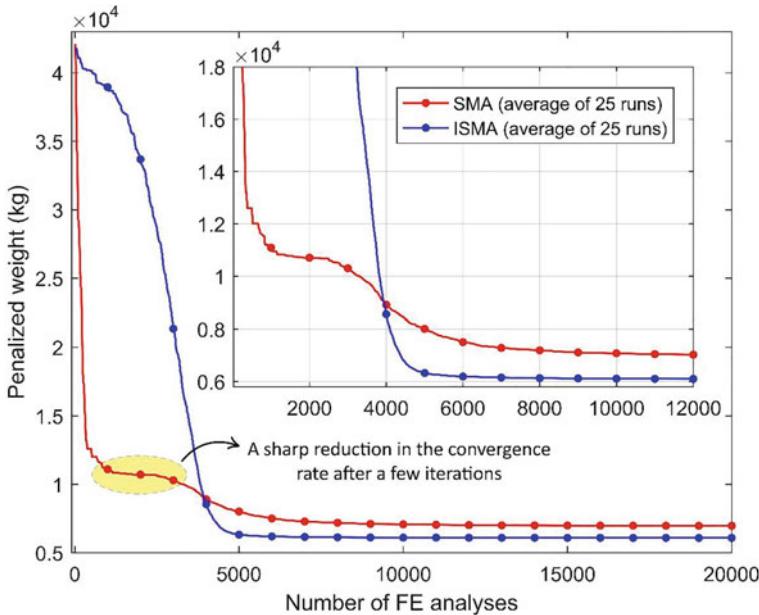


Fig. 9.8 Comparison of the convergence histories of SMA and ISMA for the 600-bar dome-like truss

terms of computational cost, the maximum number of FE analyses of ISMA is 20000, which is considerably less than those of other methods in the literature, except for ECBO-Cascade [8] and SHADE. However, the best and average weights achieved by the proposed ISMA are much better those of the ECBO-Cascade [8] and SHADE. Moreover, the computational cost of the ISMA to obtain its best weight is the smallest one compared to that of others. The optimized weights obtained by the SMA and ISMA over 25 independent runs for the 1180-bar dome-like truss are depicted in Fig. 9.10. The figure shows that, in most of the runs, the classical SMA has converged to solutions relatively far away from the near-optimal solutions achieved by the proposed ISMA. However, the proposed ISMA is found to consistently converge to near-optimal solutions. The convergence histories of the average results obtained by the SMA and ISMA over 25 independent runs are depicted in Fig. 9.11. As the figure demonstrates, similar to the previous example, the convergence rate of the classical SMA is very high in the first few iterations (i.e., before about 50-th iteration), but decreases significantly in the subsequent iterations. As mentioned earlier, this occurs due to the lack of diversity of the population in the exploration phase of the classical SMA (see Fig. 9.3). On the other hand, however, thanks to the fine balance between exploration and exploitation of the ISMA, the convergence characteristics of the proposed ISMA are significantly better than the classical SMA. Indeed, in the first few iterations (i.e., before about 50-th iteration), the ISMA converges slower than the classical SMA, which can be regarded as the result of the exploration of the whole

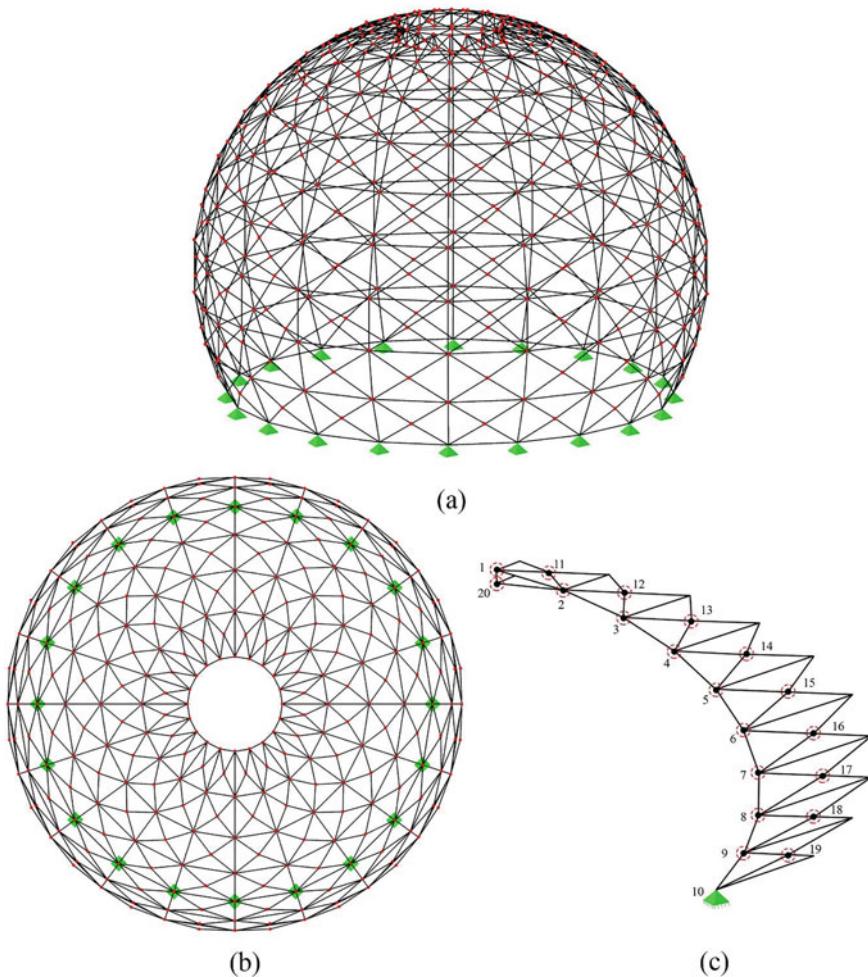


Fig. 9.9 Schematic of the 1180-bar single-layer dome: **a** perspective view; **b** top view; **c** substructure

search space. However, in the subsequent iterations (i.e., especially after about 150-th iteration), because of the elitist strategy of the ISMA, it converges faster than the classical SMA. The first five natural frequencies of the best designs of the SMA, ISMA, and other methods obtained for the 1180-bar dome-like truss are listed in Table 9.10. In addition, the first and third natural frequencies derived from our FEM code for free vibration analysis of trusses are also reported (see rows 1* and 3* in Table 9.10). As can be seen, none of the frequency constraints are violated. Similar to the previous example, the natural frequencies derived from our FEM code are quite close to those reported in the literature and very small differences (by a maximum of 0.574%) are due to rounding of the design variable values reported in the literature.

Table 9.7 Nodal coordinates (m) of the first sub-structure of the 1180-bar dome-like truss

Node number	Coordinates (x, y, z)	Node number	Coordinates (x, y, z)
1	(3.1181, 0.0, 14.6723)	11	(4.5788, 0.7252, 14.2657)
2	(6.1013, 0.0, 13.7031)	12	(7.4077, 1.1733, 12.9904)
3	(8.8166, 0.0, 12.1354)	13	(9.9130, 1.5701, 11.1476)
4	(11.1476, 0.0, 10.0365)	14	(11.9860, 1.8984, 8.8165)
5	(12.9904, 0.0, 7.5000)	15	(13.5344, 2.1436, 6.1013)
6	(14.2657, 0.0, 4.6358)	16	(14.4917, 2.2953, 3.1180)
7	(14.9179, 0.0, 1.5676)	17	(14.8153, 2.3465, 0.0)
8	(14.9179, 0.0, -1.5677)	18	(14.9179, 2.2953, -3.1181)
9	(14.2656, 0.0, -4.6359)	19	(13.5343, 2.1436, -6.1014)
10	(12.9903, 0.0, -7.5001)	20	(3.1181, 0.0, 13.7031)

9.5.3 A 1410-Bar Dome-Like Truss

The last example is a 1410-bar double-layer dome shown in Fig. 9.12. As shown in the figure, the dome is a cyclic symmetric structure composed of 30 identical substructures. The angle between two adjacent substructures is 12° . Each substructure consists of 13 nodes and 47 elements, as shown in Fig. 9.12. Table 9.11 gives the Cartesian coordinates of the nodes of the first substructure. The connectivity information between the nodes of the first substructure is also provided in Table 9.12. The cross-sectional area of each element of the substructure represents a sizing design variable. In addition, similar to the previous example, the layout of the structure remains unchanged during the design process. Thus, this is a size optimization problem with 47 design variables. A non-structural mass of 100 kg is also attached to all free nodes of the dome. The material properties, frequency constraints, and cross-sectional area bounds of the problem are given in Table 9.1.

Table 9.8 Comparison of optimal results obtained for the 1180-bar dome-like truss (cm^2)

Element number (element nodes)	DPSO [7]	ECBO-Cascade [8]	PFIA [12]	CPA [10]	VPS [23] [23]	MDVC-UVPS	Chaotic WSA [24]	This study SHADE	SMA	ISMA
1 (1-2)	7.926	8.0110	7.952	7.877	6.8743	7.3691	6.9078	7.3704	7.1807	7.2170
2 (1-11)	10.426	8.7028	10.466	11.025	10.0230	9.3399	10.7524	10.4463	11.4120	9.6995
3 (1-20)	2.115	3.1616	2.089	3.325	4.4140	2.7203	2.9439	2.0820	3.3000	2.4203
4 (1-21)	14.287	13.6820	14.219	14.672	13.5515	13.2822	13.487	14.3751	13.1888	13.7398
5 (1-40)	3.846	3.2865	3.944	2.894	1.8303	3.6758	3.3147	4.1313	2.8691	3.0521
6 (2-3)	5.921	6.0397	5.979	5.872	7.0824	6.1391	6.9318	6.1818	6.4546	6.1827
7 (2-11)	7.955	8.4370	7.775	7.026	6.3960	7.0964	7.6325	7.8981	7.1377	7.2240
8 (2-12)	6.697	6.4122	6.351	6.746	6.5646	6.0208	6.2343	6.0587	6.6151	6.7305
9 (2-20)	1.889	2.6346	1.896	1.608	2.3705	2.1225	1.3899	1.6386	1.5885	2.3985
10 (2-22)	11.881	11.7440	11.908	11.696	13.2621	12.3488	12.9919	11.4038	10.6898	12.1406
11 (3-4)	7.121	7.9272	7.241	7.523	7.0922	6.8578	6.9162	6.7799	7.6536	6.9400
12 (3-12)	6.080	5.4548	5.647	6.162	6.8079	5.7773	5.1119	5.3027	5.9965	5.5447
13 (3-13)	6.599	6.7221	6.700	6.769	6.3815	6.9931	8.7795	7.7783	6.3623	7.1585
14 (3-23)	7.772	8.1544	7.799	7.671	7.3122	7.3355	6.684	6.8207	7.2240	7.2911
15 (4-5)	9.358	9.7560	9.198	8.732	8.7221	10.5464	9.317	9.5225	9.1650	9.1486
16 (4-13)	6.213	6.5905	6.282	5.841	6.3680	6.9589	6.483	6.6967	6.2908	6.7518
17 (4-14)	8.200	7.0392	7.695	8.545	7.3159	8.0977	8.2833	8.2509	7.2519	8.0406
18 (4-24)	7.799	6.9219	7.520	7.386	11.5749	7.7738	8.0703	8.0023	7.3171	7.5381
19 (5-6)	11.752	11.6919	11.840	12.668	14.7985	12.4614	12.7141	13.5910	11.1855	12.1508
20 (5-14)	7.494	9.8890	7.230	8.733	5.5174	7.8154	7.0934	8.2096	7.7356	8.2227

(continued)

Table 9.8 (continued)

Element number (element nodes)	DPSO [7]	ECBO-Cascade [8]	PFIAs [12]	CPA [10]	VPS [23]	MDVC-UVPS [23]	Chaotic WSA [24]	This study
						SHADE	SMA	ISMA
21 (5–15)	9.696	9.3316	10.211	9.037	15.7381	10.2039	10.069	10.4828
22 (5–25)	9.177	9.1093	9.252	8.903	8.3419	8.9262	9.7217	9.0288
23 (6–7)	17.326	18.1212	17.222	17.013	17.5000	16.5275	17.2315	17.2406
24 (6–15)	11.797	10.6725	11.417	10.048	10.3084	9.0166	9.7761	9.7227
25 (6–16)	14.002	13.5340	14.196	14.057	15.1958	13.8204	13.2779	15.6928
26 (6–26)	11.562	12.0248	11.639	12.154	10.9395	11.4021	11.5212	11.4705
27 (7–8)	23.981	23.1245	24.065	24.024	24.9421	24.2631	21.2086	22.6967
28 (7–16)	12.996	15.2630	13.377	14.143	13.9614	14.5494	13.0618	13.9065
29 (7–17)	16.591	18.3075	16.469	17.894	18.4153	17.7753	17.9725	19.3482
30 (7–27)	15.910	15.2361	16.057	15.276	14.4945	15.4594	14.0147	14.7081
31 (8–9)	34.642	40.0749	34.125	36.006	36.3529	34.1372	33.5273	34.8133
32 (8–17)	19.860	18.4775	18.866	18.409	19.6608	19.1254	20.1075	18.3689
33 (8–18)	25.079	26.0689	24.600	22.644	23.7259	24.1954	23.098	23.2177
34 (8–28)	18.965	21.2213	21.103	22.121	22.0297	21.5899	22.0597	20.5161
35 (9–10)	47.514	46.3724	47.696	48.973	47.3286	49.4717	49.187	52.3873
36 (9–18)	28.133	23.6689	27.760	25.396	22.9442	26.2915	26.835	24.9848
37 (9–19)	33.023	35.0703	33.518	33.599	30.8229	33.7558	31.4569	34.4359
38 (9–29)	32.263	27.9369	31.773	31.724	33.1098	29.7608	30.2512	28.7803
39 (10–19)	33.401	34.2912	33.592	36.419	32.5526	34.0489	34.4764	35.0164

(continued)

Table 9.8 (continued)

Element number (element nodes)	DPSO [7]	ECBO-Cascade [8]	PFIAs [12]	CPA [10]	VPS [23]	[23]	MDVC-UVPS	Chaotic WSA [24]	This study
							SHADE	SMA	ISMA
40 (10–30)	1.344	1.0726	1.000	1.004	1.7363	1.0024	1.1023	1.0900	1.0041
41 (11–21)	9.327	8.5106	9.455	10.624	11.5271	9.0344	9.9842	9.2225	9.6213
42 (11–22)	7.202	6.8664	7.189	6.909	8.4571	7.5316	7.5443	6.3255	8.1146
43 (12–22)	6.792	5.8229	6.767	5.644	5.4136	6.3726	7.6993	6.4328	6.8085
44 (12–23)	6.228	5.3986	6.322	5.301	7.1832	5.7643	6.0238	6.2220	5.6267
45 (13–23)	6.601	8.0669	6.720	7.370	5.4066	6.7270	6.4087	7.5261	7.4399
46 (13–24)	6.584	6.9797	6.425	6.301	6.2534	6.7021	6.4428	5.8260	6.7557
47 (14–24)	8.320	7.2735	8.451	7.853	6.9383	7.8082	8.4255	8.6072	7.8806
48 (14–25)	8.844	9.1827	8.176	7.987	10.6872	8.1225	8.7143	9.0769	8.0373
49 (15–25)	11.254	10.6227	10.069	10.460	12.8005	10.1777	9.8677	11.3055	10.0249
50 (15–26)	12.162	11.5740	12.219	11.446	10.2216	10.1825	11.4715	10.1475	11.2971
51 (16–26)	13.854	15.5194	13.257	13.918	11.5330	13.4590	15.248	12.9694	14.2923
52 (16–27)	13.844	14.1342	13.782	14.694	11.6918	13.9788	12.9199	13.4371	12.8043
53 (17–27)	17.536	17.1612	17.573	18.227	20.7566	18.1070	19.5895	16.9140	16.4090
54 (17–28)	20.551	19.0798	19.909	19.235	18.1341	19.2212	20.1524	18.7768	18.5642
55 (18–28)	24.072	23.4414	24.019	24.421	28.2882	23.4359	24.5119	24.5356	22.7806
56 (18–29)	27.287	26.5726	27.701	22.866	24.2023	27.6479	25.8838	24.3727	27.3480
57 (19–29)	32.965	33.4104	32.918	34.344	48.0180	33.6805	36.4546	31.0984	33.8344
58 (19–30)	36.940	37.1198	37.001	36.964	35.6517	35.7035	37.7461	41.1466	34.6816

(continued)

Table 9.8 (continued)

	DPSO [7]	ECBO-Cascade [8]	PFIAs [12]	CPA [10]	VPS [23]	MDVC-UVPS [23]	Chaotic WSA [24]	This study SHADE	SMA	ISMA
Element number (element nodes)										
59 (20–40)	3.837	4.7593	3.864	5.269	5.5956	4.7617	3.7146	4.6878	3.9625	4.6791
Weight (kg)	37,779.81	37,770.71	37,695.59	37,739.57	38,699.14	37,451.77	37,642.38	37,534.17	37,543.58	37,367.47
Number of FE analyses	N/A	19,180	18,650	N/A	24,780	19,391	N/A	19,800	20,000	16,320
Average weight (kg)	38,294.45	37,885.15	37,755.05	37,813.34	38,861.82	37,545.53	37,795.53	37,637.42	37,664.32	37,432.92
Worst weight (kg)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	37,752.56	37,848.82	37,555.37
Standard deviation (kg)	550.5	133.84	58,025	61.50	385.41	64.85	165.32	59.80	82.65	45.13
Maximum number of FE analyses	50,000	20,000	25,000	80,000	45,000	30,000	30,000	20,000	20,000	20,000
Constraint violation (%)	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Number of runs	10	5	20	10	30	30	20	25 out of 62	25 out of 34	25 out of 31

Table 9.9 The Friedman rank test results for the 1180-bar dome-like truss

	DPSO [7]	ECBO-Cascade [8]	PFJA [12]	CPA [10]	VPS [23]	MDVC-UVPS [23]	Chaotic WSA [24]	This study
Friedman rank of minimum weight	10	9	6	8	7	2	5	SHADE
Friedman rank of average weight	9	8	5	7	10	2	6	SMA
Friedman rank of standard deviation	10	7	2	4	9	5	8	ISMA

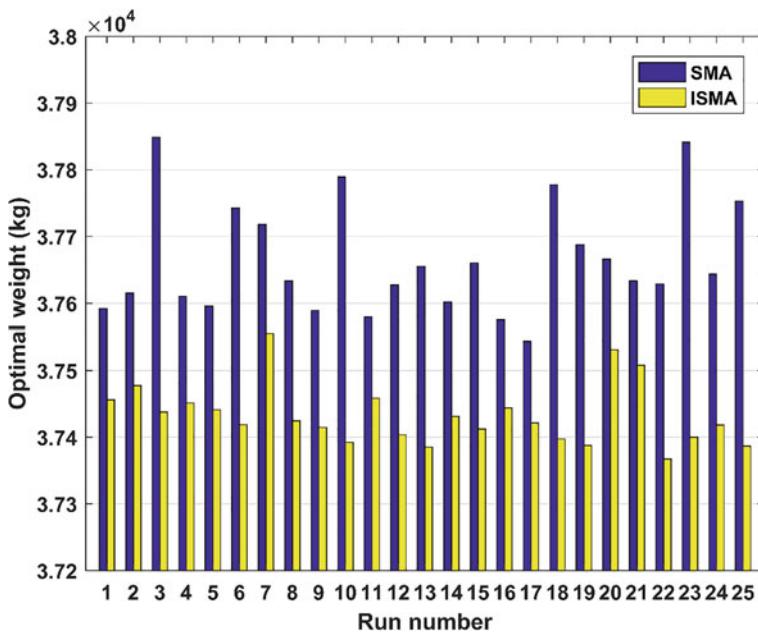


Fig. 9.10 Comparison of the optimized weights of the 1180-bar dome-like truss obtained by SMA and ISMA

