SELÇUK UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

REPUBLIC OF TÜRKİYE

SELÇUK UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

INTEGRATION STRATEGIES OF LEVY FLIGHT TECHNIQUE INTO METAHEURISTIC ALGORITHMS: A STUDY ON SIBERIAN TIGER OPTIMIZATION ALGORITHM

Md Al Amin HOSSAIN

MASTERS THESIS

Department of Information Technologies Engineering

August-2024
KONYA

TEZ KABUL VE ONAYI

Md Al Amin HOSSAIN tarafından hazırlanan "INTEGRATION STRATEGIES OF LEVY FLIGHT TECHNIQUE INTO METAHEURISTIC ALGORITHMS: A STUDY ON SIBERIAN TIGER OPTIMIZATION ALGORITHM" adlı tez çalışması 09/08/2024 tarihinde aşağıdaki jüri tarafından oy birliği ile Selçuk Üniversitesi Fen Bilimleri Enstitüsü Bilişim Teknolojileri Mühendisliği Anabilim Dalı'nda YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Jüri Üyeleri	**İmza**
Başkan Dr. Öğr. Üyesi Ahmet YILMAZ	……..
Danışman Doç. Dr. Tahir SAĞ	……..
Üye Prof. Dr. Fatih BAŞÇİFTÇİ	……..

Yukarıdaki sonucu onaylarım.

Prof. Dr. ……. ……..
FBE Müdürü

THESIS ACCEPTANCE AND APPROVAL

The thesis titled "INTEGRATION STRATEGIES OF LEVY FLIGHT TECHNIQUE INTO METAHEURISTIC ALGORITHMS: A STUDY ON SIBERIAN TIGER OPTIMIZATION ALGORITHM" prepared by Md Al Amin HOSSAIN has been unanimously accepted as a Master's Thesis in the Department of Information Technology Engineering at the Graduate School of Natural and Applied Sciences of Selçuk University on 09/08/2024 by the jury listed below.

Jury Members	**Signature**
Chair Assist. Prof. Dr. Ahmet YILMAZ	……..
Advisor Assoc. Prof. Dr. Tahir SAĞ	……..
Member Prof. Dr. Fatih BAŞÇİFTÇİ	……..

I approve the above result.

Prof. Dr.
Director of Graduate School of Natural and Applied Sciences

TEZ BİLDİRİMİ

Bu tezdeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

DECLARATION PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Md Al Amin HOSSAIN

Date:

ÖZET

YÜKSEK LİSANS TEZİ

LEVY UÇUŞU TEKNİĞİNİN METASEZGİSEL ALGORİTMALARA ENTEGRE EDİLME STRATEJİLERİ: SİBİRYA KAPLANI OPTİMİZASYON ALGORİTMASI ÜZERİNE BİR İNCELEME

Md Al Amin HOSSAIN

Selçuk Üniversitesi Fen Bilimleri Enstitüsü
Bilişim Teknolojileri Mühendisliği Anabilim Dalı

Danışman: Doç. Dr. Tahir SAĞ

2024, 144 Sayfa

Jüri
Doç. Dr. Tahir SAĞ
Prof. Dr. Fatih BAŞÇİFTÇİ
Dr. Öğr. Üyesi Ahmet YILMAZ

Levy Uçuşu (LF), Levy dağılımı tarafından yönetilen rastgele bir yörüngedir. Bu yörüngenin büyük sıçramaları, optimizasyon teknikleri kullanılarak her olası çözümün aranmasını kolaylaştırır. Bu tez, LF tekniğinin meta-sezgisel algoritmalara entegrasyon stratejilerini araştırmış ve bu tekniklerin optimizasyon algoritmalarının performansı üzerindeki etkisini değerlendirmiştir. Çalışmanın hedefleri, üstün LF entegrasyon yaklaşımlarını keşfetmeye katkıda bulunmak ve mevcut meta-sezgisel algoritmaları geliştirmektir. Sibirya Kaplanı Optimizasyonu (STO) algoritması, Sibirya kaplanlarının avlanma davranışından esinlenmiş, küresel optimizasyon görevleri için başarılı bir meta-sezgisel algoritmadır. STO, birçok optimizasyon probleminde etkinliğini kanıtlamış olsa da, genellikle optimizasyon algoritmalarını etkileyen bazı sorunları ele almakta yetersiz kalabilir. Özellikle, çok boyutlu problemlerle arama alanının yetersiz keşfi yerel optimumlarda sıkışmaya neden olabilir ve bu da alt-optimal çözümlere erken yakınsamaya yol açar. Bu zorluklar, algoritmanın küresel optimumu bulma konusundaki güvenilirliğini ve verimliliğini azaltır. Tez, LF yönteminin küresel keşif yetenekleriyle tanınan özelliklerini, STO algoritmasına rastgele bir yerel arama süreciyle birleştirerek entegrasyonunu önermektedir. Bu tez, Levy Flight tabanlı Sibirya Kaplanı Optimizasyonu (LFSTO) dahil olmak üzere dört LF ile entegre edilmiş hibrit yaklaşımı araştırdı ve bunları entegrasyon stratejilerine göre kategorize etti. İlk yöntem, doğrudan entegrasyon tekniklerine odaklanan LFSTO-direct stratejisidir. Diğer yandan, ikinci ve üçüncü stratejiler sırasıyla LFSTO-randprob ve LFSTO-trial olarak adlandırılmış olup, rastgele değer ve deneme sayısı koşullarına odaklanmaktadır. Dördüncü yöntem olan LFSTO-trap, tuzaklama mekanizmalarına yönelik stratejilere odaklanmaktadır. Dört LFSTO stratejisinin etkinliğini değerlendirmek için, iyi bilinen CEC-2017 test fonksiyonunda kapsamlı deneyler yapıldı ve LFSTO stratejilerinin yakınsama hızı ile çözüm kalitesi analiz edildi. Sonuçlar, LFSTO-randprob stratejisinin 10, 30 ve 50 boyutlu durumlarda standart STO ve diğer üç entegrasyon stratejisinden daha iyi performans gösterdiğini ve sırasıyla ortalama sıralama değerlerinin 2.03, 2.17 ve 2.44 olduğunu göstermektedir. Bununla birlikte, geleneksel STO, ortalama sıralama değerleri sırasıyla 4.68, 4.72 ve 4.00 olan diğer dört LF tabanlı strateji arasında en kötü performansı sergilemiştir. Ayrıca, kapsamlı istatistiksel değerlendirme sonuçların tutarlılığını garanti etmiştir.

Anahtar Kelimeler: Küresel optimum, Levy Uçuşu, Metaheuristik algoritmalar, Sibirya Kaplanı Optimizasyonu

ABSTRACT

MS THESIS

INTEGRATION STRATEGIES OF LEVY FLIGHT TECHNIQUE INTO METAHEURISTIC ALGORITHMS: A STUDY ON SIBERIAN TIGER OPTIMIZATION ALGORITHM

Md Al Amin HOSSAIN

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE OF SELÇUK UNIVERSITY
THE DEGREE OF MASTER OF SCIENCE
IN INFORMATION TECHNPLOGIES ENGINEERING

Advisor: Assoc. Prof. Dr. Tahir SAĞ

2024, 144 Pages

Jury
Assoc. Prof. Dr. Tahir SAĞ
Prof. Dr. Fatih BAŞÇİFTÇİ
Assistant Prof. Dr. Ahmet YILMAZ

Levy Flight (LF) is a random trajectory that is governed by the Levy distribution. Its large hops facilitate the search for every feasible solution using optimization techniques. This thesis inquired about the LF technique's several integration strategies into metaheuristic algorithms and its impact on the performance of optimization algorithms has been assessed. The study's goals are to contribute to discovering superior LF incorporation approaches and enhancing existing metaheuristic algorithms. The Siberian Tiger Optimization (STO) algorithm, inspired by the hunting behavior of Siberian tigers, is a successful metaheuristic algorithm for global optimization tasks. While STO has demonstrated its effectiveness in many optimization problems, it may fall short in addressing some issues that generally affect optimization algorithms. Specifically, insufficient exploration of the search space in multidimensional issues can get stuck in local optima, leading to premature convergence to suboptimal solutions. These challenges decrease the reliability and efficiency of the algorithm in finding the global optimum. The dissertation proposes incorporating the LF method, renowned for its global exploration capabilities, into the STO algorithm in combination with a random local search process. This thesis investigated four LF-integrated hybrid approaches, including Levy Flight-based Siberian Tiger Optimization (LFSTO), and categorized them based on their integration strategies. The first one is the LFSTO-direct strategy, which focuses on direct integration techniques. On the other hand, the second and third strategies are named LFSTO-randprob and LFSTO-trial, respectively, and focus on the random value and trial count condition. The fourth method, called LFSTO-trap, concentrates on strategies for trapping mechanisms. To evaluate the effectiveness of four LFSTO strategies, extensive experiments were conducted on a well-known CEC-2017 test function, as well as the convergence rate of the LFSTO strategies and the quality of the solution were analyzed. The results show that the LFSTO-randprob strategy outperforms the standard STO and the other three integration strategies in the 10, 30, and 50-dimension cases, with mean rank values of 2.03, 2.17, and 2.44, respectively. However, conventional STO demonstrated the worst performance of the other four LF-based strategies by achieving the mean rank value of 4.68, 4.72, and 4.00, respectively. Furthermore, the comprehensive statistical assessment guaranteed consistency of the outcomes.

Keywords: Global optima, Levy Flight, Metaheuristic algorithms, Siberian Tiger Optimization

FOREWORD

I would like to express my gratitude to my thesis advisor, Dr. Tahir SAĞ, for guiding me with his valuable time, knowledge, and experience and for always supporting me throughout my master's studies as well as specifically my thesis.

I thank my dear family and friends for always making me feel their moral support.

Md Al Amin HOSSAIN
KONYA-2024

CONTENTS

SYMBOLS AND ABBREVIATIONS

Symbols

$\odot$	:	Dot product operator
$\oplus$	:	Entry wise multiplication product
CSV	:	Data file
Trial	:	The counter of inability to develop a solution
rand()	:	A random number between 0 and 1
gbest	:	Global best value
pbest	:	Local best value
r	:	Uniform random number between 0 and 1
p	:	Random Variable
A	:	Absolute random value between -1 to 1
3D	:	Three dimensions
2D	:	Two dimensions
B	:	Stability index
C	:	Scale parameter
Lb	:	Lower bound
Ub	:	Upper bound

Abbreviations

ABC	:	Artificial Bee Colony
ACO	:	Ant Colony Optimization
ALO	:	Antlion Optimization
AO	:	Aquila Optimizer
AP	:	Awareness Probability
SI	:	Swarm Intelligence
BA	:	Bat Algorithm
BH	:	Black Hole
CEC	:	Competition on Evolutionary Computation
CEM	:	Cross-Entropy Method
CMA-ES	:	Covariance Matrix Adaptation Evolution Strategy
CMC	:	Contraceptive Method Choice
CS	:	Cuckoo Search
CSA	:	Crow Search Algorithm
CTRW	:	Continuous-Time Random Walks
DBAL	:	Discrete Bat Algorithm with Lévy Flights
DE	:	Differential Evolution
DKP	:	Discounted Knapsack Problem
EA	:	Evolutionary Algorithms
ED	:	Euclidean Distance
ES	:	Evolution Strategies
FA	:	Firefly Algorithm
FLFBA	:	Fractional Levy Flight Bat Algorithm
FS	:	Feature Selection
GA	:	Genetic Algorithms
GOA	:	Grasshopper Optimization Algorithm
GS	:	Greedy Selection

GSA	:	Gravitational Search Algorithm
GWO	:	Grey Wolf Optimizer
HA	:	Hill-climbing Algorithm
HMOMFO	:	Hybrid Multi-Objective Moth Flame Optimization
IEHO	:	Improved Elephant Herding Optimization
IKH	:	Improved Krill Herd
KH	:	Krill Herd
LF	:	Levy Flight
LFA	:	Levy Flight Firefly Algorithm
LFD	:	Lévy Flight Distribution
LFPSO	:	Levy Flight Particle Swarm Optimization
LFSS	:	LF-inspired Local Search
LFSTO	:	Levy Flight Siberian Optimization Algorithm
LJA	:	Levy flight Jaya Algorithm
LLF	:	Local Levy Flight
LS	:	Local Search
MBO	:	Monarch Butterfly Optimization
MCSO	:	Modified Chicken Swarm Optimization
MFO	:	Moth Flame Optimization
MH	:	Metaheuristic
OBL	:	Opposition-based Learning
OL	:	Orthogonal Learning
PDF	:	Probability Density Function
PhA	:	Physics-based Algorithms
PID	:	Proportional Derivative Integral
PSO	:	Particle Swarm Optimization
ROA-LF	:	Remora Optimization Algorithm-Levy Flight
SCA	:	Sine-Cosine Algorithm
SHO	:	Selfish Herd Optimizer
SMA	:	Slime Mould Algorithm
SOA	:	Seagull Optimization Algorithm
SSA	:	Salp Swarm Algorithm
STO	:	Siberian Tiger Optimization
TS	:	Tabu Search
TSP	:	Traveling Salesman Problem
WOA	:	Whale Optimization Algorithm
WSN	:	Wireless Sensor Network

1. INTRODUCTION

1.1. Overview

Optimization techniques attempt to overcome common challenges such as a wide variety of problem types, large number of user-defined parameter settings, and complex calculations. In such scenarios, effectively finding solutions that minimize or maximize the specified objective function can be quite challenging (Iacca et al., 2021). In recent decades, a large number of evolutionary algorithms (EA) have emerged to solve many engineering optimization issues, most of which are extremely chaotic with various design parameters and complicated constraints. Nowadays, metaheuristic (MH) algorithms are widely used to solve real-world problems due to their minimal time consumption and global search capabilities (Jensi and Jiji, 2016). Furthermore, MH approaches have attracted much attention and have been used to address various optimization problems. EA, Swarm Intelligence (SI) algorithms, Physics-based Algorithms, and human-based algorithms are the four broad groups into which we may divide MH algorithms (Abualigah et al., 2021). Among these, Genetic Algorithms (GA) (Holland, 1992) Artificial Bee Colony (ABC) (Karaboga and Basturk, 2007), Particle Swarm Optimization (PSO) (Kennedy, 1995), Ant Colony Optimization (ACO) (Dorigo, 1996), Firefly Algorithm (FA) (Yang, 2010b), Grey Wolf Optimizer (GWO) (Mirjalili et al., 2014), etc. are the most popular.

Several optimization techniques have been created to minimize computing costs and enhance the overall efficiency of diverse frameworks. Nevertheless, there are a number of drawbacks and restrictions with conventional optimization techniques, such as their tendency to converge to local optima and unidentified exploration areas. Furthermore, their solution is limited to a single basis (Hashim et al., 2019; Sağ, 2022). In order to address these deficiencies, several novel optimization techniques have been put out lately. Different challenges were subsequently solved using them.

The search procedure, typically consisting of two separate stages: the initial one referred to as broadening the search space (exploration) and the subsequent stage referred to as greater intensification (exploitation), is a conventional characteristic shared by all of these different MH algorithms. The MH method creates stochastic generators for exploring various exploration areas in the initial stage (Salcedo-Sanz, 2016; Abualigah et al., 2020). The optimization approach looks for the best option inside the search space in

the subsequent stage. However, these two-phase mechanisms generally face local optimality problems. To prevent being stuck at the local optimal points, an effective MH optimization algorithm must strike an equilibrium between the exploration and exploitation.

To address this issue of local minima, various techniques are employed, such as hybridizing different algorithm methods, orthogonal learning, parameter adaptation, using different location update strategies, etc. Among these, the techniques depending on Levy Flight (LF) distribution (Kamaruzaman et al., 2013), are one of the most useful approaches to improve the traditional MH algorithms. From past LF-related studies, we can see that the authors integrate LF distribution into various MH in several ways to improve standard MH algorithms. Mainly, LF is a random walk search method that has powerful characteristics to significantly avoid local optima and jump through global space with fast convergence speed.

Through various integrations of LF, this study aims to present more efficient MH optimization methods. This research enhances the Siberian Tiger Optimization (STO) algorithm that is a conventional swarm-based MH optimization technique. The STO algorithm is inspired by the hunting and fighting behavior of Siberian tigers (Trojovsky et al., 2022). This study investigated the various enhancing strategies of the STO algorithm via LF techniques and the proposed algorithms named by the Levy Flight Siberian Optimization Algorithm (LFSTO). Moreover, CEC-2017 benchmark functions are utilized to assess how well the presented integration strategies work.

1.2. Thesis Objectives and Purposes

This research indicates that adding LF to the STO algorithm and stating some strategies to combine them to make LFSTO could greatly improve the algorithm's speed of convergence, quality of solutions, and stability for the global optimization problem. The foundation of study lies in the idea that incorporating LF, a well-known stochastic technique for global exploration, into current optimization approaches can result in significant performance improvements. To look into this assumption, a set of three research questions was initially formed: (i) Does LF speed up optimization algorithm convergence relative to its non-levy counterparts? (ii) Does combining LF improve optimization algorithm solutions to a variety of benchmark problems? (iii) Which levy

integration strategy is most effective, and can LF-enriched algorithms scale better for multifaceted optimization tasks?

Purpose: The primary purpose of this research is to advance the field of global optimization by introducing an innovative hybrid algorithm, LFSTO, which combines the predatory instincts of Siberian tigers with the mathematical capabilities of Levy Flight. The motive of this combination is to circumvent the shortcomings of STO, which include rapid convergence and poor exploration of the solution space, respectively. Specifically, the research aims to assess how the incorporation of Levy Flight can enhance the efficiency and performance of these algorithms, ultimately contributing to the advancement of optimization methodologies.

Objectives: The following objectives are meant to guide the research towards achieving measurable, realistic, and doable results throughout the thesis, proving or disproving the hypothesis that adding LF to optimization algorithms makes them work better. The first goal is to look at how LF can be theoretically and practically integrated into STO using various strategies. This includes sketching out the main ideas and figuring out how LF's unique functions can work with metaheuristic algorithms, in this case, STO's search strategies. The second goal is to design, implement, and fine-tune the LF-integrated STO algorithm, focusing on optimizing algorithmic parameters and control mechanisms to ensure superior performance. The third motive is to prove that the LFSTO algorithm works by testing it on a large number of different benchmark optimization problems and discovering the best LF integration method that works well enough to overcome both local and global optima.

The ultimate goal is to use statistical experiments to determine the implications of the data gained through validation via observation. This phenomenon will determine if the performance enhancements from LF integration techniques are statistically significant as compared with regular STO. The study's premise posits that LFSTO integration strategies will outperform STO techniques in enhancing global efficiency. The objectives include conceptual creation, algorithm implementation, validation through observation, and statistical assessment to confirm the hypothesis and enhance optimization.

1.3. Organization of the Thesis

This thesis consists of eight chapters with an abstract, symbols, abbreviations, references, and an appendix section. The background, hypothesis, objectives, and

purposes of the thesis are discussed in detail in the first chapter. Chapter-2 presents the literature studies and related works of LF in MH algorithms as well as the motivation of this thesis. Chapter-3 focuses on optimization, including the mathematical model, exploration and exploitation concept, optima concept, functioning of optimization, types, and taxonomy of optimization. The background, mathematical model, features, and functional model of the LF are described in Chapter-4. Chapter-5 is composed of the STO algorithm with an overview, mathematical model, and operational framework sections. Chapter-6 elucidates the materials and method of this thesis, wherein all the proposed strategies named LFSTO-direct, LFSTO-randprob, LFSTO-trial, and LFSTO-trap are illustrated with pseudocode and flowchart. Parameter settings, experimental results, findings discussion, and convergent graph are given in Chapter-7. Last but not least, a summary of the conclusions and suggestions after conducting this thesis is explained in Chapter-8.

2. LITERATURE REVIEW

This thesis explores LF, a stochastic process known for improving several optimization algorithms. This section of the literature review covers some studies that employed strategic LF in metaheuristics and nature-inspired algorithms. To mitigate premature convergence and inadequate exploration, these combinations were explored. The study underscores a range of optimization techniques and elucidates how LF has enhanced its proficiency and efficacy. This detailed analysis of LF's optimization applications prepares for the intended study, which integrates LF into the STO method to improve globally optimized performance.

An unconventional probabilistic optimization issue was addressed under the investigation of Pavlyukevich (Pavlyukevich, 2007) by using simulated annealing with LF of the parameter stability index. This method utilized the distinct large leaps of the LF mechanism to incorporate a non-local search element into the optimization procedure. As a crucial component of the method, the LF drives non-local exploration. By shaping the course of the search, LF was especially vital to the randomized system's mechanics. The unexplored prospective landscape's ground state was quickly constrained by the polynomial rate of temperature reduction.

Yang and Deb (Yang and Deb, 2009) introduced Cuckoo Search (CS), a new metaheuristic technique that draws inspiration from cuckoos' essential offspring parasite activity as well as LF actions regarding birds and fruit fly species. The CS algorithm integrates LF during the time it generates new cuckoo solutions. For every cuckoo's novel solution, LF determines the step length and trajectory. By allowing cuckoos to carry out steps with different durations and orientations, this randomized method, under the direction of LF, adds unpredictability to the algorithm. To keep the algorithm from being stuck in local optima and to help find optimum solutions, this method combines local exploration within currently available solutions with global exploration throughout the area for searching.

Xin-She Yang (Yang, 2010a) utilized LF with the Firefly Algorithm to create the LF Firefly Algorithm (LFA) as a unique optimization strategy. LFA works on attraction and exploration. In a d-dimensional space, the LFA algorithm allows fireflies to explore by moving toward a brighter firefly when its light intensity exceeds another. The relative distance between fireflies controls their attraction through progressive decline. This method is vital for assessing novel solutions and upgrading light intensity. Fireflies attract

others based on their brightness and distance. LFA's randomness allows fireflies to delve into the searching region by carrying out occasional lengthy jumps, thus offering a promising optimization method.

The authors (Wang et al., 2013) suggested adding an LF feature to the Krill Herd (KH) method to optimize it. This led to the creation of the LF Krill Herd (LKH) algorithm. This innovative LKH method primarily integrates LF during the local search phase to enhance its balance between exploration and exploitation. In their proposed method, the Levy-flight integration within the KH algorithm occurs during the fine-tuning of individual krill positions. After the krill individuals undergo various motions, the Local LF (LLF) operator is applied to refine the positions in the search space, allowing for a more focused local search towards the end of each iteration.

Haklı and Uğuz (Haklı and Uğuz, 2014) introduced the Levy flight particle swarm optimization (LFPSO) algorithm to improve the limitations of traditional PSO. Their approach includes two critical changes. First, each particle is assigned a limit value, which increases if the particle fails to improve during an iteration. Second, particles exceeding this limit are redistributed using Levy distribution. LFPSO's operation begins with random particle distribution within the search space. Fitness values are calculated, and personal best (pbest) and global best (gbest) values are determined. If a particle surpasses its limit, it's redistributed through LF, allowing for long jumps. After redistribution, fitness is calculated, and pbest and gbest are updated. They added a limit value for particles and used adaptive LF with a randomly chosen parameter, which are the main novelties of LFPSO's method. These modifications enhance global search capabilities and solution quality.

An important improvement in their investigation (Guo et al., 2014) is the incorporation of LF into the Improved Krill Herd (IKH) algorithm. The new improved approach integrates LF into the IKH scheme during the highest-level Krill probing movement, leading to better candidate solutions. A unique information-sharing method has also been implemented among the highest-level krill, allowing for a quicker convergence to optimal solutions by establishing a group of some of the best-performing krill for collaborative learning. The krill can more effectively coordinate their movements and hunt for food because of this collaboration process.

According to the studies of Sharma et al. (Sharma et al., 2015), an interesting addition to the Artificial Bee Colony (ABC) optimization method is the incorporation of LF. The investigation and mining strengths of the algorithm are greatly improved by this

innovative method, which is called LF features-based ABC (LFABC). In the LF-inspired Local Search (LFSS) stage, LF is included in the LFABC algorithm by determining a step size based on LF principles. To keep the merging pace, particularly at elevated step sizes, the methodology updates the best solution inside the population using this step size. This prevents the algorithm from bypassing the genuine solution.

With the goal of outperforming LFPSO and other PSO variations, Jensi and Jiji (Jensi and Jiji, 2016) improved the LFPSO approach to create the PSOLF algorithm. The main modification made to the PSOLF strategy is the LF approach, which changes the particle speed and fixes problems with too much convergence and local optima that happen in standard PSO. The procedure includes determining the pbest and gbest values, evaluating fitness, and distributing random particles. Random probabilities govern the updates of particle position and velocity. The LF technique is used when the random value is smaller than 0.5, offering particles to perform large jumps in the direction of pbest and gbest. This encourages global search space exploration and variety in the swarm.

In their study, Heidari and Pahlavani (Heidari and Pahlavani, 2017) suggested an improved iteration of the Grey Wolf Optimizer (GWO) known as the LF-based GWO (LGWO) to overcome the challenges of the standard GWO method. The traditional GWO algorithm, while efficient in various scenarios, occasionally suffers from stagnation at local optima due to inadequate diversity among the wolf population. To combat these issues, LGWO incorporates Lévy Flight (LF) and a Greedy Selection (GS) strategy, primarily in the hunting phases, to improve its global search efficiency and mitigate stagnation problems. While maintaining a social hierarchy of alpha (α), beta (β), and omega (ω) wolves, LGWO eliminates the distinct role of delta wolves in various phases of the hunting process. This reconfiguration ensures a more balanced population structure and more effective searching. LF introduces randomness with the inclusion of a random β parameter, allowing for both small and long-distance jumps, thus balancing exploration and exploitation tendencies during the search.

Ling et al. (Ling et al., 2017) improved the LF trajectory-based whale optimization algorithm (LWOA) by integrating LF with WOA. The integration procedure comprises LF-affecting humpback whale location updates in WOA, dependent on several circumstances. Random variable p (range 0 to 1) less than 0.5 determines the fact that to modify wolf positions via LF and a cosine function. Also, the updating method is refined by adjusting the procedure depending on the values of the parameter `A` if the absolute

random value of A (range within -1 and 1) is less than 1. These expressions of conditionality aid GWO in constantly determining update techniques.

Feng and Wang (Feng and Wang, 2018) provide 10 new binary Moth Search methods (MS1–MS9) to solve the discounted knapsack problem (DKP), an enlarged variant of the conventional 0–1 knapsack problem. They look at how LF and fly-straight operators change the Moth Search (MS) algorithm and suggest new ways to use MS that are based on mutation operators. In this study, the whole population divided into two subpopulation state and LF is utilized to update the populations (moths) in subpopulation 1 state. Experimental results from 30 DKP cases show that the suggested approaches outperform the basic MS in solution quality and computing efficiency. MS1 performs best, with 12 best and 11 second-best values across all cases. The study's mutation operator findings may be useful for restricted knapsack and combination optimization issues.

Combining LF with the black hole (BH) technique, Abdulwahab et al. (Abdulwahab et al., 2019) present a new approach to data clustering and optimization. An excessively harsh generation of random step sizes by LF in this implementation can cause solutions to be found outside of the search region. To deal with this, a coefficient of 0.01 is employed to minimize step sizes as they get larger. When the BH algorithm is generating step sizes, LF is incorporated into it. The aforementioned step sizes replace the random number generation variable in the standard BH method for updating the star locations, which indicate solutions. The Levy distribution determines these step sizes, which improve the BH algorithm's global search capability and direct the stars as they explore the search region.

Emary et al. (Emary et al., 2019) showed a new LF-based improvement for the sine-cosine algorithm (SCA) and WOA that helps with inertia and harmony. This version enhances the investigation and utilization by substituting LF for some random motions in both optimization tools. The LF-based variations demonstrated better exploring in difficult search areas with many minima, and their usefulness was confirmed by testing them in several benchmark tasks. The suggested changes also guaranteed numerically substantial gains in efficiency while lowering the probability of inertia and early convergence.

Bejarbaneh et al. (Bejarbaneh et al., 2020) introduced a new method called a hybrid PSO algorithm that combines the SCA and LF. Two separate approaches to durable control methods that address nonlinear scenarios were derived from this hybrid

paradigm. In this modified algorithm, first check if the trial value is small or large than the specified limit value; if the trial value is greater than the limit, then check if the random value is high or low than 0.5; if the value is low, then perform a levy-based sine function; else, perform a levy-based cosine function. This work provides a new approach for governing system layout that shows promise in dealing with nonlinear system complexity.

Taking ideas from LF random walks, Houssein et al. (Houssein et al., 2020) proposed a new metaheuristic technique called Lévy flight distribution (LFD) to deal with practical optimization problems. Initially, the LFD algorithm estimates the Euclidean distance (ED) between the first two closest agents. Once the agents have completed a predetermined number of repetitions, the distance ED is then compared with a specified threshold. If the resulting distance is below the threshold, the algorithm adjusts agent placements using the LF-derived step length. There was another contrast between the random value (R) and the comparative scalar value (CSV = 0.5). The method uses LF-based location updates if CSV is greater than R; otherwise, it uses the usual location update function. Compared to popular metaheuristic algorithms, it performs better. Wireless sensor network (WSN) problems that are particularly challenging to resolve are also well-suited to the LFD method. According to the research, the LFD strategy is convergent, effective at probing and profiting, and reliable at eliminating local optima.

In his research, Mohammed Alweshah (Alweshah, 2020) used the Monarch Butterfly Optimization (MBO) algorithm to solve a feature selection (FS) dilemma. There are two changes made to the MBO: first, it employs an improved crossover operator; second, it adds the LF distribution to accelerate convergence. When MBO was combined with a modified crossover operator, classification accuracy went up to 93% across all datasets, and the FS length went down in 25 standard datasets. This study improves FS approaches for the Internet of Things, picture splitting, and sentiment analysis, contributing to data extraction and machine learning.

In their study, Wu et al. (Wu et al., 2020) introduced a finite element-based scheme update approach using the Crow Search Algorithm with LF (CSALF). Using the CSA as its foundation, crows in the LFCSA perform iterative searches for improved food sources. LF enhances the algorithm's optimization efficiency by substituting traditional random flight in CSA scenarios. Utilizing the features of LF, the crows' new positions are determined, offering both short- and long-distance navigation inside the search area. A crow (population) changes its location via the usual CSA approach to follow another crow if the random value (between 0 and 1) is equal to or greater than the awareness probability

(AP). If the random value is smaller than AP, the initial crow uses LF techniques to move. This LF combination has an impact on Crows' AP-based search space exploration.

To improve the Selfish Herd Optimizer's (SHO) global optimization features, Zhao et al. (Zhao et al., 2020) presented the Selfish Herd Optimizer with an LF Distribution Strategy (LFSHO). SHO, while theoretically robust, is prone to premature convergence and suboptimal solutions. LFSHO addresses these limitations by incorporating Levy-flight distribution. In LFSHO, LF is applied to update the prey leader's position, which significantly influences global search. By adding LF operations, the algorithm avoids stagnation and escapes local optima. The updated formula for the leader's position incorporates LF operations and imitates prey-fleeing behavior when under predator attack. This approach expands solution space exploration and enhances global search, ultimately leading to improved candidate solutions.

Boudjemaa et al. (Boudjemaa et al., 2020) created the fractional LF bat algorithm (FLFBA) by combining fractional calculus with a local search process based on Levy distribution. The goal of FLFBA is to make the bat algorithm (BA) better for global optimization. The FLFBA method creates a population of random places and assesses the objective function at every position to get the primary global optimal solution. Then, to boost solution variety and exploration, the population is split into two distinct sections. Using LF-based seeking and contrasting between two randomly chosen local optimal solutions, one partition produces new locations. By emphasizing exploration, this method aids in the algorithm's escape from local optima. The second division simulates continuous-time random walks (CTRW) by using a mix of fractional calculus and differential evolution for velocity updates. This strategy diversifies learning sessions for every individual.

The study carried out by Ba et al. (Ba et al., 2020) improved the Antlion Optimization (ALO) algorithm. It is more effective on challenging optimization issues and is known as the orthogonal learning (OL) levy ALO approach. With this enhanced version, the ALO algorithm incorporates the LF distribution approach. By comparing the recently updated Ant positions, which the LF distribution determines, the LF-incorporated strategy selects the best candidate solution for the following iteration. The ALO algorithm's searching ability is increased, and the risk of local optima in a state of being is significantly reduced by the addition of LF.

To strive to improve ACO methods for handling complicated stochastic optimization issues, Liu et al. (Liu et al., 2020) present an approach called "greedy-Levy

Ant Colony Optimization (ACO)" that combines the epsilon greedy and LF methods. LF is mostly integrated into the algorithm's candidate selection procedure. LF is used when the ACO strategy chooses a candidate node for the subsequent optimizing phase. LF adjusts the opportunity of selecting candidate nodes according to aspects such as step duration, shifting ratio, and LF threshold. The technique takes advantage of LF's distinctive features, namely its fat-tailed distribution, to maximize the choice of candidates in a manner that strikes a compromise between exploring and exploiting. Empirical investigations show that the ACO method performs more effectively when addressing complicated optimization issues after this incorporation.

A modified Slime Mould Algorithm (SMA) called LF-SMA was offered by Cui et al. (Cui et al., 2020) that incorporates LF to improve its global exploratory skills. In particular, the method for modifying particle locations reveals the combination of LF. The new position formulation utilizes LF to provide random step lengths, which may improve global search performance and reduce local optima. The factor flight range controls LF in the LF-SMA inclusion to adjust particle movement lengths. In essence, it regulates the size of the exploring step lengths, which affects how well the algorithm searches the solution space. In addition, the Hadamard product is employed to execute element-wise multiplications across matrices or vectors, which affects whether the LF optimizes particle positions.

The authors of the study, Ingle and Jatoth (Ingle and Jatoth, 2020), came up with a new way to solve the problems of local optima and population diversity by adding LF and a greedy selection scheme to the JAYA algorithm. A lot of simulations and parameter sensitivity tests show that the proposed JAYA algorithm with LF is much better than other JAYA algorithms and variants when it comes to solution quality, convergence speed, and robustness, especially when it comes to non-linear channel equalization. While successful, there is still potential for further refining the algorithm to achieve optimal solutions.

Bandopadhyay and Roy (Bandopadhyay and Roy, 2020) developed the hybrid multi-objective moth flame optimization (HMOMFO) algorithm. They coupled the particle swarm optimization (PSO) algorithm with the Lévy flying technique. HMOMFO was designed to enhance both the searching and exploitation capabilities of the organization.

Introducing a novel approach, researchers Saji and Barkatou (Saji and Barkatou, 2021) present the Discrete Bat Algorithm with Lévy Flights (DBAL) to tackle the

challenging Traveling Salesman Problem (TSP). By using LF-inspired random walks along with a neutral crossover operator, the DBAL algorithm makes it easier to find, use, and stay away from local minima. A lot of testing on different benchmark datasets and close comparisons with eight different techniques are shown in the study to show that DBAL consistently delivers better solution quality than competitors. It looks like the proposed DBAL could be a good way to deal with the TSP's complexity, as shown by how well it did in both parametric and non-parametric statistical tests compared to other methods.