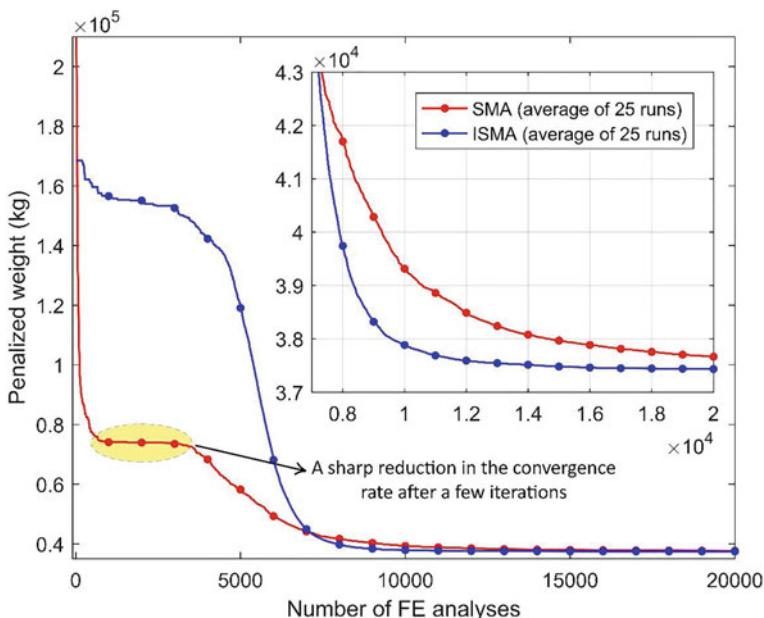


Fig. 9.11 Comparison of the convergence histories of SMA and ISMA for the 1180-bar dome-like truss

Table 9.10 First five natural frequencies (Hz) of the 1180-bar dome-like truss evaluated at optimal designs

Frequency number	DPSO [7]	ECBO-Cascade [8]	PFA [12]	CPA [10]	VPS [23]	MDVC-UVPS [23]	Chaotic WSA [24]	This study	SHADE	SMA	ISMA
1	7.000	7.000	7.0000	7.000	7.000	7.000	7.0001	7.0001	7.0002	7.0000	7.0000
1*	7.0088	7.0023	7.0084	7.0085	7.0019	7.0019	7.0001	N/A	7.0002	7.0000	7.0000
2	7.000	7.001	7.0000	7.000	7.001	7.001	7.0001	N/A	7.0002	7.0000	7.0000
3	9.000	9.002	9.0024	9.0000	9.000	9.000	9.0049	9.0049	9.0022	9.0009	9.0000
3*	9.0472	9.0245	9.0541	9.0424	9.0289	9.0289	9.0124	9.0049	9.0022	9.0009	9.0000
4	9.000	9.002	9.0024	9.0000	9.000	9.000	N/A	N/A	9.0022	9.0009	9.0000
5	9.005	9.062	9.0129	9.000	9.177	9.005	N/A	N/A	9.0158	9.0870	9.0033

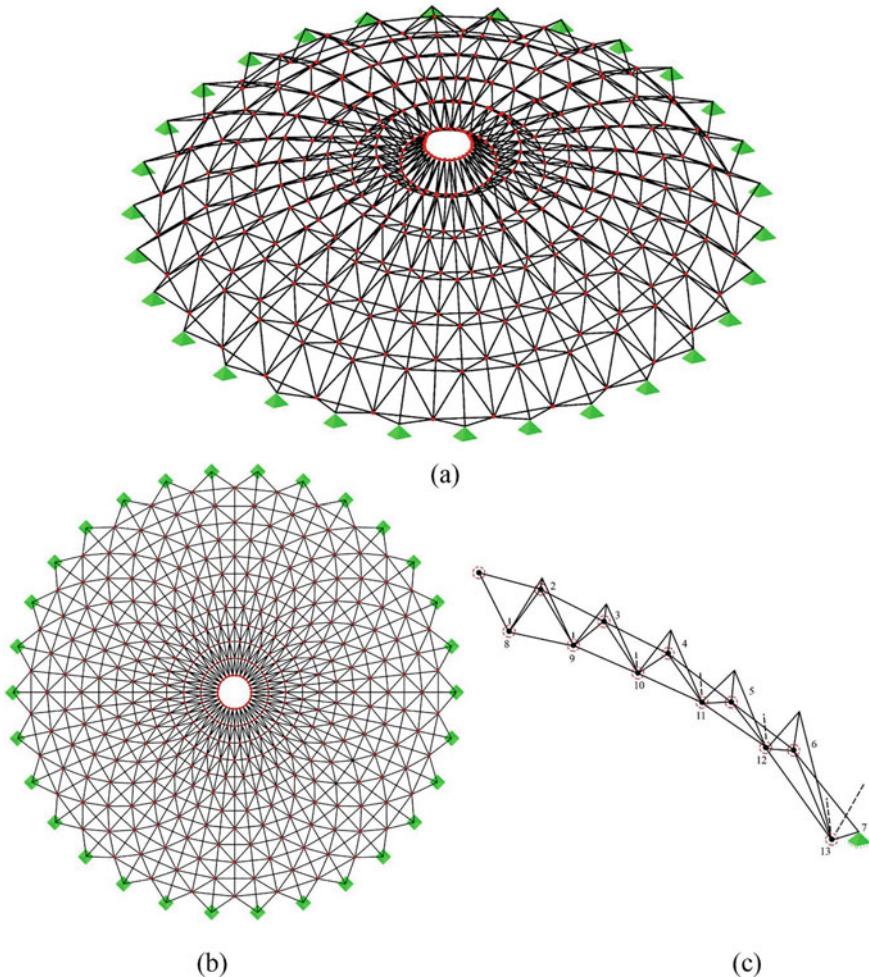


Fig. 9.12 Schematic of the 1410-bar single-layer dome: **a** perspective view; **b** top view; **c** substructure

Table 9.12 provides a comparison between the optimization results obtained by the present algorithms with those of the other methods reported in the literature. As can be seen from the results given in Table 9.12, the best weight obtained by the ISMA is 10309.41 kg, which is better than those obtained by the classical SMA and other compared methods. Moreover, after 15,900 FE analyses, the proposed ISMA has found a feasible optimum design with a structural weight of 10,323.75 kg, which is better than the best weights found by other methods. In terms of the average weight, however, the ISMA is ranked third after PFJA [12] and SHADE. On the other hand, the maximum number of FE analyses of PFJA is 5000 more than that of the ISMA. Compared with the classical SMA, the proposed ISMA showed significant

Table 9.11 Nodal coordinates (m) of the first sub-structure of the 1410-bar dome-like truss

Node number	Coordinates (x, y, z)
1	(1.0, 0.0, 4.0)
2	(3.0, 0.0, 3.75)
3	(5.0, 0.0, 3.25)
4	(7.0, 0.0, 2.75)
5	(9.0, 0.0, 2.0)
6	(11.0, 0.0, 1.25)
7	(13.0, 0.0, 0.0)
8	(1.989, 0.209, 3.0)
9	(3.978, 0.418, 2.75)
10	(5.967, 0.627, 2.25)
11	(7.956, 0.836, 1.75)
12	(9.945, 1.0453, 1.0)
13	(11.934, 1.2543, -0.5)

improvement in all four measured terms. The Friedman rank test results for the 1410-bar dome-like truss problem are presented in Table 9.13. As aforementioned, the results show that the proposed ISMA ranked first in terms of the best weight. However, PFJA [12] and SHADE showed the best performance in terms of average weight and standard deviation, respectively. It should be noted that the SHADE algorithm ranked first and second in terms of standard deviation and average weight, respectively, confirming its effectiveness to be used in the optimal design problems of truss structures with frequency constraints. Figure 9.13 illustrates the best weights obtained by the SMA and ISMA over 25 independent runs. It is observed that the classical SMA has converged to a non-optimal solution at the 21-th run. The first five natural frequencies of the best designs of the 1410-bar dome-like truss obtained by the SMA, ISMA, and other compared methods are provided in Table 9.14. The first and third natural frequencies calculated by our FEM code for free vibration analysis of truss structures are also provided (see rows 1* and 3* in Table 9.14). It is observed that the frequency constraints are satisfied by all methods. Therefore, as it is shown in Table 9.12, the constraint violations of all methods are zero. It is obvious from the table that the natural frequencies calculated by our FEM code are quite close to those reported in the literature and very small differences (by a maximum of 0.167%) are due to the rounding of the design variable values reported in the literature. In terms of computational cost, the maximum number FE analyses of the proposed ISMA is 20000, which is less than or equal to those of other compared methods. The number of FE analyses required to obtain the best weight for the ECBO-Cascade [8], PFJA [12], SHADE, and SMA are 19,460, 16,900, 19,900, and 19,980, respectively, which are less than that of the ISMA (i.e., 20,000). However, as pointed above, the ISMA requires only 15,900 FE analyses to find a feasible design with a structural weight of 10,323.75 kg, that is better than the best weights of ECBO-Cascade [8], PFJA [12], SHADE, and SMA. Figure 9.14 depicts the convergence histories of the average

Table 9.12 Comparison of optimal results obtained for the 1410-bar dome-like truss (cm^2)

Element number (element nodes)	CPA [10]	CRPSO [26]	ECBO-Cascade [8]	DPSO [7]	PFIA [12]	JA [25]	This study
	SHADE	SMA	SMA	SHADE	SMA	ISMA	
1 (1–2)	7.416	2.5	7.9969	7.209	6.1902	5.1377	5.8998
2 (1–8)	4.768	6.0	6.1723	5.006	4.4036	3.7009	5.2816
3 (1–14)	38.993	18.0	35.5011	38.446	31.2253	33.9653	31.3485
4 (2–3)	8.966	9.5	10.2510	9.458	8.4715	9.5992	9.1717
5 (2–8)	4.511	6.0	5.3727	4.313	4.8590	5.5482	4.3893
6 (2–9)	1.544	1.0	1.3488	1.494	1.5759	3.6969	2.2026
7 (2–15)	8.371	29.5	11.4427	8.455	12.9451	21.8390	14.1079
8 (3–4)	9.276	8.0	9.7157	9.488	9.3263	8.6255	8.8618
9 (3–9)	3.583	2.0	1.3005	3.480	3.2716	3.5731	3.0341
10 (3–10)	3.476	1.5	2.5046	3.495	3.2878	3.3162	3.2140
11 (3–16)	15.531	1.0	10.7849	16.037	12.6719	9.0269	10.6711
12 (4–5)	10.285	7.5	10.1954	9.796	10.0979	9.5465	8.6714
13 (4–10)	2.497	1.0	2.2300	2.413	2.5803	2.4441	2.6867
14 (4–11)	5.397	6.0	5.1186	5.681	5.3769	4.0831	4.6130
15 (4–17)	16.503	14.5	14.0053	15.806	16.0581	12.3422	16.6861
16 (5–6)	8.193	9.0	8.9713	8.078	8.6789	8.6568	8.8547
17 (5–11)	3.829	1.0	4.0756	3.931	3.3199	4.0585	3.4865
18 (5–12)	6.151	8.0	5.9211	6.099	6.4966	5.2401	5.4402
19 (5–18)	10.465	19.5	10.6915	10.771	10.8804	13.4226	10.4322
20 (6–7)	13.925	16.5	10.6220	13.775	14.0056	13.3777	13.3513

(continued)

Table 9.12 (continued)

Element number (element nodes)	CPA [10]	CRPSO [26]	ECBO-Cascade [8]	DPSO [7]	PFIA [12]	JA [25]	This study
	SHADE	SMA	ISMA				
21 (6–12)	4.415	5.0	4.5064	4.231	5.0843	6.0785	5.7078
22 (6–13)	6.863	9.0	8.4086	6.995	6.9952	8.0438	7.3410
23 (6–19)	1.769	1.0	5.8405	1.837	1.0270	1.0226	2.1391
24 (7–13)	4.339	5.0	5.0342	4.397	4.3788	4.2592	4.1984
25 (8–9)	2.115	6.5	3.8932	2.115	2.1951	2.1809	2.7969
26 (8–14)	4.951	5.5	6.1647	4.923	4.2562	3.9382	5.9349
27 (8–15)	4.147	7.0	6.8990	4.047	4.6605	5.2629	5.7570
28 (8–21)	6.044	15.5	11.6387	5.906	8.8694	12.1908	9.7276
29 (9–10)	3.222	4.5	3.8343	3.392	3.2333	4.1286	3.7985
30 (9–15)	1.970	2.5	1.4772	1.902	1.7611	3.6731	1.3983
31 (9–16)	4.290	2.5	1.3075	4.381	3.2831	2.6622	2.7460
32 (9–22)	8.020	1.0	4.4876	8.442	7.1936	4.8959	4.9768
33 (10–11)	4.857	6.0	6.0196	5.011	4.9840	5.4788	6.3077
34 (10–16)	3.689	1.0	2.6729	3.577	3.6672	3.4673	2.8174
35 (10–17)	2.831	1.0	1.6342	2.805	2.4062	2.4088	3.7421
36 (10–23)	1.985	1.0	1.8410	2.024	2.1576	1.1880	3.1237
37 (11–12)	6.373	10.0	6.8841	6.709	7.1043	7.1520	7.4503
38 (11–17)	4.865	5.5	4.1393	5.054	5.2070	4.3348	4.9775
39 (11–18)	3.412	3.5	3.3264	3.259	3.6853	2.5376	2.7562
							3.3034
							3.1729

(continued)

Table 9.12 (continued)

Element number (element nodes)	CPA [10]	CRPSO [26]	ECBO-Cascade [8]	DPSO [7]	PFIA [12]	JA [25]	This study		
							SHADE	SMA	ISMA
40 (11-24)	1.027	1.0	1.0000	1.063	1.0007	1.0364	1.5077	1.001	1.0001
41 (12-13)	6.218	7.5	6.9373	5.934	6.6302	6.2738	7.0288	6.3002	6.1117
42 (12-18)	7.342	8.5	4.4568	7.057	6.6773	5.7109	6.0357	6.5281	6.2896
43 (12-19)	5.458	7.5	4.6758	5.745	5.2167	6.1412	5.5673	4.8998	5.3444
44 (12-25)	1.140	1.0	1.0084	1.185	1.0016	1.0061	1.4738	1.002	1.0001
45 (13-19)	7.401	7.5	7.5103	7.274	8.1289	8.9413	6.9808	8.0640	7.8981
46 (13-20)	4.578	6.5	5.2449	4.798	4.5151	4.7986	4.2476	5.1254	5.1166
47 (13-26)	1.561	1.0	1.0550	1.515	1.0010	1.0556	1.2350	1.000	1.0052
Weight (kg)	10,435.47	11,044.616778	10,504.20	10,453.84	10,326.2936	10,400.079	10,412.92	10,417.87	10,309.41
Number of FE analyses	N/A	N/A	19,460	N/A	16,900	N/A	19,900	19,980	20,000
Average weight (kg)	10,658.48	13,017.899263	10,590.67	11,100.57	10,399.828	10,707.902	10,485.33	10,607.01	10,556.67
Worst weight (kg)	N/A	15,496.852679	N/A	N/A	N/A	11,202.677	10,562.73	11,066.58	10,825.00
Standard deviation (kg)	129.90	1454.812976	52.51	334.20	75.441	250.624	38.29	132.03	130.92
Maximum number of FE analyses	80,000	20,000	20,000	50,000	25,000	20,000	20,000	20,000	20,000
Constraint violation (%)	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Number of runs	10	20	5	10	20	10	25 out of 36	25 out of 33	25 out of 29

Table 9.13 The Friedman rank test results for the 1410-bar dome-like truss

	CPA [10]	CRPSO [26]	ECBO-Cascade [8]	DPSO [7]	PFJA [12]	JA [25]	This study		
							SHADE	SMA	ISMA
Friedman rank of minimum weight	6	9	8	7	2	3	4	5	1
Friedman rank of average weight	6	9	4	8	1	7	2	5	3
Friedman rank of standard deviation	4	9	2	8	3	7	1	6	5

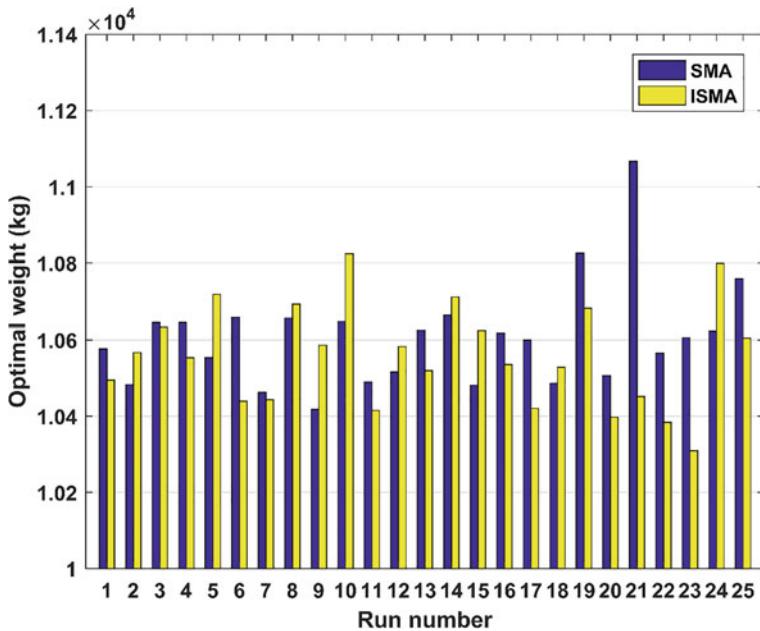


Fig. 9.13 Comparison of the optimized weights of the 1410-bar dome-like truss obtained by SMA and ISMA

results obtained by the SMA and ISMA over 25 independent runs. As can be seen in the magnified part of the figure, the proposed ISMA converges faster than the classical SMA.