An enhanced version of the salp swarm algorithm (SSA) has been suggested by Nautiyal et al. (Nautiyal et al., 2021), and it relies on Gaussian, Cauchy, and Lévy motion mutation strategies. The Gaussian mutation is used to improve the ability to use information about the neighborhood; the Cauchy mutation is used to improve the ability to search globally; and the LF mutation is used to improve the level of randomness in the search operation.

Iacca et al. (Iacca et al., 2021) improved the efficiency of the Jaya optimization method by including LF in it. They observed that LF, which resembles random movement with "massive" steps periodically, is remarkable at promoting tremendous "jumps" in the exploration process. They suggested an alteration to the genuine Jaya method to address the problem of local optima grabbing the search. To modify the algorithm's length of steps, LF was employed to sample random values rather than uniformly distributed ones. This novel method called the "LF Jaya algorithm" (LJA), enabled particles to carry out neighborhood searches along with, on certain occasions, "jumping" to other parts of the exploration area.

When used to solve complex planning problems, it has trouble exploring. Deepa and Venkataramann's research (Deepa and Venkataraman, 2021) found a novel approach to this problem; they termed it the LF technique with the WOA. It enhances coverage optimization in WSN. The use of LF allows exploratory factors to modify their positions in order to alter their search positions. This update procedure is carried out under certain circumstances, as determined by the values of parameters like the probability factor and coefficient vector. LF uses a Levy distribution and a random integer to introduce randomization. The quest for the agent's position is subject to both little alterations and sometimes significant leaps as a result of this unpredictability.

As mentioned by Singh et. al. (Singh et al., 2021), the Improved Elephant Herding Optimization (IEHO) algorithm combines LF with clan (kin) updating factors to boost

local and global search functionality. A step size adjuster is used to govern the amount of effect that LF has on the original positions, and LF characteristics are employed for updating the positions of clan individuals. This combination takes place while the position is being updated, and LF serves to explore and add randomness to find the best solution. The approach highlights the use of LF in the algorithm's clan update operators, highlighting its function in meeting both local and global search requirements.

Abu Khurma et al. (Khurma et al., 2021) developed a new method for efficient feature selection in medical diagnostics through the combination of the Moth Flame Optimization (MFO) algorithm with the LF function. A significant role of the LF function is to reduce the possibility of the MFO becoming stuck in a local minimum and foster its exploratory behavior. In particular, LF is applied to subsequent moth location updates, permitting every upgraded moth to execute an LF once and increasing the search space's degree of unpredictability. Feature selection in difficult medical diagnostic issues relies on the MFO's capacity to include multiple solutions and avoid local minima; its incorporation improves that capability.

Abualigah et al. (Abualigah et al., 2021) proposed the Aquila Optimizer (AO), a novel population-based optimization technique. It takes its cues from the way Aquila birds hunt. In particular, the algorithm uses LF and four Aquila-inspired hunting approaches to optimize. These strategies involve the highest flight with a vertical descent, contour flying with a quick-ride attack, lower flight with a gradual downhill assault, as well as navigating and capturing prey. The technique integrates LF into confined exploration (X2) and exploitation (X4) stages. The scheme employs LF to explore the target prey's specified region in X2 and exploit it in its ultimate position in X4. The LF improves solution quality by balancing exploration and exploitation during optimization using Aquila's hunting behaviors.

By adding LF and mutation operators, Ewees et al. (Ewees et al., 2022) presented a modified variant of the Seagull Optimization Algorithm (SOA) called Improved SOA (ISOA). The ISOA methodology entails updating solutions by calculating LF at each iteration and then adding a mutation operator during the initial 25% of iterations. The mutation operator enhances exploration and exploitation trade-offs, contributing to efficient convergence. Fitness evaluation is based on classification error and the number of selected features, with a balancing factor to control the trade-off. These steps are repeated iteratively until a stop condition is met. Various experiments have shown that

adding the LF and mutation operators to ISOA makes its search much faster and more precise.

Aghaee et al. (Aghaee et al., 2022) provided an enhanced genetic algorithm (GA) based on the Lévy walk to select ensembles in a semi-supervised manner. The findings from all of the studies reveal that the strategy that was presented has been successful when there have been adequate instruction instances, and the enhancement in accuracy has reached close to 11% in several of the experiments that were conducted.

Peng and Zhang (Peng and Zhang, 2022) devised a revolutionary light firefly approach that they named the LF Firefly Algorithm (FAFA) in order to discover the optimal thresholds for multilevel thresholding in pictorial segmentation. They did this by expanding the Rényi entropy. A Lévy feature-based adaptive parameter approach contributes to an improvement in the general efficacy of the FAFA.

Verma et al. (Verma et al., 2023) addressed local maxima and preliminary convergence by including LF in their Modified Chicken Swarm Optimization (MCSO) algorithm. LF assists roosters, hens, and chicks in updating their CSO positions when they face local optima or cannot find an optimal path. LF adds fluctuation and versatility to their movements. LF allows a chicken to adapt and jump long distances in the search space when confined to local optima. In MCSO, position updates are done by LF when candidate solutions are trapped in no neighboring (local) solutions. This random exploration helps hens escape local optima and find superior solutions, thereby boosting the algorithm's efficiency.

According to Abdullah Mujawib Alashjaee (Mujawib Alashjaee, 2023), LF inclusion is crucial to identifying breaches, particularly when undertaking training. In his study, LF improves the Remora Optimization Algorithm-Levy Flight's (ROA-LF) ability to explore, keeping it from getting stuck in local optima and making it easier to choose features for intrusion detection. In this phase, LF is used to iteratively update Romera positions within predefined limits, extending ROA's searchability. If Romera’s previous position is better than the updated positions, then the Whale position update is implemented. Otherwise, the Sailfish position update is implemented. To check if the update is optimal or not, a small step is taken in the test direction. The testing phase leverages the features selected during training for classifier performance evaluation, using a threshold of 0.5 to determine the optimal feature subset. The fitness value of Romera’s position and the newly tested position is calculated. If the new test direction results in a larger fitness value, then host feeding is implemented. If the test direction provides a

smaller fitness value, then Romera’s position is updated using LF. LF is introduced into the exploration phase of the ROA to enhance its exploration ability further.

The LF-incorporated Gravitational Search Algorithm (LevyGSA), created by the author (Joshi, 2023), mainly incorporates LF into its agents' location updating techniques. Both elite and non-elite agents use LF when they alter their positions inside LevyGSA. The ideal solution is made more accurate by updating the locations of elite agents. In the meantime, LF impacts the actions of non-elite agents in a roundabout way through the mechanisms that maintain equilibrium between the algorithm's exploration and exploitation dynamics. LF is added to position update techniques to improve elite agents' interior search and non-elite exploration-exploitation trade-offs. All things considered; this will significantly alter the behavior of LevyGSA searches.

To solve the shortcomings of the traditional Grasshopper Optimization Algorithm (GOA), such as limited global search efficiency and accuracy, Wu et al. (Wu et al., 2023) suggested an improved version of the GOA called the Levy Flight Grasshopper Optimization Algorithm (LFGOA). Regular GOA incorporates levy distribution by substituting uniformly distributed random integers (between 0 and 1), with the exception of the first population in the position initialization stage. Moreover, the new location is determined by multiplying the present location by the levy function. Their developed algorithm demonstrated increased effectiveness in resolving optimization issues by striking a more effective equilibrium between exploration and exploitation. The results of comparison experiments, including benchmark test functionality and reality-based engineering challenges in the real world, will prove this.

A comprehensive review of prior studies has demonstrated LF's significance and value in optimization. Table 1 presents a summary of several reviewed studies. An oversight in the research exists because no freely available program exploits LF throughout optimization techniques. This study tackles the vital requirement of finding out LF's influence on optimization algorithms. This study bridges the disparity with conceptual as well as experimental optimization remarks, demonstrating its innovation. An open-source framework for discovering LF will allow investigators from many domains to use and assess its possibilities.

Table 1. A brief description of several reviewed studies.

Sl. No.	Algorithm	LF Integration Strategy	The Employed Benchmark Test Function	Ref.
1	Simulated Annealing	LF particles are incorporated into the non-local (global) exploration stage with a stability index parameter that depends on the current position of the particles.	Numerical examples (two-dimensional potential function, Shekel function, Hartman function).	(Pavlyukevich, 2007)
2	Cuckoo Search	The Cuckoo search algorithm's original version used the LF directly in the first step of the algorithm when generating new solutions.	Various standard test functions (Michalewicz's, Schwefel's, Rosenbrock's, De Jong's, Ackley's, Rastrigin's, Griewank's, Easom's, and Shubert's).	(Yang and Deb, 2009)
3	Firefly Algorithm	The Levy distribution is element-wise multiplied directly by the standard firefly movement state.	Various standard test functions (Michalewicz's, Schwefel's, Rosenbrock's, De Jong's, Ackley's, Rastrigin's, Griewank's, Easom's, and Shubert's).	(Yang, 2010a)
4	Krill Herd	Local LF operators based on random walking are integrated only into the exploitation phase of the original KH algorithm.	Fourteen high dimensional complex benchmark functions.	(Wang et al., 2013)
5	Particle Swarm Optimization	In this approach, firstly, each particle is assigned a threshold value that increases if it fails to improve over a specified trial count. Secondly, when this threshold exceeds the trial count, particles are redistributed using the Levy distribution. The Levy distribution is directly element-wise multiplied into the particle velocity update state.	Twenty-one well-known benchmark functions.	(Haklı and Uğuz, 2014)
6	Krill Herd	Levy method added to the original KH highest-level Krill probing movement stage when generating new Krill.	Fourteen different benchmark functions.	(Guo et al., 2014)
7	Artificial Bee Colony	Standard ABC has three steps. However, in this study, LFABC first implemented an LF-inspired local search strategy and then employed this in basic ABC as a fourth step to enhance the exploitational capability of normal ABC.	Twenty test problems of different complexities and five real-world engineering optimization problems.	(Sharma et al., 2015)
8	Particle Swarm Optimization	Check if the random distribution is less than 0.5, then integrate LF via the element-by-element multiplication process to the position and velocity update stage.	Twenty-one different dimensional benchmark test functions.	(Jensi and Jiji, 2016)

Sl. No.	Algorithm	LF Integration Strategy	The Employed Benchmark Test Function	Ref.
9	Grey Wolf Optimizer	LF is directly incorporated in the position update stage of hunting phases.	Twenty-nine unconstrained problems from CEC-2005, thirty modern benchmarks from CEC-2014, and fourteen real-world test cases from CEC-2011	(Heidari and Pahlavani, 2017)
10	Whale Optimization Algorithm	When the random value is less than 0.5 and the absolute random value is less than 1, then LF is entrywise multiplied to update the humpback whale location.	Twenty-three benchmarks test functions and infinite impulse response model identification problems	(Ling et al., 2017)
11	Binary Moth Search	The LF operator is integrated into the subpopulation 1 stage to update the populations (moths) of this subcategory.	30 DKP instances: 10 uncorrelated instances (UDKP1-UDKP10), 10 weakly correlated instances (WDKP1-WDKP10), and 10 strongly correlated instances (SDKP1-SDKP10).	(Feng and Wang, 2018)
12	Black Hole	Levy step size is employed directly in the exploitation phase position update step.	six UCI machine learning datasets: Iris, Wine, Glass, Cancer, Contraceptive Method Choice (CMC), and Vowel	(Abdulwahab et al., 2019)
13	Sine-Cosine Algorithm and Whale Optimization Algorithm	LF added to the exploration phase of both SCA and WOA their search agents (candidates) position update stage, where only a portion of search agents will explore even after optimization; the rest will exploit.	Twenty-three standard benchmark functions and six composite benchmark functions from CEC-2005 test problem suite.	(Emary et al., 2019)
14	Sine-Cosine Algorithm and Particle Swarm Optimization	The LF distribution is added to the only sine-cosine algorithm differently, then applied to the position update stage of the original PSO according to two criteria.	Proportional derivative integral (PID) controller tuning problem	(Bejarbaneh et al., 2020)
15	Levy Flight Distribution	In LFD, Levy-derived step length is incorporated into the original Levy distributed algorithm to update position when ED is less than the specified threshold and CSV is greater than random value R.	Twenty-two functions of a well-known CEC-2017 benchmark suite.	(Houssein et al., 2020)
16	Monarch Butterfly Optimization	This modified algorithm first implemented the LF mutation operator, then applied it to the position update stage when the random value (between 0 and 1) was less than 0.5.	Twenty-five UCI datasets.	(Alweshah, 2020)
17	Crow Search Algorithm	When the random value is less than the AP value, the levy step is directly added to each iteration count to generate a new state of the crow.	Two real world problem: simple structure (beam) and complex structure (gearbox housing)	(Wu et al., 2020)

Sl. No.	Algorithm	LF Integration Strategy	The Employed Benchmark Test Function	Ref.
18	Selfish Herd Optimizer	Incorporate levy distribution in the prey leader position update stage.	Fifteen benchmark functions and four well known engineering examples: tension/compression spring design problem, pressure vessel design problem, speed reducer design problem and welded beam design problem	(Zhao et al., 2020)
19	Bat Algorithm	LF was added both in the exploration and exploitation phases; however, in the exploitation phase, 50% of position updates were performed by levy distribution.	Twenty-four benchmark functions from CEC-2025 benchmark test suite.	(Boudjemaa et al., 2020)
20	Antlion Optimization	In this modified approach, the first stage ant position update is performed by OL, and the second stage ant position update is performed via LF.	Thirty benchmark function from IEEE CEC-2017 test instances.	(Ba et al., 2020)
21	Ant Colony Optimization	LF is added to the candidate selection mechanism as a random variable in the original ACO.	Twelve different TSPLIB benchmark instances.	(Liu et al., 2020)
22	Slime Mould Algorithm	Levy distribution directly added to the original SMA algorithm position update stage using Hadamard product operator.	Thirteen benchmark test functions.	(Cui et al., 2020)
23	Jaya Algorithm	Employed LF on the best 50% solutions in the new candidate solution generation step.	17 frequently used unimodal and multimodal benchmark functions	(Ingle and Jatoth, 2020)
24	Moth Flame Algorithm and Particle Swarm Optimization	At first, define the LF integrated PSO strategy and then apply this approach to the location update stage of moth flame optimization.	Real world problem: Hybrid micro-grid system	(Bandopadhyay and Roy, 2020)
25	Discrete Bat Algorithm	The LF-derived random distribution is directly entry wise multiplied in the new velocity generation function.	Thirty-eight symmetric tsp benchmark instances from TSPLIB	(Saji and Barkatou, 2021)
26	Salp Swarm Algorithm	The Levy distribution function is integrated into SSA in the mutation operation phase to enhance candidate randomness during search time.	Twenty-three standard benchmark and three engineering design problems: three bar truss design, pressure vessel design, and speed reducer design problem.	(Nautiyal et al., 2021)
27	Jaya Algorithm	Added levy distribution by replacing the uniformly distributed two random numbers when particles are upgrading their positions.	CEC-2014 benchmark problems as well as five industrial optimization problems from the CEC-2011 benchmark suite.	(Iacca et al., 2021)
28	Whale Optimization Algorithm	Levy distribution directly entry-wise multiplies with the exploration agents search location update stage.	Twenty-five well known benchmark optimization functions.	(Deepa and Venkataraman, 2021)

Sl. No.	Algorithm	LF Integration Strategy	The Employed Benchmark Test Function	Ref.
29	Elephant Herding Optimization	Levy distribution-based mutation strategy integrated with the position update step.	Ninety-seven standard benchmark functions, CEC-BC-2017 test function and five real world engineering problem: welded beam design, tension/ compression spring design, pressure vessel design, gear design, and ammonia synthesis reactor problems.	(Singh et al., 2021)
30	Moth Flame Optimizer	After the position update stage, LF operators are utilized, and each Moth (candidate) that has been updated is permitted to carry out LF once.	Twenty-three medical datasets.	(Khurma et al., 2021)
31	Aquila Optimizer	The original AO algorithm is based on the LF approach and consists of four stages: two explorations and two exploitations. Levy distribution function integrate narrowed exploration and exploitation stages.	Twenty-three well known functions, thirty CEC-2017 test functions, ten CEC-2019 test functions, and seven real world problems: tension / compression spring design problem, pressure vessel design problem, welded beam design problem, 3-bar truss design problem, speed reducer problem, cantilever beam design problem, and multiple disc clutch brake problem.	(Abualigah et al., 2021)
32	Seagull Optimization Algorithm	LF is added directly to the current position step at each iteration to update current solutions.	Twenty-nine benchmark functions from CEC-2017, twenty benchmark datasets, and face pose recognition problem.	(Ewees et al., 2022)
33	Genetic Algorithm	This modified approach implemented the LF-based mutation function in the exploration phase.	airborne visible/infrared imaging spectrometer (AVIRIS) and ROSIS3 datasets.	(Aghaee et al., 2022)
34	Firefly Algorithm	LF is directly incorporated into the original Firefly algorithm as an adaptive parameter to calculate fitness values.	Six test images with different threshold namely Barbara, Birds, Fish, Owl, butterfly, and House.	(Peng and Zhang, 2022)
35	Chicken Swarm Optimization	The levy random walk is added to the rooster, hen, and chick's position update stage when they (candidates) are in a trapped condition.	Thirty benchmark test functions from CEC-2017 and pressure vessel design problem.	(Verma et al., 2023)
36	Remora Optimization Algorithm	The LF method is integrated into the exploration phase of the original ROA by the dot product ($\odot$) operator.	Three standard open-access datasets: BreastEW, Churn, and HeartEW, and three engineering problems: cantilever beam design, three-bar truss design, and pressure vessel design.	(Mujawib Alashjaee, 2023)

Sl. No.	Algorithm	LF Integration Strategy	The Employed Benchmark Test Function	Ref.
37	Gravitational Search Algorithm	When both elite and non-elite agents update their positions, the levy distribution is incorporated by multiplying it by all of the agents.	Twenty-three classical test problems and thirty CEC-2014 test problems	(Joshi, 2023)
38	Grasshopper Optimization Algorithm	Except for the first population in the position initialization stage, levy distribution is incorporated into standard GOA by replacing uniformly distributed random numbers (between 0 and 1).	Twenty-three well-known benchmark functions and seven engineering problems: Himmelblau's nonlinear optimization problem, cantilever beam design, car side impact design, gear train design, pressure vessel design, speed reducer design, and tabular column design problem.	(Wu et al., 2023)

3. OPTIMIZATION

Optimization, a fundamental principle in the pursuit of efficiency, productivity, and perfection, is an inherent notion in the fields of mathematical science, engineering, and several branches of science. This fundamental idea encompasses the skill of optimizing solutions, tasks, and strategies for the best results, enhancing usefulness while eliminating restrictions (Maier et al., 2019). Exploring the core of optimization exposes an environment rich in various approaches, mathematical foundations, and computational ability. Generally, seeking the most optimal solution to a particular problem under a series of constraints is an aspect of optimization (Nicholson, 2017). In another phrase, it refers to an approach to boosting or reducing an objective function while taking into account several criteria. Throughout optimization, the best component is selected from a set of viable choices and alternatives (Maier et al., 2019). The ultimate goal of this problem-solving or solution-finding procedure is to identify the most effective solutions to the goal-directed function or functions under constraints. The optimization process is the method of determining the "optimum" arrangement of an assortment of variables for attaining a set of goals. These variables can have real or discrete values (Kumar and Yadav, 2022). Optimization intends to provide the "best" model concerning a list of constraints or priorities. These involve maximizing aspects like output, fortitude, dependability, endurance, effectiveness, and usage (Maier et al., 2019).

3.1. Mathematical Model of Optimization

Finding the best feasible solution to an issue is done through the process of optimization. The exploration space (all possible choices) for an optimal operation problem is represented by the tandem (x, f), and the function f:X→R is represented by f. Equation (3.1) (Jeter W, 2018) illustrates the optimal x solution.

$$x_{opt} \in X \; is \; optime \; if - f(x) \leq f\left(x_{opt}\right), \forall x \in X \tag{3.1}$$

Mathematically, an optimization problem is an arrangement of inequalities and formulas that define the problem to be solved. A linear or nonlinear function of the decision variables frequently represents the objective function. A collection of linear or nonlinear inequalities serves as the constraints representation (Nicholson, 2017). Equation (2), (Jeter W, 2018) therefore, defines the generic mathematical formula for an optimization problem.

maximize or minimize of
$f(x)$
subject to:
$g_i(x) \leq 0 \; and \; h_j(x) \; = 0 \; and \; i = 1,2, \ldots m \; and \; j = 1,2, \cdots p$ (3.2)

The constraints on inequality and equality are denoted by: $g_i(x)$ and $h_j(x)$, respectively, whereby f (x) is the objective function to be optimized. The constraints on inequality and equality are represented by the numbers m and p.

In essence, optimization aims for the most effective option from a range of viable options while taking into account a number of criteria. Computational frameworks that include the objective function, decision variables, and constraints of the optimal solution are frequently used to represent this path to supremacy. A factor that is subject to the linear inequity criteria must take the format of equation (3.3) (Maier et al., 2019) if the decision variables are $(x_1, x_2, \ldots . x_n)$ and the optimal function f$(x_1, x_2, \ldots . x_n)$ is linear.

$$\begin{aligned} g_1(x_1, x_2, \ldots . x_n) &\leq b_1 \\ g_2(x_1, x_2, \ldots . x_n) &\leq b_2 \\ &\vdots \\ g_m(x_1, x_2, \ldots . x_n) &\leq b_m \end{aligned} \quad (3.3)$$

while upholding the constraints: $x_1 \geq 0, x_2 \geq 0, \ldots, x_n \geq 0$

3.2. Exploration and Exploitation Concept in Optimization

Two key concepts, exploitation, and exploration, interact in the complex world of optimization to guide strategies towards optimal solutions. These two complementary approaches capture the idea of probing the problem domain while attempting to strike a fine balance between checking out uncharted territory and sharpening current insights (Long et al., 2018). Several conscious equations encapsulate the core of optimization's exploration and exploitation (Črepinšek et al., 2013).

3.2.1. Exploration

Exploration is the method of extensively and wildly seeking as well as surveying the solution space. It entails searching for novel and uncharted areas of the problem domain to discover potentially superior solutions in the context of optimization techniques. Exploration aims to get a thorough grasp of the issue's scenery, particularly

in areas where possible solutions may exist but are still under investigation (Jie et al., 2009). An algorithm may identify alternate routes to possibly superior solutions that may not appear clear from the understanding of the situation at hand by examining multiple regions. Computationally, exploration may be defined as the search for novelties in the solution region. By prompting algorithms to depart from well-known areas and venture into unexplored areas, it aims to diversify the quest. Equation (3.4) (Jie et al., 2009) presents a prospective formula for exploration.

$$E_r = \Sigma\left(x_i - x_{avg}\right)^2 \tag{3.4}$$

where x_i represents a decision variable and x_{avg} denotes the average value of decision variables. By allowing divergences beyond the standard, the equation encourages exploration by capturing every parameter's variance via the average value.

3.2.2. Exploitation

Exploitation concentrates on fine-tuning and stepping up inquiry inside areas of the problem domain that have previously demonstrated potential for producing efficient remedies. Exploitation in optimization focuses on areas that, according to the optimization approach's present understanding, contain superior outcomes (Hussain et al., 2018). The approach seeks to enhance as well as clarify the solutions already obtained by focusing its exploration on these areas. This is essential for maximizing the outcomes inside the areas of interest and accelerating the algorithm's progression to the best or nearly optimum outcome. A formula that expresses the concept of exploitation from equation (3.5) (Jie et al., 2009).

$$E_t = \Sigma\left(f_{max} - f(x_i)\right)^2 \tag{3.5}$$

where $f(x_i)$ represents the value of the objective function at a specific point and f_{max} is the maximum value of the objective function encountered so far. The method is encouraged to improve solutions that are nearest to the result by the equation, which stresses the proportional variation between the quantity at every location and the highest possible target value.

3.2.3. Balancing Exploration and Exploitation

An optimization method must strike a balance between exploitation and exploration. When excessive exploitation could prevent the approach from moving

beyond local optima and preventing it from finding superior solutions additionally in the problem area, excessively much exploration could result in inefficiencies in the refinement of already-discovered satisfactory solutions (Kumar et al., 2023). A key problem in designing optimization strategies is finding an ideal equilibrium between exploration and exploitation. Typically, this includes adjusting algorithm factors and using adaptive techniques to change the exploration-exploitation conflict over time as the algorithm evolves. Equation (3.6) (Tan et al., 2009) is a representation of a combined expression that can be formulated to achieve a balance between exploration and exploitation.

$$E_B = \alpha * E_r - \beta * E_t \tag{3.6}$$

where α and β are weight factors controlling the emphasis on exploration and exploitation. Users may modify the functionality of the strategy to concentrate more on exploration or exploitation, depending on the demands of the challenge, by varying the parameters α and β.

These separate explorations and exploitations of mathematical terms provide a concrete illustration of the tactics used by optimization algorithms. These offer a quantitative comprehension of the way algorithms explore the range of possible solutions, searching for novel options and iterating on existing ones to find the best spots.

3.3. Optima Concept in Optimization

Optimization is a complex process that seeks the optimal solutions to a problem. Two important concepts in this process are local optima and global optima. These special principles apply to the whole area of optimization and hold significant implications for the effectiveness and efficiency of optimization methods (Nicholson, 2017).

3.3.1. Local Optima

The term “exploring neighboring levels” refers to the local optimal solution. A solution is considered to be a local optimum if it has a greater value for the target function compared to the locations that are locally nearby it, even though this value might not be the maximum value that is attainable over the entirety of the solution area (Liang and Suganthan, 2005). It is analogous to climbing an elevation and attaining the location that,

from that specific observation location, appears to be ideal, even though there may be greater heights that are obstructed from sight.

The existence of local optima presents a dilemma since algorithms for optimization, if they are not expertly led, have the potential to get trapped at such minimal points, unaware of the higher points that lie beyond them. This occurrence, which has been assigned the name "convergence to local optima," can prevent the algorithm from attaining the global optimal state, thereby limiting the algorithm's ability to discover the best possible solution (Desale et al., 2015).

3.3.2. Global Optima

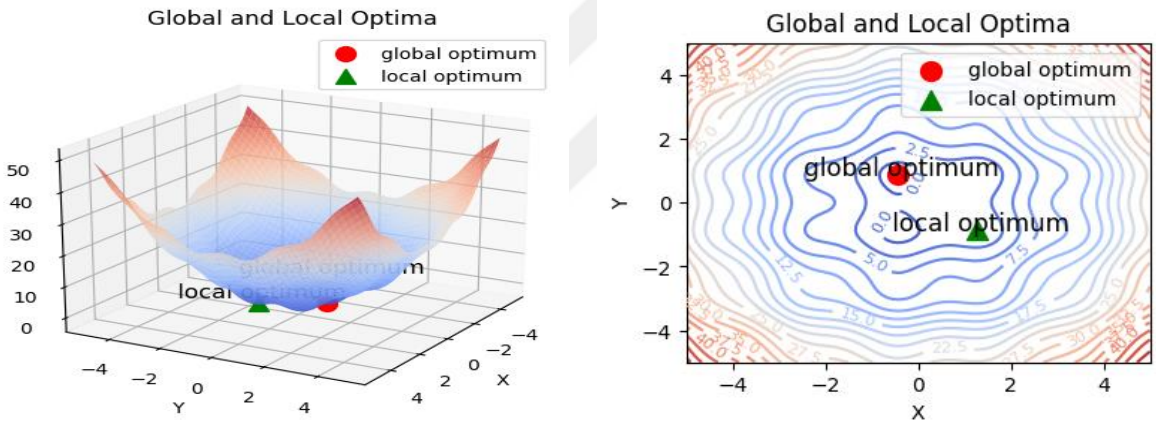
On the contrary, global optima refers to “ascending to ultimate levels”. The concept of the global optima is defined as the highest point that may be reached within the context of the optimization scenario. It exemplifies the maximum possible value of the targeted function, which is attainable over the entirety of the solution surface area, and it is the embodiment of flawlessness. Getting to the global optimum implies that we have achieved the absolute ideal result, which is superior to all other possible solutions, including local optima (Mirjalili et al., 2015). The search for the global optimum is of the utmost importance in the field of optimization because it captures the core of the problem, which is to discover the optimal solution while remaining unbound by the enticement of neighborhood allurements. The pursuit of the global optimal solution may seem like a challenging endeavor, but it can also confer unmatched benefits by opening the door to solutions that are exceptional in every aspect (Liang and Suganthan, 2005).

3.3.3. Optimality Balancing

The term "consequences for optimization" implies the process of exploring the landscape. The complicated movement that occurs inside algorithmic optimization is best shown by the simultaneous nature of both global and local optima (Liang and Suganthan, 2005). Finding an equilibrium between exploring new territory and capitalizing on existing resources is essential to achieve globally optimal performance. The exploration process guarantees that the algorithm moves beyond local optimal solutions by going into previously unknown zones, while exploitation treats alternatives to achieve convergence away from the global minimum. A conceptual trip of the greatest importance takes place

when one travels to recognize and differentiate between local and global optima. Optimization techniques, empowered through these insights, set out on a deliberate journey, managing the tricky move of balancing both local and global investigation and guiding themselves regarding solutions of unusual worth and quality (Mirjalili et al., 2015).

Figure 3.1 illustrates the local and global optimum points for the objective function using 3D and 2D plots, respectively. A green triangle sign has been set at a spot on the terrain that denotes the local optimum of the target function. Its minimal point is a local optimum in the vicinity or area, and in this scenario, a low point inside a specified neighborhood is considered to be the local optimum. In the position that denotes the local maximum (global optimal solution) of the function of the objective, a red circle sign has been used. The position at which the function obtains its ultimate minimum score over the entirety of the realm is commonly referred to as the global optimal point.



(a) 3D space (b) 2D space

Figure 3.1. Local and Global optima in Optimization

3.4. Extremum Points Avoidance Methods

The most prominent and prevalent problem in optimization is getting stuck in local optima. To overcome this problem, optimization algorithms must employ tactics that allow to explore the solution space outside the proximity of the local optima. Several methods exist in the literature to overcome the difficulties associated with local optima. This subsection presents a number of local optima problem-solving techniques that were applied in earlier, highly regarded research.

- **Crossover and mutation:** By integrating beneficial genetic elements, crossover enables the algorithm to take advantage of potential locations. As a result, superior solutions may converge in areas of the search space where global optima are more prevalent. This aids in escaping local optima by increasing the likelihood of producing a varied crop of children, some of whom may inherit desirable characteristics from both parents. The genetic operator known as "crossover" allows for the creation of novel approaches by combining the DNA of two or more parent solutions (Dang et al., 2016).

  A counterargument is that mutations in GAs and other search-based techniques contribute to variety in populations. Because mutations disrupt the existing solutions, the algorithm is free to investigate neighboring areas in the hopes of escaping local optima (Blum et al., 2011). To keep the population from becoming too homogeneous, a mutation might delay convergence in some parts of the solution arena. It encourages a more comprehensive investigation by keeping the GA from getting stuck on a small subset of potential solutions (Dang et al., 2018).
- **Niching technique:** By increasing the variety of the population, niche approaches in optimization keep solutions from gathering in one place, which slows down the process of quickly reaching the best conditions in that area. Fitness broadcasting sanctions comparable solutions, and overcrowding elects options for regeneration depending on the length (Li et al., 2017). Niching strategies avoid early convergence by encouraging various solutions in diverse locations. Higher niche radii promote wider inquiry, whereas lower niches refine solutions within an area. In GAs and PSO, niching leads to choosing, crossover, and mutation to preserve variety (Lu et al., 2021). Certain algorithms use adaptive niching to alter specs according to a solution's features, improving flexibility and performance throughout optimization sessions.
- **Opposition-based learning (OBL):** The OBL method by Tizoosh et al. (Tizhoosh, 2005) contrasts one's fitness with that of the opposing integer and progresses superior candidates toward the subsequent generation (Hussien and Amin, 2021). OBL simultaneously searches for opposing solution area orientations to avoid local optima and increase exploration. OBL optimizes by exploring alternative pathways and promoting disturbances. It improves global search, discovers areas, and uses composite approaches. With OBL's adjustable opposition, metaheuristic strategies prevent convergence and evolve individuals (Mahdavi et al., 2018). Researchers have looked into a wide range of OBL applications, including determining the action

policies for reinforcement agents, potential solutions for designing evolutionary algorithms, and the transfer functions and weights of neural networks.

- **Hybrid approaches:** To get beyond local optima, hybrid optimization methods smoothly combine several methodologies (Blum et al., 2011). Optimization algorithms can be more versatile, diversified, and successful by integrating global exploration, local modification, OBL, adaptation techniques, and aggregation approaches (Seyyedabbasi et al., 2021). These solutions tackle local optima issues in a fashion that benefits society. Combining a GA with the local searching mechanism to create a memetic algorithm serves as an illustration of a hybrid methodology. While LS focuses on improving solutions in a specific area, GA offers research on a worldwide scale (Yi et al., 2013). This combination improves the algorithm's optimal effectiveness by balancing exploration and exploitation, boosting its ability to escape local optima.
- **Parameter tuning or adaptation:** Changing algorithmic settings is part of optimizing parameter adaptation because it improves the complicated interaction between thoroughly exploring the solution domain and efficiently leveraging attractive parts (Salgotra et al., 2021). By adjusting settings like mutation rates and step sizes, the algorithm may be fine-tuned to fit the unique requirements of each task. Because of this adaptability, the algorithm can better traverse solution areas, which improves convergence towards superior global remedies and allows it to elude local maximal values (Zhou et al., 2022).
- **Local search (LS) strategies:** Numerous studies have used LS techniques to overcome local optima by improving options around the existing solution (Hussien and Amin, 2021). The hill climb method requires a methodical approach to enhancement via repeated incremental tweaks. The stochastic acceptability of less effective solutions is introduced via SA, which allows for exploration. The TS technique preserves solutions it has already traveled to, so it doesn't have to go over identical areas twice (Pavlyukevich, 2007). By using these techniques, optimization approaches explore large solution areas, finding better solutions on a global scale while avoiding local maxima.
- **Levy flight:** LF is a method developed by Paul Levy (Kamaruzaman et al., 2013), and so many optimization algorithms have been improved via this for probabilistic exploration that helps algorithms for optimization get out of local optima by using

heavy-tailed step durations. Exploring and evasion from constrained solution areas are made possible by the rare, lengthy hopping. The converse is true for unpredictable processes, such as Brownian motions and Gaussian walks (Blum et al., 2011). Autonomous steps characterize Brownian motion, which is both steady and unpredictable. In contrast to LF's infrequent large leaps, the resembling probability distribution of Gaussian walks—which are associated with the normal distribution—indicates less typical steps (Kamaruzaman et al., 2013; Li et al., 2022).

- **Orthogonal design:** In optimization, orthogonal design is a technique for getting beyond local optima. This strategy guarantees a comprehensive investigation of the solution arena by methodically altering numerous elements consecutively in an orthogonal and consistent way (Yang and Phung, 2010). By effectively exploring different sets of components, it promotes a greater impact on global search and prevents the optimization process from being trapped in neighborhood optimality (Ma et al., 2021).

3.5. Functioning of Optimization Techniques

An optimization problem may be split down into its fundamental working method, which consists of two key phases (Nicholson, 2017) :

- Modeling: The initial phase is to generate a computational illustration of the problem to be solved. To accomplish this, one must first determine the decision parameters, the objective function, and the constraints that apply.
- Solving: The computational framework is to be solved as the second phase in the process. Several different optimization techniques can be used to accomplish this goal. A simple optimization approach is shown in Figure 3.2.

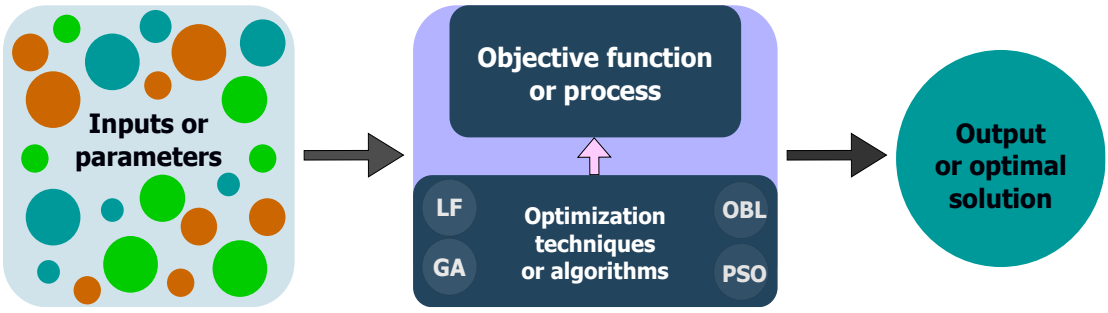


Figure 3.2. A basic optimization processes.

Figure 3.3 is an illustration of a description of the operation of optimization approaches utilizing a schematic structure. The schematic depiction provides a generic picture of how optimization approaches work, emphasizing the major phases included in the repetitive method of developing solutions. The particulars of every stage may change based on the optimization technique that is applied. The operation of this generalized method of optimization involves the steps of initializing the initial solution or population, evaluating the goal or fitness function, generating new solutions, and iteratively improving the method (Kumar and Yadav, 2022). The next paragraph provides a concise summary of these steps:

- Initialization: The approach starts with the generation of either an initial solution or a population of prospective solutions, whichever comes first. This defines an initial position from which the optimization procedure will begin.
- Evaluation: The target or fitness function is applied to every single solution to assess it. This function provides a quantitative assessment of the effectiveness or appropriateness of the solution concerning the given problems.
- Generate new solutions: Exploration (the process of seeking unknown regions) and exploitation (the process of enhancing prospective solutions) are both crucial phases in the formation of new solutions. To generate novel solutions, some different methods are used, including mutation, local exploration, and recombination.
- Update best solution: When a superior solution is identified, the current best solution (that is, the one having the greatest fitness score for maximization or the smallest value for minimization) is changed.
- Termination: The entire approach comes to an end when either an optimal solution that satisfies the conditions for termination or the highest possible number of cycles that were defined have been met. The concept of convergence, which occurs when solutions become stable or achieve the highest number of cycles, is an example of a typical threshold.

Figure 3.3. Optimization techniques functioning diagram.

3.6. Types of Optimization Problems

There are several distinct types of optimizations, each of which is designed to meet an individual set of issues, features, and purposes. The following is a rundown of the various kinds of optimization, along with brief descriptions of each one:

- Single-Objective Optimization: Single-objective optimization is concerned with improving just one objective function while taking constraints into account. The purpose is to locate the approach that improves or decreases the impact of this particular factor as much as possible. This kind is typically utilized in circumstances where there is a single objective that has to be accomplished, including the maximizing of profits, the reduction of costs, or the optimization of time (Kumar and Yadav, 2022).
- Multi-Objective Optimization: When using multiple objectives, an optimization scenario must strike an equilibrium between several conflicting objectives. The goal is to discover a group of alternatives that constitute the "Pareto front" or "Pareto frontier," which is a situation in which no solution can be made better on a single target without making a different one poorer. Whether it comes to accomplishing selections that require creating compromises, including striking a balance between price and worth or between adverse impacts and effectiveness, multi-objective optimization is an especially useful tool because of its ability to provide optimal solutions (Desale et al., 2015).
- Constrained optimization: The term "constrained optimization" refers to the process of solving optimization issues in which the outcomes need to comply with certain restraints. This restriction may take the form of equitable constraints, like the ability to manufacture, or unequal constraints, like monetary restrictions. The optimization method strives for solutions that can meet all of the requirements while also improving the overall performance of the function that they aim to optimize (Maier et al., 2019).
- Unconstrained optimization: The process of addressing optimization issues without using any formal constraints is known as unconstrained optimization. The only aspect that is of concern is finding the best way to maximize the value of the target function, and there are no restraints placed on the factors involved. It is frequently utilized in computational studies, the creation of algorithms, and the determination of the most optimal coefficients (Kumar and Yadav, 2022).

- Global optimization: The ultimate goal of global optimization is to identify the globally optimal levels, which refer to the most effective feasible solution across the whole surface, while simultaneously avoiding the identification of any local optimal value (Stork et al., 2020). Since there are several local optimal solutions for the target function and the purpose is to figure out the solution that is of the highest quality, it is essential to perform this kind of optimization (Mirjalili et al., 2015).
- Local Optimization: Local optimization aims to identify the most effective solution for a problem that is located close to its starting point. It seeks to accomplish this by repeatedly enhancing the value of the target function inside a constrained nearby environment to better perfect the solution. Since the problem domain is generally straightforward, local optimization is a viable option (Gao et al., 2021).
- Discrete Optimization: The discipline of discontinuous optimization is concerned with finding solutions to issues in which the decision parameters (variables) can only take periodic or integer forms. The majority of stochastic optimization issues fall into this category, which includes tasks like determining the optimal sequencing, timetables, or allotment (Li et al., 2022).
- Continuous Optimization: In continuum optimization, the decision parameters may each decide upon an actual value that falls within a particular band. This presents several challenging obstacles. It can be utilized in situations in which the range of possibilities is continuous, for instance, when adjusting parameters or optimizing functions (Kumar and Yadav, 2022).
- Mixed-Integer Optimization: Both continuous and discontinuous methods of decision-making are integrated into mixed-integer optimization. This kind of approach is utilized when the issue involves the integration of continuous and discrete assessments, making it suitable for tasks that are difficult in practical situations (Kumar and Yadav, 2022).
- Linear optimization: When the value of the target function and the constraints that accompany it are linear, the optimization problem can be characterized as a linear optimization problem. Manufacturing activity scheduling, resource allocation, and supply logistics control are just a few of its many uses (Maier et al., 2019).
- Nonlinear optimization: Optimization difficulties that involve nonlinear target functions or constraints can be addressed by using nonlinear optimization. This kind is frequently employed in the fields of mathematics, business, and technology and encompasses a wide spectrum of real-life issues (Maier et al., 2019).

In a nutshell, the many kinds of optimization offer a flexible toolkit that may be used to handle an extensive variety of different problems. The features, aims, and limitations of a particular issue should inform the researcher's decision regarding what kind of optimization to use; this will lead to solutions that are both beneficial and successful.

3.7. Taxonomy of Optimization Technique

The discipline of optimization is complex, consisting of many different aspects and housing a multitude of methods that are aimed at addressing a variety of problems. The aforementioned approaches are organized into hierarchies by this taxonomy, which gives a comprehensive overview of the qualities and implications of each method. Figure 3.4, which is a type of hierarchy schematic, provides a detailed overview of optimization approaches as follows:

3.7.1. Heuristic Methods

The term "heuristic" denotes techniques that are based on experience when used in the context of learning, solving problems, and searching. Even though it cannot be ensured that such tactics for figuring out solutions will produce ideal solutions, they represent an achievable strategy for handling complicated issues in an effective manner (Kumar and Yadav, 2022). Methods such as genetic algorithms (GA), greedy search algorithms (GSA), and hill-climbing (HA) all utilize guidelines of conduct to direct the hunt toward potentially useful solutions (Nicholson, 2017). Here are certain instances of stochastic optimization algorithms:

- Greedy search algorithms: These algorithms are advantageous in that they provide effectiveness; however, they cannot always ensure a solution that is optimal from a global perspective since they make decisions at every stage that are optimized locally (Maier et al., 2019).
- Hill climbing: HA is the process of iteratively altering solution parameters by heading in the direction of better target parameters, to converge to a local optimal solution (Kumar and Yadav, 2022).

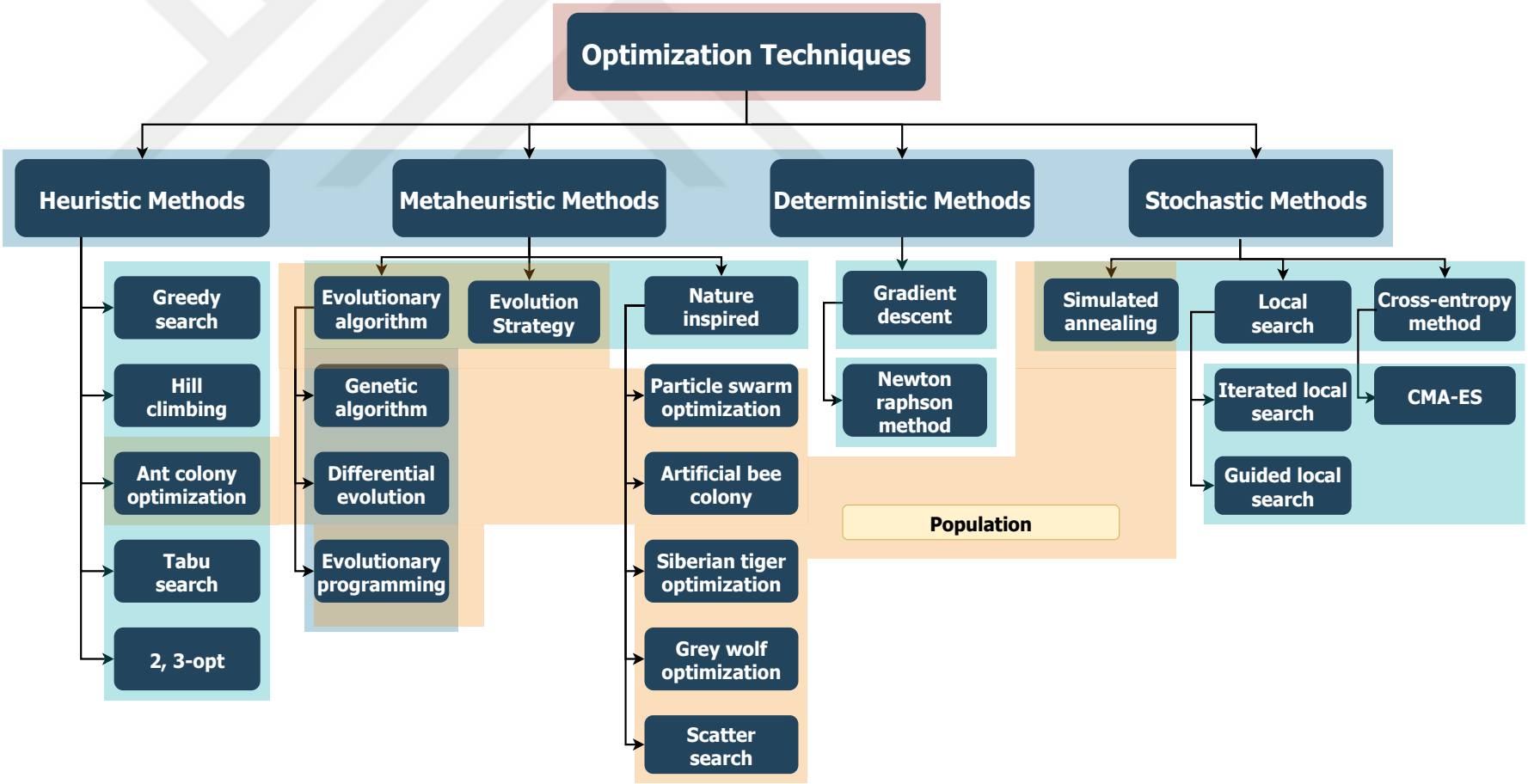


Figure 3.4. Taxonomy of optimization technique

- Ant colony optimization algorithm: ACO is a method that takes its cues from how ants forage for food. This method uses pheromone-based exchanges to determine the most efficient routes via networks (graphs) (Liu and Cao, 2020).
- Tabu Search (TS): It is a method of investigation that is memory-based and inhibits particular phases to prevent returning to results that have already been investigated (Kumar and Yadav, 2022).

3.7.2. Metaheuristic Methods

In the field of information technology, a meta-heuristic is a way to solve a problem by repeatedly trying to make a possible solution better based on a certain measure of how well it works (Kumar and Yadav, 2022). Such sophisticated tactics, which are frequently prompted by events of nature, direct the exploration mechanism to thoroughly investigate every potential remedy region. This optimization method is referred to by several names, including PSO, ABC, and ACO. These methods make use of the conflict that exists between exploration and exploitation to identify solutions that go beyond the scope of traditional methods (Hussain et al., 2018). There are several instances of metaheuristic optimization methods, such as:

- Evolution Strategies (ES): The evolutionary algorithm (EA), which is a strategy that takes its inspiration from nature and employs it for probabilistic global optimization, is one of the most effective metaheuristic optimization tools currently available. The algorithm alludes to a crucial method for creating adaptable frameworks in the arena of optimization. Even though the method is relatively straightforward, it is nevertheless capable of finding a solution that is either the best or the least possible (Stork et al., 2020).
- Genetic Algorithms (GA): The genetic algorithm known as GA is an example of an evolutionary algorithm (EA) and is developed from the biological evolution and selection process via crossover and mutation operations (Maier et al., 2019).
- Particle Swarm Optimization (PSO): This method, which optimizes solutions by altering particle velocity parameters and placements within the solution area, simulates social actions to get optimal results (Wang et al., 2022).
- Artificial Bee Colony (ABC): This approach imitates the natural actions of honeybees by utilizing bees that work and observers to search for new solutions and improve existing ones (Yonar and Yapici Pehlivan, 2020).

- ❖ Differential Evolution (DE): This method alters and reassembles solution arrays depending on the distinctions between them, promoting exploration as well as extraction (Gao et al., 2021).

3.7.3. Deterministic Methods

The goal of deterministic optimization is to locate a solution that is optimal across all scenarios, and it offers conceptual assurances to ensure the outcome that is given is, in fact, optimal across all possible scenarios. Deterministic optimization algorithms make use of special and handy aspects of the issue they are solving to achieve this goal (Fulber-Garcia, 2020). A deterministic algorithm can be misled (also known as "deceived" or "confused") due to chaotic assessment of candidate solutions or chaotic function gradients, which will cause the algorithm to jump about or become stuck (also known as failing to converge) (Gao et al., 2021). The following are some examples of deterministic optimization algorithms:

- ❖ Gradient Descent: The gradient descent algorithm uses mathematical concepts like calculus to iterate over solution parameters to reduce the value of the target function (Fulber-Garcia, 2020).
- ❖ Newton-Raphson Method: It is a methodical approach that mimics the roots of a function for the sake of optimization. This strategy relies on derivatives (Francisco et al., 2005).

3.7.4. Stochastic Methods

The term "stochastic optimization" implies applying elements of inconsistency to a method of optimization or the target function (Brownlee, 2021). Stochastic optimization algorithms offer an alternative strategy. This strategy allows for poorer optimum local decisions to be taken inside the process of searching, which may enhance the likelihood of the strategy of seeking to identify the global optimal value of the objective function (Spall, 2012). A great number of stochastic algorithms take their cues from natural or biological behaviors. These additional levels of procedures, referred to as "metaheuristics," are responsible for establishing the necessary circumstances for the

targeted exploration of the target function (Fulber-Garcia, 2020). The following are some instances of the algorithms that are used in stochastic optimization:

- Simulated Annealing: This technique searches the solution domain and gets away from local optimal solutions by allowing less privileged options with a given frequency (Bashath et al., 2022).
- Iterated Local Search (ILS): It is a technique that integrates neighborhood searches alongside disruptions and approval requirements to break out of the confines of the local optimal solution and investigate unknown regions (Gao et al., 2021).
- Cross-Entropy Method (CEM): This approach to combinatorial optimization evaluates and adjusts the pattern of distribution of effective solutions (Brownlee, 2021).
- Covariance Matrix Adaptation Evolution Strategy (CMA-ES): It is a complex derivative-free optimizing strategy that plans the distribution pattern of possible solutions and changes the covariance vector (Maier et al., 2019).

4. LEVY FLIGHT APPROACH

A random walk is a stochastic process in which the next step is chosen at random based on the one that came before it. The most well-known sort of arbitrary walk is the Brownian movement, where the step lengths are regularly appropriated (Yang and Deb, 2009). Levy Flight (LF) is a kind of irregular walk, a numerical idea used to demonstrate developments that include both little and enormous strides with a steady conveyance. It is named after the French mathematician Paul Lévy (Kamaruzaman et al., 2013), who originally concentrated on these examples in the 20th century (Houssein et al., 2020). This implies that the likelihood of making a stride of a given length corresponds to the length raised to the power of β, where β is a boundary that decides the state of the circulation (Li et al., 2022). Figure 3.5 delineates an illustration of an LF with 100 stages and a β of 1.5, which shows three critical boundaries of the LF: the path, step lengths, and directions. Here are the three parameters in Figure 3.5 and which line they correspond to:

- Direction: The direction of the LF is the deep blue line. The direction is a number between 0 and 2π, and it represents the angle of the next step.
- Step length: The step length of the LF is the red line. The step length is a positive number, and it represents the length of the next step.
- Position: The position of the LF is the blue line. The position is a two-dimensional vector, and it represents the current location of the walker.

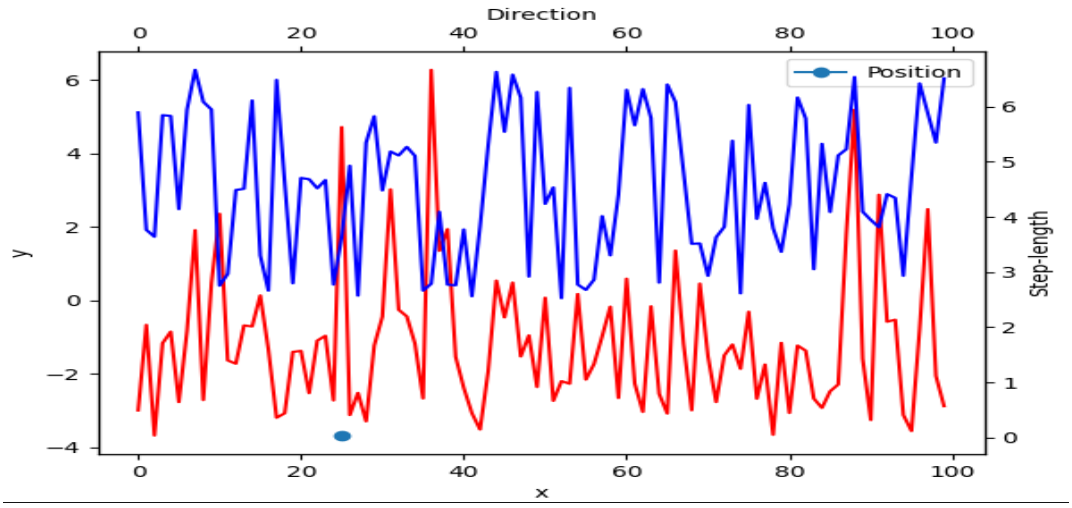


Figure 3.5. A Lévy flight with 100 steps and an β of 1.5

4.1. Levy Flight

Levy Flight, as its name suggests, is a type of random walk that the Levy distribution controls and has long jumps that make it simpler for optimization algorithms to find every possible solution (Houssein et al., 2020). Figure 4.6 visually demonstrates the concept of a Lévy flight, where each step has a length determined by the Lévy distribution and provides an intuitive visualization of how LF can behave in a two-dimensional space. In this figure, the blue line represents the trajectory of the LF, illustrating the successive steps taken in the random walk. Each step's length is determined by sampling from the Lévy flight distribution. The green point marks the initial position (0, 0), where the flight begins. The red point indicates the final position after the specified number of steps.

The heavy-tailed nature of the distribution leads to occasional long steps, resulting in the distinctive path observed in the plot. The occasional large steps introduced by LF enable the algorithm to escape local optima and increase the likelihood of finding global optima or high-quality solutions. This is in contrast to a Brownian motion and Gaussian walk, in which the probability of taking a very long step is very low (Li et al., 2022). Figure 4.7 depicts the LF, Guassian Walk, and Brownian motion trajectory with 5000 step lengths.

In the context of optimization algorithms, LF is employed to enhance the exploration capability of the algorithm. Instead of using fixed step sizes as in traditional random walks, LF introduces occasional long jumps, enabling the algorithm to efficiently explore the solution space and potentially discover better solutions, including global optima (Houssein et al., 2020). This makes LF a valuable tool for improving the performance of optimization algorithms, especially in complex and multimodal landscapes where there are many local optima to navigate around.

4.2. Mathematical Model

The Levy distribution, a probability distribution, describes LF, a particular type of random walk process. In mathematical terms, let X be a random variable representing the step size taken in each movement of the Levy Flight. Equation (4.1) (Li et al., 2022) provides the Levy distribution's probability density function (PDF):

$$f(x) = \frac{1}{|x|^{(1+\beta)}} \;;\quad 0 < \beta < 2 \tag{4.1}$$

where x is the step size and β is a constant parameter called the stability index. The stability index β determines the "fat-tailed" property of the Levy distribution. When β is close to 2 (i.e., β < 2), the distribution has long tails, and the probability of taking large steps increases, allowing for occasional long jumps during the random walk. From a mathematical viewpoint, the Levy flight formula can be simplified to the following form (Houssein et al., 2020) :

$$f(x;\mu,C) = \sqrt{\left(\frac{C}{2\Pi}\right)} * exp\left(-\frac{C}{(2(x-\mu))}\right)\frac{1}{(x-\mu)^{3/2}} \tag{4.2}$$

where x represents the step length, μ is the location parameter (mean step length), and C is the scale parameter (controls the step length distribution).

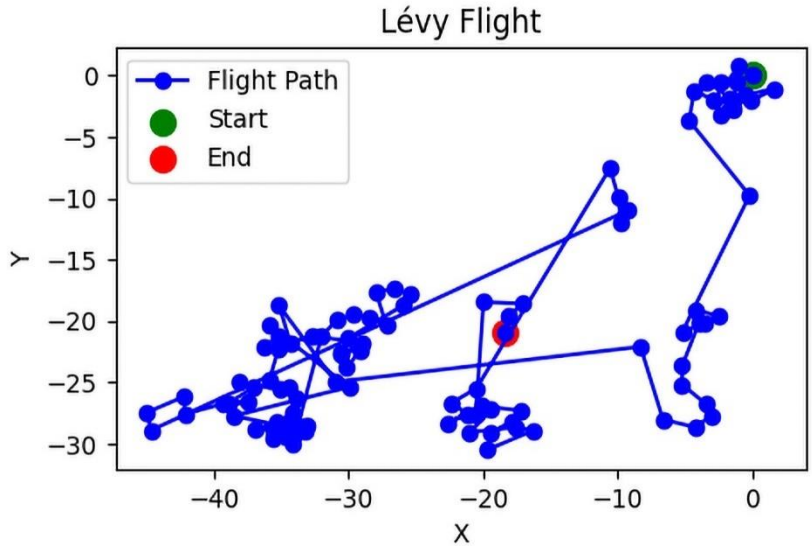


Figure 3.6. A levy flight exploration space.

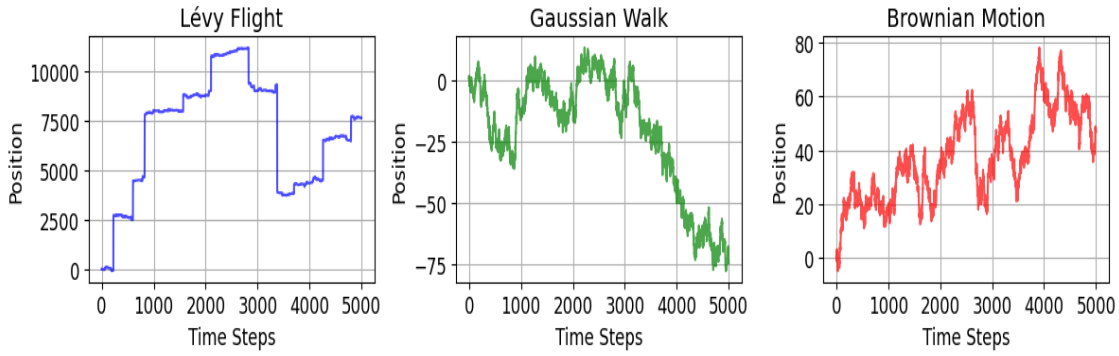


Figure 3.7. Trajectory of the Lévy flight, Guassian Walk and Brownian motion.

If α is the skewness parameter ($-1 \leq \beta \leq 1$), often set to 0 for symmetric Lévy flights, γ is the scale parameter that affects the width of the distribution, and μ is the location parameter representing the starting point of the distribution, then the PDF of the Lévy flight distribution p is defined by the Lévy-stable distribution in equation (4.3) (Houssein et al., 2020), which has a power-law tail.

$$p(x;\alpha,\beta,\gamma,\mu) = \left(\frac{1}{(2\pi)}\right) \int [\, exp(i\lambda x)\phi(\lambda;\alpha,\beta,\gamma,\mu)]\, dy \tag{4.3}$$

where $\phi(\lambda;\alpha,\beta,\gamma,\mu)$ is the characteristic function of the Lévy-stable distribution and λ is the frequency variable.

The Fourier transform of the characteristic function provides insight into the scaling properties and behavior of the LF distribution in the frequency domain. The characteristic function has a power-law tail that shows how heavy the distribution's tails are. The parameters $\alpha,\ \beta,\ \gamma, and\ \mu$ control different aspects of the distribution's properties. Equation (4.4) therefore gives the Fourier transform of the LF distribution's characteristic function (Houssein et al., 2020) :

$$\varphi(\lambda;\alpha,\beta,\gamma,\mu) = exp(i\mu\lambda - \gamma^{\alpha}|\lambda|^{a}\ [1 - i\beta\ sin(\lambda)tan(\pi\alpha/2)]\ ) \tag{4.4}$$

To get the sign of the parameter λ, we utilize the expression $sgn\ (\lambda)$ in equation (10). Mathematical notions such as the LF probability and its Fourier transform are fundamental to comprehending the statistical features and characteristics of LF in different domains.

4.3. Features of Lévy Flight

This subsection provides a detailed explanation of the three key features of LF: heavy-tailed distribution, scale invariance, and long-range correlation, along with their corresponding mathematical formulas.

4.3.1. Heavy-Tailed Distribution

Lévy flight exhibits a heavy-tailed distribution, meaning that it has a higher probability of large jumps compared to traditional distributions like the Gaussian distribution. The heavy-tailed property indicates that extreme events are more likely to occur in Lévy flight, leading to infrequent but significant changes in the system (Yang

and Deb, 2009). Figure 3.8 depicts the heavy-tailed distribution of levy flight compared with the Brownian motion distribution and flight path of other distributions.

The heavy-tailed behavior is often characterized using the PDF of the Lévy stable distribution. The PDF of the Lévy stable distribution has a power-law tail that can be described as follows (Barthelemy et al., 2008) :

$$P(x) \sim |x|^{(-\alpha-1)} \tag{4.5}$$

where α is the tail index parameter, and x represents the value of the random variable.

4.3.2. Scale Invariance

Lévy flights are scale-invariant, meaning that they do not change their shape when the step length is scaled. In other words, whether we observe LF over a short time interval or a long-time interval, the overall statistical behavior remains consistent. The reason for this is that the fixed distribution from which the step lengths are taken doesn't change when the size of the sample changes (Yang and Deb, 2009).

The scale invariance property is reflected in the characteristic function of the Lévy distribution. The characteristic function $\phi(t)$ of a Lévy stable distribution with parameters (α, β, γ, μ) satisfies the scaling property (Barthelemy et al., 2008) :

$$\phi(ct) = \phi(t)e^{(i\mu ct)-|\sigma ct|^{\alpha}[1-i\beta\, sgn(ct)\phi(\mathrm{t})]} \tag{4.6}$$

where c is a scaling factor, and other parameters have their usual meanings.

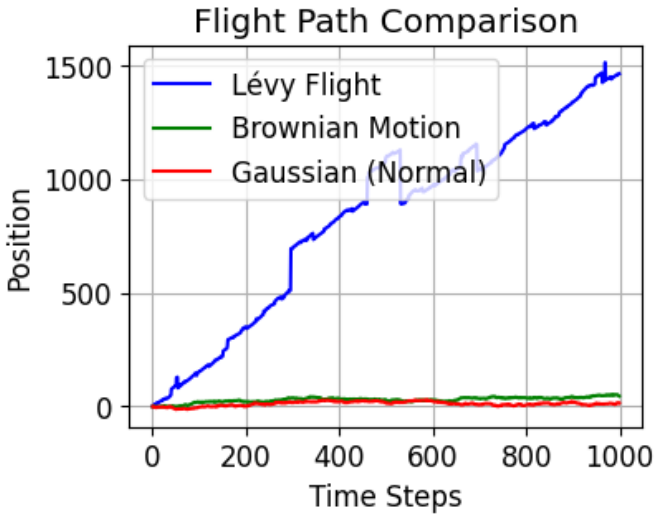


Figure 3.8. Lévy flight with heavy-tailed distribution.

4.3.3. Long-Range Correlation

Lévy flight exhibits long-range correlation, meaning that there is a persistence of influence across a wide range of time or space intervals. This property implies that the system's behavior at one time or location can affect its behavior at a distant time or location. This is because the step lengths are drawn from a stable distribution that has a long tail (Guerrero et al., 2015).

The long-range correlation property is often studied using the Hurst exponent H, which characterizes the autocorrelation behavior of a time series. In LF, the Hurst exponent typically lies in the range of $0.5 < H < 1$, indicating a memory that persists over a wide range of time intervals (Barthelemy et al., 2008).

These three features collectively make LF a unique and intriguing phenomenon in various fields, including physics, biology, and finance. It can show complex behaviors and events that regular Gaussian distributions might miss because of its heavy-tailed distribution, scale invariance, and long-range correlation. These three features are illustrated in Figure 3.9 in space.

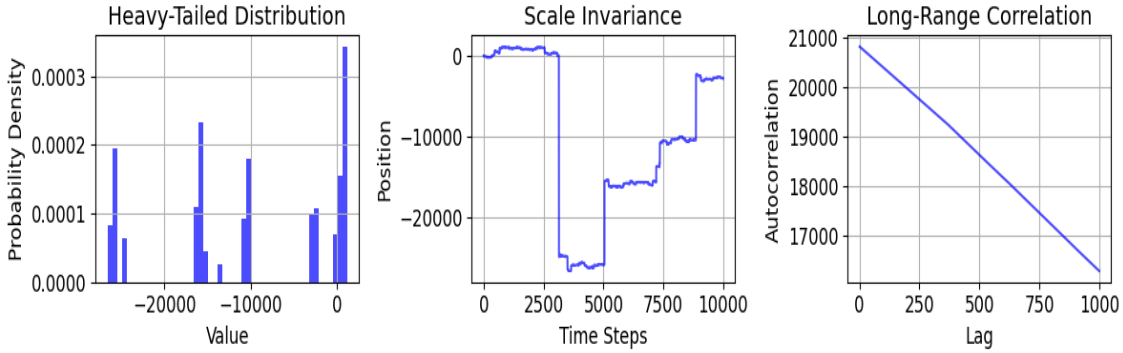


Figure 3.9. Three key features of Levy flight

4.4. Functional Model

To generate an LF, one must initially produce a range of step sizes using a generator of random numbers and then a subsequent set of trajectories or directions using the same generator. The step lengths are drawn from a stable distribution, and the directions are drawn from a uniform distribution. The steps are then added together to form a path (Barthelemy et al., 2008). The LF algorithm efficiently explores the search space by occasionally making long jumps, allowing it to quickly cover large regions. This

exploration capability is especially beneficial in optimization algorithms, where the algorithm aims to find the best possible solution in a complex and multi-dimensional search space (Li et al., 2022). The steps of this LF method are explained as follows, with their corresponding pseudocode and flowchart shown in Table 3.1 and Figure 3.10, respectively:

- Step 1: Initialize the starting point. Start the LF from an initial point in the search space, represented as ($x_{current}$, $y_{current}$). This initial point is the starting position for the random walk.
- Step 2: Generate $step\_length$ from the Levy distribution, which has a heavy-tailed property. The Levy distribution is defined by the probability density function (PDF) mentioned earlier: $\boldsymbol{f(x) = 1 / x^{(1+\beta)}}$.
- Step 3: Generate step directions. Generate a random angle θ from 0 to 2π. This angle determines the direction of the step taken in the random walk.
- Step 4: Compute the new position. Using the $step\_length$ and the random angle θ, compute the new position (x_{new}, y_{new}) as follows:

  $x_{new} = x_{current} + step\_length * cos(\theta)$
  $y_{new} = y_{current} + step\_length * sin(\theta)$
- Step 5: Check the validity of the new position. In some cases, the step length from the Levy distribution can be very large, causing the algorithm to jump far away from the search space. To prevent this, check if the new position falls within the bounds of the search space. If it goes beyond the boundaries, reject the step and repeat Step 2 to generate a new step size and direction.
- Step 6: Modify the Present Scenario. When the updated location is correct, then set the current positions to the new position.
- Step 7: Carry out the same action. Repeat steps 2–6 indefinitely or until a termination condition is satisfied, depending on the predefined number of iterations.

This iterative process of generating step sizes and directions from the Levy distribution and updating the current position constitutes the LF algorithm.

In summary, the Levy Flight algorithm works by using the Levy distribution to generate random step sizes and random angles are used to determine the direction of each step. By occasionally taking long jumps due to the heavy-tailed property of the Levy distribution, the algorithm explores distant and unexplored regions in the search space efficiently. This exploration capability is beneficial to optimization algorithms as it allows them to escape local optima and discover potentially better solutions, including global

optima, in complex and multimodal landscapes. The process continues iteratively, covering large regions of the search space and increasing the chances of finding a high-quality solution.