Table 9.14 First five natural frequencies (Hz) of the 1410-bar dome-like truss evaluated at optimal designs

Frequency number	CPA [10]	CRPSO [26]	ECBO-Cascade [8]	DPSO [7]	PFIA [12]	JA [25]	This study
					SHADE	SMA	ISMA
1	7.000	7.0008	7.0020	7.001	7.0009	7.0017	7.0000
1*	7.0113	7.0008	7.0048	7.0127	7.0125	7.0017	7.0000
2	7.000	N/A	7.0030	7.001	7.0009	7.0017	7.0000
3	9.000	9.0068	9.0010	9.003	9.0001	9.0027	9.0000
3*	9.0082	9.0068	9.0047	9.0105	9.0083	9.0027	9.0000
4	9.002	N/A	9.0010	9.005	9.0002	9.0035	9.0002
5	9.002	N/A	9.0030	9.005	9.0002	9.0035	9.0003

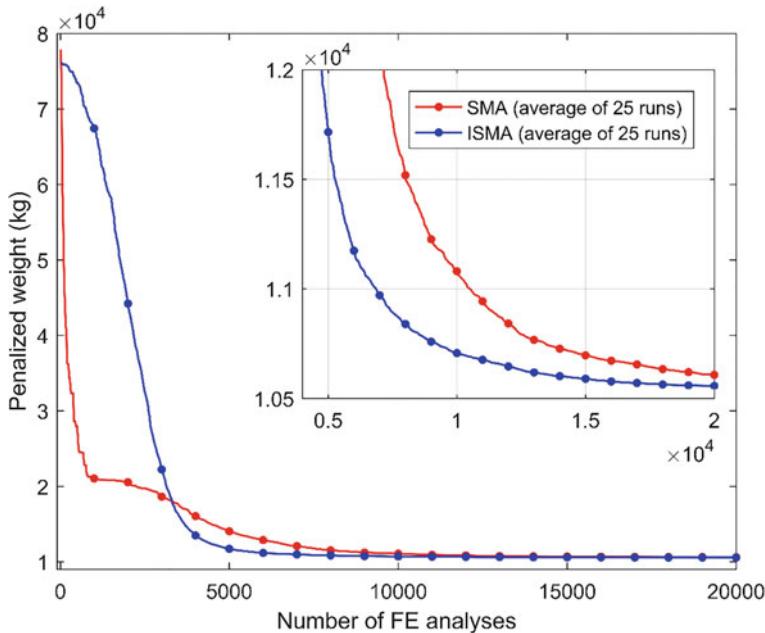


Fig. 9.14 Comparison of the convergence histories of SMA and ISMA for the 1180-bar dome-like truss

9.6 Concluding Remarks

The slime mould algorithm (SMA) is a recently proposed swarm-based metaheuristic inspired by the morphological changes of the acellular slime mould *Physarum polycephalum* while foraging and has been applied to some real-world optimization problems. However, similar to many other swarm-based metaheuristic algorithms, it often suffers from the problems of premature convergence to non-optimal solutions and slow convergence rate, especially when dealing with large-scale optimization problems. In this study, to eliminate the drawbacks of the classical SMA, the improved slime mould algorithm (ISMA) is proposed. The key contributions of the proposed ISMA are as follows: (1) an elitist strategy is adopted in the replacement phase of the classical SMA to improve the convergence rate; and (2) a slight modification is made to the exploration phase of the classical SMA to avoid premature convergence to non-optimal solutions. Both the classical SMA and the proposed ISMA are then applied to solve three large-scale dome-like truss optimization problems with natural frequency constraints. To our knowledge, this is the first time that the SMA has been applied to structural optimization. It is shown that the proposed ISMA not only escape from non-optimal solutions but also overcomes the shortcoming of the slow convergence rate of the classical SMA. The obtained results indicate that the convergence characteristics of the proposed ISMA are significantly improved

in comparison with the classical SMA. Furthermore, in all cases, the best weight, average weight, worst weight, and standard deviation obtained by ISMA are meaningfully better than those acquired by SMA. Moreover, in most cases, the obtained results by using ISMA are superior or comparable with those obtained by some other methods in the literature, indicating the high capability of ISMA for solving truss optimization problems with natural frequency constraints. Thus, the proposed method is promising to extend its application to other types of structures, such as frames, bridges, retaining walls, etc.

References

1. Kaveh A, Biabani Hamedani K, Kamalinejad M (2022) Improved slime mould algorithm with elitist strategy and its application to structural optimization with natural frequency constraints. *Comput Struct* 264:106760. <https://doi.org/10.1016/j.compstruc.2022.106760>
2. Grandhi R (1993) Structural optimization with frequency constraints-a review. *AIAA J* 31(12):2296–2303. <https://doi.org/10.2514/3.11928>
3. Bellagamba L, Yang TY (1981) Minimum-mass truss structures with constraints on fundamental natural frequency. *AIAA J* 19(11):1452–1458. <https://doi.org/10.2514/3.7875>
4. Gomes HM (2011) Truss optimization with dynamic constraints using a particle swarm algorithm. *Expert Syst Appl* 38(1):957–968. <https://doi.org/10.1016/j.eswa.2010.07.086>
5. Khatibinia M, Naseralavi SS (2014) Truss optimization on shape and sizing with frequency constraints based on orthogonal multi-gravitational search algorithm. *J Sound Vib* 333(24):6349–6369. <https://doi.org/10.1016/j.jsv.2014.07.027>
6. Sergeyev O, Mroz Z (2000) Sensitivity analysis and optimal design of 3D frame structures for stress and frequency constraints. *Comput Struct* 75(2):167–185. [https://doi.org/10.1016/S0045-7949\(99\)00088-7](https://doi.org/10.1016/S0045-7949(99)00088-7)
7. Kaveh A (2017) Optimal analysis and design of large-scale domes with frequency constraints. In *Applications of Metaheuristic Optimization Algorithms in Civil Engineering* (pp. 257–279). Springer, Cham. https://doi.org/10.1007/978-3-319-48012-1_14
8. Kaveh A, Ilchi Ghazaan M (2016) Optimal design of dome truss structures with dynamic frequency constraints. *Struct Multidisc Optim* 53(3):605–621. <https://doi.org/10.1007/s00158-015-1357-2>
9. Kaveh A, Ilchi Ghazaan M (2017) Vibrating particles system algorithm for truss optimization with multiple natural frequency constraints. *Acta Mech* 228(1):307–322. <https://doi.org/10.1007/s00707-016-1725-z>
10. Kaveh A, Zolghadr A (2018) Optimal design of cyclically symmetric trusses with frequency constraints using cyclical parthenogenesis algorithm. *Adv Struct Eng* 21(5):739–755. <https://doi.org/10.1177%2F1369433217732492>
11. Kaveh A, Biabani Hamedani K, Kamalinejad M (2021) An enhanced Forensic-Based Investigation algorithm and its application to optimal design of frequency-constrained dome structures. *Comput Struct* 256:106643. <https://doi.org/10.1016/j.compstruc.2021.106643>
12. Degertekin SO, Bayar GY, Lamberti L (2021) Parameter free Jaya algorithm for truss sizing-layout optimization under natural frequency constraints. *Comput Struct* 245:106461. <https://doi.org/10.1016/j.compstruc.2020.106461>
13. Li S, Chen H, Wang M, Heidari AA, Mirjalili S (2020) Slime mould algorithm: A new method for stochastic optimization. *Future Gener Comput Syst* 111:300–323. <https://doi.org/10.1016/j.future.2020.03.055>
14. Abdel-Basset M, Chang V, Mohamed R (2020) HSMA_WOA: A hybrid novel Slime mould algorithm with whale optimization algorithm for tackling the image segmentation problem of chest X-ray images. *Appl Soft Comput* 95:106642. <https://doi.org/10.1016/j.asoc.2020.106642>

15. Zubaidi SL, Abdulkareem IH, Hashim KS, Al-Bugharbee H, Ridha HM, Gharghan SK, Al-Qaim FF, Muradov M, Kot P, Al-Khaddar R (2020) Hybridised Artificial Neural Network Model with Slime Mould Algorithm: A Novel Methodology for Prediction of Urban Stochastic Water Demand. *Water* 12(10):2692. <https://doi.org/10.3390/w12102692>
16. Kumar C, Raj TD, Premkumar M, Raj TD (2020) A new stochastic slime mould optimization algorithm for the estimation of solar photovoltaic cell parameters. *Optik* 223:165277. <https://doi.org/10.1016/j.ijleo.2020.165277>
17. Nguyen TT, Wang HJ, Dao TK, Pan JS, Liu JH, Weng S (2020) An Improved Slime Mold Algorithm and its Application for Optimal Operation of Cascade Hydropower Stations. *IEEE Access* 8:226754–226772. <https://doi.org/10.1109/ACCESS.2020.3045975>
18. De Jong KA (1975) Analysis of the behavior of a class of genetic adaptive systems.
19. Kaveh A, Zolghadr A (2014) Democratic PSO for truss layout and size optimization with frequency constraints. *Comput Struct* 130:10–21. <https://doi.org/10.1016/j.compstruc.2013.09.002>
20. Tanabe R, Fukunaga A (2013) Success-history based parameter adaptation for differential evolution. In 2013 IEEE congress on evolutionary computation Jun 20 (pp. 71–78). IEEE. <https://doi.org/10.1109/CEC.2013.6557555>
21. Zhang J, Sanderson AC (2009) JADE: adaptive differential evolution with optional external archive. *IEEE Transactions on evolutionary computation*. Aug 18;13(5):945–958. <https://doi.org/10.1109/TEVC.2009.2014613>
22. Naik MK, Panda R, Abraham A (2020) Normalized square difference based multilevel thresholding technique for multispectral images using leader slime mould algorithm. *J King Saud Univ - Comput Inf Sci*. <https://doi.org/10.1016/j.jksuci.2020.10.030>
23. Kaveh A, Ilchi Ghazaan M (2018) A new hybrid meta-heuristic algorithm for optimal design of large-scale dome structures. *Eng Optim* 50(2):235–252. <https://doi.org/10.1080/0305215X.2017.1313250>
24. Kaveh A, Amirsoleimani P, Dadras Eslamlou A, Rahmani P (2021) Frequency-constrained optimization of large-scale dome-shaped trusses using chaotic water strider algorithm. *Structures* 32:1604–1618. <https://doi.org/10.1016/j.istruc.2021.03.033>
25. Kaveh A, Biabani Hamedani K, Joudaki A, Kamalinejad M (2021) Optimal analysis for optimal design of cyclic symmetric structures subject to frequency constraints. *Structures* 33:3122–3136. <https://doi.org/10.1016/j.istruc.2021.06.054>
26. Carvalho JP, Lemonge AC, Carvalho EC, Hallak PH, Bernardino HS (2018) Truss optimization with multiple frequency constraints and automatic member grouping. *Struct Multidisc Optim* 57(2):547–577. <https://doi.org/10.1007/s00158-017-1761-x>

Chapter 10

Improved Arithmetic Optimization Algorithm



10.1 Introduction

In this chapter, an improved version of the arithmetic optimization algorithm (AOA) called improved arithmetic optimization algorithm (IAOA) is developed to overcome the drawbacks of the standard AOA [1]. The performance of the proposed IAOA is demonstrated through four well-known structural optimization problems with discrete design variables. The results obtained by IAOA are compared to those of the standard AOA and other optimization methods in the literature. The results show that the proposed IAOA meaningfully surpasses the standard AOA in both terms of robustness and convergence speed.

Over the past decades, structural optimization has attracted intensive efforts both in theoretical and practical aspects, due to its potential to reduce the structural weight and cost while satisfying the design requirements. Structural optimization not only provides an effective way to reduce the structural weight and cost, but also allows sensitivity analysis to improve the structural integrity. Accordingly, a great number of optimization methods have been proposed and successfully utilized in a large variety of structural optimization problems. Examples of classical optimization techniques applied to structural optimization include the integer linear programming (ILP) [2] and sequential quadratic programming (SQP) [3]. It is worth mentioning that most of the above-mentioned studies include only continuous design variables. However, in real-world structural optimization problems, the structural elements must usually be selected from a list of commercially available cross-sections. Moreover, real-world structural optimization problems often involve multiple highly nonlinear constraints. Because of these reasons, classical optimization approaches were found not suitable or easy to deal with real-world structural optimization problems. However, some classical optimization methods integrated with the round-off techniques have been proposed for solving discrete structural optimization problems. Nevertheless, the rounded-off techniques may result in solutions that are far from the optimal point,

or even infeasible solutions, especially when dealing with a large number of design variables.

In order to overcome the shortcomings of classical optimization methods, a large number of stochastic optimization techniques, known as metaheuristic optimization algorithms, have been proposed in recent decades. In contrast to most of the classical optimization methods, metaheuristic algorithms do not need the derivatives of the objective and constraint functions, which make them easy to be implemented. Most of the metaheuristic algorithms are inspired by nature and are hence called nature-inspired metaheuristic algorithms. Based on the source of inspiration, nature-inspired metaheuristic algorithms are classified in four major categories [4]: evolutionary-based, swarm-based, human-based, and physics-based. Evolutionary-based metaheuristics simulate the mechanisms of biological evolution such as mutation, recombination, generation, selection, etc. Swarm-based optimization algorithms draw their inspiration from the social behaviors of animals and other biological organisms. Physics-based optimization algorithms mimic the physical phenomena in nature. Human-based optimization algorithms draw inspiration from behavior and activities of human systems. In Fig. 10.1, some examples of the four major categories of nature-inspired metaheuristic algorithms are presented. In recent years, population-based metaheuristic algorithms have been widely applied to solve a large variety of discrete structural optimization problems, such as artificial bee colony (ABC) [5], firefly algorithm (FA) [6], teaching–learning-based optimization (TLBO) [7], etc. However, it should be noted that metaheuristic techniques may require high computational effort to provide good performance in terms of effectiveness, especially when the number of design variables is high. Accordingly, some design-driven heuristic approaches such as guided stochastic search (GSS) [8] have been proposed in recent years to tackle with challenging structural optimization problems having a large number of design variables.

Although the aforementioned works have overcome many of the restrictions of classical optimization techniques in solving discrete structural optimization problems, structural designers are increasingly faced with the need to adopt more robust and efficient methods for optimizing structural systems with discrete design variables. The arithmetic optimization algorithm is a newly proposed metaheuristic algorithm inspired by the four basic arithmetic operations of subtraction, addition, multiplication, and division [9]. Up to now, the AOA has been employed for solving some real-world optimization problems, including model identification [10], image segmentation [11], etc. However, it has been found that the standard AOA suffers from poor exploration and is prone to trap into local optima [10, 11]. Thus, aiming to overcome the drawbacks mentioned above, this study develops an improved variant of the arithmetic optimization algorithm, called improved AOA (IAOA), to solve discrete structural optimization problems. To our knowledge, the AOA is still not applied to structural design optimization. The main contributions of the proposed IAOA, as compared with the standard AOA, are as follows: (1) Both the exploration and exploitation capabilities of the standard AOA are enhanced to overcome its drawbacks. (2) The proposed IAOA requires less algorithm-specific parameters compared to the standard AOA, which makes it easy to be applied. The performance

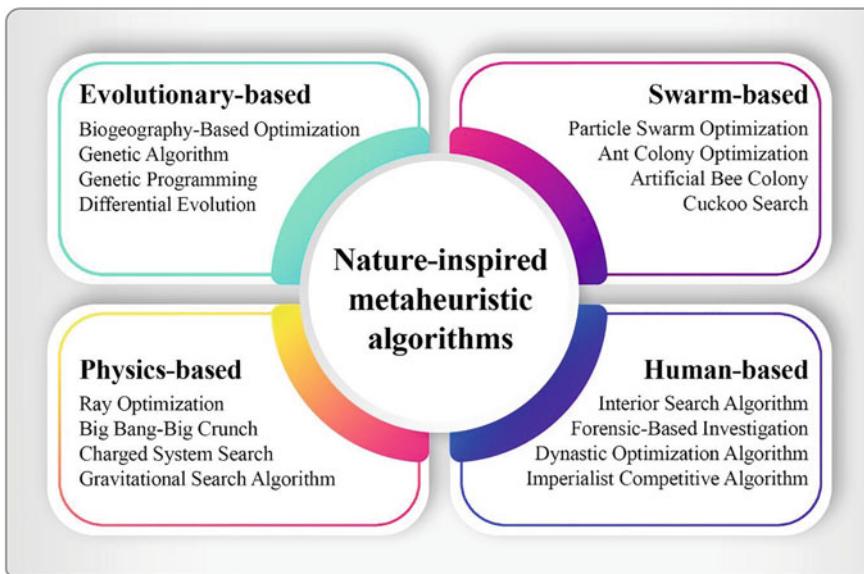


Fig. 10.1 Major categories of nature-inspired metaheuristic algorithms

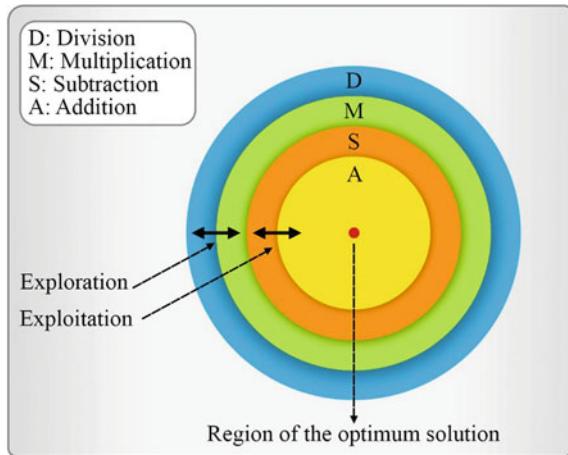
of the proposed IAOA is demonstrated through three benchmark examples of discrete structural optimization problems. The results achieved by the AOA and IAOA are compared with each other and with those of some other state-of-the-art optimization methods existing in the literature.

The rest of this chapter is structured as follows: Sect. 10.2 provides a discussion of the standard AOA. In Sect. 10.3, the improved arithmetic optimization algorithm (IAOA) is presented. In Sect. 10.4, the mathematical model of discrete structural optimization is presented. In Sect. 10.5, three benchmark discrete structural optimization problems are investigated and optimization results are discussed. In Sect. 10.6, both AOA and IAOA are applied to a very high-dimensional structural optimization problem. Finally, Sect. 10.7 draws the concluding remarks.

10.2 Overview of the Arithmetic Optimization Algorithm (AOA)

The arithmetic optimization algorithm (AOA) is a population-based metaheuristic algorithm developed by Abualigah et al. in 2021 [9]. Arithmetic is a branch of mathematics dealing with numbers and their addition, subtraction, multiplication, and division. Accordingly, AOA simulates the distribution characteristics of the basic arithmetic operations of subtraction (S), addition (A), division (D), and multiplication (M). Like other metaheuristic algorithms, the search process of AOA consists

Fig. 10.2 The order of arithmetic operations and their supremacy (from the outside to the inside)



of two major phases: exploration and exploitation. In the exploration phase, the position of the search agents (i.e., candidate solutions) are updated based on multiplication and division operators, whereas the exploitation phase deals with addition and subtraction operators. Figure 10.2 illustrates the order of arithmetic operations and their supremacy from the outside to the inside. The mathematical formulation of AOA is briefly presented as follows.

10.2.1 Initialization Phase

Like other population-based metaheuristics, AOA starts the search process by a population of randomly distributed solutions (X), as shown in Eq. (10.1). Note that, whenever the best solution of the current iteration is more fitted than the best-so-far solution, then the best-so-far solution is updated.

$$X = \begin{bmatrix} x_{1,1} & \dots & x_{1,j} & \dots & x_{1,n} \\ \dots & \dots & \dots & \dots & \dots \\ x_{i,1} & \dots & x_{i,j} & \dots & x_{i,n} \\ \dots & \dots & \dots & \dots & \dots \\ x_{N,1} & \dots & x_{N,j} & \dots & x_{N,n} \end{bmatrix} \quad (10.1)$$

where N denotes the number of candidate solutions of the population; n stands for the number of design variables of the problem; and $x_{i,j}$ represents the j -th design variable of the i -th candidate solution of the initial population.

The AOA employs a dynamic function, named Math Optimizer Accelerated (*MOA*), for switching between the exploitation and exploration phases during the

search process. The MOA function is defined by the following equation:

$$MOA(C_{Iter}) = Min + C_{Iter} \times \left(\frac{Max - Min}{M_{Iter}} \right) \quad (10.2)$$

where M_{Iter} denotes the maximum number of iterations; C_{Iter} stands for the current iteration number and varies between 1 and M_{Iter} ; $MOA(C_{Iter})$ denotes the value of the function MOA at the current iteration; and Min and Max are the minimum and maximum values of MOA function, respectively. In this work, Min and Max are set to 0.2 and 0.9, respectively, as suggested by Abualigah et al. [9].

When $MOA(C_{Iter}) < r_1$, the search space is explored using the multiplication (M) and division (D) operators, but when $MOA(C_{Iter}) \geq r_1$, the promising areas of the search space located in the exploration phase are exploited using the addition (A) and subtraction (S) operators. Note that r_1 is a uniformly distributed pseudorandom number varies from 0 to 1. As can be observed from Eq. (10.2), MOA starts from $Min + (Max - Min)/M_{Iter}$ at the first iteration (i.e., $C_{Iter} = 1$) and increases linearly until it reaches Max at the last iteration (i.e., $C_{Iter} = M_{Iter}$). As a result, $MOA(C_{Iter})$ can smoothly switch between the exploration and exploitation phases.