Table 3.1. Pseudocode of LF algorithm

1. Initialize the Starting Point Start
2. Generate step size from Levy distribution: step_length: $\boldsymbol{f(x) = 1 \,/\, x^{(1+\beta)}}$
3. Generate random angle θ between 0 and 2π
4. Compute new position:

$$\boldsymbol{x_{new} = x_{current} + step\_length * cos(\theta)}$$
$$\boldsymbol{y_{new} = y_{current} + step\_length * sin(\theta)}$$

5. Check Validity of New Position
   if $(\boldsymbol{x_{new}, y_{new}})$ is within the search space boundaries.
   Accept the step.
   else
   Repeat from Step 2
6. Update current position: position $(\boldsymbol{x_{current}, y_{current}}) = (\boldsymbol{x_{new}, y_{new}})$
7. Repeat Steps 2 to 6 for a predetermined number of iterations or termination criterion is met

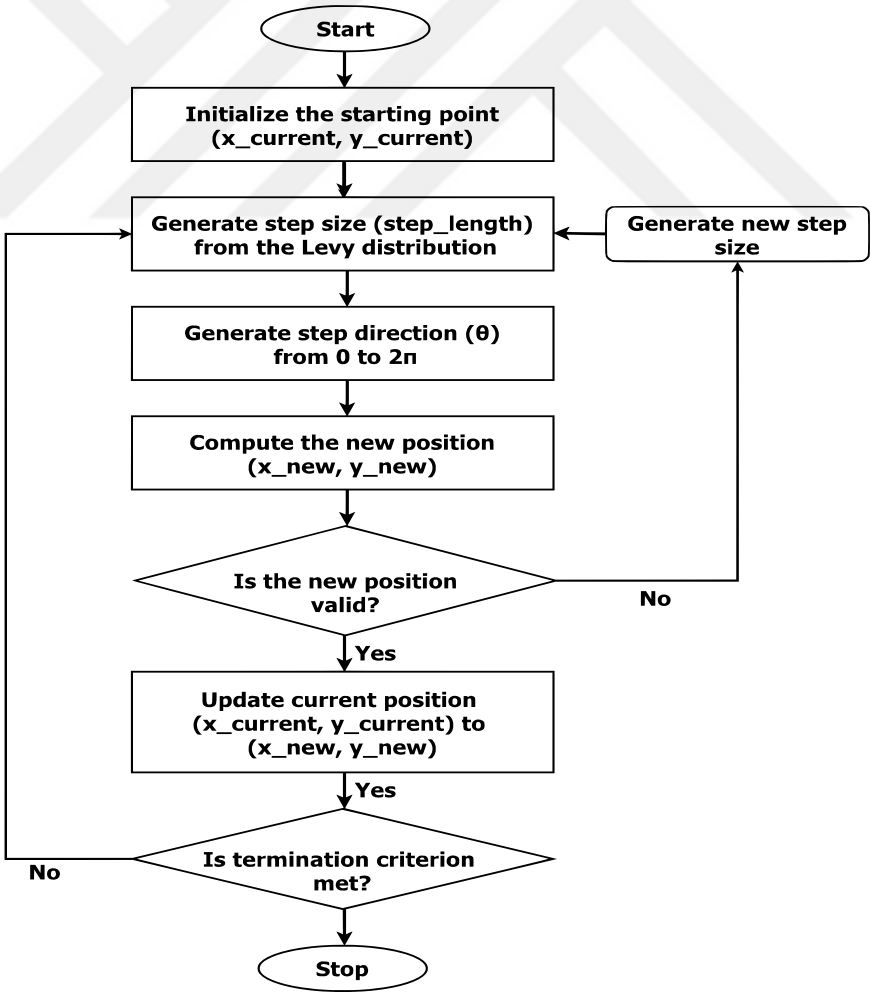


Figure 3.10. Flowchart of LF algorithm

5. SIBERIAN TIGER OPTIMIZATION ALGORITHM

The formation of metaheuristic algorithms has made use of a wide variety of resources for motivation, some of which include natural events, the laws of physics, the rules of games, human interactions, and the field of biology. Strategies that are physics-based, human-based, swarm-based, evolutionary-based, and game-based are the five categories that are applied to categorize metaheuristic algorithms (Trojovsky et al., 2022). In this section, a novel swarm-based method called Siberian Tiger Optimization (STO) is discussed, including its background, mathematical model, features, and implementation steps with a functioning model.

5.1. Overview

The Amur tiger is another name for the Siberian tiger, which is native to the Russian Far East, North Korea, and Northeast China. The Siberian tiger is a member of the Panthera tigris family (Dale G. et al., 2010). According to the many regions in which they are found, the Siberian tiger is also known as the "Amur tiger," "Ussurian tiger," "Korean tiger," and "Manchurian tiger." These titles have been given to the animal under a variety of distinct monikers (Yang et al., 2018). The STO algorithm is an integral part of a category of nature-inspired optimization strategies that have been developed recently (Zhang and Chen, 2023). The Siberian tiger, one of the world's most spectacular and evasive large animals, is the source of both the title and the motivation for the item itself. STO, involving its predecessor, was initially developed to demonstrate the shrewdness and operational aptitude of these magnificent beasts in the context of optimization. This is because STO shares its original name with the game StarCraft II: The Old Republic.

At the beginning of the 21st century, a time frame that was characterized by an increasing fascination with bio-inspired optimization approaches, the STO algorithm was developed. In 2017, Seyedali Mirjalili and Amirhossein Karimi came up with the idea for this algorithm (Zhang and Chen, 2023) and Pavel Trojovsky, Mohammad Dehghani, and Pavel Hanus were the ones to put it into practice (Trojovsky et al., 2022). The predatory behaviors and combat techniques of Siberian tigers served as a source of motivation for them. It was established as a remedy to the demand for cutting-edge algorithms that were suited to tackling difficult and multifaceted optimization issues, and that requirement inspired its creation. The first step in STO's adventure was making the discovery that the

hunting techniques utilized by Siberian tigers are an affirmation of the natural world's skills of optimization. These dominant predators have spent decades perfecting an individual blend of encompassing and hunting attitudes, which allows them to successfully hunt evasive prey in situations that are immense and difficult to navigate.

5.2. Mathematical Model

Mathematical models that mimic Siberian tigers' natural hunting behaviors are the STO algorithm's foundation. It introduces key variables, including the positions of tigers, to navigate the search space efficiently. These positions are organized into vectors, as shown in Equation (5.1). Additionally, the algorithm introduces velocity vectors to signify the direction of movement within the search space.

$$X = \begin{bmatrix} X_1 \\ \vdots \\ X_i \\ \vdots \\ X_N \end{bmatrix}_{N \times m} = \begin{bmatrix} x_{1,1} & \cdots & x_{1,j} & \cdots & x_{1,m} \\ \vdots & \ddots & \vdots & \iddots & \vdots \\ x_{i,1} & \cdots & x_{i,j} & \cdots & x_{i,m} \\ \vdots & \iddots & \vdots & \ddots & \vdots \\ x_{N,1} & \cdots & x_{N,j} & \cdots & x_{N,m} \end{bmatrix}_{N \times m} \tag{5.1}$$

In this context, X symbolizes the population matrix of Siberian tigers' locations, where each X_i represents an individual Siberian tiger, essentially a candidate solution within the optimization process. The parameter denotes the total count of Siberian tigers in the population.

In the STO algorithm, the initial positions of the Siberian tigers within the search space are randomized as the algorithm begins its execution. This randomness is achieved through the utilization of Equation (5.2):

$$x_{i,j} = lb_j + r_{i,j} \cdot \left(ub_j - lb_j\right) \tag{5.2}$$

The symbol $x_{i,j}$ in this formula denotes the jth degree of the location of the tiger in the exploring area. In this case, j is from 1 to m ($j = 1, 2, \ldots, m$) and d is from 1 to N ($d = 1, 2, \ldots, N$). The sample's tiger count is denoted by N, and the entire number of issue variables is indicated by m. Random integers from the range [0, 1] are used to create the variables $r_{i,j}$. Additionally, lb_j and ub_j stand for the lower and upper limits of the jth problem variable, respectively.

Every iteration of the STO algorithm involves constant modification of the tiger positions and movements, mostly through intricate mathematical formulas. A precise

mathematical structure serves as the basis for the STO algorithm's operation, and it may be generally split into two key phases:

A. Prey hunting phase: In the Prey Hunting Phase of the STO algorithm, the Siberian tigers actively search for prey within the defined search space. Their primary objective is to identify and target prey with high fitness values, as these represent potentially optimal solutions. This phase involves a two-stage process for updating the positions of the tigers:

Position Update Stage-1: During this stage, the positions of the tiger population are adjusted based on their selection of and subsequent attack on prey. In the STO framework, each Siberian tiger selects potential prey positions from other members of the population that possess superior objective function values compared to the selecting tiger. This selection process is captured in Equation (5.3):

$$PP_i = \{X_k | k \in \{1,2,\dots,N\} \wedge F_k < F_i\} \cup \{x_{best}\} \tag{5.3}$$

In this equation, x_{best} represents the position of the best candidate solution, essentially the top performing STO member. Here, N stands for the total number of STO members, F_k for the vector of objective function values, and F_i for the objective function value that the i—the Siberian tiger—obtained. So, we can represent F as a vector function according to equation (5.4):

$$F = \begin{bmatrix} F_1 \\ \vdots \\ F_i \\ \vdots \\ F_N \end{bmatrix}_{N \times 1} = \begin{bmatrix} F(X_1) \\ \vdots \\ F(X_i) \\ \vdots \\ F(X_N) \end{bmatrix}_{N \times 1} \tag{5.4}$$

Member positions are altered in the STO algorithm based on the objective function improvement principle. This concept is expressed mathematically in Equation (5.5):

$$X_i = \begin{cases} X_i^{P1S1}, & F_i^{P1S1} < F_i; \\ X_i, & else, \end{cases} \tag{5.5}$$

Within this formula, F_i^{P1S1} represents the objective function value for the current position X_i^{P1S1} of the i-th member. If $TP_{i,j}$ denotes the jth dimension of the tiger position TP_i, while X_i^{P1S1} signifies the new position of the i-th member, this new position is determined by simulating the attack on prey, as specified in Equation (5.6).

$$X_{i,j}^{P1S1} = x_{i,j} + r_{i,j} \cdot (TP_{i,j} - I_{i,j} \cdot x_{i,j}) \tag{5.6}$$

Where $i = 1, 2, \dots, N$, $j = 1, 2, \dots, M$ and $r_{i,j}$ represents random numbers within the interval [0, 1]. Furthermore, $I_{i,j}$ comprises random numbers drawn from the set {1,2}.

Position Update Stage 2: In the second stage of position update within the STO algorithm, the population members adjust their positions through a chase process. During this process, a Siberian tiger modifies its location while actively engaging with its prey. To replicate this chase, the algorithm follows a two-step approach:

Step 1: First, the algorithm calculates a new position for the Siberian tiger in proximity to the location of the prey it is pursuing. This new position is determined using Equation (5.7):

$$X_{i,j}^{P1S2} = x_{i,j} + \frac{r_{i,j} \cdot (ub_j - lb_j)}{t}, \; i = 1, 2, \ldots, N \;, \; j = 1, 2, \ldots, m \text{ and } \; t = 1, 2, \ldots, T \quad (5.7)$$

where $X_{i,j}^{P1S2}$ denotes the new position of the ith Siberian tiger in the jth dimension, and t represents the iteration counter of the algorithm.

Step 2: Following the calculation of the new position, the algorithm checks whether this change results in an improved objective function value, as per Equation (5.8):

$$X_i = \begin{cases} X_i^{P1S2}, & F_i^{P1S2} < F_i; \\ X_i, & else, \end{cases} \quad (5.8)$$

In this equation, X_i represents the position of the ith Siberian tiger. If the objective function value F_i^{P1S2} for the newly calculated position is better than the previous value F_i, the algorithm adopts the new position. Otherwise, it retains the tiger's current position.

B. Fighting Phase: In the Fighting Phase of the STO algorithm, Siberian tigers engage in combat with black bears. These confrontations arise from disputes over prey and the need to safeguard their territory and lives. Winning these fights grants the tigers more opportunities to secure prey and assert dominance over larger areas. This phase comprises three distinct stages:

i. Selection: During this stage, the tigers are chosen for combat based on their fitness values. Tigers with higher fitness values are more likely to be selected for battle. This selection process induces significant and abrupt changes in the positions of STO members, thereby augmenting the algorithm's global search and exploration capabilities. To replicate these concepts, a new position is initially computed for the i-th STO member, where i ranges from 1 to N, following Equation (5.9):

$$x_{i,j}^{P2S1} = \begin{cases} x_{i.j} + r_{i,j} \cdot \left(x_{k,j} - I_{i,j} \cdot x_{i,j}\right), & F_k < F_i; \\ x_{i.j} + r_{i,j} \cdot (x_{i,j} - I_{i,j} \cdot x_{k,j}), & else, \end{cases} \quad (5.9)$$

Here $x_{k,j}$ represents the j-th dimension of a bear's location, with j ranging from 1 to m. The variable k is randomly selected from the set $\{1,2, \ldots, i-1, i+1, \ldots, N\}$, x_i^{P2S1} signifies the new position of the i-th member based on the 1st stage of the 2nd phase of

STO, and $r_{i,j}$ represents random numbers within the interval [0, 1]. Furthermore, $I_{i,j}$ comprises random numbers drawn from the set {1,2}. Subsequently, in the calculation of the new position, the algorithm employs Equation (5.10) to determine whether the objective function value (F_i^{P2S1}) for the 2nd phase 1st stage position is improved. If improvement is detected, the newly calculated position supplants the previous position of the corresponding member.

$$X_i = \begin{cases} X_i^{P2S1}, & F_i^{P2S1} < F_i; \\ X_i, & else, \end{cases} \tag{5.10}$$

ii. Fight: During the 2nd stage of the Fighting Phase in the STO algorithm, the positions of the population members undergo updates to simulate conflicts that occur during combat. This behavior induces subtle adjustments in the positions of the population members, resulting in enhanced local search capability and exploitation ability for STO. To replicate this behavior, a new position is first computed near the location of the confrontation using Equation (5.11):

$$X_{i,j}^{P2S2} = x_{i,j} + \frac{r_{i,j} \cdot (ub_j - lb_j)}{t}, \; i = 1, 2, \ldots, N \;, \; j = 1, 2, \ldots, m \text{ and } \; t = 1, 2, \ldots, T \tag{5.11}$$

In this equation, X_i^{P2S2} reflects the new position of the i-the Siberian tiger in the j-th dimension, and t represents the iteration counter of the algorithm.

iii. Update of Tigers: After determining the new position, the algorithm evaluates if it is appropriate for the update procedure based on whether it raises the value of the objective function, as shown by Equation (5.12):

$$X_i = \begin{cases} X_i^{P1S2}, & F_i^{P2S2} < F_i; \\ X_i, & else, \end{cases} \tag{5.12}$$

Where X_i denotes the position of the i-th Siberian tiger. The algorithm adopts the newly calculated position X_i^{P2S2} if the objective function value F_i^{P2S2} for that position is better than the previous value F_i .

5.3. Operational Framework

This subsection describes the functioning of STO, including the steps of the STO algorithm, its operation of optimization, the pseudocode of the STO algorithm, and its flowchart. Mainly, STO operates through a series of iterations. In each iteration, the algorithm simulates the encircling and attacking behaviors. It evaluates the fitness of each solution and determines which solutions are eligible for "attacking" (Trojovsky et al., 2022).The algorithm updates the positions and velocities of solutions based on these

evaluations, gradually converging toward optimal solutions. This process balances exploration to escape local optima and exploitation to refine solutions.

The STO pseudocode initializes a population of tigers and iteratively improves their placement during prey hunting and fighting. Tigers alter their positions based on prey selection and hunting behaviors during the food hunt-seeking phase and battles during combat. The iterative process continues until a limit is reached. Algorithm 5.1. illustrates the pseudocode of the STO algorithm.

The processes involved in the STO algorithm are depicted in a flowchart for easier comprehension. It offers a graphical description of how the algorithm moves from one stage to another, leading readers step-by-step through the process of optimization, as depicted in Figure 5.1.

Algorithm 5.1. Pseudocode of the STO Algorithm

Input:

$\boldsymbol{N}$: Number of Siberian tigers
$\boldsymbol{T}$: Maximum number of iterations
$\boldsymbol{lb_j}$: Lower bound for problem variable $\boldsymbol{j}$
$\boldsymbol{ub_j}$: Upper bound for problem variable $\boldsymbol{j}$
$\boldsymbol{F_i}$: Objective function value obtained by the $\boldsymbol{i}$ -th Siberian tiger

Initialization:

Initialize the population matrix of Siberian tigers' locations X randomly within problem constraints.
Set the iteration counter $t = 1$.

Evaluation and Updating:

While $\boldsymbol{t \leq T}$ (where T is the maximum number of iterations).

a. Prey Hunting Phase:
   i. Stage-1: Position Update
   For each Siberian tiger $\boldsymbol{i}$ in the population:
   - Calculate a new position $P1S1$ based on the 1st stage of the 1st phase using Equation (5.6):

$$X_{i,j}^{P1S1} = x_{i,j} + r_{i,j} \cdot (TP_{i,j} - I_{i,j} \cdot x_{i,j})$$

   - If $F_i^{P1S1} < F_i$, update the position to $P1S1$ as per Equation (5.5):

$$X_i = \begin{cases} X_i^{P1S1}, & F_i^{P1S1} < F_i; \\ X_i, & else, \end{cases}$$

   ii. Stage-2: Position Update
   For each Siberian tiger i in the population:
   - Calculate a new position $P1S2$ based on the 2nd stage of the 1st phase using Equation (5.7):

$$X_{i,j}^{P1S2} = x_{i,j} + \frac{r_{i,j} \cdot (ub_j - lb_j)}{t}, \; i = 1, 2, \dots, N \;, j = 1, 2, \dots, m \text{ and } \; t = 1, 2, \dots, T$$

   - If $F_i^{P1S2} < F_i$, update the position to $P1S2$ as per Equation (5.8):

$$X_i = \begin{cases} X_i^{P1S2}, & F_i^{P1S2} < F_i; \\ X_i, & else, \end{cases}$$

b. Fighting Phase:
   i. Stage-1: Selection
   For each Siberian tiger i in the population:
   - Calculate a new position $P2S1$ based on the 1st stage of the 2nd phase using Equation (5.9):

$$x_{i,j}^{P2S1} = \begin{cases} x_{i.j} + r_{i,j} \cdot \left(x_{k,j} - I_{i,j} \cdot x_{i,j}\right), & F_k < F_i; \\ x_{i.j} + r_{i,j} \cdot (x_{i,j} - I_{i,j} \cdot x_{k,j}), & else, \end{cases}$$

   - If $F_i^{P2S1} < F_i$, update the position to $P2S1$ as per Equation (5.10):

$$X_i = \begin{cases} X_i^{P2S1}, & F_i^{P2S1} < F_i; \\ X_i, & else, \end{cases}$$

   ii. Stage-2: Fight
   For each Siberian tiger i in the population:
   - Calculate a new position $P2S2$ based on the 2nd stage of the 2nd phase using Equation (5.11).

$$X_{i,j}^{P2S2} = x_{i,j} + \frac{r_{i,j} \cdot (ub_j - lb_j)}{t}, \; i = 1, 2, \dots, N \;, \; j = 1, 2, \dots, m \text{ and } \; t = 1, 2, \dots, T$$

   - If $F_i^{P2S2} < F_i$, update the position to $P2S2$ as in Equation (5.12).

$$X_i = \begin{cases} X_i^{P1S2}, & F_i^{P2S2} < F_i; \\ X_i, & else, \end{cases}$$

c. Increment Iteration Counter:
   Increase iteration counter t by $\boldsymbol{t = t + 1}$

EndWhile

Output:

$\boldsymbol{X_{best}}$: Best position found in the population.
$\boldsymbol{F_{best}}$: Fitness value of the best solution found

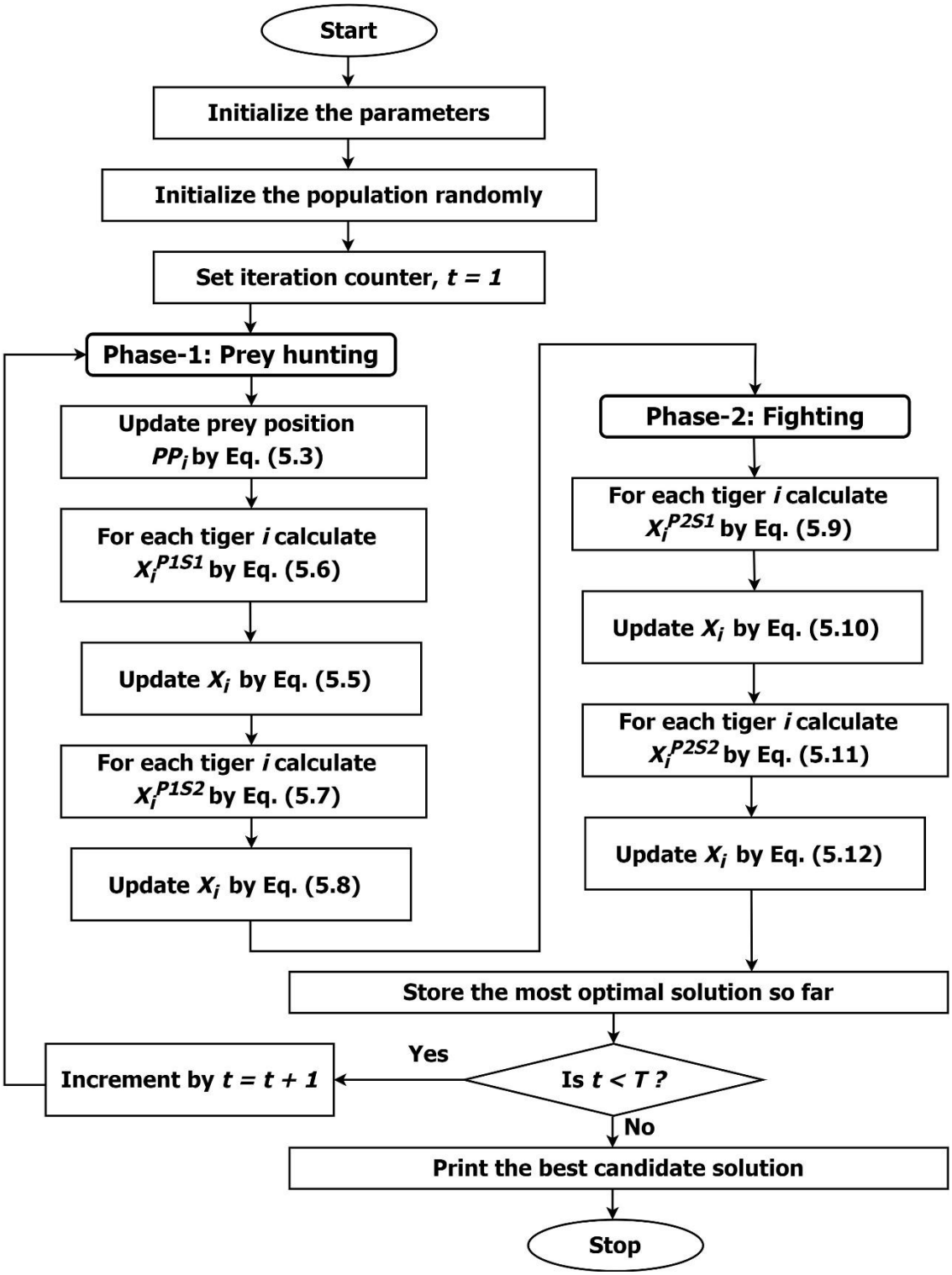


Figure 5.1. Flowchart of STO

6. MATERIAL AND METHOD

The methodological basis of this thesis is based on the hybridization of the LF technique and the STO algorithm. LF was chosen as an activator due to its ability to introduce diversity in the candidate solution vector. This chapter provides information on the resources that were utilized in this study and an in-depth exposition of the selected methodologies will be offered, with the goals of illuminating their harmony with the research aims and describing their functions within the several phases of the dissertation, respectively.

6.1. Materials

6.1.1. CEC2017 Benchmark Function

The proposed LF-integrated STO strategy is put through its pace to see how well it can solve optimization problems using common benchmark functions from the CEC-2017 test suite. The thirty functions that make up the CEC-2017 (Wu et al., 2017) have been broken down into four categories: unimodal (F1-F3), multimodal (F4-F11), hybrid (F12-F20), and lastly, composition functions (F21-F30) (Abualigah et al., 2021; Trojovsky et al., 2022). Table 6.1. (Awad et al., 2016; Abualigah et al., 2021) contains an outline of the CEC-2017 test function, including its specifications, where N denotes the problem dimensionality, which refers to the total number of dimensions or variables that are contained within the function, Fi is commonly employed to express the number of instances or index of the function that is contained inside the benchmark set, and function type provides information on the characteristic or classification of the function (Awad et al., 2016; Kreischer and Barbosa, 2017; Piotrowski et al., 2023; Sharma and Raju, 2023).

6.2. Proposed Strategies

This subsection constitutes the focal point of this study, wherein an exhaustive elucidation of the proposed strategies is presented alongside comprehensive details regarding all offered procedures. The proposed strategies are categorized into four separate approaches which comprise LFSTO-direct, LFSTO-randprob, LFSTO-trial, and LFSTO-trapped. While LFSTO-randprob investigates random value approaches, LFSTO-direct is concerned with direct integration techniques. While LFSTO-trapped focuses on strategies for trapping mechanisms, LFSTO-trial is devoted to trial-based

approaches. The experimental technique relies on these tactics, each suited to the study's aims and findings.

Table 6.1. Overview of CEC-2017 benchmark test function problems

No.	Function Name	N	Fi	Function Type
F1	Shifted and Rotated Bent Cigar Function	-	100	Unimodal Functions
F2	Shifted and Rotated Sum of Differential Power Function	-	200	
F3	Shifted and Rotated Zakharov Function	-	300	
F4	Shifted and Rotated Rosenbrock's Function	-	400	Simple Multimodal Functions
F5	Shifted and Rotated Rastrigin's Function	-	500	
F6	Shifted and Rotated Expanded Scaffer's F6 Function	-	600	
F7	Shifted and Rotated Lunacek Bi-Rastrigin Function	-	700	
F8	Shifted and Rotated Non-Continuous Rastrigin's Function	-	800	
F9	Shifted and Rotated Levy Function	-	900	
F10	Shifted and Rotated Schwefel's Function	-	1000	
F11	Hybrid Function 1	3	1100	Hybrid Functions
F12	Hybrid Function 2	3	1200	
F13	Hybrid Function 3	3	1300	
F14	Hybrid Function 4	4	1400	
F15	Hybrid Function 5	4	1500	
F16	Hybrid Function 6	4	1600	
F17	Hybrid Function 6	5	1700	
F18	Hybrid Function 6	5	1800	
F19	Hybrid Function 6	5	1900	
F20	Hybrid Function 6	6	2000	
F21	Composition Function 1	3	2100	Composition Functions
F22	Composition Function 2	3	2200	
F23	Composition Function 3	4	2300	
F24	Composition Function 4	4	2400	
F25	Composition Function 5	5	2500	
F26	Composition Function 6	5	2600	
F27	Composition Function 7	6	2700	
F28	Composition Function 8	6	2800	
F29	Composition Function 9	3	2900	
F30	Composition Function 10	3	3000	

6.2.1. LFSTO-direct

The proposed LFSTO-direct algorithm is a direct LF integration method that combines the best features of two different algorithms, the STO algorithm and Levy Flight distributions, to make a new and effective optimization strategy that makes the most of each one's strengths. We have reviewed several studies related to LF with metaheuristic algorithms in Section 2 of our study. We can see that in most of the studies, authors improved or modified the original algorithms utilized by LF using the direct incorporation method. Especially, Cui et al. (Cui et al., 2020), Heidari and Pahlavani (Heidari and Pahlavani, 2017), Mujawib Alashjaee Abdullah (Mujawib Alashjaee, 2023), Ba et al. (Ba et al., 2020), Ling et al. (Ling et al., 2017), and Addulwahab et al. (Abdulwahab et al., 2019), etc. employed LF in the position update step using the entry wise multiplication ($\oplus$) operator.

In general, STO has two phases, namely prey hunting and the fighting phase. In the prey hunting (exploration) phase, candidates proactively alter solutions depending on neighboring better solutions, employing randomized motions and directions from the population's optimal solutions to discover new locations. Fighting (exploitation) with bear phase, where candidates replicate the predator-prey movement and modify their responses according to other candidates' performance. This stage promotes exploitation by expanding solutions via rivals and evolution to surpass the population.

Fundamentally, LFSTO-direct is a technique that iteratively moves proactively across solution regions. A large number of different possible solutions are generated during its startup phase by using a uniform random distribution within certain limitations. Through a complex and multi-step procedure, this population is continuously refined to get the desired results. In this proposed strategy, we added LF only in the prey hunting phase of STO. According to the STO principle, the prey hunting phase has two position-update stages, so in the LFSTO algorithm, update the new position of the first stage of the first phase as in Equation (6.1) by modifying Equation (5.6).

$$X_{i,j}^{P1S1} = x_{i,j} + r_{i,j} \cdot \left(TP_{i,j} - I_{i,j} \cdot x_{i,j}\right) \bigoplus levy(step) \tag{6.1}$$

In Equation (6.1), the LF function is integrated directly with the traditional position update stage of STO. In this circumstance, the value of the LF distribution may cross the limit or bounds, and the effect of this on the $X_{i,j}$ can produce values beyond its bounds. To guarantee that population variables endure within predetermined boundaries stipulated

by the lower bound (lb) and upper bound (ub), it is imperative to consistently utilize the clip operation during the experiment. In this case, $levy(step)$ represents the LF random distribution and is expressed as:

$$levy(step) = \frac{U}{V^{1/\beta}} \tag{6.2}$$

Where β is the stability index, V is a random number derived from the distribution of Gaussian values $(0,1)$, and U is a random number selected from the Gaussian distribution $(0, \sigma^2)$, and standard deviation σ defined as:

$$\sigma = \left[\frac{\gamma(1+\beta) * \sin\left(\frac{\pi+\beta}{2}\right)}{\gamma\left(\frac{1+\beta}{2}\right) * \beta * 2^{\frac{\beta-1}{2}}} \right]^{1/\beta} \tag{6.3}$$

where γ is a representation of the gamma function.

In the same way, update the new position of the second stage of the first phase as in Equation (6.4) by modifying Equation (5.7) via Levy distribution.

$$X_{i,j}^{P1S2} = \left[x_{i,j} + \frac{r_{i,j} \cdot (ub_j - lb_j)}{t} \right] \oplus levy(step) \,, \; i = 1, 2, \dots, N \;\; , j = 1, 2 \,, \dots, m \quad \text{and} \;\; t = 1, 2 \,, \dots, T \tag{6.4}$$

where $\oplus$ is an entrywise multiplication product. Once more, for Equation (6.4), after integrating the operation, the clip function is applied to control and produce the values of LF as well as $X_{i,j}$ within the aforementioned boundaries.

The LFSTO-direct method's prey-hunting (exploration) phase is based on the real-life actions of prey animals that seek out more advantageous situations. Employing randomized movements derived from LF and combining guidelines from the most optimal solutions in the overall population, every potential solution dynamically adapts itself. By combining the best solutions with the unpredictability of LF, we may explore unexplored territory while still making use of the best solutions. This strikes a compromise between both investigation and extraction. At the same time, a competitive advantage is introduced during the battling with bears phase. Candidates compete to copy or defend against poor solutions. This stage creates a dynamic adaptation process that adjusts solutions based on performance and pool interactions. Because of this, the algorithm is able to escape the local optima more easily. However, the LFSTO-direct approach does not integrate the LF approach into the STO second phase.

The key point is that LFSTO-direct maintains a well-balanced exploration-exploitation ratio by dynamically adjusting the LF step size throughout iterations. To maximize the algorithm's performance in complex solution environments, the step size

Algorithm 6.1. Pseudocode of the LFSTO-direct

Input:

$\boldsymbol{N}$: Number of Siberian tigers
$\boldsymbol{T}$: Maximum number of iterations
$\boldsymbol{lb_j}$: Lower bound for problem variable $\boldsymbol{j}$
$\boldsymbol{ub_j}$: Upper bound for problem variable $\boldsymbol{j}$
$\boldsymbol{F_i}$: Objective function value obtained by the $\boldsymbol{i}$ -th Siberian tiger
$\boldsymbol{\beta}$: Stability index
$\boldsymbol{V}$: Gaussian distribution $(0,\ 1)$,
$\boldsymbol{U}$: the Gaussian distribution $(0, \sigma^2)$

Initialization:

Initialize the population matrix of Siberian tigers' locations X randomly within problem constraints.
Set the iteration counter $t = 1$.
Initialize the $step\ size$

Evaluation and Updating:

While $\boldsymbol{t \leq T}$ (where T is the maximum number of iterations).

a. Prey Hunting Phase:
   i. Stage-1: Position Update
   For each Siberian tiger $\boldsymbol{i}$ in the population:
   - Calculate a new position $P1S1$ based on the 1st stage of the 1st phase using Equation (6.1):

$$X_{i,j}^{P1S1} = x_{i,j} + r_{i,j} \cdot (TP_{i,j} - I_{i,j} \cdot x_{i,j}) \oplus levy(step)$$

   Check solution within boundaries or not by clip $(P1S1, lb_j, ub_j)$
   - If $F_i^{P1S1} < F_i$, update the position to $P1S1$ as per Equation (5.5):

$$X_i = \begin{cases} X_i^{P1S1}, & F_i^{P1S1} < F_i; \\ X_i, & else, \end{cases}$$

   ii. Stage-2: Position Update
   For each Siberian tiger i in the population:
   - Calculate a new position $P1S2$ based on the 2nd stage of the 1st phase using Equation (6.4):

$X_{i,j}^{P1S2} = \left[\boldsymbol{x_{i,j}} + \frac{\boldsymbol{r_{i,j}} \cdot (\boldsymbol{ub_j} - \boldsymbol{lb_j})}{\boldsymbol{t}}\right] \oplus levy(step)$, $i = 1, 2, \dots, N$, $j = 1, 2, \dots, m$ and $t = 1, 2, \dots, T$

   Check solution within boundaries or not by clip $(P1S2, lb_j, ub_j)$
   - If $F_i^{P1S2} < F_i$, update the position to $P1S2$ as per Equation (5.8):

$$X_i = \begin{cases} X_i^{P1S2}, & F_i^{P1S2} < F_i; \\ X_i, & else, \end{cases}$$

b. Fighting Phase:
   i. Stage-1: Selection
   For each Siberian tiger i in the population:
   - Calculate a new position $P2S1$ based on the 1st stage of the 2nd phase using Equation (5.9):

$$x_{i,j}^{P2S1} = \begin{cases} x_{i.j} + r_{i,j} \cdot \left(x_{k,j} - I_{i,j} \cdot x_{i,j}\right), & F_k < F_i; \\ x_{i.j} + r_{i,j} \cdot (x_{i,j} - I_{i,j} \cdot x_{k,j}), & else, \end{cases}$$

   - If $F_i^{P2S1} < F_i$, update the position to $P2S1$ as per Equation (5.10):

$$X_i = \begin{cases} X_i^{P2S1}, & F_i^{P2S1} < F_i; \\ X_i, & else, \end{cases}$$

   ii. Stage-2: Fight
   For each Siberian tiger i in the population:
   - Calculate a new position $P2S2$ based on the 2nd stage of the 2nd phase using Equation (5.11).

$X_{i,j}^{P2S2} = x_{i,j} + \frac{r_{i,j} \cdot (ub_j - lb_j)}{t}$, $i = 1, 2, \dots, N$, $j = 1, 2, \dots, m$ and $t = 1, 2, \dots, T$

   - If $F_i^{P2S2} < F_i$, update the position to $P2S2$ as in Equation (5.12)

Algorithm 6.1. Pseudocode of the LFSTO-direct (continued)

$$X_i = \begin{cases} X_i^{P1S2}, & F_i^{P2S2} < F_i \\ X_i, & else, \end{cases};$$

c. Increment Iteration Counter:
Increase iteration counter t by $\boldsymbol{t = t + 1}$

EndWhile

Output:

$\boldsymbol{X_{best}}$: Best position found in the population.
$\boldsymbol{F_{best}}$: Fitness value of the best solution found

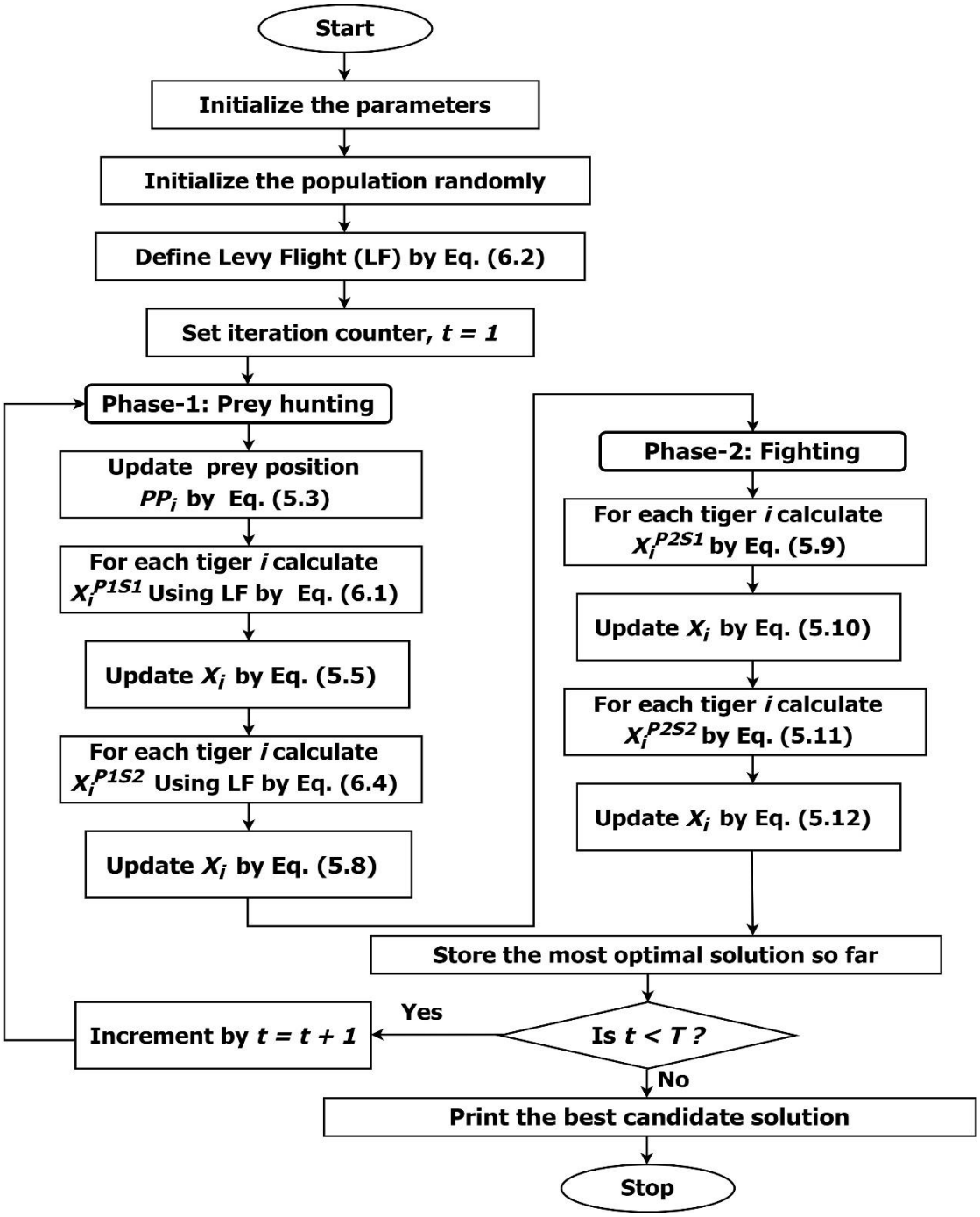


Figure 6.1. Flowchart of LFSTO-direct strategy

adaptation is utilized to achieve a smooth transition from global exploration to local exploitation. The pseudocode of LFSTO-direct is illustrated in Algorithm 6.1, and the corresponding flowchart is depicted in Figure 6.1. A thorough evaluation of the convergence qualities of the LFSTO-direct integration approach is possible since it operates across several runs and iterations. The method overcomes the limits of traditional STO by gathering the best solutions and their fitness values, then repeatedly improving itself until it converges to optimum or near-optimal solutions.

6.2.2. LFSTO-randprob

In the second proposed LFSTO-randprob strategy, the exploration-exploitation decision-making process plays a pivotal role in balancing search space exploration (via LF) and exploiting promising regions (through a rival collaboration). This decision-making is probabilistic and uses a random number comparison (random probability = 0.5) to choose between two distinct phases of the original STO. The authors (Jensi and Jiji, 2016) implemented the PSOLF algorithm by modifying the LFPSO algorithm (Haklı and Uğuz, 2014) via the LF method and applying this conditional approach. Like the LFSTO-direct approach, LFSTO-randprob added Levy distribution directly to only the first phase of standard STO. However, the single difference is that the second proposed algorithm used a random value condition.

The algorithm considers updating the tigers (populations) velocity and position similarly to the original STO with a condition based on a random probability check. When the generated random probability is greater than or equal to 0.5, the standard update equations (Eqs. 5.6 and 5.7) from the STO are employed. However, if the randomly generated value is less than 0.5, it prompts a different position update for the first stage (p1s1) of the hunting phase using Equation (6.1) and the second stage (p1s2) of the hunting phase using Equation (6.4). This update involves utilizing the LF method, which enables the member to execute a more extensive jump toward its local and global best. This significant flight assists in enhancing the diversity within the flock, enabling the algorithm to explore the entire search space more extensively. Consequently, it facilitates the algorithm's efforts for global exploration.

Thus, this criterion is applied in the scenario of LFSTO-randprob to generate a random selection between two alternatives with a 50% probability for each step of the first phase. It proceeds to the LF integration step if the requirement is satisfied and to the

traditional STO step otherwise. Algorithms may be made to pick between several pathways or actions by introducing unpredictability or randomness through stochastic decision-making processes. In addition, the iterative process constantly tweaks the adaptive step size that is employed in LF. The number of iterations determines the dynamic adaptation. The continual fine-tuning of the Levy distribution function is the goal of the adjustment process. This adaptable technique enhances the algorithm's ability to thoroughly investigate the exploratory area and converge on optimal or nearly optimal solutions. Algorithm 6.2 and Figure 6.2 provide the pseudocode and related flowchart of this method, respectively.

6.2.3. LFSTO-trial

The LFSTO-trial approach gets around standard STO's local optima by adding a trial mechanism that changes how candidate solutions search based on how well they've worked in the past. This study went through several LF distribution-based metaheuristic algorithms in the literature and investigated the LF incorporation technique with those swarm-based algorithms. Haklı and Uğuz (2014) work stands out as particularly important from all of that research (Holland, 1992; Kennedy, 1995; Krishnanand, 2005; Karaboga and Basturk, 2007; Yang, 2010b; Yang and Deb, 2013). They created a new LFPSO algorithm by combining levy distribution with trial and limit value mechanisms.

This LFSTO-trial algorithm incorporates a trial mechanism to dynamically adjust candidate solutions' hunting (exploration) strategy based on their success or failure. Similar to LFSTO-direct, this thesis work proposed a technique that involved randomly distributing LF directly to the prey-hunting phase of the original STO only. The trial mechanism is governed by a counter variable, `Trial`, which is initialized to zero for each candidate solution, and the limit value is set at the same time. The algorithm employs this counter to distinguish between two conditional steps within the two-position update stage of the prey-hunting phase of STO.

First, a typical hunting strategy without LF random movements is used if the 'Trial' counter is below a certain limit. This implies that regular position upgrading is carried out by applying Equation (5.6) for the first stage (P1S1) and Equation (5.7) for the second stage (P1S2). If it is determined that the new solution represents an improvement, the counter is reset. In contrast, the method moves on to the second conditional step and uses

modified LF distributions to explore the solution space if the counter 'Trial' goes higher than the limit.

Algorithm 6.2. Pseudocode of the LFSTO-randprob

Input:

$\boldsymbol{N}$: Number of Siberian tigers
$\boldsymbol{T}$: Maximum number of iterations
$\boldsymbol{lb_j}$: Lower bound for problem variable $\boldsymbol{j}$
$\boldsymbol{ub_j}$: Upper bound for problem variable $\boldsymbol{j}$
$\boldsymbol{F_i}$: Objective function value obtained by the $\boldsymbol{i}$ -th Siberian tiger.
$\boldsymbol{\beta}$: Stability index
$\boldsymbol{V}$: Gaussian distribution $(0,\ 1)$,
$\boldsymbol{U}$: The Gaussian distribution $(0, \sigma^2)$

Initialization:

Initialize the population matrix of Siberian tigers' locations X randomly within problem constraints.
Set the iteration counter $t = 1$.
Initialize the $step\ size$

Evaluation and Updating:

While $\boldsymbol{t \leq T}$ (where T is the maximum number of iterations).

a. Prey Hunting Phase:
   i. Stage-1: Position Update
   For each Siberian tiger $\boldsymbol{i}$ in the population:
   - If rand () < 0.5 then
     Calculate a new position $P1S1$ based on the 1st stage of the 1st phase using Equation (6.1):
     $$X_{i,j}^{P1S1} = x_{i,j} + r_{i,j} \cdot (TP_{i,j} - I_{i,j} \cdot x_{i,j}) \oplus levy(step)$$
     Check solution within boundaries or not by clip $(P1S1, lb_j, ub_j)$
   - Else
     Calculate a new position $P1S1$ based on the 1st stage of the 1st phase using Equation (5.6):
     $$X_{i,j}^{P1S1} = x_{i,j} + r_{i,j} \cdot (TP_{i,j} - I_{i,j} \cdot x_{i,j})$$
   - End If
   - If $F_i^{P1S1} < F_i$, update the position to $P1S1$ as per Equation (5.5):
     $$X_i = \begin{cases} X_i^{P1S1}, & F_i^{P1S1} < F_i; \\ X_i, & else, \end{cases}$$

   ii. Stage-2: Position Update
   For each Siberian tiger i in the population:
   - If rand () < 0.5 then
     Calculate a new position $P1S2$ based on the 2nd stage of the 1st phase using Equation (6.4):
     $X_{i,j}^{P1S2} = \left[\boldsymbol{x_{i,j}} + \frac{\boldsymbol{r_{i,j}} \cdot (\boldsymbol{ub_j} - \boldsymbol{lb_j})}{\boldsymbol{t}}\right] \oplus levy(step)$, $i = 1, 2, \dots, N$, $j = 1, 2, \dots, m$ and $t = 1, 2, \dots, T$
     Check solution within boundaries or not by clip $(P1S2, lb_j, ub_j)$
   - Else
     Calculate a new position $P1S2$ based on the 2nd stage of the 1st phase using Equation (5.7):
     $X_{i,j}^{P1S2} = \left[\boldsymbol{x_{i,j}} + \frac{\boldsymbol{r_{i,j}} \cdot (\boldsymbol{ub_j} - \boldsymbol{lb_j})}{\boldsymbol{t}}\right]$, $i = 1, 2, \dots, N$, $j = 1, 2, \dots, m$ and $t = 1, 2, \dots, T$
   - End If
   - If $F_i^{P1S2} < F_i$, update the position to $P1S2$ as per Equation (5.8):
     $$X_i = \begin{cases} X_i^{P1S2}, & F_i^{P1S2} < F_i; \\ X_i, & else, \end{cases}$$

b. Fighting Phase:
   i. Stage-1: Selection

Algorithm 6.2. Pseudocode of the LFSTO-randprob (continued)

For each Siberian tiger i in the population:

- Calculate a new position $P2S1$ based on the 1st stage of the 2nd phase using Equation (5.9):

$$x_{i,j}^{P2S1} = \begin{cases} x_{i.j} + r_{i,j} \cdot \left(x_{k,j} - I_{i,j} \cdot x_{i,j}\right), & F_k < F_i; \\ x_{i.j} + r_{i,j} \cdot (x_{i,j} - I_{i,j} \cdot x_{k,j}), & else, \end{cases}$$

- If $F_i^{P2S1} < F_i$, update the position to $P2S1$ as per Equation (5.10):

$$X_i = \begin{cases} X_i^{P2S1}, & F_i^{P2S1} < F_i; \\ X_i, & else, \end{cases}$$

ii. Stage-2: Fight

For each Siberian tiger i in the population:

- Calculate a new position $P2S2$ based on the 2nd stage of the 2nd phase using Equation (5.11).

$X_{i,j}^{P2S2} = x_{i,j} + \frac{r_{i,j} \cdot (ub_j - lb_j)}{t}$, $i = 1, 2, \ldots, N$, $j = 1, 2, \ldots, m$ and $t = 1, 2, \ldots, T$

- If $F_i^{P2S2} < F_i$, update the position to $P2S2$ as in Equation (5.12).

$$X_i = \begin{cases} X_i^{P1S2}, & F_i^{P2S2} < F_i; \\ X_i, & else, \end{cases}$$

c. Increment Iteration Counter:
Increase iteration counter t by $\boldsymbol{t = t + 1}$

EndWhile

Output:
$\boldsymbol{X_{best}}$: Best position found in the population.
$\boldsymbol{F_{best}}$: Fitness value of the best solution found

In this way, tigers update their position by using Equation (6.1) for the first stage (P1S1) and Equation (6.4) for the second stage (P1S2), respectively. Once more, the trial is reset whenever a more effective option is discovered. In both scenarios, the strategy seeks the optimal fitness value by comparing the current solution's fitness to that of the new candidate solution. The trial counter is set back to zero, and the best solution (X_{best}) is updated if the current solution is superior. When the current solution remains unchanged, the algorithm gains insight into failed attempts by incrementing the candidate's trial number by 1. This allows it to adaptively tweak its method for searching the solution space.

This trial system allows candidate solutions to experiment with new techniques depending on their prior performance, changing their search in real-time. This facilitates the algorithm's exploration and exploitation of intricate optimization environments. This approach also uses the adaptive levy step size mechanism, similar to LFSTO-direct and LFSTO-randprob. Algorithm 6.3 contains the LFSTO-trial pseudocode, and Figure 6.3 depicts its flow diagram. During the prey hunting phase, the 'Trial' counter directs the algorithm through both exploration and exploitation stages, allowing for a balanced approach to conquering local minima. The trial counter ('Trial') refreshes whenever a possible solution improves during the prey hunting phase, promoting exploration of other

pathways and limiting stubbornness with impoverished techniques. This technique for resetting encourages the discovery of new areas, which helps to escape local optima by encouraging other search tactics.

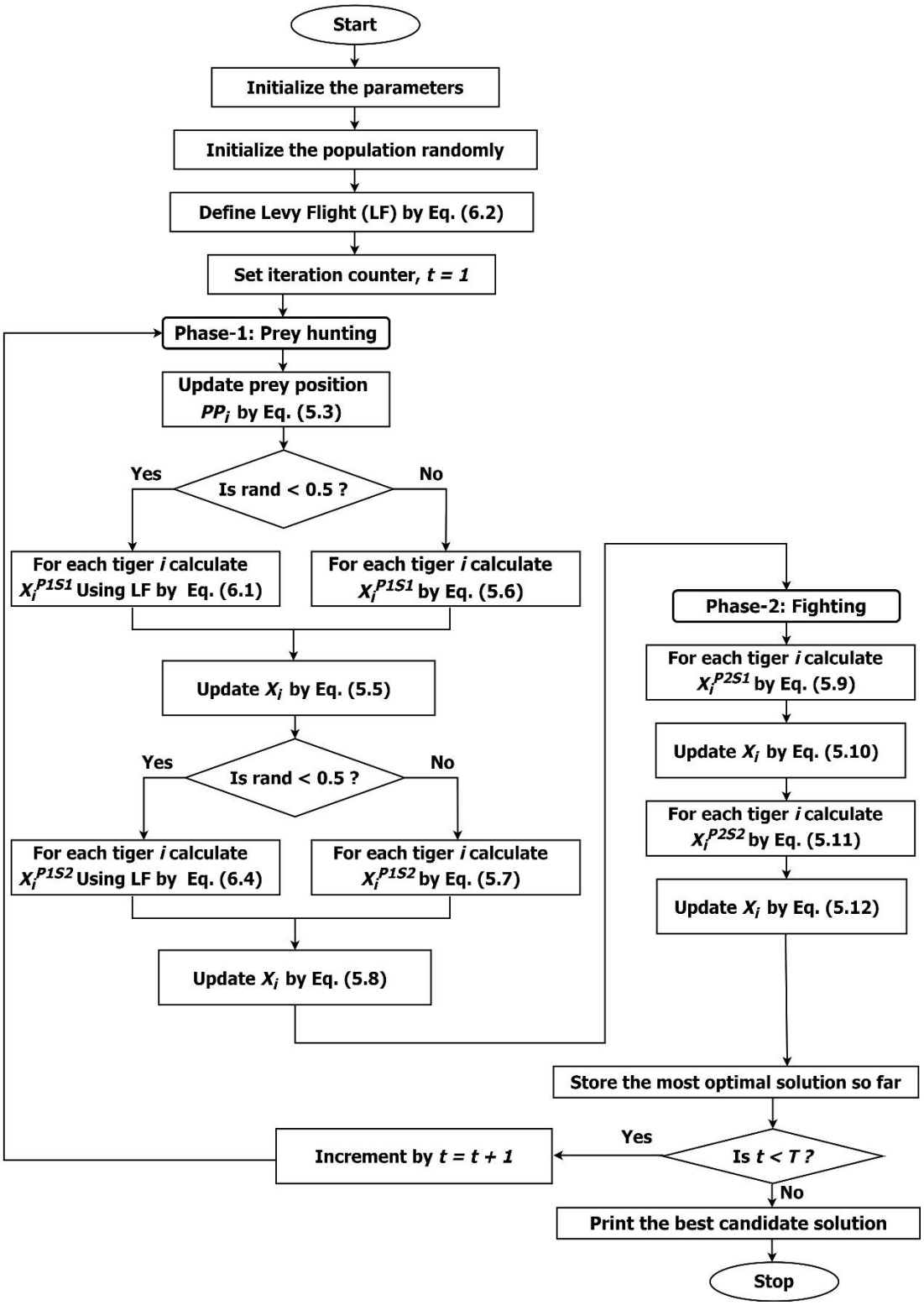


Figure 6.2. Flowchart of LFSTO-randprob strategy

Algorithm 6.3. Pseudocode of the LFSTO-trial

Input:

$\boldsymbol{N}$: Number of Siberian tigers
$\boldsymbol{T}$: Maximum number of iterations
$\boldsymbol{lb_j}$: Lower bound for problem variable $\boldsymbol{j}$
$\boldsymbol{ub_j}$: Upper bound for problem variable $\boldsymbol{j}$
$\boldsymbol{F_i}$: Objective function value obtained by the $\boldsymbol{i}$ -th Siberian tiger
$\boldsymbol{\beta}$: Stability index
$\boldsymbol{V}$: Gaussian distribution $(0,\ 1)$,
$\boldsymbol{U}$: The Gaussian distribution $(0, \sigma^2)$
$\boldsymbol{Trial}$: Initialize the trial counter as 0
$\boldsymbol{Limit}$: Set the limit value

Initialization:

Initialize the population matrix of Siberian tigers' locations X randomly within problem constraints.
Set the iteration counter $t = 1$.
Initialize the $step\ size$

Evaluation and Updating:

While $\boldsymbol{t \leq T}$ (where T is the maximum number of iterations).

a. Prey Hunting Phase:
   i. Stage-1: Position Update
   For each Siberian tiger $\boldsymbol{i}$ in the population:
   - If Trial[i] < Limit, then
     Calculate a new position $P1S1$ based on the 1st stage of the 1st phase using Equation (6.1):
     $$X_{i,j}^{P1S1} = x_{i,j} + r_{i,j} \cdot (TP_{i,j} - I_{i,j} \cdot x_{i,j})$$
   - Else
     Calculate a new position $P1S1$ based on the 1st stage of the 1st phase using Equation (5.6):
     $$X_{i,j}^{P1S1} = x_{i,j} + r_{i,j} \cdot (TP_{i,j} - I_{i,j} \cdot x_{i,j}) \oplus levy(step)$$
     Check solution within boundaries or not by clip $(P1S1, lb_j, ub_j)$
   - End If
   - If $F_i^{P1S1} < F_i$, update the position to $P1S1$ as per Equation (5.5):
     $$X_i = \begin{cases} X_i^{P1S1} \ and\ Trial[i] = 0, & F_i^{P1S1} < F_i; \\ X_i \ and\ Trial[i] = Trial[i] + 1, & else, \end{cases}$$

   ii. Stage-2: Position Update
   For each Siberian tiger i in the population:
   - If Trial[i] < Limit, then
     Calculate a new position $P1S2$ based on the 2nd stage of the 1st phase using Equation (6.4):
     $X_{i,j}^{P1S2} = \left[\boldsymbol{x_{i,j}} + \frac{\boldsymbol{r_{i,j} \cdot (ub_j - lb_j)}}{\boldsymbol{t}}\right]$, $i = 1, 2, \dots, N$, $j = 1, 2, \dots, m$ and $t = 1, 2, \dots, T$
   - Else
     Calculate a new position $P1S2$ based on the 2nd stage of the 1st phase using Equation (5.7):
     $X_{i,j}^{P1S2} = \left[\boldsymbol{x_{i,j}} + \frac{\boldsymbol{r_{i,j} \cdot (ub_j - lb_j)}}{\boldsymbol{t}}\right] \oplus levy(step)$, $i = 1, 2, \dots, N$, $j = 1, 2, \dots, m$ and $t = 1, 2, \dots, T$
     Check solution within boundaries or not by $clip(P1S2, lb_j, ub_j)$
   - End If
   - If $F_i^{P1S2} < F_i$, update the position to $P1S2$ as per Equation (5.8):
     $$X_i = \begin{cases} X_i^{P1S2} \ and\ Trial[i] = 0, & F_i^{P1S2} < F_i; \\ X_i \ and\ Trial[i] = Trial[i] + 1, & else, \end{cases}$$

b. Fighting Phase:
   i. Stage-1: Selection
   For each Siberian tiger i in the population:
   - Calculate a new position $P2S1$ based on the 1st stage of the 2nd phase using Equation (5.9):

Algorithm 6.3. Pseudocode of the LFSTO-trial (continued)

$$x_{i,j}^{P2S1} = \begin{cases} x_{i.j} + r_{i,j} \cdot \left(x_{k,j} - I_{i,j} \cdot x_{i,j}\right), & F_k < F_i; \\ x_{i.j} + r_{i,j} \cdot \left(x_{i,j} - I_{i,j} \cdot x_{k,j}\right), & else, \end{cases}$$

- If $F_i^{P2S1} < F_i$, update the position to $P2S1$ as per Equation (5.10):

$$X_i = \begin{cases} X_i^{P2S1}, & F_i^{P2S1} < F_i; \\ X_i, & else, \end{cases}$$

ii. Stage-2: Fight

For each Siberian tiger i in the population:

- Calculate a new position $P2S2$ based on the 2nd stage of the 2nd phase using Equation (5.11).

$$X_{i,j}^{P2S2} = x_{i,j} + \frac{r_{i,j} \cdot (ub_j - lb_j)}{t}, \; i = 1, 2, \ldots, N \;,\; j = 1, 2, \ldots, m \text{ and } \; t = 1, 2, \ldots, T$$

- If $F_i^{P2S2} < F_i$, update the position to $P2S2$ as in Equation (5.12).

$$X_i = \begin{cases} X_i^{P1S2}, & F_i^{P2S2} < F_i; \\ X_i, & else, \end{cases}$$

c. Increment Iteration Counter:

Increase iteration counter t by $\boldsymbol{t = t + 1}$

EndWhile

Output:

$\boldsymbol{X_{best}}$: Best position found in the population.

$\boldsymbol{F_{best}}$: Fitness value of the best solution found

6.2.4. LFSTO-trap

By including a trapping condition, the LFSTO-trap strategy allows for the technique of search to be adaptively adjusted during the prey-hunting phase of the conventional STO. This fourth offered tactic draws inspiration from the MCSA algorithm (Verma et al., 2023) and the Dragonfly Algorithm (DA) (Mirjalili, 2015), in which both studies used the LF-based random walk searching technique for location updates while trapped in no local solutions. This method checks whether the algorithm got stuck in a trap or not in the local search space via adaptive window size and trap value parameters.

Generally, window size is the measurement or analysis of the number of data elements or objects in a given range or time interval. Signal processing (Kammoun et al., 2022), statistical analysis of time series (Keelawat et al., 2021), and optimization techniques (Serbet and Kaya, 2023) are just a few of the many areas that frequently employ it. To avoid calculations and oversights, the window size is adaptively set to use an appropriate portion of the data for analysis (Katkovnik et al., 2002) The window size in the LFSTO-trap method and trap function refers to the number of latest iterations or fitness values used to evaluate a trapping circumstance. In cases with fewer fitness values than iteration numbers, this evolving adjustment prevents index mistakes and excessive computing by limiting the window size. The trap function is crucial in the LFSTO-trap

approach, modifying search strategies depending on progress in fitness during the hunting phase. Following is a brief explanation of how this adaptive function works, including an example.

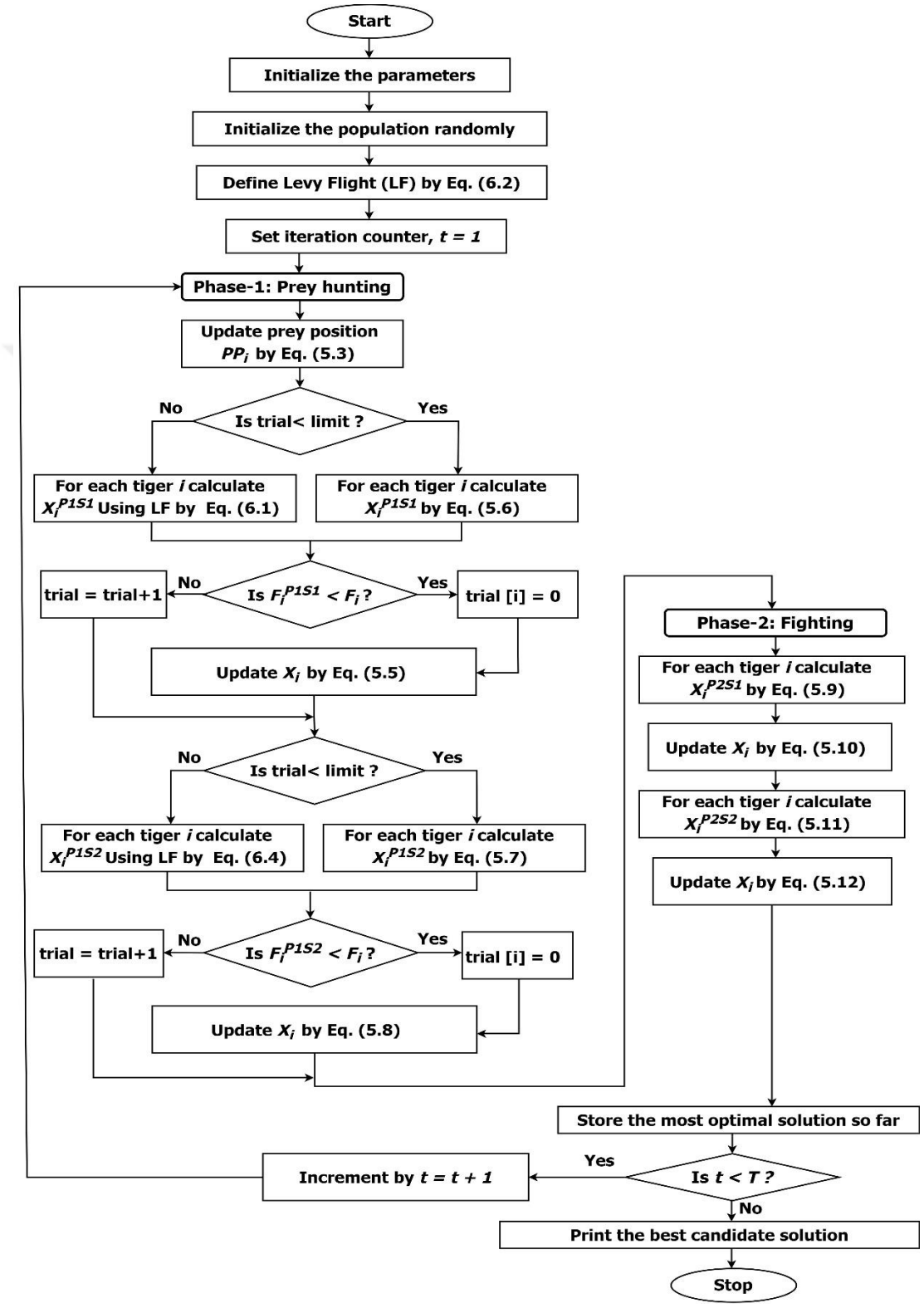


Figure 6.3. Flowchart of LFSTO-trial strategy

The size of the window is constantly changed to the minimum of the most recent iteration count and the total length of the best fitness values. This maintains the window width within fitness values. Step size, iteration number, and the square root of dimension determine the threshold value. For trapping situations, it is an adaptive threshold factor. The method checks for trapping circumstances if the present iteration is larger than the adapted window size, suggesting sufficient fitness values for evaluation. The mean fitness improvement is calculated by subtracting subsequent fitness values within the window. Find out if the most recent improvement exceeds the previously estimated threshold value, indicating a trapping condition. If not, the return value is false.

Assume that there are 10 iterations, a step size of 0.01, and 3 dimensions (dim), and the optimal (best) fitness values to use are [100, 90, 85, 80, 78, 76, 75, 74, 72, 70]. At this step, the function adjusts the window width to 10 (the current iteration). The threshold factor was computed as follows in Equation (6.5).

$$thereshold = step\ size\ (0.01)/\left(window\ size(10) * \sqrt{\dim\ (3)}\right) \approx 0.000192 \quad (6.5)$$

The mean of the most recent improvements inside the specified window ([-10, -5, -5, -2, -2, -1, -1, -2, -2]) is -7.222, after which the calculation is completed. The assertion is valid, indicating the possibility of a trapping condition, because its most recent improvement (-7.222) is less than the threshold value (0.000192).

The criterion '-7.222 < 0.000192' evaluates to be valid in the provided instance. This indicates that the adaptive threshold factor that was computed is greater than the recent improvement in fitness values within the given window width. That is why the function would give a true result, suggesting that the algorithm could be stuck in a trapped state. As a result of the trapping condition, the algorithm might do better with adding a random levy walk during the prey-hunting phase of STO. This would encourage more exploratory movement and could help it break out of local optima.

The LFSTO trap technique includes trap criteria to adapt the search strategy during the hunting phase. Like the previous three proposed methods, this strategy integrates the levy random walk only into the two-position update stage of the first phase (the hunting phase). At first, set the trap function within the N population, then check whether. If the defined trap criteria are true, then the algorithm is in stuck mode and then executes the position update of the first stage (P1S1) via integrating levy walk utilizing Equation (6.1). If the criteria are not met, then the algorithm normally updates the position using Equation (5.6). In the same way, in the second stage of the first phase (P1S2), if the

criteria are met, update its position using Equation (6.4) and otherwise use Equation (5.7) to update its position. This tactic makes use of adjustable step sizes, much like the first three. The algorithm's performance has been improved by avoiding local minima and expanding global search using this trap-condition-based LFSTO-trap method. Algorithm 6.4 presents the pseudocode for this strategy, while Figure 6.4 illustrates the corresponding flowchart.

Algorithm 6.4. Pseudocode of the LFSTO-trap

Input:

$\boldsymbol{N}$	: Number of Siberian tigers
$\boldsymbol{T}$	: Maximum number of iterations
$\boldsymbol{lb_j}$	: Lower bound for problem variable $\boldsymbol{j}$
$\boldsymbol{ub_j}$	: Upper bound for problem variable $\boldsymbol{j}$
$\boldsymbol{F_i}$	: Objective function value obtained by the $\boldsymbol{i}$ -th Siberian tiger
$\boldsymbol{\beta}$	: Stability index
$\boldsymbol{V}$	: Gaussian distribution $(0,\ 1)$,
$\boldsymbol{U}$	: The Gaussian distribution $(0, \sigma^2)$
$\boldsymbol{Window\ size}$	: Set the *window_size* as current iteration.
$\boldsymbol{Threshold}$	: Calculate *threshold_factor* by using Eq.(6.5)

Initialization:

Initialize the population matrix of Siberian tigers' locations X randomly within problem constraints.

Set the iteration counter $t = 1$.

Initialize the $step\ size$

Evaluation and Updating:

While $\boldsymbol{t \leq T}$ (where T is the maximum number of iterations).

Set the $trap\_condition$

a. Prey Hunting Phase:

i. Stage-1: Position Update

For each Siberian tiger $\boldsymbol{i}$ in the population:

- If $trap\_condition$ is true, then

  Calculate a new position $P1S1$ based on the 1st stage of the 1st phase using Equation (6.1):

  $$X_{i,j}^{P1S1} = x_{i,j} + r_{i,j} \cdot (TP_{i,j} - I_{i,j} \cdot x_{i,j}) \oplus levy(step)$$

  Check solution within boundaries or not by clip $(P1S1, lb_j, ub_j)$

- Else

  Calculate a new position $P1S1$ based on the 1st stage of the 1st phase using Equation (5.6):

  $$X_{i,j}^{P1S1} = x_{i,j} + r_{i,j} \cdot (TP_{i,j} - I_{i,j} \cdot x_{i,j})$$

- End If
- If $F_i^{P1S1} < F_i$, update the position to $P1S1$ as per Equation (5.5):

$$X_i = \begin{cases} X_i^{P1S1}, & F_i^{P1S1} < F_i; \\ X_i, & else, \end{cases}$$

ii. Stage-2: Position Update

For each Siberian tiger i in the population:

- If $trap\_condition$ is true, then

  Calculate a new position $P1S2$ based on the 2nd stage of the 1st phase using Equation (6.4):

  $X_{i,j}^{P1S2} = \left[\boldsymbol{x_{i,j}} + \frac{\boldsymbol{r_{i,j} \cdot (ub_j - lb_j)}}{\boldsymbol{t}}\right] \oplus levy(step)$, $i = 1, 2, \dots, N$, $j = 1, 2, \dots, m$ and $t = 1, 2, \dots, T$

  Check solution within boundaries or not by clip $(P1S2, lb_j, ub_j)$

- Else

Algorithm 6.4. Pseudocode of the LFSTO-trap (continued)

Calculate a new position $P1S2$ based on the 2nd stage of the 1st phase using Equation (5.7):

$$X_{i,j}^{P1S2} = \left[\boldsymbol{x_{i,j}} + \frac{\boldsymbol{r_{i,j} \cdot (ub_j - lb_j)}}{\boldsymbol{t}} \right], i = 1, 2, \dots, N \ , j = 1, 2, \dots, m \text{ and } t = 1, 2, \dots, T$$

- End If
- If $F_i^{P1S2} < F_i$, update the position to $P1S2$ as per Equation (5.8):

$$X_i = \begin{cases} X_i^{P1S2}, & F_i^{P1S2} < F_i; \\ X_i, & else, \end{cases}$$

b. Fighting Phase:

i. Stage-1: Selection

For each Siberian tiger i in the population:

- Calculate a new position $P2S1$ based on the 1st stage of the 2nd phase using Equation (5.9):

$$x_{i,j}^{P2S1} = \begin{cases} x_{i.j} + r_{i,j} \cdot \left(x_{k,j} - I_{i,j} \cdot x_{i,j}\right), & F_k < F_i; \\ x_{i.j} + r_{i,j} \cdot (x_{i,j} - I_{i,j} \cdot x_{k,j}), & else, \end{cases}$$

- If $F_i^{P2S1} < F_i$, update the position to $P2S1$ as per Equation (5.10):

$$X_i = \begin{cases} X_i^{P2S1}, & F_i^{P2S1} < F_i; \\ X_i, & else, \end{cases}$$

ii. Stage-2: Fight

For each Siberian tiger i in the population:

- Calculate a new position $P2S2$ based on the 2nd stage of the 2nd phase using Equation (5.11).

$$X_{i,j}^{P2S2} = x_{i,j} + \frac{r_{i,j} \cdot (ub_j - lb_j)}{t}, i = 1, 2, \dots, N \ , \ j = 1, 2, \dots, m \text{ and } t = 1, 2, \dots, T$$

- If $F_i^{P2S2} < F_i$, update the position to $P2S2$ as in Equation (5.12).

$$X_i = \begin{cases} X_i^{P1S2}, & F_i^{P2S2} < F_i; \\ X_i, & else, \end{cases}$$

c. Increment Iteration Counter:

Increase iteration counter t by $\boldsymbol{t = t + 1}$

EndWhile

Output:

$\boldsymbol{X_{best}}$: Best position found in the population.