10.2.2 Exploration Phase

As aforementioned, arithmetic operations of multiplication (M) and division (D) are employed in the exploration phase of AOA to direct the exploration of the search space. Because of the highly scattering properties of D and M operators, diverse areas of the search space could be explored. These operators are not able to thoroughly exploit the promising areas of the search space. However, they are required to ensure that all the search space is explored enough to provide a reliable estimate of the global optimum. The exploration phase of the AOA is executed based on the following position update rule:

$$x'_{i,j} = \begin{cases} x_{Best,j} \div (MOP + \varepsilon) \times ((UB_j - LB_j) \times \mu + LB_j), & r_2 > 0.5 \\ x_{Best,j} \times MOP \times ((UB_j - LB_j) \times \mu + LB_j), & \text{otherwise} \end{cases} \quad (10.3)$$

where $x'_{i,j}$ is the updated value of the j -th design variable of the i -th candidate solution; $x_{Best,j}$ denotes the j -th design variables the best-so-far solution; UB_j and LB_j are the upper and lower bounds of the j -th design variables, respectively; ε is a very small floating-point number (i.e., 2^{-52}) to avoid singularity; r_2 is a uniformly distributed pseudorandom number varies from 0 to 1; and μ is a constant parameter to control the search process, which is set to 0.5 as suggested by Abualigah et al. [9]. Also, MOP denotes a function called Math Optimizer Probability and is established as follows:

$$MOP(C_{Iter}) = 1 - \left(\frac{C_{Iter}}{M_{Iter}} \right)^{1/\alpha} \quad (10.4)$$

where $MOP(C_{Iter})$ denotes the value of the coefficient MOP at the current iteration number, and α is a sensitive parameter reflecting the accuracy of exploitation over the iterations, which is set to 5 as suggested by Abualigah et al. [9].

It can be seen from Eq. (10.3) that the division (D) operator is conditioned by $r_2 > 0.5$ and the multiplication (M) operator is ignored until the division (D) operator completes its search task; otherwise, the multiplication (M) operator is adopted to perform the search process instead of the D .

10.2.3 Exploitation Phase

In contrast with division (D) and multiplication (M) operators that produce large step sizes which result in highly scattered population of candidate solutions, subtraction (S) and addition (A) operators produce small step sizes which lead to highly dense population of candidate solutions. As a result, these operators can easily intensify the search process in the promising areas of the search space detected in the exploration phase. The exploitation phase of the AOA is performed based on the following position update rule:

$$x'_{i,j} = \begin{cases} x_{Best,j} - MOP \times ((UB_j - LB_j) \times \mu + LB_j), & r_3 > 0.5 \\ x_{Best,j} + MOP \times ((UB_j - LB_j) \times \mu + LB_j), & \text{otherwise} \end{cases} \quad (10.5)$$

where r_3 is a uniformly distributed pseudorandom number varies from 0 to 1.

From Eq. (10.5), it can be seen that the subtraction (S) operator is conditioned by $r_3 > 0.5$ and the addition (A) operator is neglected until the subtraction (S) operator performs its search task; otherwise, the Addition (A) operator is employed to guide the search process instead of the S .

It is noted that, in both exploitation and exploration phases, if the penalized objective function value of the updated position of the i -th candidate solution is more fitted than that of its current position, the new position will replace its corresponding current position. For a minimization problem, this can be stated mathematically as follows:

$$x_i = \begin{cases} x'_i, & p(x'_i) < p(x_i) \\ x_i, & \text{otherwise} \end{cases} \quad (10.6)$$

where $p(x_i)$ denotes the penalized objective function value of the i -th solution.

The pseudocode of the standard AOA is shown in Fig. 10.3 [9].

Pseudocode of the standard AOA

```

1: Initialization phase
2: Initialize the algorithm parameters:  $N$ ,  $Min$ ,  $Max$ ,  $M_{Iter}$ ,  $\mu$ , and  $\alpha$ 
3: Generate the initial population of solutions randomly:  $X = [x_1; x_2; \dots; x_N]$ 
4: Identify the best solution of initial population:  $x_{Best}$ 
5:  $C_{Iter} = 1$ 
6: while ( $C_{Iter} \leq M_{Iter}$ ) do
7:    $MOA(C_{Iter}) = Min + C_{Iter} \times ((Max - Min)/M_{Iter})$ 
8:    $MOP(C_{Iter}) = 1 - (C_{Iter}/M_{Iter})^{(1/\alpha)}$ 
9:   for  $i = 1$  to  $N$  do
10:    for  $j = 1$  to  $n$  do
11:      if  $MOA(C_{Iter}) < r_1$  then
12:        if  $r_2 > 0.5$  then (Exploration phase)
13:           $x'_{i,j} = x_{Best,j} \div MOP \times ((UB_j - LB_j) \times \mu + LB_j)$ 
14:        else
15:           $x'_{i,j} = x_{Best,j} \times MOP \times ((UB_j - LB_j) \times \mu + LB_j)$ 
16:        end if
17:      else
18:        if  $r_3 > 0.5$  then (Exploitation phase)
19:           $x'_{i,j} = x_{Best,j} - MOP \times ((UB_j - LB_j) \times \mu + LB_j)$ 
20:        else
21:           $x'_{i,j} = x_{Best,j} + MOP \times ((UB_j - LB_j) \times \mu + LB_j)$ 
22:        end if
23:      end if
24:    end for
25:    Enforce design variable constraints
26:    if  $p(x'_i) < p(x_i)$  then
27:       $x_i = x'_i$ 
28:    end if
29:    Update the best solution found so far:  $x_{Best}$ 
30:  end for
31:   $C_{Iter} = C_{Iter} + 1$ 
32: end while
33: return  $x_{Best}$ 

```

Fig. 10.3 Pseudocode of the standard arithmetic optimization algorithm

10.3 Proposed Improved Arithmetic Optimization Algorithm (IAOA)

The standard AOA represents an interesting source in inspiration for exploring the search space. However, as observed in previous studies, the standard AOA suffers from premature stagnation in non-optimal solutions [10, 11]. As will be discussed

below, this may be due to the rapid loss of population diversity in the early stages of the search process, caused by poor exploration ability of AOA. As it was shown in the previous section, the exploitation and exploration phases of the standard AOA are clearly distinguished from each other by their unique position updating rules (i.e., Eqs. (10.3) and (10.5) for exploration and exploitation phases, respectively). Exploration means the ability to produce diverse solutions so as to explore the search space globally. However, it can be deduced from Eq. (10.3) that the exploration phase of the AOA consists of focusing the search around the best solution found so far, which may rapidly reduce the population diversity during the initial stages of the search process. As a result, the standard AOA suffers from the problem of poor exploration and may converge prematurely to non-optimal solutions. On the other hand, the position updating rule given by Eq. (10.3) involves the bounds of design variables, which means that the exploration phase of the standard AOA depends on the bounds of design variables. This may cause difficulties in convergence, especially when bounds of the design variables are too narrow or too wide. In fact, when the bounds of the design variables are too conservative, large step sizes are generated for the movement of solutions in the search space, which may result in exceeding the bounds of the search space. On the other hand, when the lower and upper bounds are very close to each other, small step sizes are generated for the movement of solutions in the search space, which may lead to premature convergence to non-optimal solutions. In addition, in the case where the design variables have the same upper and lower bounds, the standard AOA suffers from another serious drawback. This drawback will be discussed below in more detail.

Consider the case where all design variables have the same bounds (i.e., $UB_1 = UB_2 = \dots = UB_n$ and $LB_1 = LB_2 = \dots = LB_n$). This is the case of discrete structural optimization where the design variables are selected from a predefined list of standard sections. In such cases, it follows from Eq. (10.3) that, in each iteration, when $r_2 > 0.5$, all the design variables of the best-so-far solution (i.e., $x_{Best,j}$, $j = 1, 2, \dots, n$) are scaled by the same factor $(MOP + \varepsilon) \times ((UB_j - LB_j) \times \mu + LB_j)$. Similarly, in each iteration, when $r_2 \leq 0.5$, all the design variables of the best-so-far solution are multiplied by the same factor $MOP \times ((UB_j - LB_j) \times \mu + LB_j)$. As a result, in the exploration phase of each iteration of the standard AOA, all the design variables of the best solution found so far are scaled (i.e., multiplied or divided) by only two factors, which leads to the exploration of a very limited area of the search space. This may cause the loss of population diversity, which may result in slow convergence and premature convergence to non-optimal solutions. This is also true for the exploitation phase of the standard AOA (see Eq. (10.5)), which will be discussed later. Consequently, in this study, to address the drawbacks of the exploration phase of the standard AOA, on the basis of division and multiplication operators, a new position updating rule is proposed for the exploration phase, as follows:

$$x'_{i,j} = \begin{cases} x_{i,j} \div (1 + (-1)^{\text{randi}([1,2])} \times 0.5 \times \text{rand} \times \overline{MOP}), & r_2 > 0.5 \\ x_{i,j} \times (1 + (-1)^{\text{randi}([1,2])} \times 0.5 \times \text{rand} \times \overline{MOP}), & \text{otherwise} \end{cases} \quad (10.7)$$

where $x_{i,j}$ is the current value of the j -th design variable of the i -th candidate solution; $rand$ is a uniformly distributed pseudorandom number varies from 0 to 1; $randi([1, 2])$ produces a pseudorandom scalar integer from 1 to 2; and \overline{MOP} is a parameter-free version of the function MOP and is proposed as follows:

$$\overline{MOP}(C_{Iter}) = \left(1 - \frac{C_{Iter}}{M_{Iter}}\right)^{randi([1,2])} \quad (10.8)$$

In contrast to the standard AOA, the exploration phase of the IAOA implies focusing the search around the current position of solutions, as can be seen from Eq. (10.7). In other words, during the exploration phase of the IAOA, each solution is updated with respect to its current position, thus allowing a broader exploration of the search space to avoid diversity loss during the search process. In addition, the random numbers used in Eqs. (10.7) and (10.8) cause that different step sizes are generated more the movement of solutions in the search space, which can enhance the exploration and maintain the diversity of the population. It can be seen that Eq. (10.7) does not depend on the bounds of design variables, which may help to avoid convergence difficulties.

As Eq. (10.5) suggests, the position updating rule of solutions in the exploitation phase of the standard AOA, similar to its exploration phase, involves focusing the search around the best-so-far solution. As aforementioned, the major drawback of the exploitation phase of the standard AOA arises when all the design variables have the same upper and lower bounds. In this case, as Eq. (10.5) suggests, in each iteration, the same factor $MOP \times ((UB_j - LB_j) \times \mu + LB_j)$ is subtracted from (when $r_3 > 0.5$) or added to (when $r_3 \leq 0.5$) all the design variables of the best solution found so far. As noted earlier, this means that, in each iteration of the exploitation phase of the standard AOA, all the design variables of the best solution found so far are scaled (i.e., added or subtracted) by only two factors. As a result, exploitation does not take place effectively. Moreover, it is seen that the exploitation phase of the AOA depends also on the bounds of design variables, which may cause convergence difficulties. In this study, to overcome the drawbacks mentioned above, a new position updating rule based on subtraction and addition operators is proposed for the exploitation phase, as follows:

$$x'_{i,j} = \begin{cases} x_{Best,j} - x_{Best,j} \times rand \times \overline{MOP}, & r_3 > 0.5 \\ x_{Best,j} + x_{Best,j} \times rand \times \overline{MOP}, & \text{otherwise} \end{cases} \quad (10.9)$$

By using Eqs. (10.8) and (10.9), different step sizes are generated for the movement of solutions in the search space, thus allowing a stronger exploitation of the best-so-far solution. The formulation of the standard AOA involves the four algorithm-specific parameters of Min , Max , α , and μ that should be tuned for any specific application. From Eqs. (10.7), (10.8), and (10.9), it is seen that the parameters α and μ are eliminated from the formulation of the IAOA, which makes it easier to be implemented compared with the standard AOA. It is noted that IAOA proposed in this study is designed to solve optimization problems of skeletal structures with discrete

search spaces. However, it has the potential to extend its application to other types of structural optimization problems, including shells, plates, etc. To our knowledge, this is the first attempt to apply AOA for structural optimization. The pseudocode and flowchart of IAOA are shown in Figs. 10.4 and 10.5, respectively.

Pseudocode of the proposed IAOA

```

1: Initialization phase
2: Initialize the algorithm parameters:  $N$ ,  $Min$ ,  $Max$ , and  $M_{Iter}$ 
3: Generate the initial population of solutions randomly:  $X = [x_1; x_2; \dots; x_N]$ 
4: Identify the best solution of initial population:  $x_{Best}$ 
5:  $C_{Iter} = 1$ 
6: while ( $C_{Iter} \leq M_{Iter}$ ) do
7:    $MOA(C_{Iter}) = Min + C_{Iter} \times ((Max - Min)/M_{Iter})$ 
8:   for  $i = 1$  to  $N$  do
9:     for  $j = 1$  to  $n$  do
10:       $\overline{MOP}(C_{Iter}) = (1 - C_{Iter}/M_{Iter})^{randi([1,2])}$ 
11:      if  $MOA(C_{Iter}) < r_1$  then
12:        if  $r_2 > 0.5$  then (Exploration phase)
13:           $x'_{i,j} = x_{i,j} \div (1 + (-1)^{randi([1,2])} \times 0.5 \times rand \times \overline{MOP})$ 
14:        else
15:           $x'_{i,j} = x_{i,j} \times (1 + (-1)^{randi([1,2])} \times 0.5 \times rand \times \overline{MOP})$ 
16:        end if
17:      else
18:        if  $r_3 > 0.5$  then (Exploitation phase)
19:           $x'_{i,j} = x_{Best,j} - x_{Best,j} \times rand \times \overline{MOP}$ 
20:        else
21:           $x'_{i,j} = x_{Best,j} + x_{Best,j} \times rand \times \overline{MOP}$ 
22:        end if
23:      end if
24:    end for
25:    Enforce design variable constraints
26:    if  $p(x'_i) < p(x_i)$  then
27:       $x_i = x'_i$ 
28:    end if
29:    Update the best solution found so far:  $x_{Best}$ 
30:  end for
31:   $C_{Iter} = C_{Iter} + 1$ 
32: end while
33: return  $x_{Best}$ 

```

Fig. 10.4 Pseudocode of the improved arithmetic optimization algorithm

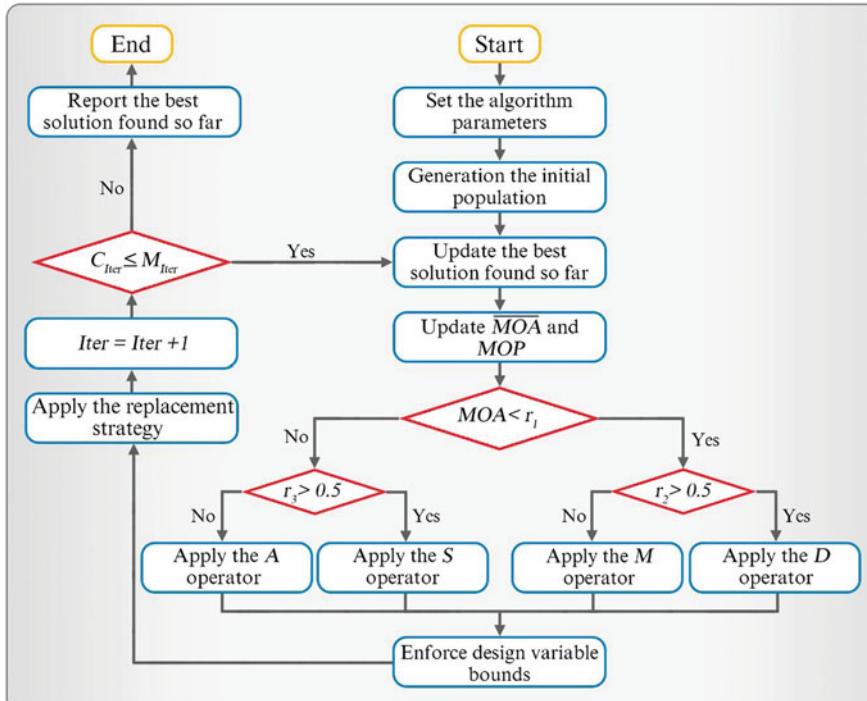


Fig. 10.5 Flowchart of the improved arithmetic optimization algorithm

10.4 Structural Optimization with Discrete Design Variables

The general purpose of structural optimization is to achieve a design that not only minimizes the structural weight but also satisfies the design constraints as well as the design variable bounds. In structural sizing optimization problems with discrete search spaces, the structural elements are chosen from a list of predefined available sections. In such problems, because of engineering practice demands and simplicity, the structural members are divided into groups of elements having the same design variables. In other words, all structural members of each group are assigned the same cross-sectional area. Accordingly, the number of element groups is equal to the number of design variables. Therefore, the discrete sizing optimization problem of skeletal structures is mathematically formulated as follows [12]:

$$\text{Find: } \{A\} = \{A_1, A_2, \dots, A_d\} \quad (10.10)$$

$$\text{to minimize: } f(\{A\}) = \sum_{i=1}^m \rho_i \times A_i \times L_i \quad (10.11)$$

$$\text{subject to: } g_n(\{A\}) \leq 0; n = 1, 2, \dots, c \quad (10.12)$$

where $\{A\}$ represents the vector of sizing design variables; d denotes the number of design variables (i.e., element groups); m represents the total number of members of the structure; ρ_i , L_i , and A_i are the material density, length, and cross-sectional area of the i -th structural element; $f(\{A\})$ denotes the objective function (i.e., the total weight of the structure); $g_n(\{A\})$ represents the n -th design constraint of the problem; and c denotes the total number of design constraints of the problem.

As aforementioned, in discrete structural optimization problems, each design variable can only take its value from a list of predefined available sections. This can be expressed mathematically as follows:

$$A_i \in D_i = \{d_{i,j}; j = 1, 2, \dots, k_i\}; i = 1, 2, \dots, d \quad (10.13)$$

where k_i is the number of available sections for the i -th design variable; D_i is the discrete set of available sections for the i -th design variable; and $d_{i,j}$ is the j -th section available for the i -th design variable.

The above problem is a constrained optimization problem. To deal effectively with design constraints, the penalty function method, perhaps the most well-known constraint-handling approach, is utilized. In the penalty function method, the original constrained optimization problem is transformed into an unconstrained one, by incorporating a penalty term in the objective function. In this study, the following dynamic multiplicative penalty function, known as the Kaveh-Zolghadr technique, is adopted [13]:

$$f_{\text{penalty}}(A) = (1 + \varepsilon_1 \times v)^{\varepsilon_2}; v = \sum_{n=1}^c v_i \quad (10.14)$$

where $f_{\text{penalty}}(A)$ is the penalty function; v stands for the sum of constraint violations; c is the number of design constraints; and ε_1 and ε_2 are the parameters of the penalty function. These parameters are adjusted based on the exploitation and exploration properties of the search process. In this study, in all design examples, ε_1 is set to a constant value, and ε_2 increases linearly with the number of iterations. This implies that, as the search progresses, a higher penalty is imposed on infeasible solutions. As a consequence, in the early iterations, highly infeasible solutions are also admissible, and therefore, the search agents are allowed to move freely through the search space, but as the search progresses, feasible solutions are preferred over infeasible ones [13]. In the case when the i -th design constraint is violated, then the value of v_i depends on the severity of the violation; otherwise, it is set to 0. This can be stated mathematically as follows:

$$v_i = \begin{cases} g_n(\{A\}), & \text{the } n\text{-th design constraint is violated} \\ 0, & \text{otherwise} \end{cases} \quad (10.15)$$

Consequently, the optimization problem is rewritten as follows:

$$\text{Minimize: } p(A) = f(A) \times f_{penalty}(A) \quad (10.16)$$

where $p(A)$ is the penalized objective function.