$\boldsymbol{F_{best}}$: Fitness value of the best solution found

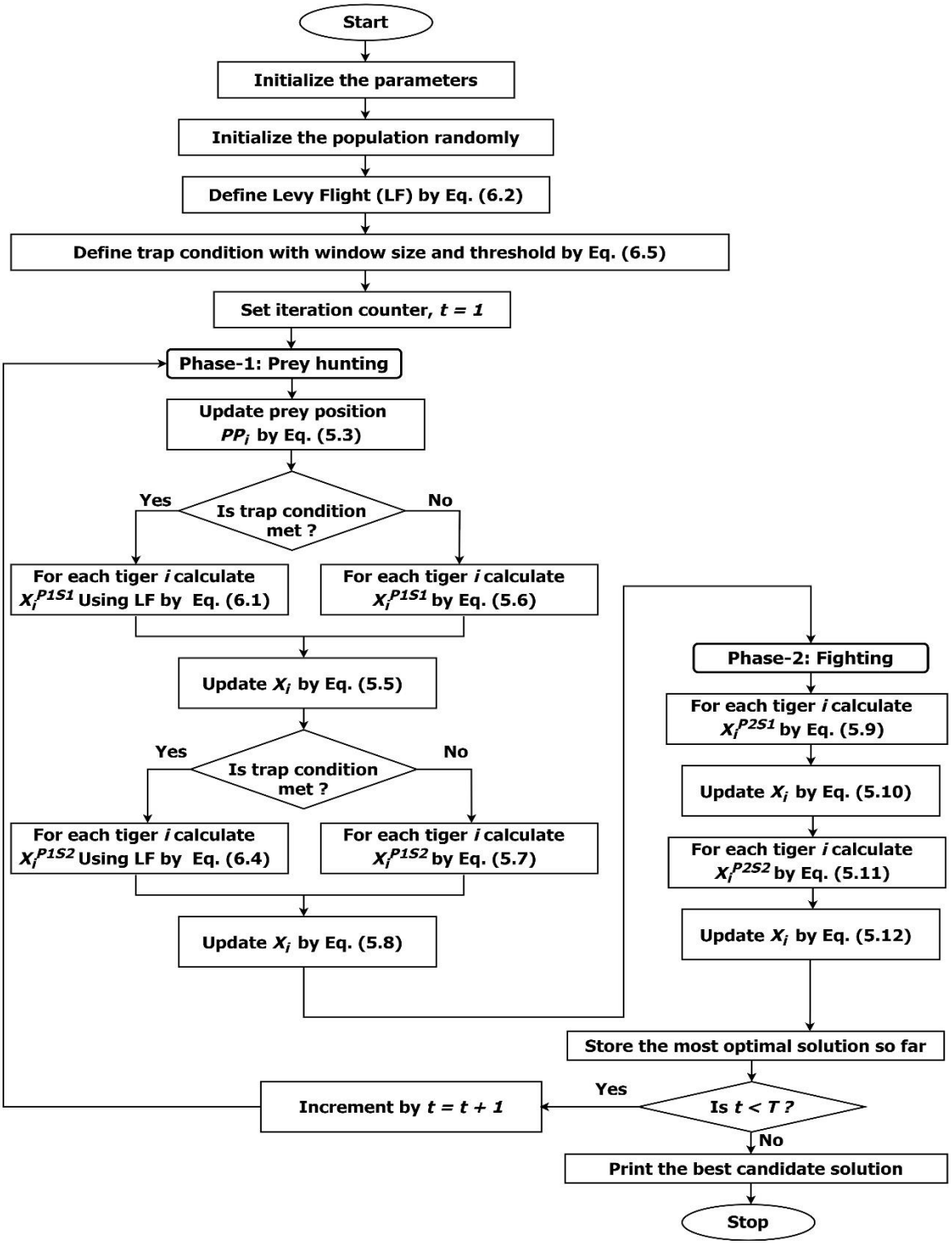


Figure 6.4. Flowchart of LFSTO-trap strategy

7. EXPERIMENTAL RESULTS AND DISCUSSION

In this chapter, all statistical results obtained from the experiments are presented and then the findings are analyzed according to these results. The organization of this chapter also includes parameter settings, evaluation metrics, step-by-step results assessments for all proposed methods, and finally, a detailed discussion of the overall findings of this study.

7.1. Parameter Settings

Most practical scenarios involve highly complex objective function calculations, which take a significant amount of time to complete. The computational difficulty of STO may be computed by counting the number of function evaluations (FEs) required. The estimated difficulty of the STO initialization procedure is FEs = N, where N is the total number of Siberian tigers. Every STO member receives updates at two distinct stages of each cycle. There are two steps to determine the objective function during each of these stages. Hence, the suggested STO's process of updating population members is complicated, with FEs = $(4N \times T)$, where T is the total iterations, and the T calculation formula obtained by the equation (7.1) as follows (Trojovsky et al., 2022):

$$T = \frac{MFEs - N}{4 \times N} \tag{7.1}$$

where MFEs is the total number of FEs and is equal to $(4T + 1) \times N$.

The experiments have been conducted on the Google Colab premium version, which utilizes a 64-bit Intel (R) Core i5 processor with 2.20 GHz and 8 GB of RAM. The standard STO algorithms and all proposed hybrid strategies are employed on the CEC-2017 benchmark test functions for 10, 30, and 50 dimensions to evaluate their effectiveness. Nevertheless, the F2 of the CEC-2017 benchmark suite is frequently dismissed by authors due to its inconsistent behavior (Trojovsky et al., 2022) and notable performance disparities for the same algorithm performed in MATLAB (Awad et al., 2016) as well as in Python, particularly for large dimensions. In this experiment, all strategies have used a population of 30 Siberian tigers, i.e., N = 30, along with 10000 maximum FEs (thereby, MFEs = $10000 \times m$, where m is the number of variables). However, in this case, m is the dimension of the problem. All benchmark functions in

each experiment were performed 30 times individually to get statistical findings on the performance of the proposed STO. In the LF context, to determine the step size, this study used equation 7.2 as follows:

$$step\ size = 0.01 \times levy(step) \tag{7.2}$$

In this case, the value of 0.01 comes from the assumption that the normal step size of travel would be $L/100$, where L represents the typical length scale. If this is not the case, then LFs might become overly aggressive, causing new solutions to leap out of the conceptual area and commit assessments (Jensi and Jiji, 2016). The summary of all the parameter settings is represented in Table 7.1.

Table 7.1. Summary of the parameter Settings

Parameter	Values	Strategies
Population Size	30	STO LFSTO-direct LFSTO-randprob LFSTO-randprob (step_size = 0.1) LFSTO-trial LFSTO-trap
Number of Runs	30	
Iteration Numbers	833, 2499, 4166	
Dimension	10, 30, and 50	
Step Size	0.01	
Number of test functions	29	
Range	-100 to +100	

7.2. Evaluation Metrics

To test and prove the strength of the algorithm's performance, this research considers several statistical evaluation metrics. such as,

- Best solution: Based on the optimization requirements of the issue at hand, it indicates the ultimate desirable value or the minimum attainable cost. The technique's ability to acquire optimal or nearly ideal solutions is assessed using the best solution as a standard.
- Worst solution: In a similar vein, the poorest possible solution is the one that maximizes costs while minimizing objectives or the smallest feasible objective value given the restrictions of the situation.
- Mean: The average metric shows how well the optimization process performs throughout every iteration or run. It indicates how well the approach performs altogether, along with how well it offers solutions that strike an

equilibrium between scalability and optimality in various issue scenarios and is calculated as equation (7.3).

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{7.3}$$

where n is the number of data point in the list, x_i represents each data point in the list, and μ is the mean of the data list.

- Standard Deviation (SD): The SD calculates the variance, or dispersion, of the solution grade across the average. A minimal SD implies that the quality of the algorithm's solutions is more consistent and stable, whereas a large SD shows that there is a lot of variation in the efficiency of the algorithm between runs or issue examples, as its calculation formula represents, as shown in equation (7.4).

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2} \tag{7.4}$$

where σ is the standard deviation of the data list.

- Median: The optimization process yielded a hierarchy of alternative attributes; the median value indicates the midpoint value of that hierarchy. Unlike the average value, which can be influenced by exceptionally high values, the median provides a trustworthy estimate of a central trend that is less susceptible to deviations or high solution properties, and the median value is calculated by equation 7.5.

$$if\ n = odd, then\ \ median = x_{n+1/2}$$
$$if\ n = even, then\ \ median = \frac{x_{n/2} + x_{n+1/2}}{2} \tag{7.5}$$

where x is the ordered data list.

- Time: Time value indicates a benchmark function or solution takes how much time to execute the problem within the total number of iterations and runs.
- Rank: Rank is an effective measure to evaluate the significance or utility of an item, as it indicates where it is in contrast to other items in a sorted list. In this experimental evaluation, rank is computed by the mean value of the functions.

- Mean rank: A statistic known as mean rank establishes the average rank of objects within a collection of observations. It serves as an overview of each item's overall performance or positioning within a ranking.

Several evaluation metrics provide insightful information about how optimization methods operate and execute when dealing with challenging optimization issues.

7.3. Evaluation of Experimental Results of Standard STO

Based on the results presented in Table 7.2 for dimension 10, STO performs better than the other five for the functions F1, F29, and F30; however, for F3 and F11–F20, the value is the same as the other five regarding the best solution metrics. For the functions F29 and F30, STO outperforms the other five approaches in the cases of the worst solution and mean value. According to standard deviation metrics, STO shows better results only for the function F29, and for the median value metrics, functions F1, F11–F20 with the other five approaches, and F29 perform better. Finally, in the case of the execution time standard STO, it takes less time than the other five strategies for the functions F1, F3, F4, F6, F8 to F13, F22 to F24, and F29. So, for dimension 10 statistical results, STO gained rank 1 only for the F29 and F30, and the final rank is 6, which is the last rank of this evaluation.

According to Table 7.3's performance findings for dimension 30, STO outperforms the other five for functions F1, F21, and F26; nevertheless, when it comes to F11–F20, the value is identical to the other five in terms of best solution metrics. STO outperforms the other five techniques in worst-case solutions only for functions F30. For the F8 and F30 functions, STO outperforms the others when looking at the mean value. Standard deviation measurements reveal that STO performs superiorly for functions F10 and F30, while median value measures endorse functions F8, F11–F20 with the other five techniques, F21 and F30. Regarding execution time, standard STO takes less time than the other five methods for functions F1, F3–F9, F16, and F27–F30. So, according to the statistics for dimension 30, STO ranked first for both the F8 and the F30, and it accomplished number six, much like the dimension 10 scenario, which was the last rank in this assessment.

The results shown in Table 7.4 for dimension 50 represent that STO does better than the other five for functions F21, F25, F26, and F30 in terms of the best solution metric; however, the value is identical for functions F11–F20 as it is for the other five. In

terms of worst-case solutions, STO is superior to the other five methods, only for function F30. Focusing on the mean value, STO performs better than the others for the F7-F9, f25, and F30 functions. Based on standard deviation values, STO does better with functions F10 only. In contrast, functions F7, F11–F20—which have the same values as the other five methods, F21, and F26—perform better when the median value is considered. Regarding execution time, standard STO takes less time than the other five methods for functions F1, F17–F19, and F22–F26. As a result, STO ranked first for both the F7-F9, F25, and F30 and overall outranked LFSTO-trap strategies in dimension 30's statistics.

7.4. Evaluation of Experimental Results of LFSTO-direct

From Table 7.2 of the statistical results for dimension 10, the LFSTO-direct approach individually outperforms the other five approaches for the functions F4, F5, and F9 in terms of the best optimal solution, F3, F5, F15, F17, F19, F21, and F24-F26 in terms of the worst solution, only F5 in terms of mean, F3, F10, F12, F15, F17, F19, and F23-F27 in terms of standard deviation, F4, F5, and F9 in terms of median value, and F5, F7, F14, F15, F17, F19, F20, and F26-F28 in terms of less execution time, respectively. Whatever the best solution and the median value case, LFSTO-direct obtained the same results as the other five methods for the functions F3 as well as F11-F20 and F11-F20, respectively. Moreover, LFSTO-direct gained the same results as LFSTO-randprob for the functions F11 in the worst solution case and F11-F20 in the mean value case, respectively. Finally, LFSTO-direct acquired rank first individually for the functions F5 and together with LFSTO-randprob for the functions F11–F20, and overall rank is second, which is superior to all other approaches except LFSTO-randprob.

The results in Table 7.3 for dimension 30 show that the LFSTO-direct approach does better than the other five approaches for each function: F4 for the best optimal solution; F3, F8, F10, F12, F14, F16-F19, F21, F22, and F24-F28 for the worst solution; F10 and F28 for the mean; F3, F7, F14, F16-F19, F21, F22, and F25-F27 for the standard deviation; F10 and F28 for the median value; and F10, F11, and F13-F15 for the execution time. Regardless of the optimal solution and median value situation, LFSTO-direct achieved identical outcomes to the other five approaches for functions F11–F20 as well as F28, and F11–F20, respectively. Additionally, LFSTO-direct gained the same results as LFSTO-randprob for the functions F11–F20 and as LFSTO-randprob, LFSTO-randprob (step_size=0.1), and LFSTO-trap for the function F29 in the mean value case.

So, LFSTO-direct got the top spot for functions F10 and F28, along with LFSTO-randprob for functions F11–F20 and for function F29; it also got the first rank along with

Table 7.2. The statistical results of CEC-2017 functions for dimension 10

Function	Metrics	STO	LFSTO-direct	LFSTO-randprob	LFSTO-randprob (step_size=0.1)	LFSTO-trial	LFSTO-trap
	best sol.	**2.7885E+02**	7.2800E+03	8.0846E+02	2.7199E+03	2.7798E+03	3.4810E+03
	worst	2.7344E+11	1.8204E+11	1.5922E+11	1.5696E+11	**9.1428E+10**	2.1294E+11
	mean	3.3463E+09	2.3289E+09	4.1771E+09	2.9747E+09	**1.2437E+09**	3.1289E+09
F1	std. dev.	1.9935E+10	1.2822E+10	1.6343E+10	1.3972E+10	**1.9523E+09**	1.6644E+10
	median	**5.0850E+02**	7.2877E+03	1.5224E+03	2.7739E+05	4.8614E+03	3.4988E+03
	time (s)	**1.9056E+02**	2.0473E+02	2.1818E+02	2.1454E+02	2.3016E+02	2.1342E+02
	rank	5	2	6	3	1	4
	best sol.	**3.0000E+02**	**3.0000E+02**	**3.0000E+02**	**3.0000E+02**	**3.0000E+02**	**3.0000E+02**
	worst	4.3757E+05	**1.4153E+04**	1.9141E+04	5.6143E+04	1.5285E+04	7.7334E+04
	mean	6.8582E+03	1.2221E+03	**1.2110E+03**	1.6145E+03	2.3777E+03	2.9201E+03
F3	std. dev.	2.5519E+04	**2.2087E+03**	2.5568E+03	3.8901E+03	3.7203E+03	7.9807E+03
	median	3.3990E+02	3.0489E+02	3.0074E+02	3.0068E+02	3.5425E+02	**3.0017E+02**
	time (s)	**2.4038E+02**	2.4573E+02	2.7363E+02	2.6631E+02	2.7680E+02	2.6318E+02
	rank	6	2	1	3	4	5
	best sol.	4.0478E+02	**4.0068E+02**	4.0180E+02	4.0221E+02	4.0353E+02	4.0377E+02
	worst	1.4392E+03	1.8599E+03	1.8983E+03	**8.0545E+02**	2.6082E+03	1.9311E+03
	mean	4.1294E+02	4.2857E+02	4.1938E+02	**4.0867E+02**	4.2994E+02	4.2276E+02
F4	std. dev.	5.6946E+01	1.3254E+02	1.1780E+02	**3.6998E+01**	1.4414E+02	1.1305E+02
	median	4.0537E+02	**4.0069E+02**	4.0204E+02	4.0252E+02	4.0381E+02	4.0433E+02
	time (s)	**2.1866E+02**	2.2654E+02	2.4953E+02	2.3443E+02	2.5504E+02	2.3104E+02
	rank	2	5	3	1	6	4
	best sol.	5.3236E+02	**5.2089E+02**	5.3980E+02	5.2786E+02	5.2288E+02	5.2786E+02
	worst	6.4288E+02	**6.3423E+02**	6.8293E+02	6.7265E+02	5.7723E+02	6.7421E+02
	mean	5.4795E+02	**5.2541E+02**	5.4450E+02	5.3340E+02	5.2546E+02	5.3535E+02
F5	std. dev.	1.6909E+01	1.3730E+01	1.2070E+01	1.2894E+01	**1.0450E+01**	1.6181E+01
	median	5.4423E+02	**5.2089E+02**	5.3989E+02	5.2792E+02	5.2288E+02	5.2835E+02
	time (s)	2.1438E+02	**2.1093E+02**	2.4027E+02	2.2878E+02	4.2890E+02	2.3022E+02
	rank	6	1	5	3	2	4
	best sol.	6.0286E+02	6.0076E+02	**6.0004E+02**	6.0005E+02	6.1023E+02	6.0067E+02
	worst	6.8926E+02	7.0068E+02	6.7543E+02	**6.0005E+02**	7.0083E+02	7.2837E+02
	mean	6.0783E+02	6.0512E+02	6.0254E+02	**6.0005E+02**	6.1741E+02	6.0555E+02
F6	std. dev.	1.3059E+01	1.1921E+01	8.1576E+00	1.0904E+01	**1.0810E+01**	1.3401E+01
	median	6.0311E+02	6.0077E+02	**6.0004E+02**	6.0005E+02	6.1496E+02	6.0076E+02
	time (s)	**2.3083E+02**	2.4422E+02	2.7072E+02	2.5147E+02	2.7544E+02	2.5464E+02
	rank	5	3	2	1	6	4
	best sol.	7.3535E+02	7.3581E+02	7.6653E+02	7.5084E+02	7.4518E+02	**7.2523E+02**
	worst	1.1286E+03	8.6869E+02	**8.6817E+02**	8.9781E+02	9.3084E+02	1.0669E+03
	mean	7.4746E+02	7.5050E+02	7.7720E+02	7.6429E+02	7.5148E+02	**7.4169E+02**
F7	std. dev.	2.5375E+01	2.0039E+01	1.5739E+01	2.1502E+01	**1.3340E+01**	2.6984E+01
	median	7.4077E+02	7.4076E+02	7.6824E+02	7.5383E+02	7.4713E+02	**7.3184E+02**
	time (s)	3.6414E+02	**3.5006E+02**	4.0512E+02	3.5460E+02	3.8529E+02	3.5477E+02
	rank	2	3	6	5	4	1
	best sol.	8.2043E+02	8.2786E+02	8.2885E+02	**8.1293E+02**	8.2602E+02	8.2519E+02
	worst	9.2880E+02	9.1737E+02	9.2032E+02	**8.8753E+02**	8.9606E+02	9.6629E+02
	mean	8.3092E+02	8.3085E+02	8.3184E+02	**8.1946E+02**	8.3462E+02	8.3403E+02
F8	std. dev.	1.2540E+01	9.0679E+00	**8.4326E+00**	1.1280E+01	1.1134E+01	9.4160E+00
	median	8.2540E+02	8.2786E+02	8.2886E+02	**8.1298E+02**	8.3155E+02	8.3253E+02
	time (s)	**2.4956E+02**	2.6552E+02	2.8777E+02	2.7008E+02	2.9346E+02	2.7783E+02
	rank	3	2	4	1	6	5
	best sol.	9.0579E+02	**9.0000E+02**	**9.0000E+02**	9.0018E+02	9.0054E+02	9.0045E+02
	worst	5.6462E+03	3.0573E+03	**2.3515E+03**	3.1031E+03	3.7538E+03	4.9189E+03
	mean	9.6752E+02	9.2616E+02	**9.1605E+02**	9.2508E+02	9.2892E+02	9.3673E+02
F9	std. dev.	2.9440E+02	1.2106E+02	**9.3410E+01**	1.3697E+02	1.7519E+02	2.3976E+02
	median	9.0759E+02	**9.0000E+02**	**9.0000E+02**	9.0018E+02	9.0054E+02	9.0045E+02
	time (s)	**2.8269E+02**	2.8836E+02	3.2159E+02	2.9496E+02	3.2227E+02	3.0346E+02
	rank	6	3	1	2	4	5
	best sol.	2.0278E+03	2.3185E+03	**1.5464E+03**	1.8073E+03	2.0578E+03	2.1861E+03
	worst	3.4638E+03	3.4369E+03	**3.1616E+03**	3.8381E+03	3.5876E+03	3.8491E+03
	mean	2.4793E+03	2.4578E+03	2.0277E+03	**2.0206E+03**	2.2156E+03	2.4225E+03
F10	std. dev.	2.2504E+02	**1.6654E+02**	3.9723E+02	2.9187E+02	2.2123E+02	1.7386E+02
	median	2.4123E+03	2.4379E+03	1.8741E+03	1.9249E+03	2.2313E+03	2.4117E+03
	time (s)	**2.0523E+02**	3.1174E+02	3.5489E+02	3.2429E+02	4.1687E+02	3.3175E+02
	rank	6	5	2	1	3	4

Table 7.2. The statistical results of CEC-2017 functions for dimension 10 (continued)

Function	Metrics	STO	LFSTO-direct	LFSTO-randprob	LFSTO-randprob (step_size=0.1)	LFSTO-trial	LFSTO-trap
	best sol.	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**
	worst	5.9555E+03	**1.1006E+03**	**1.1006E+03**	1.1773E+03	7.0833E+03	1.6314E+04
	mean	1.1213E+03	**1.1000E+03**	**1.1000E+03**	1.1001E+03	1.1122E+03	1.1273E+03
F11	std. dev.	2.5467E+02	2.1371E-02	**2.0799E-02**	2.6752E+00	2.2894E+02	5.5529E+02
	median	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**
	time (s)	**3.0022E+02**	3.1623E+02	3.4069E+02	3.2333E+02	3.3835E+02	3.1003E+02
	rank	5	1	1	3	4	6
	best sol.	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**
	worst	1.1363E+04	**1.2004E+03**	1.2018E+03	1.2895E+03	6.8907E+03	8.2098E+03
	mean	1.2298E+03	**1.2000E+03**	**1.2000E+03**	1.2001E+03	1.2108E+03	1.2268E+03
F12	std. dev.	4.4538E+02	**1.5012E-02**	6.3746E-02	3.1008E+00	2.1118E+02	3.6121E+02
	median	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**
	time (s)	**2.9308E+02**	3.2198E+02	3.3174E+02	3.2027E+02	3.4294E+02	3.1143E+02
		6	1	1	3	4	5
	best sol.	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**
	worst	6.2580E+03	1.3009E+03	1.3006E+03	1.3397E+03	**1.3000E+03**	8.3695E+03
	mean	1.3239E+03	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**	1.3147E+03
F13	std. dev.	3.0109E+02	3.2674E-02	1.9343E-02	1.3739E+00	**2.2737E-11**	2.7294E+02
	median	**1.3000E+02**	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**
	time (s)	**2.9825E+02**	3.1252E+02	3.3092E+02	3.2804E+02	5.8409E+02	3.1333E+02
	rank	6	1	1	1	1	5
	best sol.	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**
	worst	1.2771E+04	1.4005E+03	**1.4002E+03**	1.4933E+03	7.3641E+03	1.0305E+04
	mean	1.4257E+03	**1.4000E+03**	**1.4000E+03**	1.4001E+03	1.4133E+03	1.4240E+03
F14	std. dev.	4.4196E+02	1.8177E-02	**7.4650E-03**	3.2311E+00	2.2648E+02	3.5150E+02
	median	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**
	time (s)	3.2282E+02	**3.2260E+02**	3.5284E+02	3.4371E+02	3.5521E+02	3.3255E+02
	rank	6	1	1	3	4	5
	best sol.	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**
	worst	7.3291E+03	**1.5004E+03**	1.5012E+03	1.6097E+03	8.3547E+04	1.1097E+04
	mean	1.5159E+03	**1.5000E+03**	**1.5000E+03**	1.5001E+03	1.5158E+03	1.5366E+03
F15	std. dev.	2.5047E+02	**1.4836E-02**	4.2124E-02	3.7997E+00	2.5472E+02	4.6046E+02
	median	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**
	time (s)	3.2411E+02	**3.1321E+02**	3.5185E+02	3.4439E+02	3.5345E+02	3.2969E+02
	rank	5	1	1	3	4	6
	best sol.	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**
	worst	8.1973E+03	1.6005E+03	**1.6002E+03**	1.7853E+03	4.4236E+03	8.8371E+03
	mean	1.6204E+03	**1.6000E+03**	**1.6000E+03**	1.6002E+03	1.6071E+03	1.6172E+03
F16	std. dev.	3.0322E+02	1.5753E-02	**5.2941E-03**	6.4168E+00	1.1434E+02	2.9062E+02
	median	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**
	time (s)	3.4122E+02	3.3252E+02	3.5297E+02	3.4251E+02	3.5394E+02	**3.3146E+02**
	rank	6	1	1	3	4	5
	best sol.	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**
	worst	1.5463E+04	**1.7006E+03**	1.7007E+03	1.7961E+03	9.8587E+03	1.3630E+04
	mean	1.7340E+03	**1.7000E+03**	**1.7000E+03**	1.7001E+03	1.7197E+03	1.7324E+03
F17	std. dev.	5.7901E+02	**2.1619E-02**	2.3192E-02	3.3275E+00	3.1392E+02	4.9325E+02
	median	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**
	time (s)	3.5945E+02	**3.4527E+02**	3.7916E+02	3.6471E+02	3.8987E+02	3.5389E+02
	rank	6	1	1	3	4	5
	best sol.	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**
	worst	1.1362E+04	**1.8002E+03**	1.8003E+03	1.8665E+03	7.7448E+03	1.1522E+04
	mean	1.8268E+03	**1.8000E+03**	**1.8000E+03**	1.8001E+03	1.8190E+03	1.8282E+03
F18	std. dev.	3.5846E+02	6.8677E-02	**1.0473E-02**	2.3035E+00	2.5367E+02	4.0988E+02
	median	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**
	time (s)	3.6169E+02	3.5220E+02	3.7475E+02	3.6251E+02	4.4881E+02	3.5262E+02
	rank	5	1	1	3	4	6
	best sol.	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**
	worst	1.0390E+04	**1.9006E+03**	1.9009E+03	1.9726E+03	3.5885E+03	1.0256E+04
	mean	1.9161E+03	**1.9000E+03**	**1.9000E+03**	1.9000E+03	1.9051E+03	1.9236E+03
F19	std. dev.	3.0205E+02	**1.9471E-02**	3.1935E-02	2.5135E+00	6.6008E+01	3.6141E+02
	median	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**
	time (s)	3.6450E+02	**3.4015E+02**	3.7362E+02	3.8354E+02	4.0201E+02	3.5538E+02
	rank	5	1	1	1	4	6
	best sol.	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**
	worst	1.1897E+04	2.0007E+03	**2.0004E+03**	2.0661E+03	7.8318E+03	1.2007E+04
	mean	2.0279E+03	**2.0000E+03**	**2.0000E+03**	2.0001E+03	2.0114E+03	2.0378E+03
F20	std. dev.	4.3050E+02	2.3536E-02	**1.5272E-02**	2.2881E+00	2.2342E+02	4.9732E+02
	median	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**
	time (s)	3.7866E+02	**3.7350E+02**	3.9814E+02	4.2180E+02	5.0484E+02	3.7724E+02
	rank	5	1	1	3	4	6

Table 7.2. The statistical results of CEC-2017 functions for dimension 10 (continued)

Function	Metrics	STO	LFSTO-direct	LFSTO-randprob	LFSTO-randprob (step_size=0.1)	LFSTO-trial	LFSTO-trap
	best sol.	**2.1742E+03**	2.1781E+03	2.1795E+03	2.1779E+03	2.2081E+03	2.1806E+03
	worst	9.2883E+03	**2.2290E+03**	2.2319E+03	2.2720E+03	1.1218E+04	7.8349E+03
	mean	2.2189E+03	2.1856E+03	**2.1826E+03**	2.1852E+03	2.2628E+03	2.2159E+03
F21	std. dev.	3.7050E+02	7.9190E+00	**4.9314E+00**	9.4159E+00	3.7575E+02	2.8413E+02
	median	2.1884E+03	2.1828E+03	**2.1820E+03**	2.1852E+03	2.2133E+03	2.1846E+03
	time (s)	6.7435E+02	7.0545E+02	7.0610E+02	6.9320E+02	7.3752E+02	7.3379E+02
	rank	5	3	1	2	6	4
	best sol.	2.2905E+03	2.2748E+03	**2.2740E+03**	2.2755E+03	2.4742E+03	2.2943E+03
	worst	2.5531E+03	2.3335E+03	**2.2817E+03**	2.4517E+03	2.6093E+03	2.6063E+03
	mean	2.3522E+03	2.2884E+03	**1.2315E+03**	2.2898E+03	2.4773E+03	2.3796E+03
F22	std. dev.	8.1212E+01	1.3858E+01	2.2740E+01	2.0205E+01	**1.1405E+01**	9.0530E+01
	median	2.3050E+03	2.2952E+03	**2.2740E+03**	2.2755E+03	2.4742E+03	2.3271E+03
	time (s)	**6.8846+02**	7.2295E+02	7.1369E+02	7.0087E+02	7.4162E+02	7.4269E+02
	rank	4	2	1	3	6	5
	best sol.	2.4145E+03	2.4227E+03	2.4159E+03	2.4248E+03	2.4668E+03	**2.3977E+03**
	worst	9.5902E+03	2.4927E+03	2.4957E+03	**2.4872E+03**	4.6938E+03	1.2995E+04
	mean	2.4583E+03	2.4290E+03	**2.4270E+03**	2.4288E+03	2.4960E+03	2.4535E+03
F23	std. dev.	3.2131E+02	**1.0083E+01**	1.0409E+01	2.4288E+03	1.5189E+02	4.0468E+02
	median	2.4254E+03	2.4234E+03	2.4254E+03	6.9714E+00	2.4683E+03	**2.4057E+03**
	time (s)	**8.7062E+02**	9.1908E+02	9.0159E+02	2.4248E+03	9.4411E+02	8.9109E+02
	rank	5	3	1	2	6	4
	best sol.	2.5209E+03	2.5192E+03	**2.5020E+03**	2.5033E+03	2.5533E+03	2.5099E+03
	worst	6.1505E+03	**2.5543E+03**	2.5675E+03	2.5920E+03	6.3427E+03	3.2905E+03
	mean	2.5465E+03	2.5244E+03	2.5218E+03	**2.5118E+03**	2.5960E+03	2.5273E+03
F24	std. dev.	1.8525E+02	**8.4151E+00**	1.2166E+01	1.0633E+01	2.0300E+02	5.1428E+01
	median	2.5320E+03	2.5196E+03	2.5243E+03	2.5132E+03	2.5688E+03	**2.5099E+03**
	time (s)	**8.0883E+02**	8.4407E+02	8.1213E+02	8.1535E+02	8.6247E+02	8.2818E+02
	rank	5	3	2	1	6	4
	best sol.	2.8194E+03	2.7727E+03	**2.7281E+03**	2.7306E+03	2.9697E+03	2.7479E+03
	worst	4.8134E+03	**2.8935E+03**	2.9137E+03	3.5124E+03	6.0098E+03	5.9394E+03
	mean	2.8804E+03	2.7776E+03	**2.7504E+03**	2.7536E+03	3.0722E+03	2.8162E+03
F25	std. dev.	2.0651E+02	**1.0952E+01**	2.3580E+01	3.4023E+01	1.8845E+02	1.8113E+02
	median	2.8225E+03	2.7727E+03	2.7518E+03	**2.7405E+03**	3.0442E+03	2.7833E+03
	time (s)	9.5021E+02	9.7692E+02	9.4076E+02	**9.3285E+02**	1.0045E+03	9.6118E+02
	rank	5	3	1	2	6	4
	best sol.	2.7670E+03	2.7734E+03	**2.7532E+03**	2.7636E+03	2.8395E+03	2.7719E+03
	worst	3.7525E+04	**2.8314E+03**	2.8575E+03	2.8494E+03	2.3512E+04	3.5437E+04
	mean	2.9100E+03	2.7786E+03	**2.7655E+03**	2.7741E+03	3.1953E+03	2.9337E+03
F26	std. dev.	1.6742E+03	**8.0863E+00**	1.1865E+01	8.9596E+00	1.3115E+03	1.8668E+03
	median	2.7670E+03	2.7734E+03	**2.7629E+03**	2.7728E+03	2.9563E+03	2.7738E+03
	time (s)	9.7653E+02	**8.2281E+02**	9.0720E+02	9.1124E+02	9.7457E+02	9.2827E+02
	rank	4	3	1	2	6	5
	best sol.	4.3548E+03	2.9718E+03	2.9594E+03	**2.9528E+03**	4.2747E+03	4.7014E+03
	worst	8.2769E+03	3.0799E+03	3.1469E+03	**3.0781E+03**	8.3712E+03	1.0161E+04
	mean	5.2658E+03	2.9752E+03	**2.9649E+03**	2.9652E+03	4.9377E+03	5.0164E+03
F27	std. dev.	7.2548E+02	**1.1221E+01**	1.95516E+01	2.0272E+01	7.4295E+02	7.1351E+02
	median	5.5193E+03	2.9718E+03	2.9594E+03	**2.9588E+03**	5.3235E+03	4.7014E+03
	time (s)	1.1568E+03	**8.7757E+02**	1.0520E+03	1.0674E+03	1.1808E+03	1.0877E+03
	rank	6	3	1	2	4	5
	best sol.	3.0288E+03	3.0156E+03	3.0213E+03	**3.0151E+03**	3.0600E+03	3.0264E+03
	worst	4.4474E+03	3.0616E+03	3.0624E+03	**3.0611E+03**	4.5410E+03	5.9204E+03
	mean	3.0470E+03	3.0214E+03	3.0239E+03	**3.0187E+03**	3.0760E+03	3.0441E+03
F28	std. dev.	7.9949E+01	7.6382E+00	**3.9093E+00**	6.4903E+00	7.9823E+01	1.4048E+02
	median	3.0380E+03	3.0156E+00	3.0223E+03	**3.0151E+03**	3.0600E+03	3.0264E+03
	time (s)	1.1514E+03	**8.6507E+02**	1.0567E+03	1.0588E+03	1.1788E+03	1.0738E+03
	rank	4	2	3	1	6	5
	best sol.	**1.4000E+03**	**1.4004E+03**	1.4004E+03	1.4004E+03	1.4001E+03	**1.4000E+03**
	worst	**1.4005E+03**	1.4014E+03	1.4014E+03	1.4012E+03	1.4024E+03	1.4006E+03
	mean	**1.4000E+03**	1.4004E+03	1.4004E+03	1.4004E+03	1.4001E+03	1.4001E+03
F29	std. dev.	**2.5148E-02**	4.6015E-02	7.3040E-02	3.6523E-02	1.0348E-01	4.4614E-02
	median	**1.4000E+03**	1.4004E+03	**1.4000E+03**	1.4004E+03	1.4001E+03	**1.4000E+03**
	time (s)	**7.0906E+02**	7.1233+02	7.4562E+02	7.4629E+02	7.6388E+02	7.1964E+02
	rank	1	4	4	4	2	2
	best sol.	**1.3000E+03**	1.3294E+03	1.3003E+03	1.3002E+03	**1.3000E+03**	**1.3000E+03**
	worst	**1.3005E+03**	1.3588E+03	1.3010E+03	1.3009E+03	1.3021E+03	1.3006E+03
	mean	**1.3000E+03**	1.3311E+03	1.3003E+03	1.3004E+03	1.3001E+03	**1.3000E+03**
F30	std. dev.	4.8171E-02	3.1269E+00	6.4458E-02	1.0575E-01	9.6879E-02	**3.5754E-02**
	median	1.3001E+03	1.3294E+03	1.3003E+03	1.3004E+03	1.3001E+03	**1.3000E+03**
	time (s)	7.1223E+02	7.3308E+02	7.4414E+02	7.4167E+02	7.7711E+02	**7.1167E+02**
	rank	1	6	4	5	3	1
	mean rank	4.68	2.34	2.03	2.41	4.27	4.48
	final rank	6	2	1	3	4	5

Table 7.3. The statistical results of CEC-2017 functions for dimension 30

function	Metrics	STO	LFSTO-direct	LFSTO-randprob	LFSTO-randrand (step_size=0.1)	LFSTO-trial	LFSTO-trap
	best sol.	**1.9980E+02**	1.1167E+04	2.3149E+03	2.6031E+02	1.0020E+03	3.5307E+03
	worst	1.0507E+12	8.1783E+11	7.4617E+11	**7.0371E+11**	7.7677E+11	8.4748E+11
	mean	1.8840E+10	1.7014E+10	1.4319E+10	1.4438E+10	1.4059E+10	**1.3816E+10**
F1	std. dev.	6.6260E+10	8.2624E+10	6.3636E+10	**6.1149E+10**	6.4562E+10	6.4606E+10
	median	1.2803E+07	1.3263E+05	3.5560E+04	1.0306E+05	**4.2290E+03**	1.1645E+05
	time (s)	**6.2929E+02**	7.5259E+02	7.4665E+02	7.5755E+02	7.9888E+02	8.0566E+02
	rank	6	5	3	4	2	1
	best sol.	1.1422E+04	1.8477E+04	8.5695E+03	1.1062E+04	**6.5396E+03**	2.1525E+04
	worst	1.0419E+06	**9.3487E+04**	1.3938E+05	2.8998E+05	2.0276E+08	9.2690E+06
	mean	5.9495E+04	4.2201E+04	**3.6777E+04**	3.7895E+04	1.9696E+05	6.9847E+04
F3	std. dev.	5.4350E+04	**1.8519E+04**	2.3441E+04	2.2547E+04	5.7329E+06	1.8737E+05
	median	4.7819E+04	3.6698E+04	3.1342E+04	3.2815E+04	**2.2601E+04**	5.6767E+04
	time (s)	**7.8970E+02**	9.1376E+02	9.1302E+02	9.2426E+02	9.7135E+02	9.8324E+02
	rank	4	3	1	2	6	5
	best sol.	4.8163E+02	**4.7964E+02**	5.1455E+02	5.1293E+02	4.7995E+02	5.3865E+02
	worst	4.0012E+04	2.5832E+04	2.7135E+04	**2.0451E+04**	2.2239E+04	4.3459E+04
	mean	8.2658E+02	8.3032E+02	9.2960E+02	7.2712E+02	**6.8107E+02**	7.8922E+02
	std. dev.	1.8940E+03	1.5819E+03	1.7086E+03	1.1996E+03	**1.1983E+03**	1.4029E+03
F4	median	5.2390E+02	4.8053E+02	5.1618E+02	5.1438E+02	**4.8031E+02**	5.4074E+02
	time (s)	**7.1577E+02**	8.2370E+02	8.3668E+02	8.6360E+02	8.9021E+02	8.9289E+02
	rank	4	5	6	2	1	3
	best sol.	6.6629E+02	6.9501E+02	7.2110E+02	7.0496E+02	**6.2138E+02**	6.7611E+02
	worst	1.0797E+03	1.0517E+03	1.0366E+03	1.0061E+03	**7.3344E+02**	1.0874E+03
	mean	6.9417E+02	7.0497E+02	7.6473E+02	7.2407E+02	**6.4054E+02**	6.8671E+02
F5	std. dev.	4.9095E+01	3.9276E+01	4.7308E+01	5.2748E+01	**3.5841E+01**	3.7669E+01
	median	6.7112E+02	6.9501E+02	7.5094E+02	7.0496E+02	**6.2138E+02**	6.7611E+02
	time (s)	**6.8539E+02**	7.9613E+02	8.1412E+02	8.2771E+02	1.4817E+03	8.5022E+02
	rank	3	4	6	5	1	2
	best sol.	6.5440E+02	6.4984E+02	6.5434E+02	6.6339E+02	**6.3271E+02**	6.5034E+02
	worst	7.5517E+02	7.5095E+02	7.5675E+02	7.3248E+02	**7.3188E+02**	7.6547E+02
	mean	6.6504E+02	6.5611E+02	6.6495E+02	6.6868E+02	**6.4855E+02**	6.6198E+02
F6	std. dev.	1.1857E+01	1.0651E+01	1.0333E+01	**8.1727E+00**	1.8766E+01	9.4017E+00
	median	6.6075E+02	6.5205E+02	6.6164E+02	6.6609E+02	**6.4188E+02**	6.5981E+02
	time (s)	**8.0037E+02**	9.5396E+02	9.0905E+02	9.3989E+02	9.7314E+02	8.9318E+02
	rank	5	2	4	6	1	3
	best sol.	1.0112E+03	1.1094E+03	1.0097E+03	9.8738E+02	**8.6806E+02**	9.2760E+02
	worst	2.7272E+03	1.5656E+03	1.5952E+03	**1.4969E+03**	2.4413E+03	2.6588E+03
	mean	1.0461E+03	1.1305E+03	1.0908E+03	1.0187E+03	**9.1417E+02**	9.7856E+02
F7	std. dev.	8.9341E+01	**4.7794E+01**	7.7361E+01	8.1742E+01	9.8624E+01	9.6353E+01
	median	1.0141E+03	1.1150E+03	1.0728E+03	9.8744E+02	**8.6817E+02**	9.3040E+02
	time (s)	**1.1479E+03**	1.2662E+03	1.2668E+03	1.3096E+03	1.3262E+03	1.2397E+03
	rank	4	6	5	3	1	2
	best sol.	9.2138E+02	9.6318E+02	9.9103E+02	9.5958E+02	9.2245E+02	**8.9881E+02**
	worst	1.3705E+03	**1.2340E+03**	1.2697E+03	1.2538E+03	1.2467E+03	1.3273E+03
	mean	**9.3887E+02**	9.7369E+02	9.9746E+02	9.7367E+02	9.6943E+02	9.4970E+02
F8	std. dev.	4.5997E+01	3.0451E+01	**2.5556E+01**	3.2377E+01	4.6714E+01	6.1294E+01
	median	**9.2139E+02**	9.6335E+02	9.9104E+02	9.6263E+02	9.6670E+02	9.2880E+02
	time (s)	**8.3177E+02**	9.6772E+02	9.7472E+02	9.7909E+02	1.0128E+03	9.3729E+02
	rank	1	5	6	4	3	2
	best sol.	3.8555E+03	7.6406E+03	2.2119E+03	5.3952E+03	**1.6359E+03**	2.3429E+03
	worst	3.0402E+04	1.9551E+04	1.8688E+04	**1.8606E+04**	1.8728E+04	2.7610E+04
	mean	4.8765E+03	8.5481E+03	3.5156E+03	6.4419E+03	**3.0334E+03**	3.7707E+03
F9	std. dev.	1.8027E+03	8.5481E+03	1.7805E+03	**1.1037E+03**	1.7028E+03	2.1602E+03
	median	4.0873E+03	7.8282E+03	3.1474E+03	6.3274E+03	3.1850E+03	**3.0314E+03**
	time (s)	**9.6376E+02**	1.0547E+03	1.0551E+03	1.0929E+03	1.1206E+03	1.0491E+03
	rank	4	6	2	5	1	3
	best sol.	8.6345E+03	4.9629E+03	5.0872E+03	**4.9258E+03**	7.7570E+03	8.0080E+03
	worst	1.1549E+04	**1.0231E+04**	1.0852E+04	1.0475E+04	1.1174E+04	1.0650E+04
	mean	8.9022E+03	**5.1497E+03**	5.3651E+03	5.2961E+03	8.2019E+03	8.2936E+03
F10	std. dev.	**2.9455E+02**	5.3723E+02	8.0655E+02	8.2438E+02	3.1939E+02	5.0898E+02
	median	8.9300E+03	**4.9689E+03**	5.0882E+03	4.9835E+03	8.2728E+03	8.0100E+03
	time (s)	1.0863E+03	**9.7451E+02**	1.1724E+03	1.2127E+03	1.5314E+03	1.1747E+03
	rank	6	1	3	2	4	5

Table 7.3. The statistical results of CEC-2017 functions for dimension 30 (continued)

function	Metrics	STO	LFSTO-direct	LFSTO-randprob	LFSTO-randrand (step_size=0.1)	LFSTO-trial	LFSTO-trap
	best sol.	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**
	worst	3.8923E+04	1.1055E+03	**1.1053E+03**	2.2296E+03	2.4134E+04	3.5777E+04
	mean	1.1428E+03	**1.1000E+03**	**1.1000E+03**	1.1005E+03	1.1419E+03	1.1462E+03
F11	std. dev.	9.4798E+02	1.1023E-01	**1.0683E-01**	2.2591E+01	6.3212E+02	1.0281E+03
	median	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**
	time (s)	1.1736E+03	**1.1221E+03**	1.3239E+03	1.3627E+03	1.3592E+03	1.2693E+03
	rank	5	1	1	3	4	6
	best sol.	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**
	worst	4.5966E+04	**1.2084E+03**	1.2103E+03	1.6522E+03	2.2388E+04	3.3370E+04
	mean	1.2494E+03	**1.2000E+03**	**1.2000E+03**	1.2002E+03	1.2219E+03	1.2444E+03
F12	std. dev.	1.2087E+03	1.6778E-01	**2.0568E-02**	9.0436E+00	5.0004E+02	8.7741E+02
	median	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**
	time (s)	1.1746E+03	1.1183E+03	**9.0156E+02**	1.3438E+03	1.3448E+03	1.2786E+03
	rank	6	1	1	3	4	5
	best sol.	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**
	worst	4.9411E+04	1.3059E+03	1.3107E+03	2.4011E+03	**1.3000E+03**	3.6325E+04
	mean	1.3480E+03	**1.3000E+03**	**1.3000E+03**	1.3004E+03	**1.3000E+03**	1.3396E+03
F13	std. dev.	1.1638E+03	1.1764E-01	2.1390E-01	2.2023E+01	**1.4404E-12**	9.4550E+02
	median	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**
	time (s)	1.1823E+03	**1.1203E+03**	1.3240E+03	1.3519E+03	2.2973E+03	1.2919E+03
	rank	6	1	1	4	1	5
	best sol.	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**
	worst	4.3719E+04	**1.4049E+03**	1.4098E+03	1.9725E+03	2.9981E+04	4.0123E+04
	mean	1.4372E+03	**1.4000E+03**	**1.4000E+03**	1.4002E+03	1.4380E+03	1.4490E+03
F14	std. dev.	9.8343E+02	**9.7186E-02**	1.9507E-01	1.1450E+01	7.4926E+02	1.0440E+03
	median	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**
	time (s)	1.2399E+03	**1.2047E+03**	1.3972E+03	1.4269E+03	1.4227E+03	1.3848E+03
	rank	4	1	1	3	5	6
	best sol.	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**
	worst	4.5664E+04	1.5058E+03	**1.5042E+03**	2.6164E+03	2.8764E+04	3.7273E+04
	mean	1.5495E+03	**1.5000E+03**	**1.5000E+03**	1.5004E+03	1.5469E+03	1.5311E+03
F15	std. dev.	1.0944E+03	1.1533E-01	**8.3638E-02**	2.2327E+01	7.1363E+02	8.2375E+02
	median	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**
	time (s)	1.2324E+03	**1.2069E+03**	1.3965E+03	1.4265E+03	1.4194E+03	1.3757E+03
	rank	6	1	1	3	5	4
	best sol.	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**
	worst	3.5427E+04	**1.6077E+03**	1.6102E+03	2.1847E+03	2.2670E+04	4.1231E+04
	mean	1.6593E+03	**1.6000E+03**	**1.6000E+03**	1.6002E+03	1.6266E+03	1.6540E+03
F16	std. dev.	1.0723E+03	**1.5425E-01**	2.0336E-01	1.1693E+01	5.6123E+02	1.1529E+03
	median	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**
	time (s)	**1.2358E+03**	1.3629E+03	1.3844E+03	1.4196E+03	1.4218E+03	1.3731E+03
	rank	6	1	1	3	4	5
	best sol.	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**
	worst	3.9835E+04	**1.7052E+03**	1.7067E+03	2.5416E+03	2.9655E+04	3.3118E+04
	mean	1.7447E+03	**1.7000E+03**	**1.7000E+03**	1.7003E+03	1.7279E+03	1.7639E+03
F17	std. dev.	9.7307E+02	**1.0390E-01**	1.3415E-01	1.6832E+01	6.5170E+02	1.0494E+03
	median	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**
	time (s)	1.3627E+03	1.2602E+03	**1.0172E+03**	1.4808E+03	1.4602E+03	1.4406E+03
	rank	5	1	1	3	4	6
	best sol.	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**
	worst	3.8092E+04	**1.8049E+03**	1.8060E+03	2.8111E+03	3.4514E+04	3.7507E+04
	mean	1.8345E+03	**1.8000E+03**	**1.8000E+03**	1.8004E+03	1.8336E+03	1.8356E+03
F18	std. dev.	8.4818E+02	**9.7483E-02**	1.1942E-01	2.0221E+01	8.0963E+02	9.0324E+02
	median	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**
	time (s)	1.3616E+03	1.2696E+03	**1.0165E+03**	1.4873E+03	1.4599E+03	1.4518E+03
	rank	5	1	1	3	4	6
	best sol.	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**
	worst	4.5671E+04	**1.9048E+03**	1.9098E+03	2.3514E+03	2.3902E+04	4.6128E+04
	mean	1.9556E+03	**1.9000E+03**	**1.9000E+03**	1.9002E+03	1.9334E+03	1.9607E+03
F19	std. dev.	1.1585E+03	**9.6803E-02**	1.9633E-01	9.0284E+00	6.0364E+02	1.2274E+03
	median	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**
	time (s)	1.3391E+03	1.2570E+03	**1.0079E+03**	1.4575E+03	1.4408E+03	1.4411E+03
	rank	5	1	1	3	4	6
	best sol.	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**
	worst	3.8227E+04	2.0038E+03	**2.0020E+03**	2.8139E+03	2.9988E+04	4.5353E+04
	mean	2.0394E+03	**2.0000E+03**	**2.0000E+03**	2.0003E+03	2.0331E+03	2.0397E+03
F20	std. dev.	8.9923E+02	7.6339E-02	**4.0666E-02**	1.6278E+01	6.5115E+02	9.9353E+02
	median	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**
	time (s)	1.4206E+03	1.3141E+03	**1.0583E+03**	1.5537E+03	1.8627E+03	1.5056E+03
	rank	5	1	1	3	4	6

Table 7.3. The statistical results of CEC-2017 functions for dimension 30 (continued)

function	Metrics	STO	LFSTO-direct	LFSTO-randprob	LFSTO-randrand (step_size=0.1)	LFSTO-trial	LFSTO-trap
	best sol.	**2.2422E+03**	2.2537E+03	2.2463E+03	2.2446E+03	3.3067E+03	2.2393E+03
	worst	7.4344E+04	**2.4244E+03**	2.5087E+03	6.6850E+03	1.2535E+05	1.6452E+05
	mean	2.4029E+03	2.2634E+03	**2.2531E+03**	2.2551E+03	3.7513E+03	2.4161E+03
F21	std. dev.	2.5849E+03	**9.6444E+00**	1.2426E+01	9.4591E+01	3.3122E+03	3.8705E+03
	median	**2.2435E+03**	2.2606E+03	2.2519E+03	2.2508E+03	3.3691E+03	2.2393E+03
	time (s)	2.4060E+03	2.2779E+03	**1.7462E+03**	2.5629E+03	2.5630E+03	2.5115E+03
	rank	4	3	1	2	6	5
	best sol.	3.1985E+03	2.3950E+03	2.3812E+03	**2.3773E+03**	3.1848E+03	3.1523E+03
	worst	3.6190E+03	**2.5843E+03**	2.5964E+03	3.8813E+03	3.9129E+03	3.8694E+03
	mean	3.2175E+03	2.4019E+03	**2.3860E+03**	2.3972E+03	3.1914E+03	3.1873E+03
F22	std. dev.	5.2043E+01	**1.1427E+01**	1.2142E+01	7.9565E+01	2.5547E+01	6.6101E+01
	median	3.2007E+03	2.4015E+03	**2.3812E+03**	2.3820E+03	3.1848E+03	3.1523E+03
	time (s)	2.4825E+03	2.3683E+03	**1.7792E+03**	2.5674E+03	2.6680E+03	2.5953E+03
	rank	6	3	1	2	5	4
	best sol.	2.4949E+03	2.4979E+03	2.4945E+03	2.4961E+03	4.7890E+03	**2.4941E+03**
	worst	8.0607E+04	2.7456E+03	2.6994E+03	**2.6260E+03**	6.9721E+04	8.3021E+04
	mean	2.6315E+03	2.5023E+03	2.5040E+03	**2.4995E+03**	5.1350E+03	2.6313E+03
F23	std. dev.	2.3128E+03	1.0380E+01	9.1783E+00	**8.9494E+00**	2.2463E+03	2.5097E+03
	median	2.4978E+03	2.4979E+03	2.5065E+03	**2.4974E+03**	4.9422E+03	2.4941E+03
	time (s)	3.1591E+03	3.0231E+03	**2.2548E+03**	3.2592E+03	3.3453E+03	3.2751E+03
	rank	5	2	3	1	6	4
	best sol.	2.5973E+03	2.6003E+03	**2.5821E+03**	2.5900E+03	3.2954E+03	2.5927E+03
	worst	6.0980E+04	**2.7129E+03**	2.7738E+03	2.7270E+03	3.3795E+04	7.2708E+04
	mean	2.7157E+03	2.6058E+03	**2.5850E+03**	2.5960E+03	3.4669E+03	2.6727E+03
F24	std. dev.	1.9796E+03	9.1912E+00	1.1249E+01	**8.6086E+00**	1.2073E+03	1.6621E+03
	median	2.5998E+03	2.6046E+03	**2.5821E+03**	2.5965E+03	3.2954E+03	2.5949E+03
	time (s)	2.9537E+03	2.7803E+03	**2.0732E+03**	3.0333E+03	3.1237E+03	3.0434E+03
	rank	5	3	1	2	6	4
	best sol.	3.0339E+03	3.0811E+03	3.0251E+03	**3.0247E+03**	9.8909E+03	3.0328E+03
	worst	2.0754E+04	**3.0811E+03**	3.7214E+03	9.0464E+03	4.5832E+04	5.1769E+04
	mean	3.2498E+03	3.0927E+03	**3.0376E+03**	3.0618E+03	1.0799E+04	3.2802E+03
F25	std. dev.	1.4733E+03	**3.4301E+01**	4.2424E+01	1.4285E+02	1.5757E+03	1.6838E+03
	median	3.0345E+03	3.0811E+03	**3.0251E+03**	3.0548E+03	9.8909E+03	3.0328E+03
	time (s)	3.4536E+03	3.0811E+03	**2.4052E+03**	3.6646E+03	3.6317E+03	3.6729E+03
	rank	5	4	1	2	3	6
	best sol.	**2.8093E+03**	2.8314E+03	2.8103E+03	2.8154E+03	7.6783E+03	2.8147E+03
	worst	5.4993E+05	**3.0872E+03**	3.5962E+03	3.8263E+03	5.9615E+05	8.0840E+05
	mean	3.6069E+03	2.8412E+03	**2.8178E+03**	2.8226E+03	1.0313E+04	3.8100E+03
F26	std. dev.	1.5429E+04	**1.6013E+01**	2.0761E+01	2.7474E+01	1.5317E+04	2.0897E+04
	median	2.8159E+03	2.8363E+03	**2.8103E+03**	2.8196E+03	9.2349E+03	2.8201E+03
	time (s)	3.3159E+03	3.4847E+03	**2.3251E+03**	3.4113E+03	3.6118E+03	3.5459E+03
	rank	4	3	1	2	6	5
	best sol.	4.2412E+04	3.2865E+03	3.2439E+03	**3.2134E+03**	4.1478E+04	3.8027E+04
	worst	8.8496E+04	**8.8515E+03**	1.7521E+04	6.5091E+04	7.1414E+04	8.2003E+04
	mean	4.5376E+04	3.3222E+03	**3.3071E+03**	3.3495E+03	4.3500E+04	4.3583E+04
F27	std. dev.	4.8899E+03	**1.9114E+02**	3.7564E+02	2.4411E+03	2.7184E+03	6.2895E+03
	median	4.2412E+04	3.3027E+03	**3.2439E+03**	3.2543E+03	4.3110E+04	4.3013E+04
	time (s)	**2.5755E+03**	2.6430E+03	4.0249E+03	4.0026E+03	4.2288E+03	4.1771E+03
	rank	6	2	1	3	4	5
	best sol.	4.1776E+03	**3.0463E+03**	3.0481E+03	3.0523E+03	3.5414E+03	3.0465E+03
	worst	2.7200E+04	**3.1475E+03**	3.1665E+03	3.2159E+03	3.5933E+04	2.7662E+04
	mean	4.3299E+03	**3.0487E+03**	3.0537E+03	3.0558E+03	3.6331E+03	3.1198E+03
F28	std. dev.	6.2926E+02	8.6736E+00	**8.5700E+00**	8.8692E+00	9.9823E+02	8.4907E+02
	median	4.3020E+03	**3.0463E+03**	3.0518E+03	3.0523E+03	3.5414E+03	3.0546E+03
	time (s)	**2.5648E+03**	2.6222E+03	4.0114E+03	4.0938E+03	4.2036E+03	4.1995E+03
	rank	6	1	2	3	5	4
	best sol.	1.4021E+03	1.4018E+03	1.4018E+03	1.4018E+03	1.4021E+03	**1.4016E+03**
	worst	1.4048E+03	1.4086E+03	1.4066E+03	1.4070E+03	1.4079E+03	**1.4046E+03**
	mean	1.4023E+03	**1.4018E+03**	**1.4018E+03**	**1.4018E+03**	1.4025E+03	**1.4018E+03**
F29	std. dev.	2.8809E-01	1.5013E-01	**1.0578E-01**	1.1316E-01	3.9906E-01	3.5065E-01
	median	1.4023E+03	1.4018E+03	1.4018E+03	1.4018E+03	1.4023E+03	**1.4016E+03**
	time (s)	**1.7611E+03**	1.8018E+03	2.4607E+03	2.6062E+03	2.5743E+03	2.4522E+03
	rank	5	1	1	1	6	1
	best sol.	1.3018E+03	1.3501E+03	1.3052E+03	1.3018E+03	**1.3014E+03**	1.3020E+03
	worst	**1.3046E+03**	1.3717E+03	1.3052E+03	1.3072E+03	1.3094E+03	1.3051E+03
	mean	**1.3019E+03**	1.3508E+03	1.3052E+03	1.3031E+03	1.3022E+03	1.3021E+03
F30	std. dev.	**1.9763E-01**	1.4209E+00	0.0000E+00	1.0225E+00	5.1308E-01	3.0542E-01
	median	**1.3018E+03**	1.3501E+03	1.3052E+03	1.3036E+03	1.3021E+03	1.3020E+03
	time (s)	**1.7609E+03**	1.7837E+03	2.4372E+03	2.5247E+03	2.5613E+03	2.4607E+03
	rank	1	6	5	4	3	2
	mean rank	4.72	2.58	2.17	2.96	3.75	4.17
	final rank	**6**	**2**	**1**	**3**	**4**	**5**

LFSTO-randprob, LFSTO-randprob (step_size=0.1), and LFSTO-trap. Finally, LFSTO-direct acquired the second position, beating all other methods except LFSTO-randprob.

Table 7.4 shows that for dimension 50, the LFSTO-direct method outperforms the other five methods for every function: F3, F7, F11, F14-F19, F21-F26, and F28 for the worst possible solution; F11-F20 together with LFSTO-randprob for the mean; F3, F6, F8, F11, F14-F19, and F21-F26 for the standard deviation; F11-F20 along with the other five strategies for the median value; and F10, F11, and F8-F11, F13-F16, and F20 for the execution time. Thus, LFSTO-direct and LFSTO-randprob shared first place for functions F11–F20. In the end, LFSTO-direct came in third place, outperforming every other approach except LFSTO-randprob and LFSTO-randprob (step_size=0.1).

In general, this study used step size 0.01 in the Levy function; however, step size is a significant and aggressive parameter for the LF distribution. To check the step size uncertainty and impact on the optimization techniques, this research utilized step size 0.1 for the LFSTO-randprob and named it LFSTO-randprob (step_size=0.1). So, when considering LFSTO-randprob (step_size = 0.1) for dimensions 10 and 30 in Tables 7.2 and 7.3, respectively, the results show a significant difference from LFSTO-randprob for all the metrics. In both scenarios, this approach gained rank third, where LFSTO-randprob is first and outperforms the rest of the strategies. Whatever is in dimension 50, as shown in Table 7.4, the results are near the solution to LFSTO-rand. For this reason, this approach got the second overall rank, which is just after the first position of LFSTO-rand strategy.

7.5. Evaluation of Experimental Results of LFSTO-randprob

The given statistical results are in Table 7.2 for dimension 10, where LFSTO-randprob shows better performance than the other five strategies in the case of the best solution for the functions F3, F6, F9 (along with LFSTO-direct), F10, F22, and F24-F26, but obtained the same results as the other five for the functions F11-F20. In terms of the worst solution, the functions F7, F9-F11, F14, F16, F20, and F22 show better results than other strategies. In the average (mean) value scenario, LFSTO-randprob outperforms individually for functions F3, F9, F10, F21-F23, and F25-F27, as well as along with LFSTO-direct for the functions F11-F20. The functions F8, F9, F11, F14, F16, F18, F20, F21, and F28 perform better in a standard deviation context. In the median value term,

the functions F6, F9, F11–F20 (along with other methods), F22, and F29 outperform all five strategies. So, in this dimension 10 scenario, LFSTO-randprob got the first position in single for the functions F3, F9, F21-F23, and F25-F27, as well as along with LFSTO-direct for function F11-F20. The final rank is first, which reveals this strategy is better and superior to other integration strategies.

The results are shown in Table 7.3 for dimension 30, where LFSTO-randprob got the same results as the other five strategies when it came to functions F11–F20 in the best optimal solution case; however, this is the better result. In terms of the worst solution, the functions F11, F15, and F20 show better results than other strategies.

In the average value case, LFSTO-randprob does better for functions F3, F21, F22, and F24–F27 on its own, with LFSTO-direct for functions F11–F20, and with LFSTO-direct, LFSTO-randprob (step_size = 0.1), and LFSTO-trap for F29. The functions F8, F11, F12, F15, F20, and F28 perform better in a standard deviation context, and the functions F11–F20 (along with other methods), F22, and F24–F27 outperform all five strategies in the median value case. The LFSTO-randprob approach takes less time for execution than the rest of the strategies for F12 and F17–F26. Lastly, LFSTO-randprob got the first position in single for the functions F3, F21, F22, and F24-F27, as well as along with LFSTO-direct for functions F11-F20 and with LFSTO-direct, LFSTO-randprob (step_size = 0.1), and LFSTO-trap for F29. Also, out of all the integration procedures, LFSTO-randprob came out on top in the 30-dimension scenario, proving its superiority.

To determine the optimal solution for functions F4, F10, F22, F23, and F27, LFSTO-randprob outperforms the other five techniques for dimension 50, as shown in Table 7.4. On the other hand, functions F11–F20 gave the same results as the other five techniques, including LFSTO-randprob (step_size=0.1) for F29. Regarding the worst-case scenario, F12 and F20 functions outperform other approaches. In the mean value scenario, LFSTO-randprob works better on its own for functions F21–F23 and F28 and when combined with LFSTO-direct for functions F11–F20. In a standard deviation context, F9, F12, F20, and F28-F30 do better than all five strategies. But for the functions F1, F4, F11–F20 (along with other methods), and F27, they outperform all five strategies in the median value case. Execution time for the LFSTO-randprob method is lower for F3-F7 and F27-F30 than the other five methods. Finally, for functions F11–F20 (along with LFSTO-direct), F21–F23, and F28, LFSTO–randprob ranked first. Furthermore, in

the 50-dimension scenario, LFSTO-randprob demonstrated its superiority over all other integration processes by acquiring its final position first.