10.5 Numerical Examples

In this section, to examine the effectiveness and efficiency of the proposed IAOA, three weight minimization problems of skeletal structures with discrete search spaces are investigated. The results achieved by the proposed IAOA are compared with those of the standard AOA and some other state-of-the-art optimization methods. The examples include a 72-bar space truss structure with 16 design variables, a 384-bar double-layer barrel vault with 31 design variables, and a 3-bay 15-story steel frame structure with 11 design variables. In all examples, for both the AOA and IAOA, the population size is to $N = 20$. In addition, the maximum number of iterations (M_{Iter}) is set to 200 for the 72-bar truss, 1000 for the 384-bar double-layer barrel vault, and 1000 for the 3-bay 15-story steel frame. These values were selected on the basis of the results of preliminary experiments and other similar works in the literature. In addition, the parameters Min , Max , α , and μ are fixed at 0.2, 0.9, 5, and 0.5, as suggested by Abualigah et al. [9]. Due to the stochastic nature of the optimization procedure, ten independent optimization runs were carried out for each example. The statistical results of the standard AOA and IAOA are provided in terms of the best, worst, average, and standard deviation of the optimized weights obtained over ten independent runs. The maximum number of finite element analyses (Max_{NFE}) of different methods are also provided. The finite element method (FEM) model for structural analysis of structures and the optimization algorithms were programmed in Matlab R2016b software.

10.5.1 A 72-Bar Space Truss

As the first design example, the optimization problem of a space truss structure shown in Fig. 10.6 is considered. This problem was previously solved by Kaveh and Ilchi Ghazaan [14], Kaveh and Bakhshpoori [15], Jalili and Hosseinzadeh [16], Le et al. [17], Ho-Huu et al. [18], etc. The modulus of elasticity of the material is $E = 10000$ ksi and the material density is $\rho = 0.1$ lb/in³. The elements of the truss are grouped into 16 groups (see Table 10.3). The cross-sectional area of each element

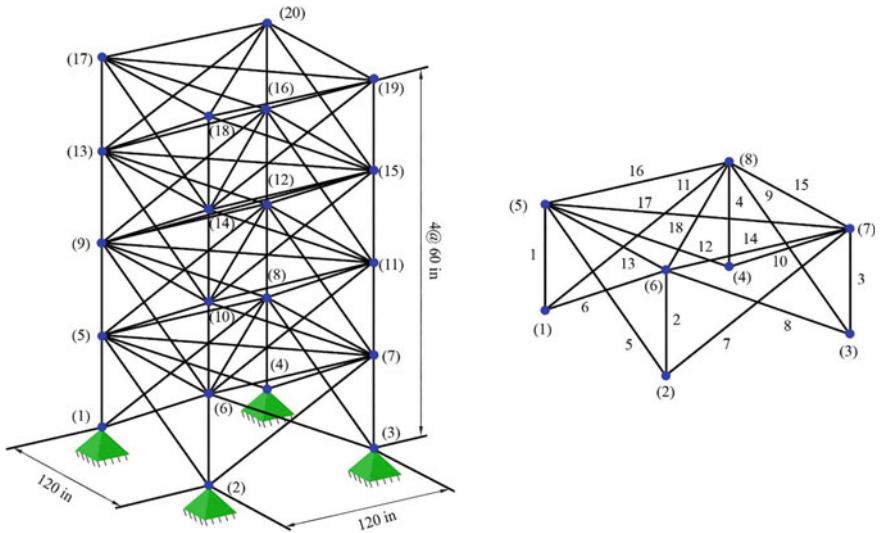


Fig. 10.6 Schematic of the 72-bar space truss

group is considered as a discrete sizing design variable. Hence, this is an optimization problem with 16 sizing design variables. The design variables must be selected from set of discrete values listed in Table 10.1. The structure should be designed for two different loading conditions shown in Table 10.2. For all free nodes, the maximum nodal displacements are restricted to ± 0.25 in in all directions. All the members are subjected to stress limitations of ± 25 ksi.

The optimization results achieved by the standard AOA and IAOA are compared in Table 10.3 with those of the colliding bodies optimization (CBO) and enhanced CBO (ECBO) [14], basic water evaporation optimization (WEO) and accelerated WEO (AWEO) [15], biogeography-based optimization and differential evolution (BBO-DE) [16], electromagnetism-like firefly algorithm (EFA) [17], and adaptive elitist differential evolution (aeDE) [18]. In this case, it seems that the standard AOA has prematurely converged to non-optimal solutions. In detail, the best design obtained by the AOA after 4000 FE analyses corresponds to a structural weight of 470.8550 lb, which is much heavier (i.e., about 21%) than those achieved by IAOA and some methods reported in the literature (389.3342 lb). Furthermore, the extremely high standard deviation of the solutions gained by AOA (83.3287 lb) indicates the instability of the original algorithm. In contrast, IAOA gives the best results in terms of worst weight, average weight, and standard deviation. The worst weight obtained by IAOA after 4000 FE analyses has a structural weight of 390.3524 lb, which is even better than average weights of other reported methods. Considering the computational cost, the maximum number of FE analyses (Max_{NFE}) of IAOA is 4000, which is lower than those reported in the literature, except for EFA (3123). Figure 10.7 illustrates the optimal weights obtained by AOA and IAOA over ten runs. It can easily be observed that AOA converged to non-optimal solutions in all runs, as

Table 10.1 Cross-sectional areas provided in the AISC code [17]

Section number	Area (in ²)						
1	0.111	17	1.563	33	3.840	49	11.500
2	0.141	18	1.620	34	3.870	50	13.500
3	0.196	19	1.800	35	3.880	51	13.900
4	0.250	20	1.990	36	4.180	52	14.200
5	0.307	21	2.130	37	4.220	53	15.500
6	0.391	22	2.380	38	4.490	54	16.000
7	0.442	23	2.620	39	4.590	55	16.900
8	0.563	24	2.630	40	4.800	56	18.800
9	0.602	25	2.880	41	4.970	57	19.900
10	0.766	26	2.930	42	5.120	58	22.000
11	0.785	27	3.090	43	5.740	59	22.900
12	0.994	28	3.130	44	7.220	60	24.500
13	1.000	29	3.380	45	7.970	61	26.500
14	1.228	30	3.470	46	8.530	62	28.000
15	1.266	31	3.550	47	9.300	63	30.000
16	1.457	32	3.630	48	10.850	64	33.500

Table 10.2 Load cases for the 72-bar space truss problem [17]

Load case	Node	Direction of loading		
		P_x (lb)	P_y (lb)	P_z (lb)
1	17	5000	5000	-5000
2	17	0	0	-5000
	18	0	0	-5000
	19	0	0	-5000
	20	0	0	-5000

aforementioned. However, IAOA consistently converged to near-optimal solutions, confirming its high stability. The convergence weight histories of the standard AOA and IAOA are plotted in Fig. 10.8. From the figure, it is evident that IAOA converges much faster than the standard AOA. Furthermore, it is observed that AOA converged prematurely within the first 91 iterations (i.e., the first 1820 FE analyses) and no improvement was occurred in the optimized weight, indicating that AOA has a poor exploration and easily trapped in non-optimal solutions. Figure 10.9 compares the average number of solution replacements observed in the search process of AOA and IAOA. It is apparent that IAOA can much more effectively explore the search space than the standard AOA. In detail, the average number of solution replacements occurred within the 4000 FE analyses of IAOA is 558.6 which is much more

Table 10.3 Comparison of optimal results obtained for the 72-bar space truss structure

Element group (elements)	Cross-sectional areas (in^2)						This study AOA	IAOA
	CBO [14]	ECBO [14]	WEO [15]	AWEO [15]	BBO-DE [16]	EFA [17]		
1 (1–4)	2.130	1.990	1.990	1.990	1.990	1.990	3.090	1.990
2 (5–12)	0.563	0.563	0.563	0.563	0.563	0.563	0.391	0.563
3 (13–16)	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111
4 (17–18)	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111
5 (19–22)	1.228	1.228	1.228	1.228	1.228	1.228	1.800	1.228
6 (23–30)	0.442	0.442	0.442	0.442	0.442	0.442	0.307	0.563
7 (31–34)	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111
8 (35–36)	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111
9 (37–40)	0.442	0.563	0.563	0.563	0.563	0.563	1.457	0.563
10 (41–48)	0.563	0.563	0.563	0.563	0.563	0.563	0.766	0.442
11 (49–52)	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111
12 (53–54)	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111
13 (55–58)	0.196	0.196	0.196	0.196	0.196	0.196	0.196	0.196
14 (59–66)	0.563	0.563	0.563	0.563	0.563	0.563	0.766	0.563
15 (67–70)	0.391	0.391	0.391	0.391	0.391	0.391	0.766	0.391
16 (71–72)	0.563	0.563	0.563	0.563	0.563	0.563	0.307	0.563
Weight (lb)	391.23	389.33	389.33	389.33	389.33	389.33	470.8550	389.3342
Number of FE analyses	4620	17,010	10,510	7490	5840	2700	4160	3900
Average weight (lb)	456.69	391.59	391.20	390.94	390.62	391.376	390.913	389.6421

(continued)

Table 10.3 (continued)

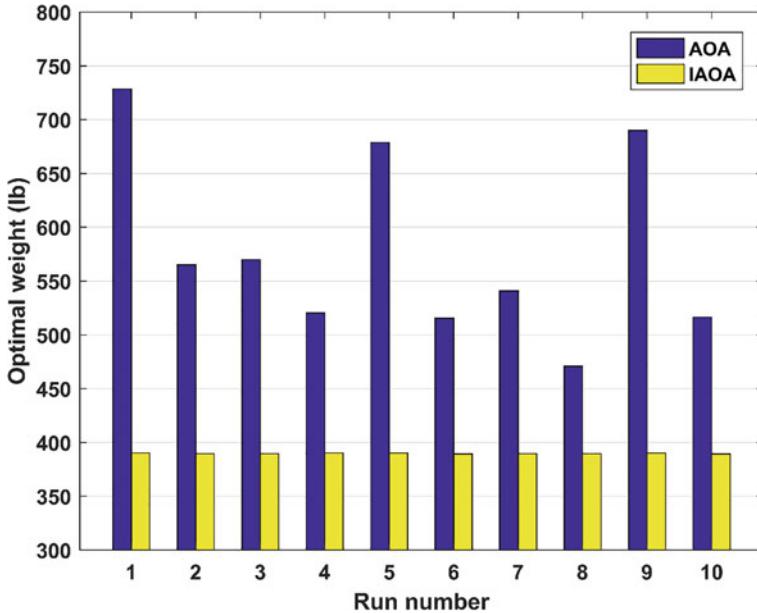


Fig. 10.7 Diversity of the optimal weights of the 72-bar space truss obtained by AOA and IAOA

(about 140%) than that of the standard AOA (232.7). The average number of solution replacements observed in the first half of the iterations (i.e., exploration phase) of the AOA is 13.7, which is about 3.55% of that of the IAOA (385.5). This confirms that the exploration capability of the IAOA is enhanced significantly in comparison to the AOA. The member stress ratios for the optimal design found by the IAOA for the 72-bar space truss are shown in Fig. 10.10. It can be seen that all stress constraints are met. Figure 10.11 shows the nodal displacements for the optimal design found by the IAOA for the 72-bar space truss. As expected, none of the displacement constraints are violated. From the figures, it can be concluded that displacement constraints control the design.

10.5.2 A 384-Bar Double-Layer Barrel Vault

For the second example, size optimization of a 384-bar double-layer braced barrel vault shown in Fig. 10.12 is considered. This problem has been previously solved using various metaheuristic techniques, such as CBO [19], ECBO [19], vibrating particles system (VPS) [19], a hybrid algorithm named MDVC-UVPS [19], ABC [20], cuckoo search (CS) [20], particle swarm optimization (PSO) [20], etc. The structure consists of a two-way bottom layer and a two-way top layer connected to each other by web elements. The structure consists of 111 ball joints and 384 members with

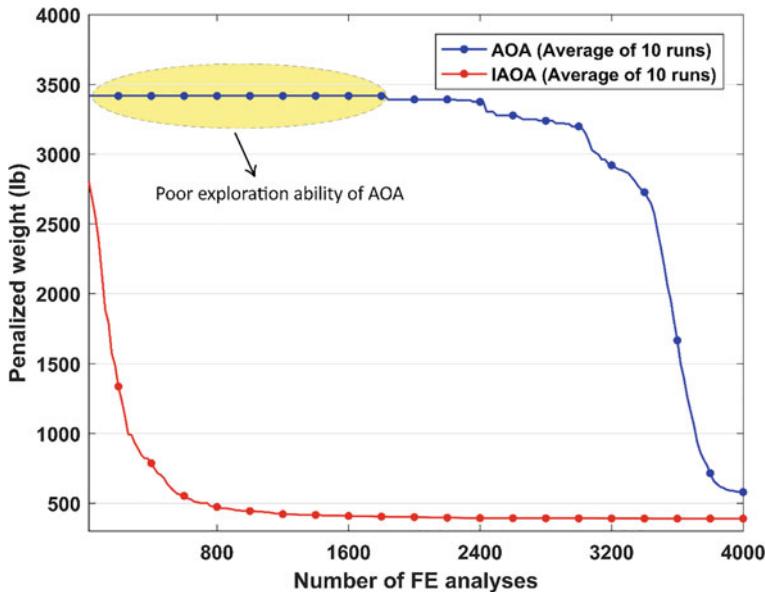


Fig. 10.8 Average weight convergence histories for the 72-bar space truss

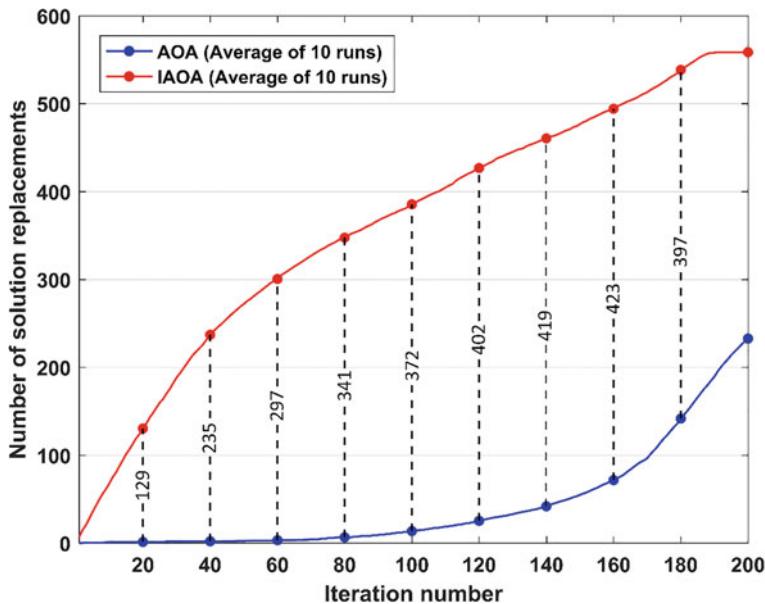


Fig. 10.9 Number of solution replacements observed in the AOA and IAOA (72-bar space truss)

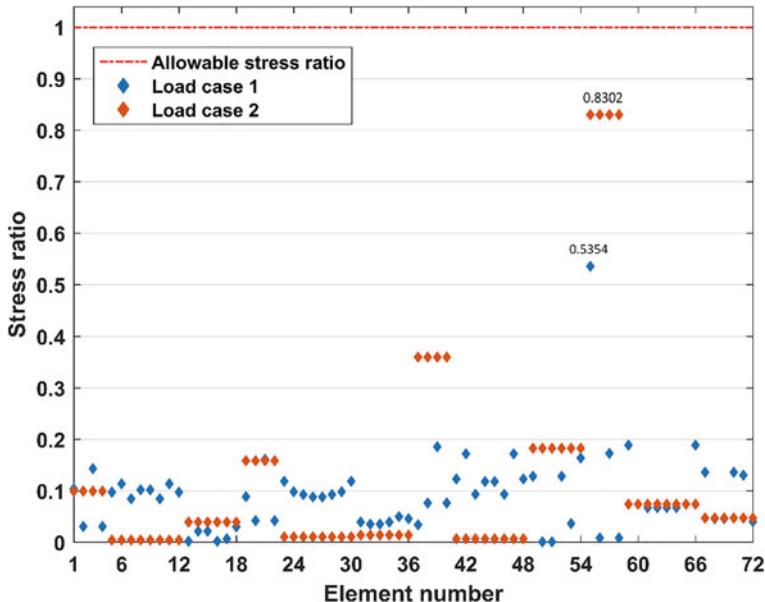


Fig. 10.10 Stress ratios of the 72-bar truss problem evaluated at the optimal design obtained by IAOA

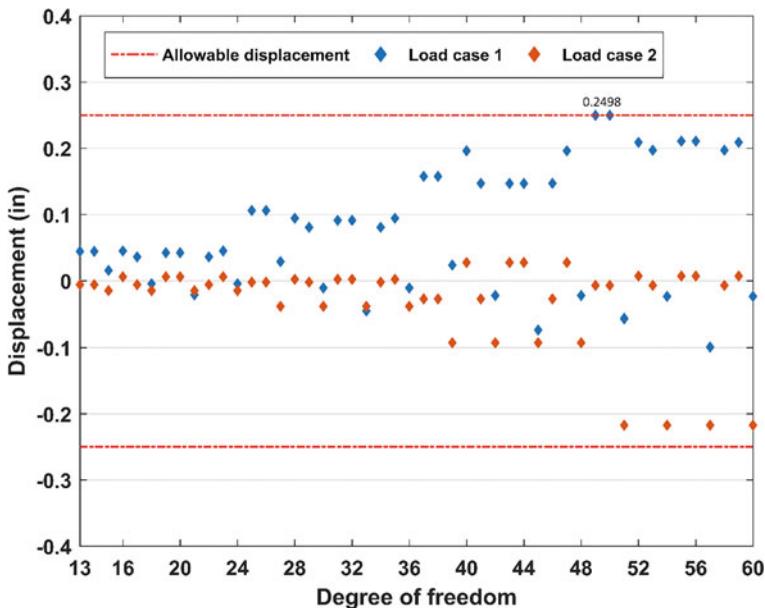


Fig. 10.11 Nodal displacements of the 72-bar truss problem evaluated at the optimal design obtained by IAOA

a span of 24.82 m, a length of 26.67 m, and a rise of 5.12 m. The depth of the barrel vault (i.e., the distance between the bottom and top layers) is equal to 1.35 m. The end connections of the structure are assumed to be hinged. As depicted in Fig. 10.12, the hinge supports are located along the two sides of the structure. With respect to the symmetry of the barrel vault about x and y axes, the 384 members are categorized into 31 element groups, as can be seen in Fig. 10.12. The cross-sectional area of each element group is taken as a sizing design variable. The layout of the structure remains fixed during the design procedure. Hence, this is an optimization problem with 31 sizing design variables. The design variables are selected from a standard list of 37 steel pipe sections given by the American Institute of Steel Construction-Load and Resistance Factor Design (AISC-LRFD) specification [21]. These sections are presented in Table 10.4. The abbreviations STD, XS, and XXS stand for standard weight, extra standard, and double-extra standard, respectively. The structural material properties are taken as follows: The elasticity modulus and the yield stress of steel are equal to $E = 30450$ ksi and $F_y = 58$ ksi, respectively, and the material density is taken as $\rho = 0.288$ lb/in³. A concentrated downward load of magnitude 20 kips is applied at each free node of the top layer of the barrel vault. The structure should be designed subject to strength and slenderness limitations according to the provisions of American Institute of Steel Construction-Allowable Stress Design (AISC-ASD) specification [22]. All free nodes in all directions are also subjected to the displacement constraints of ± 5 mm. The slenderness constraint is expressed in terms of the following equation [22]:

$$\begin{cases} L_i/r_i \leq 300, \text{ tension members } (\sigma_i \geq 0) \\ KL_i/r_i \leq 200, \text{ compression members } (\sigma_i < 0) \end{cases} \quad (10.17)$$

where K is the effective length factor and is set as unity for the pin-connected compression members; L_i is the length of the i -th member; and r_i is the radius of gyration of the i -th member. Based on the above equation, for members whose design is based on tensile force, the slenderness ratio should preferably not be greater than 300. In addition, for members whose design is based on compressive force, the slenderness ratio should preferably not be greater than 200.

Based on the provisions of AISC-ASD specification [22], the allowable tensile stress (F_t) and the allowable compressive stress (F_a) are determined as follows:

$$\begin{cases} \sigma_i \leq F_t = 0.6F_y, \text{ tension members } (\sigma_i \geq 0) \\ \sigma_i \leq F_a, \text{ compression members } (\sigma_i < 0) \end{cases} \quad (10.18)$$

$$F_{ai} = \begin{cases} \left[1 - \frac{(KL_i/r_i)^2}{2C_c^2} \right] F_y / \left(\frac{5}{3} + \frac{3(KL_i/r_i)}{8C_c} - \frac{(KL_i/r_i)^3}{8C_c^3} \right), & KL_i/r_i < C_c \\ \frac{12\pi^2 E}{23(KL_i/r_i)^2}, & KL_i/r_i > C_c \end{cases} \quad (10.19)$$

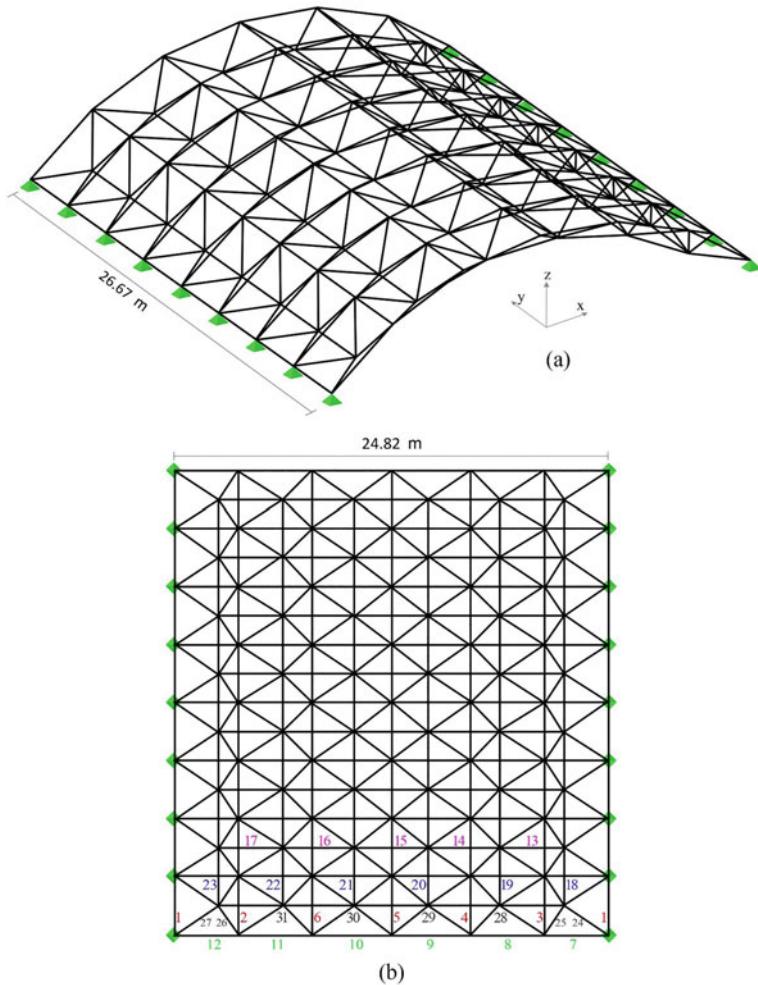


Fig. 10.12 Schematic of the 384-bar double-layer barrel vault: **a** perspective view; **b** top view

$$C_c = \sqrt{\frac{2\pi^2 E}{F_y}} \quad (10.20)$$

where C_c is the critical slenderness ratio.

In this example, the effectiveness of the modifications made in the proposed IAOA will be discussed in detail. Table 10.5 compares the optimal results achieved by the present algorithms with those of the other methods existing in the literature. The correctness of the FEM model was confirmed by comparing the results with those obtained by commercial finite element package SAP2000. It is seen that

Table 10.4 Steel pipe sections given by AISC-LRFD specification [21]

Section number	Type	Nominal diameter (in)	Area (in ²)	Moment of inertia (in ⁴)	Radius of gyration (in)
1	STD	½	0.250	0.017	0.261
2	XS	½	0.320	0.020	0.250
3	STD	¾	0.333	0.037	0.334
4	XS	¾	0.433	0.045	0.321
5	STD	1	0.494	0.087	0.421
6	XS	1	0.639	0.106	0.407
7	STD	1¼	0.669	0.195	0.540
8	STD	1½	0.799	0.310	0.623
9	XS	1¼	0.881	0.242	0.524
10	STD	2	1.07	0.666	0.787
11	XS	1½	1.07	0.391	0.605
12	XS	2	1.48	0.868	0.766
13	STD	2½	1.70	1.53	0.947
14	STD	3	2.23	3.02	1.16
15	XS	2½	2.25	1.92	0.924
16	XXS	2	2.66	1.31	0.703
17	STD	3½	2.68	4.79	1.34
18	XS	3	3.02	3.89	1.14
19	STD	4	3.17	7.23	1.51
20	XS	3½	3.68	6.28	1.31
21	XXS	2½	4.03	2.87	0.844
22	STD	5	4.30	15.2	1.88
23	XS	4	4.41	9.61	1.48
24	XXS	3	5.47	5.99	1.05
25	STD	6	5.58	28.1	2.25
26	XS	5	6.11	20.7	1.84
27	XXS	4	8.10	15.3	1.37
28	STD	8	8.40	72.5	2.94
29	XS	6	8.40	40.5	2.19
30	XXS	5	11.3	33.6	1.72
31	STD	10	11.9	161	3.67
32	XS	8	12.8	106	2.88
33	STD	12	14.6	279	4.38
34	XXS	6	15.6	66.3	2.06
35	XS	10	16.1	212	3.63
36	XS	12	19.2	362	4.33
37	XXS	8	21.3	162	2.76

IAOA achieved the best performance in terms of the average weight, best weight, worst weight, and standard deviation, while the standard AOA showed the worst performance in terms of the best optimized weight. The reason, as mentioned earlier, is may be due to the premature convergence of the standard AOA to non-optimal solutions. The IAOA obtained a feasible optimal solution with a structural weight of 62,639.08 lb after 19,300 FE analyses, which is lighter than the best weights gained by all the other algorithms, namely 69,448.52 lb for CBO, 62,486.02 lb for ECBO, 62,445.30 lb for VPS, 62,735.42 lb for MDVC-UVPS, 65,420.66 lb for ABC, 63,995.51 lb for CS, 64,859.22 lb for PSO, and 79,379.29 lb for AOA. The worst weight obtained by IAOA over ten independent runs is 64180.30 lb, which is even better than the average weights of the others. It should be noted out that all the optimal designs reported in [19] and [20] clearly violate the slenderness constraint of Eq. (10.17). The slenderness ratio of the first element group is 502.87 for CBO, 502.87 for ECBO, 392.96 for VPS, 502.87 for MDVC-UVPS, 322.48 for ABC, 502.87 for CS, and 502.87 for PSO. However, the optimal designs provided by AOA and IAOA satisfy the slenderness constraint of Eq. (10.5). In terms of the convergence speed, the maximum number of finite element (FE) analyses of the proposed IAOA is 20000, which is equal to or less than those of other reported methods. Compared to the standard AOA, IAOA showed significant improvements in all aspects. The IAOA obtained a feasible optimal solution with a structural weight of 77,263.13 lb after only 11,680 FE analyses, which is better than that of the standard AOA after 2000 FE analyses. Figure 10.13 illustrates the diversity of the optimal weights found by the standard AOA and IAOA over ten independent runs. As it is seen from the figure, AOA converged prematurely to non-optimal solutions in all runs. In contrast, the proposed IAOA consistently converged to near-optimal solutions, confirming its high robustness. The average convergence histories of the standard AOA and IAOA are also plotted for comparison in Fig. 10.14. It demonstrates clearly that IAOA converges faster than the standard AOA. On the other hand, it is observed that, after a few iterations, there is a sharp reduction in the convergence speed of the standard AOA and a slight improvement was observed in the average optimized weight until about iteration 750. This comes from the fact that the standard AOA suffers severely from the rapid loss of population diversity and is prone to stagnation in non-optimal solutions. However, thanks to the strong exploration of the search space in the proposed IAOA, it significantly improves the convergence rate. Figure 10.15 compares the average number of solution replacements occurred in AOA and IAOA. It is noted that the solution replacement means the current position of a solution is replaced by its corresponding updated position. It is seen that the average number of solution replacements observed in IAOA after 20,000 FE analyses is about 821, which is much more than about 603 of the standard AOA. This can be attributed to a better and more effective search process in the proposed IAOA compared with the standard AOA. Figures 10.16 and 10.17 show the member stress ratios and nodal displacements of the design found by the proposed IAOA for the 384-bar double-layer barrel vault, respectively. It is seen that both the strength and displacement constraints of the problem are satisfied.

Table 10.5 Comparison of optimal results obtained for the 384-bar double-layer barrel vault

Element group	Optimal pipe sections						PSO [20]	This study	AOA	IAOA
	CBO [19]	ECBO [19]	VPS [19]	MDVC-UVPS [19]	ABC [20]	CS [20]				
1	STD $\frac{1}{2}$	STD $\frac{1}{2}$	STD $\frac{3}{4}$	STD $\frac{1}{2}$	XS 1	STD $\frac{1}{2}$	STD $\frac{1}{2}$	STD $\frac{1}{2}$	STD $\frac{1}{4}$	STD $\frac{1}{4}$
2	XS 2	STD $2\frac{1}{2}$	XS $2\frac{1}{2}$	XS 2	STD $2\frac{1}{2}$	XS $1\frac{1}{2}$	STD 3	XS 8	XS 8	XS 2
3	XS 2	XS 2	XS $2\frac{1}{2}$	XS 2	XS 3	STD $2\frac{1}{2}$	XS $1\frac{1}{2}$	XS 3	XS 3	XS 2
4	STD 3	STD $1\frac{1}{2}$	XS $1\frac{1}{2}$	STD $1\frac{1}{2}$	XS $1\frac{1}{2}$	STD $1\frac{1}{2}$	STD $1\frac{1}{2}$	XS 2	STD $\frac{1}{4}$	STD $\frac{1}{4}$
5	XXS 2 $\frac{1}{2}$	XS 4	XXS 3	XXS 3	XXS 3	XS $2\frac{1}{2}$	XS 5	XS 3	XS 3	XXS 3
6	STD $2\frac{1}{2}$	STD $1\frac{1}{2}$	XS 2	XS 2	STD $\frac{1}{4}$					
7	STD 12	STD 12	STD 12	STD 12	STD 12	STD 12	XXS 8	STD 10	STD 10	STD 12
8	XXS 4	STD 10	XS 8	XXS 5	STD 10	XXS 4	XS 8	XXS 6	XXS 6	XXS 5
9	XXS 5	STD 12	XS 10	XS 10	XS 10	XS 12	XXS 6	XXS 8	XXS 8	XS 10
10	STD 12	XXS 8	XS 10	XS 10	STD 12	XS 8	XXS 8	XS 10	XS 10	XS 10
11	XXS 5	XXS 5	XXS 5	XXS 5	STD 10	XXS 5	XS 6	STD 8	STD 8	XXS 5
12	XXS 6	XS 8	XXS 5	STD 12	STD 10	XXS 6	XS 8	XS 10	XS 10	XS 8
13	XXS 3	STD 6	STD 6	STD 6	XS 5	STD 6	STD 6	XS 6	XS 6	XXS 3
14	XS 3 $\frac{1}{2}$	XXS 3	STD 4	XXS 3	XXS 2 $\frac{1}{2}$	XS 4	XS 3 $\frac{1}{2}$	XS 3	STD 4	STD 4
15	STD $2\frac{1}{2}$	STD $2\frac{1}{2}$	STD $2\frac{1}{2}$	XS $2\frac{1}{2}$	STD 3	STD $2\frac{1}{2}$	STD $2\frac{1}{2}$	STD $2\frac{1}{2}$	STD $2\frac{1}{2}$	STD $2\frac{1}{2}$
16	XS 6	STD 5	STD 5	STD 4	STD 4	XXS 3	STD 4	STD 4	STD 4	STD 5
17	XS 6	XS 4	XXS 3	STD 6	XS 5	XXS 2 $\frac{1}{2}$	XS 5	XS 5	XS 5	XS 4
18	XS 2	XS 1 $\frac{1}{2}$	XS 1 $\frac{1}{2}$	XS 1 $\frac{1}{2}$	XXS 2	XS $1\frac{1}{4}$	STD $2\frac{1}{2}$	STD 12	STD 12	XS 2
19	XS 2	STD 1 $\frac{1}{4}$	STD 1 $\frac{1}{4}$	STD 1 $\frac{1}{4}$	XS 2	STD 1 $\frac{1}{4}$	STD 1 $\frac{1}{4}$	STD 1 $\frac{1}{2}$	STD 1 $\frac{1}{2}$	XS 1 $\frac{1}{4}$
20	XS 2 $\frac{1}{2}$	XS 1 $\frac{1}{2}$	XS 1 $\frac{1}{2}$	XS 1 $\frac{1}{2}$	XS 2	STD 1 $\frac{1}{4}$	XS 2	STD 2 $\frac{1}{2}$	XS 2	(continued)

Table 10.5 (continued)

Element group	Optimal pipe sections						PSO [20]	This study	AOA	IAOA
	CBO [19]	ECBO [19]	VPS [19]	MDVC-UVPS [19]	ABC [20]	CS [20]				
21	XS 4	XS 1½	XS 1½	XS 1½	STD 2½	XS 1½	STD 2½	XS 6	STD 2½	XS 2
22	STD 3½	STD 1¼	XS 1½	STD 1¼	XS 2	STD 1¼	STD 1¼	STD 4	STD 4	XS 2
23	XS 1½	XS 1½	XS 1½	XS 1½	XS 1½	XS 1½	XS 1½	XXS 2	XXS 2	STD 2
24	STD 3½	XS 2½	XS 2½	STD 3½	STD 3	XS 2½	XS 3	XXS 2½	XXS 2½	STD 3½
25	STD 2½	XS 2½	XS 2½	XS 2	STD 2½	STD 2½	STD 2½	XS 1¼	STD 2½	STD 2½
26	XXS 4	STD 2½	XS 1½	XS 2	STD 2½	XS 2	STD 2½	STD 1½	STD 1½	XS 2
27	XS 3	XXS 2	STD 3	STD 3½	XXS 2	XS 3	STD 2½	XXS 2½	XXS 2½	STD 3
28	XS 2	XS 1½	XS 1½	XS 2	STD 3	STD 3	XS 1½	XS 2	XS 2	XS 2
29	STD 2½	STD 2½	XS 2	XS 2	XS 2	STD 2½	XS 2	XS 2½	XS 2½	XS 2
30	STD 3	XS 1½	XS 2	XS 2	XS 2	XS 2½	XS 1½	XS 2	XS 2	XS 2
31	STD 2½	XS 1½	XS 1½	XS 2	STD 2½	XS 2	XS 1½	STD 3	STD 3	XS 2
Weight (lb)	69,448.52	62,486.02	62,445.30	62,735.42	65,420.66	63,995.51	64,859.22	79,379.29	79,379.29	62,326.90
Number of FE analyses	4320	15,980	12,780	3460	N/A	N/A	N/A	18,880	18,880	19,360
Average weight (lb)	123,397	65,785	67,900	65,738	67,231.74	66,872.09	76,268.36	89,286.24	89,286.24	63,189.43
Worst weight (lb)	N/A	N/A	N/A	N/A	68,315.15	68,151.43	98,622.48	103,539.05	103,539.05	64,180.30
Standard deviation (lb)	103,837	3386	2913	2882	908.81	1178.48	10,368.10	7995.62	7995.62	631.29
Average number of FE analyses	20,000	20,000	20,000	20,000	30,000	30,000	30,000	20,000	20,000	20,000
Number of runs	30	30	30	30	20	20	20	10	10	10

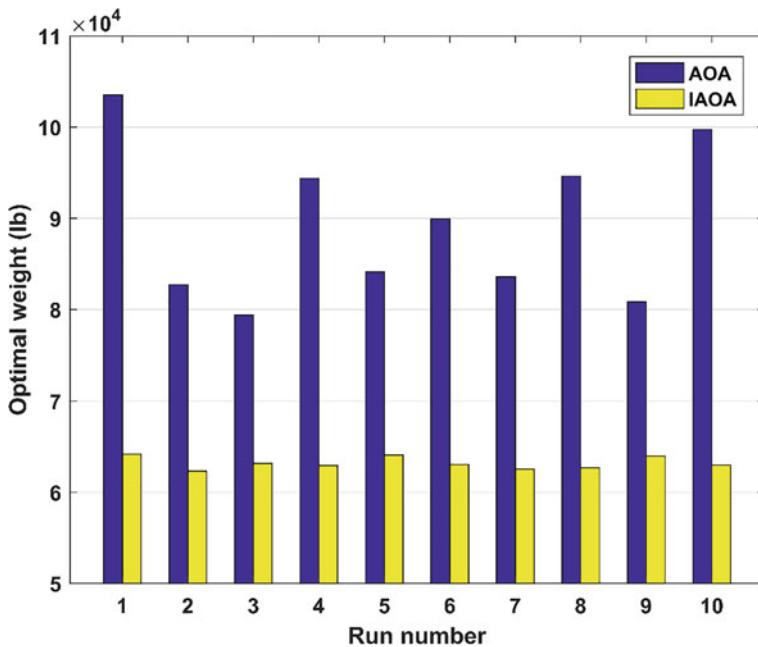


Fig. 10.13 Diversity of the optimal weights of the 384-bar double-layer barrel vault obtained by AOA and IAOA

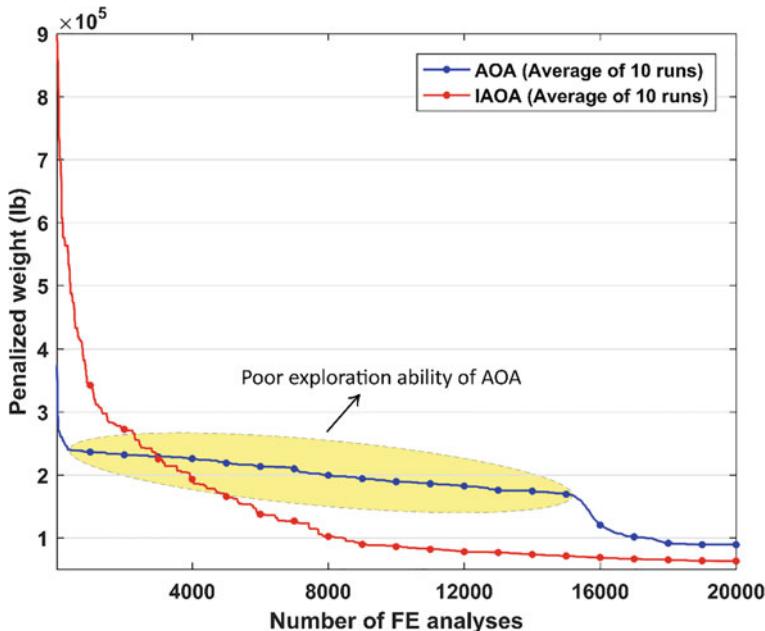


Fig. 10.14 Average weight convergence histories for the 384-bar double-layer barrel vault

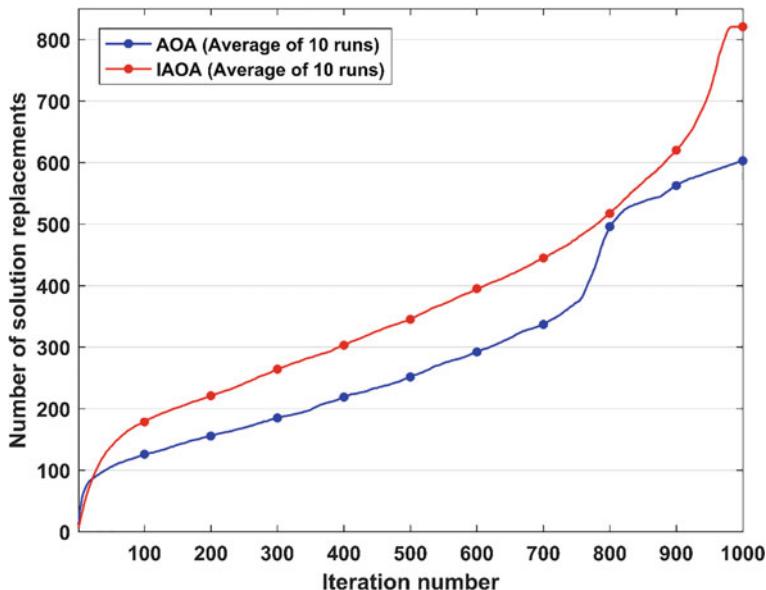


Fig. 10.15 Number of solution replacements observed in the AOA and IAOA (384-bar double-layer barrel vault)

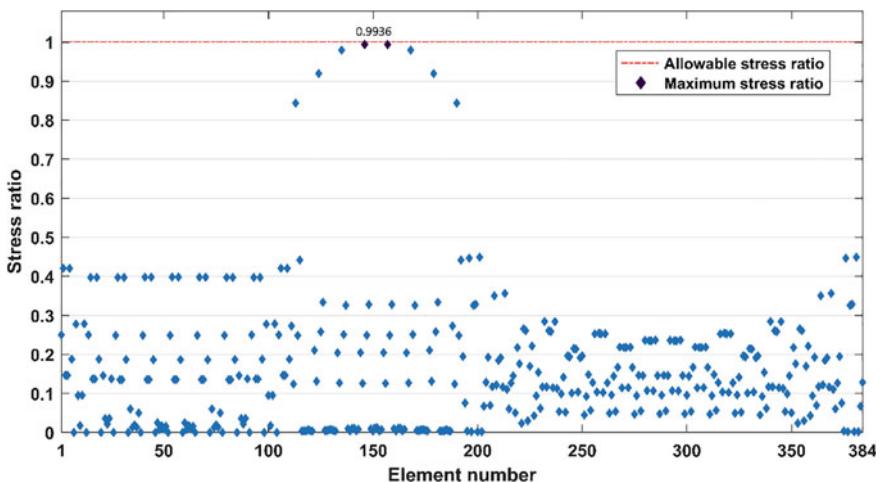


Fig. 10.16 Stress ratios of the 384-bar barrel vault problem evaluated at the optimal design obtained by IAOA

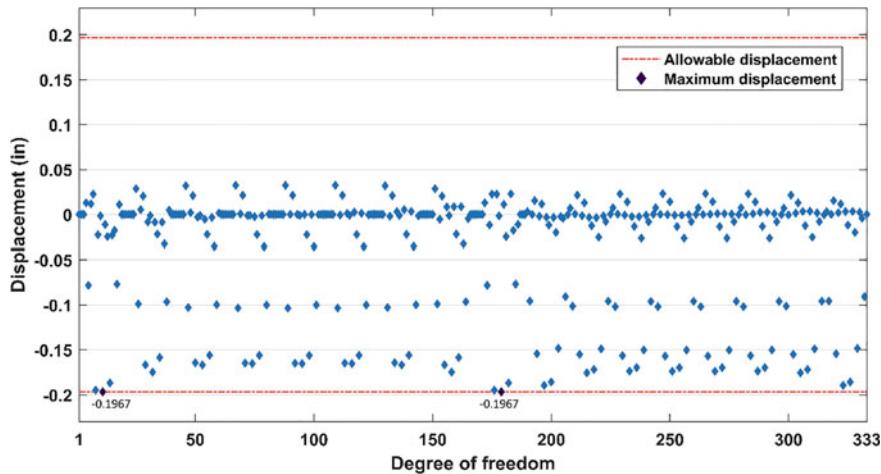


Fig. 10.17 Nodal displacements of the 384-bar barrel vault problem evaluated at the optimal design obtained by IAOA

10.5.3 A 3-Bay 15-Story Steel Frame

The last example is a 3-bay 15-story steel frame structure consisting of 105 structural members including 60 column and 45 beam elements. This problem has been previously investigated by means of several methods such as CBO and ECBO by Kaveh and Ilchi Ghazaan [14], WEO and AWEO by Kaveh and Bakhshpoori [15], GSS by Kazemzadeh Azad and Hasançebi [23], etc. The configuration of the structure, the elements grouping, and the loading conditions are depicted in Fig. 10.18. Because of engineering practice demands, the structural elements are divided into 11 element groups, leading to 11 discrete sizing design variables (see Fig. 10.18). The sections of all the column and beam element groups have to be chosen from all 267 W-shaped sections given by the AISC-LRFD specification [21]. The elasticity modulus of the material is $E = 29000$ ksi and the yield stress is $F_y = 36$ ksi. The structure should be designed subject to strength and serviceability constraints as specified by the AISC-LRFD specification [21]. The maximum lateral displacement of the top story should also not exceed 9.25 in [14]. The unbraced length of each beam element is taken as one-fifth of the length of the span, while for each column element, the unbraced length is the same as its height. The out-of-plane effective length factor is also taken as $K_y = 1$. In sway-permitted frames, the effective length factors of the members are specified as $K_x \geq 1$ and can be determined by the following approximate equation proposed by Dumonteil [24]:

$$K_x = \sqrt{\frac{1.6G_A G_B + 4(G_A + G_B) + 7.5}{G_A + G_B + 7.5}} \quad (10.21)$$

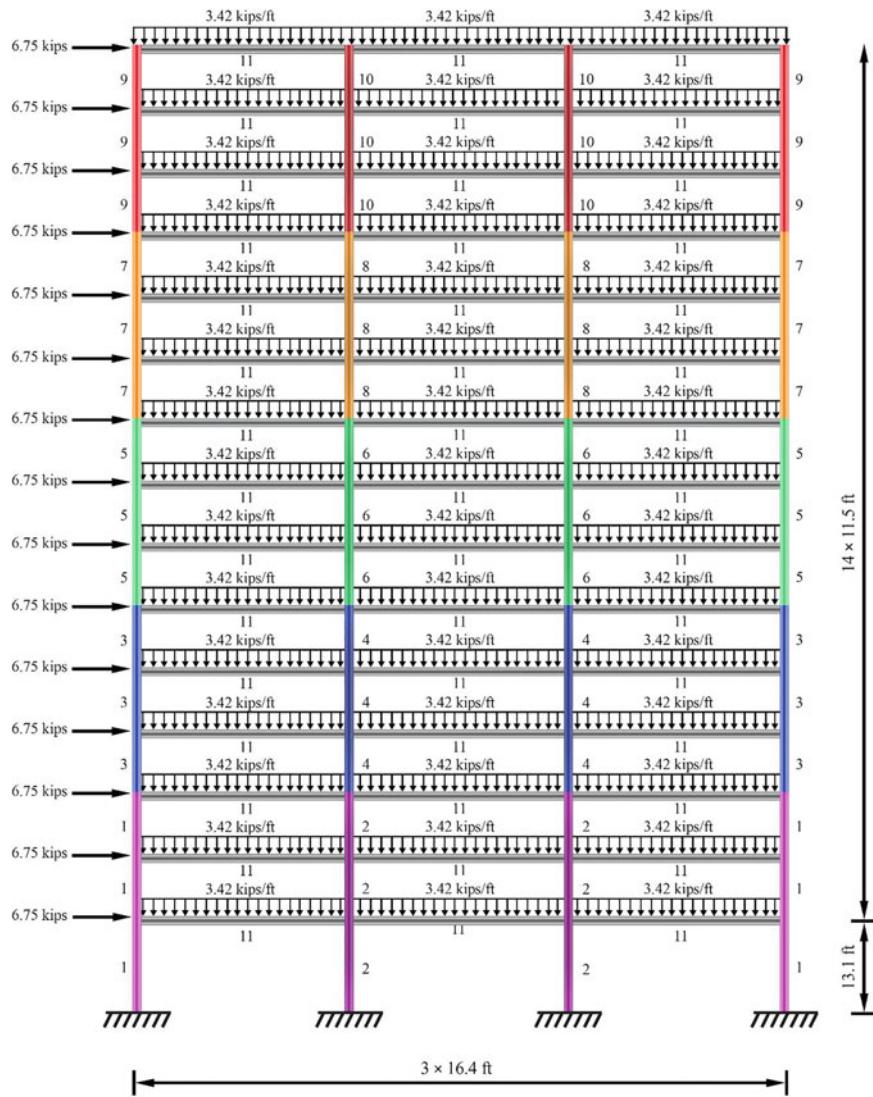


Fig. 10.18 Schematic of the 3-bay 15-story steel frame structure

where G_B and G_A are stiffness ratios of girders and columns at end joints, B and A , of the column section, respectively.

The strength and serviceability constraints of the problem are briefly given below:

(1) Maximum lateral displacement

$$\frac{\Delta_T}{H} - R \leq 0 \quad (10.22)$$

where H denotes the height of the frame structure to that level; R is the maximum allowable drift ratio, which is considered to be 1/300; and Δ_T is the maximum lateral displacement at the top of the frame structure.

(2) Inter-story displacements

$$\frac{d_i}{h_i} - R_I \leq 0, i = 1, 2, \dots, ns \quad (10.23)$$

where h_i denotes the height of the i -th story of the frame structure; d_i stands for the inter-story drift of the i -th story of the frame structure; ns denotes the total number of stories of the structure; and R_I is the maximum allowable inter-story drift ratio, which is set to 1/300 [21].

(3) Strength constraints

$$\begin{cases} \frac{P_r}{2\phi_c P_n} + \left(\frac{M_{rx}}{\phi_b M_{nx}} + \frac{M_{ry}}{\phi_b M_{ny}} \right) \leq 1, & \frac{P_r}{\phi_c P_n} < 0.2 \\ \frac{P_r}{\phi_c P_n} + \frac{8}{9} \left(\frac{M_{rx}}{\phi_b M_{nx}} + \frac{M_{ry}}{\phi_b M_{ny}} \right) \leq 1, & \frac{P_r}{\phi_c P_n} \geq 0.2 \end{cases} \quad (10.24)$$

where P_r is the required axial strength; P_n is the nominal axial strength; ϕ_b is the resistance factor for flexure ($\phi_b = 0.9$); M_{nx} and M_{rx} are the nominal flexural strength and the required flexural strength about x axis, respectively; M_{ny} and M_{ry} are the nominal flexural strength and the required flexural strength about y axis, respectively; ϕ_c is the resistance factor ($\phi_c = 0.9$ for tension, $\phi_c = 0.85$ for compression).

The optimal designs found by the standard AOA and IAOA with those of some other methods reported in the literature are given in Table 10.6. Similar to the previous examples, it is seen that the standard AOA is converged to non-optimal solutions. In detail, the best optimal design obtained by AOA over ten independent runs has a structural weight of 102,675.20 lb, which is much heavier than that of other methods. This is also true for the average weight of the AOA designs, indicating that the standard AOA suffers from the premature convergence problem. However, the proposed IAOA could effectively overcome the above-mentioned difficulty, resulting in consistently convergence to near-optimal designs. In detail, the average weight of IAOA designs is 88256.52 lb, which is better than those acquired by methods reported in the literature, namely 90,649.49 lb for WEO, 88,893.09 lb for AWEO, and 88,410 lb for ECBO. It is also observed that IAOA achieves the best weight of 87,261.95 lb which is better than others, except for ECBO (86,986 lb). In terms of the computational cost, the maximum number of FE analyses (Max_{NFE}) of the IAOA is identical to that of the ECBO. It can be seen that although the IAOA requires more FE analyses than WEO, AWEO, and GSS, it has obtained better results in terms of the average and best weights. Figure 10.19 illustrates the weight of the designs found by the AOA and IAOA. It is easily seen that, in all of the runs, the standard AOA converged prematurely to non-optimal solutions. However, thanks to the modifications made in the

IAOA, it consistently converged to solutions near the global minimum. The average convergence curves of the AOA and IAOA are provided in Fig. 10.20. It reveals that the IAOA converges much faster than the standard AOA. In addition, it is seen that the AOA converged prematurely within the first 414 iterations (i.e., the first 8280 FE analyses) and no improvement was observed in the average optimized weight. This results from the rapid loss of population diversity in the standard AOA, which results in premature convergence to non-optimal solutions. Figure 10.21 compares the average number of solution replacements observed in the AOA and IAOA. The average number of solution replacements after 20,000 FE analyses is about 568 for the IAOA, which is much better than that of the AOA (about 403). This mean that the IAOA could explore the search space more effectively than the AOA. Figure 10.22 depicts the stress ratio of members for the optimal design found by the IAOA. In this figure, it is seen that the optimal design of the IAOA satisfy the stress constraints of the problem. The maximum stress ratio is equal to 99.47%. The lateral displacement at the top story is 5.44 in, which is lower than the maximum allowable value of 9.25 in. Figure 10.23 depicts the inter-story drift ratios for the optimal design found by the IAOA. The maximum inter-story drift ratio is 0.3152% which is less than the allowable limit (0.3333%).

10.6 Application to High-Dimensional Structural Optimization Problems

In this section, in order to investigate the performance of the standard AOA in solving high-dimensional structural optimization problems, both the AOA and IAOA methods, as well as the standard PSO algorithm, are applied to solve the optimization problem of ISCSO 2017 [25]. This problem is size and shape optimization of the three-dimensional steel truss structure shown in Fig. 10.24. The total structure is composed of 52 nodes and 198 members. The topology of the truss is assumed to be unchangeable. The structure is designed for the following load cases: (1) horizontal loads of 10 kN applied in the positive x -direction, (2) horizontal loads of 10 kN applied in the positive y -direction, and (3) vertical loads of 10 kN applied in the negative z -direction. The loads are applied at all unsupported nodes of the truss. The displacements of all nodes in x , y , and z directions are limited to a maximum value of ± 100 mm. The material density, elastic modulus, and yield stress of the truss material are 7.85 ton/m³, 200 GPa, and 248.2 MPa, respectively. The structure is designed in accordance with the regulations of AISC-LRFD 1994 [26]. To this end, each member is checked considering the limit states of tensile yielding and compressive buckling. This problem includes 198 sizing variables (which represent the cross-sections of the truss members) as well as 13 shape variables (which are the z -coordinates of the top blue nodes as well as the x -coordinates of the side orange nodes of the structure) resulting in a total of 211 design variables. It should be emphasized that each pair of blue nodes with the same x -coordinate will have the same height. Similarly, each

Table 10.6 Comparison of optimal results obtained for the 3-bay 15-story steel frame structure

Element group	Optimal W-shaped section					
	ECBO [14]	WEO [15]	AWEO [15]	GSS [23]	This study	
					AOA	IAOA
1	W14 × 99	W14 × 90	W14 × 99	W36 × 135	W24 × 104	W14 × 99
2	W27 × 161	W36 × 170	W27 × 161	W27 × 146	W36 × 182	W27 × 161
3	W27 × 84	W30 × 90	W27 × 84	W24 × 94	W18 × 86	W27 × 84
4	W24 × 104	W24 × 104	W24 × 104	W14 × 109	W30 × 132	W24 × 104
5	W14 × 61	W24 × 68	W14 × 61	W24 × 68	W27 × 102	W21 × 68
6	W30 × 90	W12 × 87	W30 × 90	W21 × 93	W18 × 97	W18 × 86
7	W14 × 48	W8 × 48	W16 × 50	W30 × 90	W8 × 67	W8 × 48
8	W14 × 61	W14 × 68	W21 × 68	W18 × 65	W30 × 90	W12 × 65
9	W14 × 30	W10 × 33	W14 × 34	W16 × 36	W14 × 30	W8 × 28
10	W12 × 40	W16 × 45	W8 × 35	W16 × 40	W18 × 50	W10 × 39
11	W21 × 44	W21 × 44	W21 × 44	W21 × 44	W18 × 50	W21 × 44
Weight (lb)	86,986	88,710.97	87,537.96	93,756.57	102,675.20	87,261.95
Number of FE analyses	9000	13,580	10,670	N/A	19,520	19,300
Average weight (lb)	88,410	90,649.49	88,893.09	95,885.51	111,296.30	88,256.52
Worst weight (lb)	N/A	N/A	N/A	98,356.16	123,712.51	88,977.35
Standard deviation (lb)	N/A	N/A	N/A	1773.74	5823.09	489.79
Average number of FE analyses	20,000	14,000	14,000	600	20,000	20,000
Number of runs	20	20	20	10	10	10

pair of orange nodes with the same height will have the same x -coordinate. For more information, the readers can refer to [26]. As stated in the problem specification, the maximum number of iterations (M_{Iter}) is set to 7500. In addition, the population size is set to 40 for PSO and 20 for both the standard AOA and IAOA.

Figure 10.25 compares the penalized weight convergence curves of the AOA, IAOA, and PSO. It should be noted that none of the methods have been able to converge to any feasible solution. From the figure it is observed that the standard AOA has prematurely converged to a non-feasible solution too far away from the solution reported by the winner of ISCCSO 2017 (i.e., 44,090.4356 kg). This is due to the fact that the standard AOA suffers from poor exploration and is prone to stagnation in non-optimal solutions. On the other hand, it can be observed from the figure that both the PSO and IAOA methods have converged much faster than

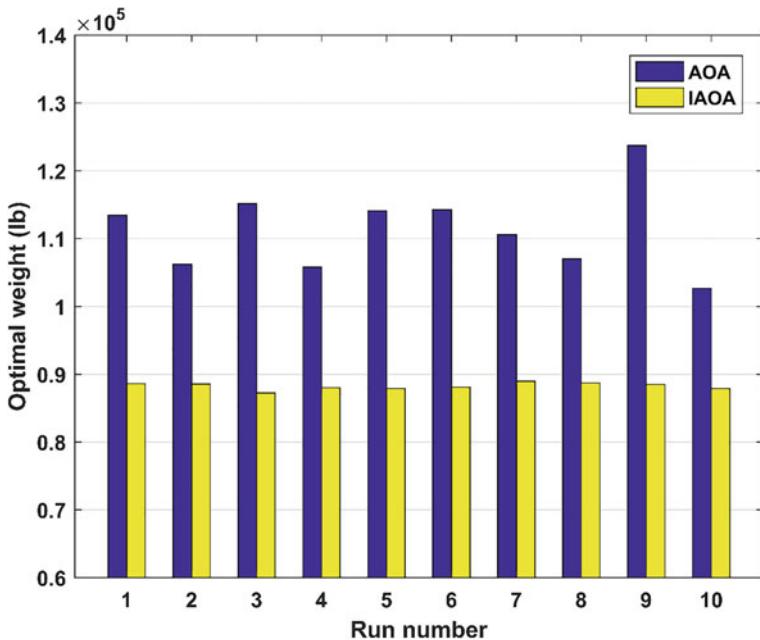


Fig. 10.19 Diversity of the optimal weights of the 3-bay 15-story steel frame structure obtained by AOA and IAOA

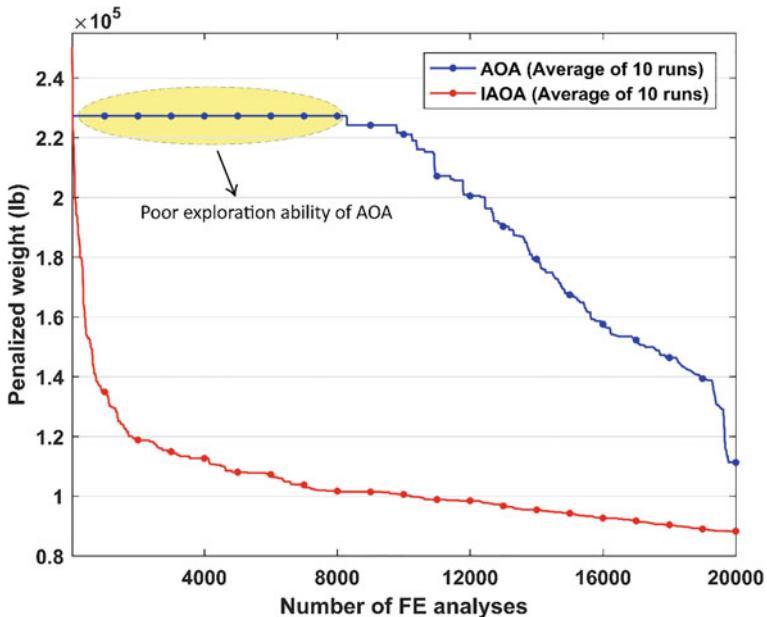


Fig. 10.20 Average weight convergence histories for the 3-bay 15-story steel frame structure

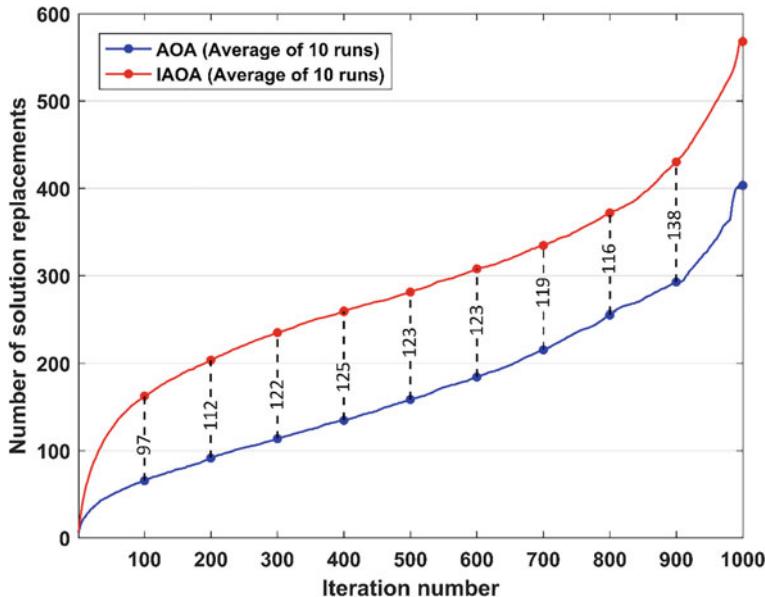


Fig. 10.21 Number of solution replacements observed in the AOA and IAOA (3-bay 15-story steel frame structure)

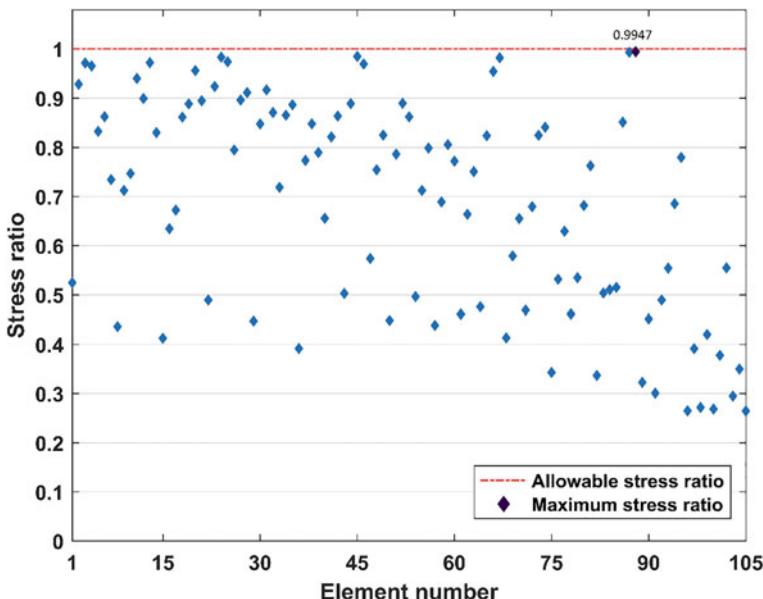


Fig. 10.22 Stress ratios of the 3-bay 15-story steel frame problem evaluated at the optimal design obtained by IAOA

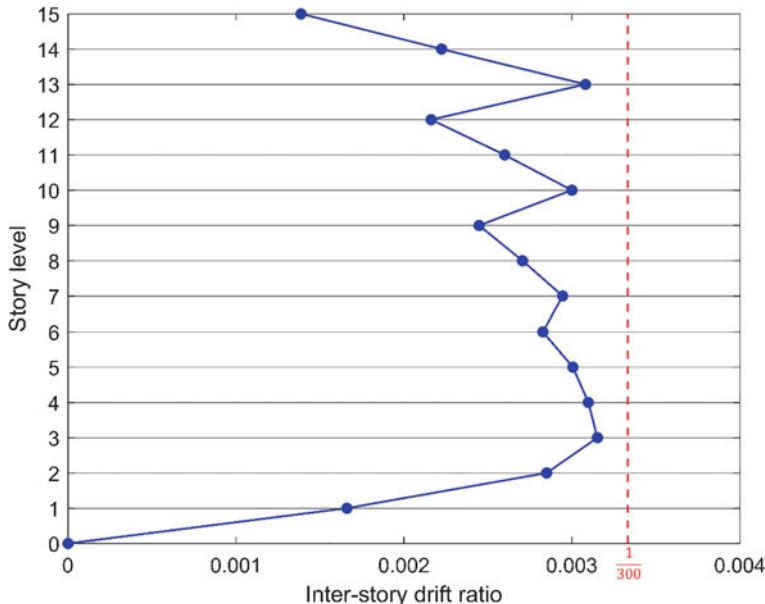


Fig. 10.23 Inter-story drift ratios of the 3-bay 15-story steel frame problem evaluated at the optimal design obtained by IAOA

the standard AOA. The best penalized weights obtained by IAOA and PSO are 52,434.5391 kg and 62,342.41 kg, respectively, which are much better than that of the standard AOA (i.e., 121,384.13 kg), showing the high effectiveness of both the PSO and IAOA methods in comparison with the standard AOA. Nevertheless, it seems that the proposed IAOA is still not strong enough to solve very high-dimensional structural optimization problems.

10.7 Concluding Remarks

This study has successfully developed an improved version of the arithmetic optimization algorithm (AOA), called IAOA, to solve discrete structural optimization problems. The arithmetic optimization algorithm (AOA) is a newly proposed meta-heuristic algorithm inspired by the distribution characteristics of the four basic arithmetic operations of subtraction, addition, division, and multiplication, and has been applied to a number of optimization problems. The standard AOA, however, due to its poor exploration capability, suffers severely from the problems of poor exploration and premature convergence to non-optimal solutions, especially when faced with high-dimensional optimization problems. The IAOA has proposed to overcome the disadvantages of the standard AOA. The major contributions of the proposed IAOA

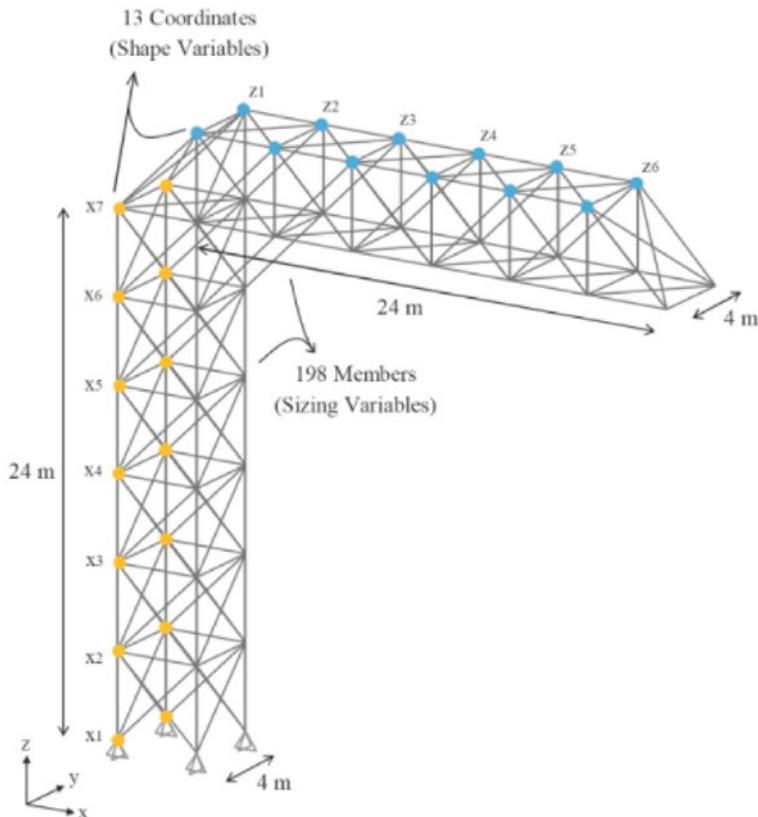


Fig. 10.24 Three-dimensional steel truss structure with 198 members

include the following points: (1) In the proposed IAOA, both exploitation and exploration phases of the standard AOA are modified to improve the convergence speed and the exploration capability. In contrast to the standard AOA in which the position update rule of the exploration phase is based on the position of the best solution found so far, the exploration phase of the proposed IAOA consists of focusing the search around the current position of solutions. Moreover, the exploitation phase of the proposed IAOA is formulated in such a way that it provides a better exploitation of the best solution found so far when compared to the standard AOA. (2) Compared to the standard AOA, the IAOA requires fewer algorithm-specific parameters, which makes it easier to implement. Finally, to demonstrate the efficiency and effectiveness of the IAOA, three benchmark discrete structural optimization problems have been examined. To our knowledge, this is the first attempt to apply the AOA for structural optimization. The results achieved by the IAOA have been compared with those of the standard AOA and other optimization algorithms existing in the literature. The results have demonstrated that the IAOA meaningfully surpasses the standard AOA

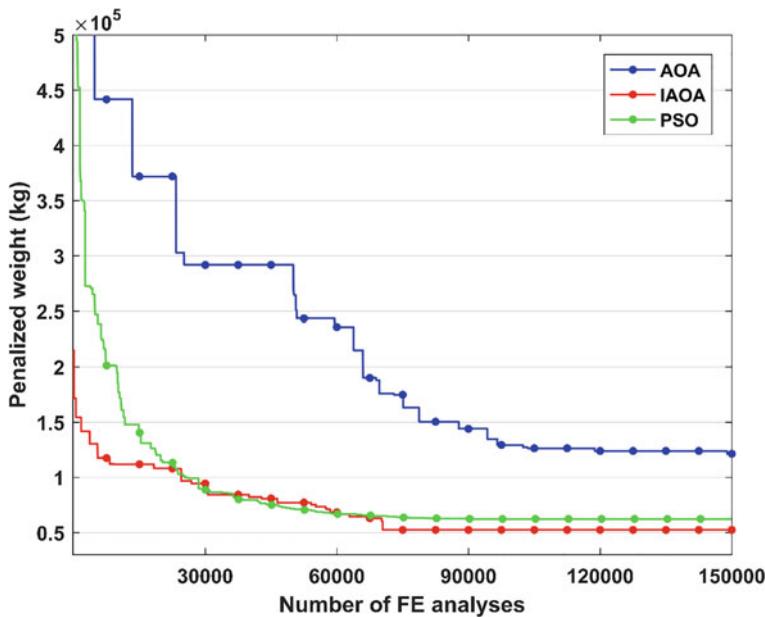


Fig. 10.25 Average weight convergence histories for the 198-bar steel truss structure obtained by PSO, AOA, and IAOA

in both terms of robustness and convergence speed. It has also been observed that the convergence characteristics of the proposed IAOA are greatly enhanced compared with the standard AOA. It has been indicated that the IAOA can effectively overcome the shortcomings of the standard AOA, such as poor exploration capability and slow convergence rate. In addition, the result comparisons with other methods in the literature have shown that the IAOA is comparable to or better than many other existing methods. Accordingly, the proposed IAOA offers a great potential to extend the applications to other types of structural optimization problems, including shells, domes, plates, etc. Nevertheless, it seems that the proposed IAOA is still not strong enough to solve very high-dimensional structural optimization problems. Therefore, it would be suggested, as future work, to further improve the performance of AOA to solve high-dimensional structural optimization problems.

References

1. Kaveh A, Biabani Hamedani K (2022) Improved arithmetic optimization algorithm and its application to discrete structural optimization. *Structures* 35:748–764. <https://doi.org/10.1016/j.istruc.2021.11.012>
2. Sivapuram R, Picelli R (2018) Topology optimization of binary structures using integer linear programming. *Finite Elem Anal Des* 139:49–61. <https://doi.org/10.1016/j.finel.2017.10.006>

3. Zhang W, Wang X, Wang Z, Yuan S (2014) Structural optimization of cylinder-crown integrated hydraulic press with hemispherical hydraulic cylinder. *Procedia Eng* 81:1663–1668. <https://doi.org/10.1016/j.proeng.2014.10.209>
4. Mirjalili S, Lewis A (2016) The whale optimization algorithm. *Adv Eng Softw* 95:51–67. <https://doi.org/10.1016/j.advengsoft.2016.01.008>
5. Sonmez M (2011) Discrete optimum design of truss structures using artificial bee colony algorithm. *Struct Multidisc Optim* 43(1):85–97. <https://doi.org/10.1007/s00158-010-0551-5>
6. Gandomi AH, Yang XS, Alavi AH (2011) Mixed variable structural optimization using firefly algorithm. *Comput Struct* 89(23–24):2325–2336. <https://doi.org/10.1016/j.compstruc.2011.08.002>
7. Dede T (2014) Application of teaching-learning-based-optimization algorithm for the discrete optimization of truss structures. *KSCE J Civil Eng* 18(6):1759–1767. <https://doi.org/10.1007/s12205-014-0553-8>
8. Azad SK, Aminbakhsh S (2021) High-dimensional optimization of large-scale steel truss structures using guided stochastic search. *Structures* 33:1439–1456. <https://doi.org/10.1016/j.istruc.2021.05.035>
9. Abualigah L, Diabat A, Mirjalili S, Abd Elaziz M, Gandomi AH (2021) The arithmetic optimization algorithm. *Comput Methods Appl Mech Eng* 376:113609. <https://doi.org/10.1016/j.cma.2020.113609>
10. Xu YP, Tan JW, Zhu DJ, Ouyang P, Taheri B (2021) Model identification of the Proton Exchange Membrane Fuel Cells by Extreme Learning Machine and a developed version of Arithmetic Optimization Algorithm. *Energy Rep* 7:2332–2342. <https://doi.org/10.1016/j.egyr.2021.04.042>
11. Abualigah L, Diabat A, Sumari P, Gandomi AH (2021) A novel evolutionary arithmetic optimization algorithm for multilevel thresholding segmentation of covid-19 ct images. *Processes* 9(7):1155. <https://doi.org/10.3390/pr9071155>
12. Stolpe M (2016) Truss optimization with discrete design variables: a critical review. *Struct Multidisc Optim* 53(2):349–374. <https://doi.org/10.1007/s00158-015-1333-x>
13. Kaveh A, Zolghadr A (2014) Democratic PSO for truss layout and size optimization with frequency constraints. *Comput Struct* 130:10–21. <https://doi.org/10.1016/j.compstruc.2013.09.002>
14. Kaveh A, Ilchi Ghazaan M (2015) A comparative study of CBO and ECBO for optimal design of skeletal structures. *Comput Struct* 153:137–147. <https://doi.org/10.1016/j.compstruc.2015.02.028>
15. Kaveh A, Bakhshpoori T (2016) An accelerated water evaporation optimization formulation for discrete optimization of skeletal structures. *Comput Struct* 177:218–228. <https://doi.org/10.1016/j.compstruc.2016.08.006>
16. Jalili S, Hosseinzadeh Y (2018) Design optimization of truss structures with continuous and discrete variables by hybrid of biogeography-based optimization and differential evolution methods. *Struct Des Tall Spec Build* 27(14):e1495. <https://doi.org/10.1002/tal.1495>
17. Le DT, Bui DK, Ngo TD, Nguyen QH, Nguyen-Xuan H (2019) A novel hybrid method combining electromagnetism-like mechanism and firefly algorithms for constrained design optimization of discrete truss structures. *Comput Struct* 212:20–42. <https://doi.org/10.1016/j.compstruc.2018.10.017>
18. Ho-Huu V, Nguyen-Thoi T, Vo-Duy T, Nguyen-Trang T (2016) An adaptive elitist differential evolution for optimization of truss structures with discrete design variables. *Comput Struct* 165:59–75. <https://doi.org/10.1016/j.compstruc.2015.11.014>
19. Kaveh A, Ilchi Ghazaan M (2018) Optimal design of double-layer barrel vault space structures. In: Meta-heuristic algorithms for optimal design of real-size structures. Springer, Cham, pp 85–99
20. Kaveh A, Mahjoubi S (2018) Optimum design of double-layer barrel vaults by lion pride optimization algorithm and a comparative study. *Structures* 13:213–229. <https://doi.org/10.1016/j.istruc.2018.01.002>
21. American Institute of Steel Construction (AISC) (2001) Manual of Steel Construction, Load & Resistance Factor Design, 3rd edn. IL, USA, Chicago

22. American Institute of Steel Construction (AISC) (1989) Manual of steel construction, Allowable Stress Design, 9th edn. IL, USA, Chicago
23. Azad SK, Hasançebi OĞ (2015) Computationally efficient discrete sizing of steel frames via guided stochastic search heuristic. Comput Struct 156:12–28. <https://doi.org/10.1016/j.compstruc.2015.04.009>
24. Dumonteil P (1992) Simple equations for effective length factors. Eng J AISC 29(3):111–115
25. https://www.brightoptimizer.com/problem_iscco2017/. Accessed 4 Nov 2021
26. American Institute of Steel Construction (AISC) (1994) Manual of Steel Construction, Load & Resistance Factor Design, 2nd edn. IL, USA, Chicago