Table 7.4. The statistical results of CEC-2017 functions for dimension 50

function	Metrics	STO	LFSTO-direct	LFSTO-randprob	LFSTO-randprob (step_size=0.1)	LFSTO-trial	LFSTO-trap
	best sol.	3.0625E+04	7.3792E+04	9.5320E+03	**4.0499E+03**	1.6502E+05	1.2748E+04
	worst	2.1193E+12	1.3237E+12	1.3164E+12	**1.2903E+12**	1.8668E+12	1.6087E+12
	mean	3.3853E+10	6.2449E+10	3.9992E+10	3.2763E+10	3.4899E+10	**3.2299E+10**
F1	std. dev.	1.2844E+11	1.6275E+11	1.4955E+11	1.2831E+11	1.3012E+11	**1.2127E+11**
	median	1.3490E+07	1.1279E+08	**1.6742E+06**	5.3732E+06	7.2132E+08	2.7711E+06
	time (s)	**1.0717E+03**	1.1882E+03	9.2768E+02	1.2804E+03	1.3024E+03	1.2485E+03
	rank	3	6	5	2	4	1
	best sol.	7.1727E+04	**1.1882E+03**	8.5742E+04	6.5316E+04	5.5406E+04	1.2159E+05
	worst	1.5987E+06	**5.7356E+05**	1.4013E+09	2.9130E+07	8.8756E+05	2.8097E+09
	mean	1.7871E+05	1.3140E+05	4.8586E+05	1.6801E+05	**1.1890E+05**	9.1776E+05
F3	std. dev.	1.1002E+05	**5.8962E+04**	2.1705E+07	6.9525E+05	5.9401E+04	4.3522E+07
	median	1.6169E+05	1.2289E+05	1.4596E+05	1.4208E+05	**1.1051E+05**	2.2657E+05
	time (s)	1.3302E+03	1.4642E+03	**1.1255E+03**	1.5685E+03	1.5896E+03	1.5540E+03
	rank	4	2	5	3	1	6
	best sol.	5.5025E+02	5.3977E+02	**4.7837E+02**	5.5576E+02	5.5402E+02	5.9773E+02
	worst	7.8188E+04	5.2675E+04	5.3212E+04	**4.9177E+04**	6.5210E+04	7.4984E+04
	mean	1.2030E+03	1.2179E+03	1.3219E+03	**1.0259E+03**	1.1351E+03	1.1088E+03
F4	std. dev.	3.3677E+03	3.1633E+03	3.9005E+03	**2.1462E+03**	2.8930E+03	2.7872E+03
	median	6.6123E+02	5.5288E+02	**4.8730E+02**	6.3880E+02	5.9640E+02	6.2003E+02
	time (s)	1.2846E+03	1.3290E+03	**1.0359E+03**	1.4369E+03	1.4644E+03	1.3659E+03
	rank	4	5	6	1	3	2
	best sol.	8.0446E+02	8.9798E+02	8.7609E+02	8.4253E+02	**7.2685E+02**	8.3928E+02
	worst	1.5747E+03	1.2904E+03	1.3171E+03	**1.2575E+03**	7.2685E+02	1.5553E+03
	mean	8.2495E+02	9.0818E+02	8.8937E+02	8.8833E+02	**7.2685E+02**	8.7462E+02
F5	std. dev.	6.6821E+01	3.7440E+01	4.8789E+01	7.0894E+01	**1.5928E-09**	7.4622E+01
	median	8.0446E+02	8.9799E+02	8.7610E+02	8.4744E+02	**7.2685E+02**	8.4141E+02
	time (s)	1.1859E+03	1.2885E+03	**9.9597E+02**	1.4111E+03	2.4392E+03	1.3441E+03
	rank	2	6	5	4	1	3
	best sol.	6.6112E+02	6.8557E+02	6.5946E+02	6.5678E+02	**6.4719E+02**	6.6690E+02
	worst	8.1982E+02	7.6327E+02	7.7280E+02	**7.5611E+02**	7.6194E+02	8.0346E+02
	mean	6.6578E+02	6.9502E+02	6.7488E+02	6.6385E+02	**6.5548E+02**	6.7822E+02
F6	std. dev.	1.1665E+01	**9.2608E+00**	1.8309E+01	1.3847E+01	1.4768E+01	1.3470E+01
	median	6.6238E+02	6.9215E+02	6.6734E+02	6.5794E+02	**6.4889E+02**	6.7103E+02
	time (s)	1.3907E+03	1.4809E+03	**1.1382E+03**	1.5995E+03	1.6116E+03	1.5244E+03
	rank	3	6	4	2	1	5
	best sol.	1.1703E+03	1.7652E+03	1.6977E+03	1.3029E+03	**1.1999E+03**	1.3614E+03
	worst	4.4645E+03	**2.1609E+03**	2.1779E+03	2.2037E+03	3.4353E+03	3.9779E+03
	mean	**1.2025E+03**	1.7771E+03	1.7145E+03	1.3352E+03	1.2411E+03	1.4020E+03
F7	std. dev.	1.2491E+02	3.7624E+01	4.7263E+01	8.9071E+01	1.3092E+02	**1.1477E+02**
	median	**1.1704E+03**	1.7662E+03	1.6978E+03	1.3047E+03	1.1999E+03	1.3617E+03
	time (s)	1.9396E+03	2.0431E+03	**1.6081E+03**	2.1860E+03	2.1920E+03	2.0985E+03
	rank	1	6	5	3	2	4
	best sol.	1.0786E+03	1.2308E+03	1.1622E+03	1.1693E+03	1.1094E+03	**1.0672E+03**
	worst	1.8767E+03	1.6478E+03	1.6713E+03	**1.5760E+03**	1.7179E+03	1.6687E+03
	mean	**1.1038E+03**	1.2423E+03	1.1735E+03	1.2074E+03	1.1414E+03	1.1093E+03
F8	std. dev.	7.0441E+01	**4.0966E+01**	4.5110E+01	6.9231E+01	8.1252E+01	7.9091E+01
	median	1.0787E+03	1.2308E+03	1.1623E+03	1.1737E+03	1.1096E+03	**1.0744E+03**
	time (s)	1.4387E+03	**1.0933E+03**	1.2174E+03	1.6441E+03	1.7135E+03	1.5768E+03
	rank	1	6	4	5	3	2
	best sol.	1.6984E+04	2.1423E+04	2.6672E+04	1.8094E+04	**1.5309E+04**	1.7297E+04
	worst	5.9325E+04	5.6100E+04	5.8673E+04	**5.5645E+04**	6.1505E+04	8.5604E+04
	mean	**1.8404E+04**	2.5560E+04	2.7397E+04	2.2710E+04	1.8479E+04	1.9886E+04
F9	std. dev.	2.5479E+03	4.5792E+03	**1.9423E+03**	4.0450E+03	2.9927E+03	4.5189E+03
	median	1.7925E+04	2.3995E+04	2.6672E+04	2.1250E+04	**1.7752E+04**	1.9717E+04
	time (s)	1.5954E+03	**1.1929E+03**	1.3097E+03	1.8518E+03	1.8653E+03	1.8185E+03
	rank	1	5	6	4	2	3
	best sol.	1.5161E+04	1.3308E+04	**1.2575E+04**	1.4106E+04	7.6643E+03	1.3874E+04
	worst	1.7388E+04	1.8491E+04	1.7490E+04	**1.7298E+04**	1.8086E+04	1.7766E+04
	mean	1.5173E+04	1.3694E+04	1.3750E+04	1.4371E+04	**1.0898E+04**	1.4304E+04
F10	std. dev.	**1.1307E+02**	8.1475E+02	9.9959E+02	3.9896E+02	2.2807E+03	6.6921E+02
	median	1.5163E+04	1.3357E+04	1.3346E+04	1.4112E+04	**1.0910E+04**	1.3886E+04
	time (s)	1.8743E+03	**1.3644E+03**	1.4912E+03	2.1258E+03	2.6025E+03	2.1252E+03
	rank	6	2	3	5	1	4

Table 7.4. The statistical results of CEC-2017 functions for dimension 50 (continued)

function	Metrics	STO	LFSTO-direct	LFSTO-randprob	LFSTO-randprob (step_size=0.1)	LFSTO-trial	LFSTO-trap
F11	best sol.	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**
	worst	6.7004E+04	**1.1189E+03**	1.1204E+03	1.7850E+03	5.1951E+04	7.0851E+04
	mean	1.1563E+03	**1.1000E+03**	**1.1000E+03**	1.1002E+03	1.1317E+03	1.1596E+03
	std. dev.	1.3787E+03	**2.9323E-01**	3.1655E-01	1.0612E+01	9.3479E+02	1.5352E+03
	median	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**	**1.1000E+03**
	time (s)	2.2603E+03	**1.6259E+03**	1.7747E+03	2.5626E+03	2.5659E+03	2.5971E+03
	rank	5	1	1	3	4	6
F12	best sol.	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**
	worst	7.3443E+04	1.2115E+03	**1.2111E+03**	2.4610E+03	2.2388E+04	6.4670E+04
	mean	1.2691E+03	**1.2000E+03**	**1.2000E+03**	1.2003E+03	1.2219E+03	1.2534E+03
	std. dev.	1.8079E+03	1.7861E-01	**1.7266E-01**	1.9534E+01	5.0004E+02	1.5588E+03
	median	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**
	time (s)	2.2651E+03	1.6209E+03	1.7679E+03	2.5234E+03	**1.3448E+03**	2.6150E+03
		6	1	1	3	4	5
F13	best sol.	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**
	worst	7.9015E+04	1.3096E+03	1.3257E+03	2.2662E+03	**1.3000E+03**	6.7533E+04
	mean	1.3573E+03	**1.3000E+03**	**1.3000E+03**	1.3002E+03	**1.3000E+03**	1.3564E+03
	std. dev.	1.5452E+03	1.4846E-01	3.9813E-01	1.4967E+01	**7.1902E-14**	1.4818E+03
	median	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**	**1.3000E+03**
	time (s)	2.2670E+03	**1.6352E+03**	2.5830E+03	2.5321E+03	3.9968E+03	2.5745E+03
	rank	6	1	1	4	1	5
F14	best sol.	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**
	worst	7.0908E+04	**1.4203E+03**	1.4213E+03	2.7551E+03	4.4250E+04	6.0332E+04
	mean	1.4636E+03	**1.4000E+03**	**1.4000E+03**	1.4003E+03	1.4346E+03	1.4450E+03
	std. dev.	1.4576E+03	**3.1446E-01**	3.3044E-01	2.0993E+01	8.3052E+02	1.3152E+03
	median	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**	**1.4000E+03**
	time (s)	2.3946E+03	**1.7047E+03**	2.7081E+03	2.7309E+03	2.4560E+03	2.7108E+03
	rank	6	1	1	3	4	5
F15	best sol.	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**
	worst	7.4946E+04	**1.5129E+03**	1.5226E+03	4.2024E+03	4.3605E+04	6.1278E+04
	mean	1.5553E+03	**1.5000E+03**	**1.5000E+03**	1.5006E+03	1.5331E+03	1.5428E+03
	std. dev.	1.5499E+03	**2.0003E-01**	3.4963E-01	4.1863E+01	8.7102E+02	1.2315E+03
	median	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**	**1.5000E+03**
	time (s)	2.3884E+03	**1.6965E+03**	2.7027E+03	2.6787E+03	2.4299E+03	2.7199E+03
	rank	6	1	1	3	4	5
F16	best sol.	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**
	worst	7.1235E+04	**1.6159E+03**	1.6175E+03	3.0854E+03	6.0767E+04	6.7529E+04
	mean	1.6346E+03	**1.6000E+03**	**1.6000E+03**	1.6004E+03	1.6414E+03	1.6547E+03
	std. dev.	1.2108E+03	**2.4580E-01**	2.7106E-01	2.3012E+01	1.1479E+03	1.4683E+03
	median	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**	**1.6000E+03**
	time (s)	2.4060E+03	**1.7022E+03**	2.7088E+03	2.7149E+03	2.4343E+03	2.5077E+03
	rank	4	1	1	3	5	6
F17	best sol.	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**
	worst	3.9835E+04	**1.7138E+03**	1.7221E+03	3.5528E+03	4.9049E+04	7.3072E+04
	mean	1.7447E+03	**1.7000E+03**	**1.7000E+03**	1.7004E+03	1.7333E+03	1.7704E+03
	std. dev.	9.7307E+02	**2.1386E-01**	3.4299E-01	2.8703E+01	9.4043E+02	1.6909E+03
	median	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**	**1.7000E+03**
	time (s)	**1.3627E+03**	1.7818E+03	2.8277E+03	2.8542E+03	2.5304E+03	2.6530E+03
	rank	5	1	1	3	4	6
F18	best sol.	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**
	worst	3.8092E+04	**1.8091E+03**	1.8151E+03	3.2631E+03	4.6086E+04	7.8846E+04
	mean	1.8345E+03	**1.8000E+03**	**1.8000E+03**	1.8004E+03	1.8320E+03	1.8478E+03
	std. dev.	8.4818E+02	**1.4100E-01**	2.3421E-01	2.2666E+01	8.3603E+02	1.4930E+03
	median	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**	**1.8000E+03**
	time (s)	**1.3616E+03**	1.7712E+03	1.8827E+03	2.8259E+03	2.5243E+03	2.7379E+03
	rank	5	1	1	3	4	6
F19	best sol.	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**
	worst	8.3881E+04	**1.9091E+03**	1.9218E+03	4.7511E+03	5.4451E+04	7.7697E+04
	mean	1.9770E+03	**1.9000E+03**	**1.9000E+03**	1.9007E+03	1.9387E+03	1.9574E+03
	std. dev.	1.7233E+03	**1.4138E-01**	3.3736E-01	4.4167E+01	1.0446E+03	1.6467E+03
	median	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**	**1.9000E+03**
	time (s)	**2.4699E+03**	2.7162E+03	2.7734E+03	2.8504E+03	2.7726E+03	2.7532E+03
	rank	6	1	1	3	4	5
F20	best sol.	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**
	worst	4.5207E+04	2.0134E+03	**2.0116E+03**	3.8873E+03	4.2847E+04	7.6304E+04
	mean	2.0238E+03	**2.0000E+03**	**2.0000E+03**	2.0005E+03	2.0264E+03	2.0604E+03
	std. dev.	8.4015E+02	2.0710E-01	**1.7973E-01**	2.9237E+01	8.2926E+02	1.6502E+03
	median	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**	**2.0000E+03**
	time (s)	2.5753E+03	**2.2018E+03**	2.9811E+03	2.9397E+03	3.5545E+03	2.7428E+03
	rank	4	1	1	3	5	6

Table 7.4. The statistical results of CEC-2017 functions for dimension 50 (continued)

function	Metrics	STO	LFSTO-direct	LFSTO-randprob	LFSTO-randprob (step_size=0.1)	LFSTO-trial	LFSTO-trap
	best sol.	**2.3004E+03**	2.3596E+03	2.3122E+03	2.3092E+03	1.2919E+04	2.3037E+03
	worst	4.5099E+05	**2.9772E+03**	3.2231E+03	2.3351E+04	2.3967E+05	4.6081E+05
	mean	2.5771E+03	2.3666E+03	**2.3162E+03**	2.3233E+03	1.4428E+04	2.5391E+03
F21	std. dev.	8.1913E+03	**1.9910E+01**	2.4078E+01	3.2782E+02	5.8749E+03	7.7350E+03
	median	**2.3008E+03**	2.3596E+03	2.3122E+03	2.3170E+03	1.4568E+04	2.3037E+03
	time (s)	4.7816E+03	**3.7798E+03**	5.1148E+03	5.0265E+03	5.1107E+03	4.8411E+03
	rank	5	3	1	2	6	4
	best sol.	4.2772E+03	2.5280E+03	**2.4446E+03**	2.4565E+03	4.3412E+03	4.3449E+03
	worst	5.5753E+03	**3.0238E+03**	3.0791E+03	5.0769E+03	5.5336E+03	4.9762E+03
	mean	4.3918E+03	2.5293E+03	**2.4596E+03**	2.5023E+03	4.4210E+03	4.4089E+03
F22	std. dev.	1.2358E+02	**1.4935E+01**	2.2225E+01	2.7487E+02	7.8893E+01	7.8041E+01
	median	4.4194E+03	2.5280E+03	**2.4568E+03**	2.4610E+03	4.4100E+03	4.3836E+03
	time (s)	**4.8543E+03**	5.0995E+03	5.3794E+03	5.3076E+03	5.2716E+03	4.9467E+03
	rank	4	3	1	2	6	5
	best sol.	2.5477E+03	2.5910E+03	**2.5419E+03**	2.5480E+03	9.9913E+03	2.5445E+03
	worst	1.6858E+05	**3.2609E+03**	3.4103E+03	3.5180E+03	2.2757E+05	2.6184E+05
	mean	2.7334E+03	2.5945E+03	**2.5502E+03**	2.5555E+03	1.0587E+04	2.7853E+03
F23	std. dev.	3.7450E+03	**1.6658E+01**	2.9824E+01	2.0489E+01	4.7082E+03	5.5144E+03
	median	2.5481E+03	2.5915E+03	2.5493E+03	2.5547E+03	9.9913E+03	**2.5447E+03**
	time (s)	**6.2628E+03**	6.4699E+03	7.0947E+03	6.6212E+03	6.6376E+03	6.3839E+03
	rank	4	3	1	2	6	5
	best sol.	2.6452E+03	2.7102E+03	2.6570E+03	2.6496E+03	5.7583E+03	**2.6350E+03**
	worst	1.3885E+05	**3.0457E+03**	3.1235E+03	3.3267E+03	2.3294E+05	2.1673E+05
	mean	2.7998E+03	2.7242E+03	2.6600E+03	**2.6561E+03**	6.2403E+03	2.9125E+03
F24	std. dev.	3.2990E+03	**1.4987E+01**	1.8313E+01	1.8335E+01	4.8805E+03	5.2723E+03
	median	2.6527E+03	2.7250E+03	2.6570E+03	2.6497E+03	5.9942E+03	**2.6476E+03**
	time (s)	**5.7752E+03**	6.1243E+03	5.9272E+03	6.1683E+03	6.2575E+03	6.0215E+03
	rank	4	3	2	1	6	5
	best sol.	**3.0339E+03**	3.5367E+03	3.3367E+03	3.4350E+03	3.0367E+04	3.3503E+03
	worst	2.0754E+04	**6.1648E+03**	8.3810E+03	1.9703E+04	1.5547E+05	9.6874E+04
	mean	**3.2498E+03**	3.5664E+03	3.4053E+03	3.4749E+03	3.1940E+04	3.8907E+03
F25	std. dev.	1.4733E+03	**7.8269E+01**	1.0162E+02	3.6863E+02	3.0996E+03	4.8967E+03
	median	**3.0345E+03**	3.5649E+03	3.3945E+03	3.4350E+03	3.2190E+04	3.3503E+03
	time (s)	**3.4536E+03**	7.2099E+03	7.4215E+03	7.0670E+03	7.2564E+03	6.9560E+03
	rank	1	4	2	3	6	5
	best sol.	**2.8093E+03**	2.9271E+03	2.8259E+03	2.8392E+03	4.4947E+04	2.8357E+03
	worst	5.4993E+05	**4.8821E+03**	5.1297E+03	5.2350E+03	1.2579E+06	2.0495E+06
	mean	3.6069E+03	2.9579E+03	2.8592E+03	**2.8500E+03**	5.4216E+04	4.5413E+03
F26	std. dev.	1.5429E+04	**4.8372E+01**	7.6621E+01	5.6503E+01	3.2160E+04	4.4018E+04
	median	**2.8159E+03**	2.9542E+03	2.8523E+03	2.8472E+03	5.0149E+04	2.8357E+03
	time (s)	**3.3159E+03**	6.9875E+03	7.1942E+03	7.0786E+03	7.2623E+03	7.0538E+03
	rank	4	3	2	1	6	5
	best sol.	1.4250E+05	5.8319E+03	**3.5878E+03**	3.6183E+03	1.3395E+05	1.2911E+05
	worst	2.4610E+05	1.8056E+05	1.7094E+05	**9.0011E+03**	2.5002E+05	2.7428E+05
	mean	1.4998E+05	6.6023E+03	3.9421E+03	**3.6815E+03**	1.4017E+05	1.4363E+05
F27	std. dev.	1.3397E+04	3.5289E+03	5.4003E+03	**3.0126E+02**	8.6010E+03	1.5306E+04
	median	1.4250E+05	6.3655E+03	**3.6105E+03**	3.6712E+03	1.3767E+05	1.4147E+05
	time (s)	8.0205E+03	8.2510E+03	**5.3383E+03**	8.2825E+03	8.3103E+03	8.1087E+03
	rank	4	3	2	1	5	6
	best sol.	6.1411E+03	3.0749E+03	3.0545E+03	**3.0520E+03**	6.4937E+03	3.0582E+03
	worst	1.4990E+05	**3.2961E+03**	3.5948E+03	3.4052E+03	7.4296E+04	6.4662E+04
	mean	6.2215E+03	3.0848E+03	**3.0574E+03**	3.0587E+03	6.6855E+03	3.1597E+03
F28	std. dev.	2.7558E+03	1.3910E+01	**1.1694E+01**	1.2717E+01	1.4370E+03	1.7134E+03
	median	6.1411E+03	3.0844E+03	3.0563E+03	**3.0524E+03**	6.5456E+03	3.0584E+03
	time (s)	8.0033E+03	7.2961E+03	**5.2772E+03**	8.2800E+03	8.3480E+03	8.2595E+03
	rank	5	3	1	2	6	4
	best sol.	1.4056E+03	1.4033E+03	**1.4032E+03**	**1.4032E+03**	8.3480E+03	1.4042E+03
	worst	1.4119E+03	1.4151E+03	1.4111E+03	1.4122E+03	1.4100E+03	**1.4099E+03**
	mean	1.4060E+03	1.4033E+03	1.4033E+03	**1.4032E+03**	1.4053E+03	1.4044E+03
F29	std. dev.	4.4344E-01	1.9042E-01	**1.3080E-01**	1.6159E-01	7.2198E-01	7.2110E-01
	median	1.4061E+03	1.4033E+03	1.4033E+03	**1.4032E+03**	1.4050E+03	1.4042E+03
	time (s)	4.6600E+03	4.6443E+03	**3.5059E+03**	4.9470E+03	4.9556E+03	4.8107E+03
	rank	6	2	2	1	5	4
	best sol.	**1.3051E+03**	1.3571E+03	1.3084E+03	1.3074E+03	**1.3051E+03**	1.3053E+03
	worst	**1.3084E+03**	1.3875E+03	1.3102E+03	1.3126E+03	1.3097E+03	1.3100E+03
	mean	**1.3056E+03**	1.3575E+03	1.3084E+03	1.3085E+03	**1.3056E+03**	**1.3056E+03**
F30	std. dev.	5.4335E-01	8.9395E-01	**2.8410E-02**	6.2661E-01	6.2719E-01	4.8009E-01
	median	1.3058E+03	1.3571E+03	1.3084E+03	1.3086E+03	1.3058E+03	**1.3053E+03**
	time (s)	4.6545E+03	4.6734E+03	**3.4967E+03**	4.8942E+03	4.8964E+03	4.7914E+03
	rank	1	6	4	5	1	1
	mean rank	4.00	3.17	2.44	2.75	3.79	4.44
	final rank	5	3	1	2	4	6

7.6. Evaluation of Experimental Results of LFSTO-trial

For dimension 10, as presented in Table 7.2, the LFSTO trial shows better results for the functions F3 and F11–F20, which are the same as other approaches in terms of the best solution. This strategy performs better than other strategies for F1 and F13 in the worst-case scenario. When considering the mean value, only two functions, namely F1 and F13, outperform other strategies, which is why the LFSTO-trial method was ranked only two times for the aforementioned functions. F1, F5-F7, and F22 functions acquire superior results in the standard deviation context, and functions F11-F20 reveal superiority along with the rest of the methods in the median value case. Finally, this strategy conquered the fourth position among the six strategies.

The LFSTO-trial method does better than the five others for all of the following functions in Table 7.3 for dimension 30: F3, F5-F7, F9, F11-F20 (along with the other approaches), and F30 for the best optimal solution; F5, F6, and F13 for the worst solution; F4-F7, F9, and F13 for the average value; F4, F5, and F13 for the standard deviation; and F1, F3-F7, and F11-F20 (along with the other five methods) for the median value. So, this method acquired first place for the total six functions, including F4-F7, F9, and F13, and final rank 4, which only outperforms LFSTO-trap and STO techniques.

Among the six functions in Table 7.4 for dimension 50, the LFSTO-trial method outperforms the other five in the following functions: F5-F7, F9, F11-F20, and F30 for the best optimal solution; F13 for the worst optimal solution; F3, F5, F6, F10, F13, and F30 for the mean value case; F5, and F13 for the standard deviation; and F3, F5, F6, F9, F10, and F11-F20 for the median value, along with the other five methods. For all six functions—F3, F5, F6, F10, F13, and F30—this method achieved first place; it also achieved final rank 4, surpassing both LFSTO-trap and STO strategies.

7.7. Evaluation of Experimental Results of LFSTO-trap

For dimension 10, Table 7.2 shows the twenty-nine functions of six strategies. From this result table, the LFSTO-trap method does better than the other five in F3, F7, F11–F20, like the other five, F23, F29, and F30 for the best optimal solution; F7 and F30 for the mean value case, which made first rank this method only for these two functions; only F30 for the standard deviation; and F3, F7, F23, F24, F29, F30, and F11–F20 along

with the other five methods for the median value. No function outperforms others in terms of worst value and execution time. Overall, this method achieved a final rank of 5, surpassing only standard STO strategies.

When looking at dimension 30, Table 7.3 shows that the LFSTO-trap outperforms the other five in F8, F23, and F29 for finding the best optimal solution; however, the same results are found for F11–F20. The function F29 shows a better result for the worst-case solution. It also does better than the other five in F1 and F29 for the mean value case, putting it at the top of the list for those two functions. There are only two functions, F29 and F11–F20 (along with the other five methods), for the median value. There is no function that outperforms others in terms of standard deviation and execution time. Overall, this method achieved a final rank of 5, outpaced only STO strategies.

In terms of dimension 50, the LFSTO-trap performs superiorly to the other five in F8 and F24 in terms of finding the best optimal solution; however, this is also true for F11–F20, which is the same solution as the other five approaches, as shown in Table 7.3. In the worst-case scenario, the function F29 produces a more favorable outcome. When it comes to the mean value case, it outperforms the other five in both F1 and F30, placing it at the top of both functions. F7, F8, F23, F24, F30, and F11-F20, along with the other five approaches, for the median value. Regarding execution time and standard deviation, no function performs better than the others. In the end, this method secured the sixth spot, which did not outperform any other integration strategies.

7.8. Non-parametric Statistical Evaluations

The Wilcoxon rank sum test (Wilcoxon, 1992) was used on the 10, 30, and 50 dimensions of the statistical results to see if there was a statistically significant difference between standard STO and the other five integration strategies. The p-value and statistics value were utilized in this non-parametric test to check the level of significance. The magnitude of disparity among each group being assessed can be quantified by the statistic's value, which indicates the test statistic generated from the data being analyzed. This aids in finding out if the techniques' effectiveness differs significantly. In the p-value case, this test used a 0.05 threshold value as the significance level (Wilcoxon, 1947). If the p-value is less than the threshold, then there is a significant difference between the two methods.

According to the test results, which are presented in Table 7.5, in 10-dimension cases, LFSTO-direct, LFSTO-randprob, and LFSTO-randprob (step_size = 0.1) outperform STO because all three cases have a p-value less than the threshold; however, there is no significant difference between STO and LFSTO-trial or LFSTO-trap. In both the 30- and 50-dimensional cases, the results reveal that LFSTO-direct, LFSTO-randprob, LFSTO-randprob (step_size = 0.1), and LFSTO-trap have significant statistical superiority to STO. However, no statistically significant difference was found between the STO and LFSTO-trial strategies. In Table 7.5, statistical significance is denoted by the "Yes" and "No" indicators.

Table 7.5. Comparison of STO with other five strategies using Wilcoxon rank sum test

Compared Strategies	CEC-2017 Benchmark Functions								
	Dim=10			Dim=30			Dim=50		
	statistics	**p-value**	**significance**	**statistics**	**p-value**	**significance**	**statistics**	**p-value**	**significance**
STO vs. LFSTO-direct	2482.0	3.229838e-10	Yes	2670.5	2.726101e-09	Yes	3355.0	0.000002	Yes
STO vs. LFSTO-randprob	2869.0	6.054466e-08	Yes	2106.5	3.291793e-12	Yes	3428.0	0.000005	Yes
STO vs. LFSTO-randprob (step_size=0.1)	2455.0	2.357356e-10	Yes	3166.5	6.984915e-07	Yes	3807.5	0.000098	Yes
STO vs. LFSTO-trial	5445.5	4.978736e-01	No	5395.0	4.408669e-01	No	4955.0	0.145787	No
STO vs. LFSTO-trap	5602.5	9.103677e-01	No	4864.5	4.662952e-02	Yes	4291.5	0.003584	Yes

7.9. Discussion

This thesis mainly investigates the integration strategies of LF into metaheuristic algorithms that were most employed in previous literature. For investigation and experimentation, this research used the STO algorithm, which is a novel metaheuristic algorithm, and the CEC2017 benchmark suite to show the effectiveness of the combination approach. So, after analyzing the results, we can see that for dimensions 10 and 30, the results are almost similar. In the 10-dimension case, the LFSTO-randprob strategy shows superior results to other methods, where this method gained a 2.03 mean rank and, in contrast, the standard STO obtained a 4.68 mean rank as the worst rank. The LFSTO-direct integration and LFSTO-randprob (step_size=0.1) approaches are the closest results of the LFSTO-randprob method, where they acquire second and third positions by gaining 2.34 and 2.41 mean rank, respectively. The findings further express that the LFSTO-trial and LFSTO-trap strategies keep their positions fourth and fifth by acquiring mean rank 4.27 and 4.48, respectively. Those values have a significant difference from the other first three strategies. Those values have a significant difference from the other first three strategies, and this difference in the convergent plot is also depicted in Figure 7.1-7.29.

In the 30-dimension case again, the LFSTO-randprob approach surpasses the other five methods, where its mean rank value is 2.17, and by obtaining the mean rank of 2.58 and 2.96, LFSTO-direct and LFSTO-randprob (step_size = 0.1) are the nearest competitors, respectively. On the other hand, the standard STOs mean rank is 4.72, and this strategy gained sixth place again. In like 10 dimensions, LFSTO-trial and LFSTO-trap continue their positions fourth and fifth with a mean rank of 3.75 and 4.17, respectively. However, in both cases, standard STO shows poorer quality results than other integration strategies, and the convergence curve in Figure 7.30-7.58 also expresses this degree of fluctuation. Further, in the 50-dimension scenario, LFSTO-randprob gains better results, and LFSTO-randprob (step_size = 0.1) shows just a near result of this approach, which means second-best results with their mean rank values of 2.44 and 2.75, respectively. It reveals that although the step size changes, the LFSTO-randprob approach is the most effective among LF integration strategies. However, in this case, LFSTO-direct acquired the third position with its mean rank value of 3.17, and standard STO produces better results than only the LFSTO-trap method, where traditional STO's mean rank is 4.00 and LFSTO-trap's mean rank is 4.44. Also, in this case, the LFSTO-trial

strategy keeps its fourth place with a mean rank of 3.79. The convergence change graph of all of these integration approaches for 50 dimensions is represented in Figure 7.59-7.88. Moreover, the non-parametric statistical test exhibited that for all the dimensions, several incorporation strategies significantly differed from standard STO, and this is also true for the convergence changes. In the end, after all evaluations of the results, this study expresses that the integration approach shows better results than standard STO only except in the dimension 50 case, where only the LFSTO-trap method is worse than STO.

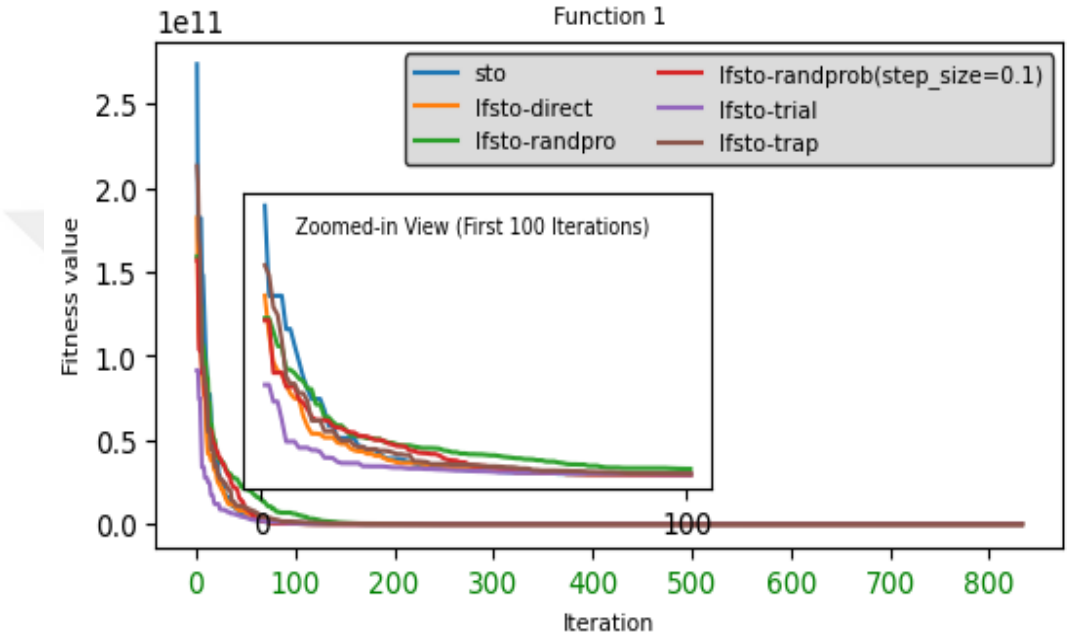


Figure 7.1. Convergence graph of LF integration strategies on CEC-2017 test functions F1 for dimension 10

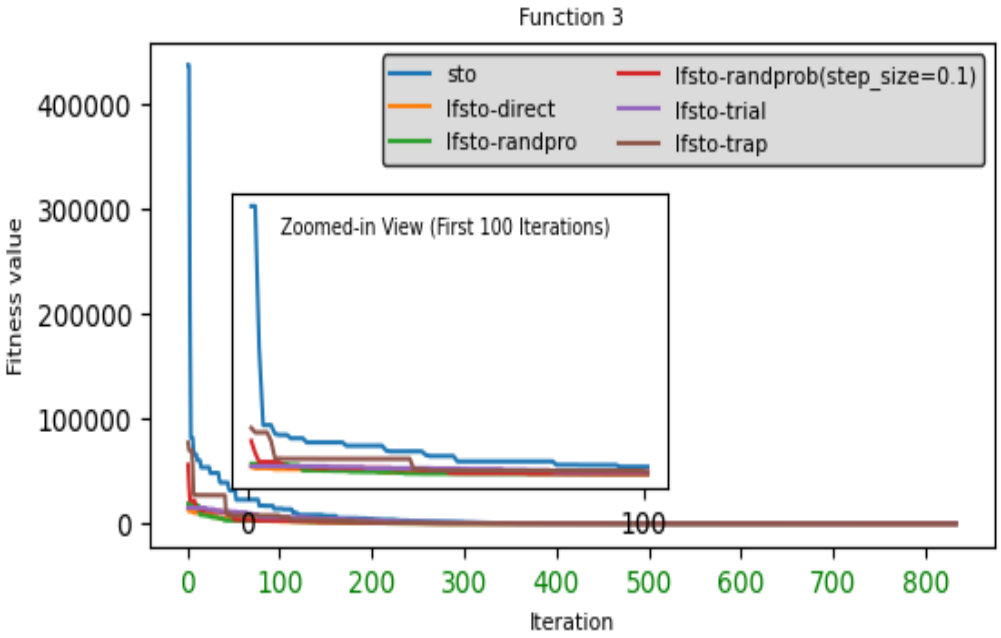


Figure 7.2. Convergence graph of LF integration strategies on CEC-2017 test functions F3 for dimension 10

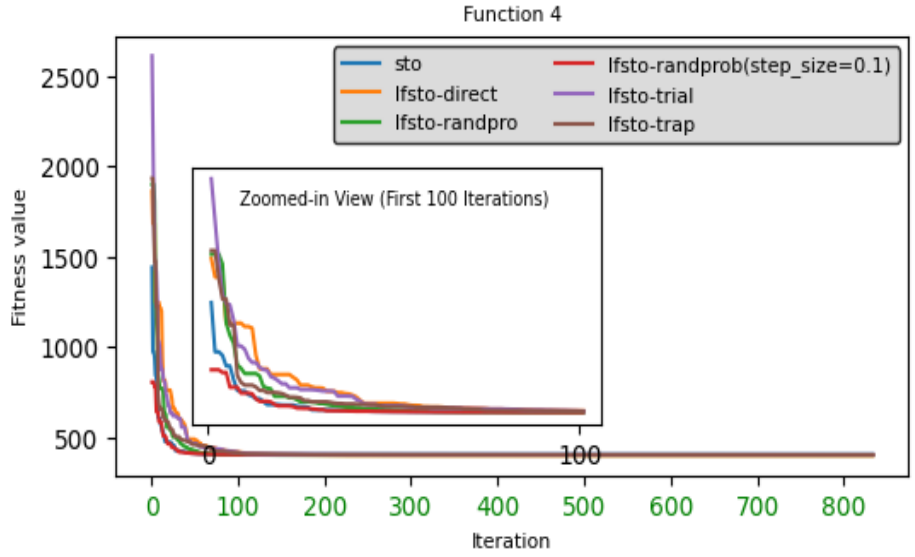


Figure 7.3. Convergence graph of LF integration strategies on CEC-2017 test functions F4 for dimension 10

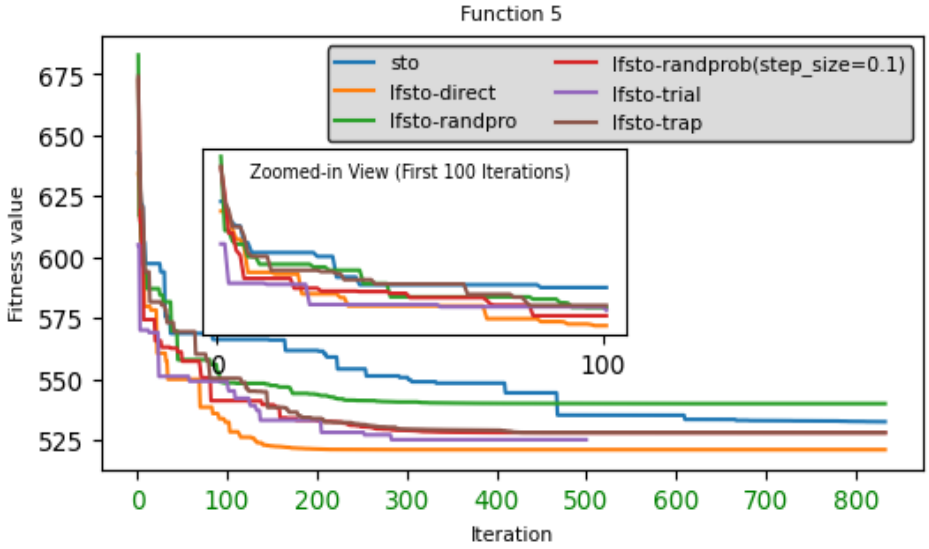


Figure 7.4. Convergence graph of LF integration strategies on CEC-2017 test functions F5 for dimension 10

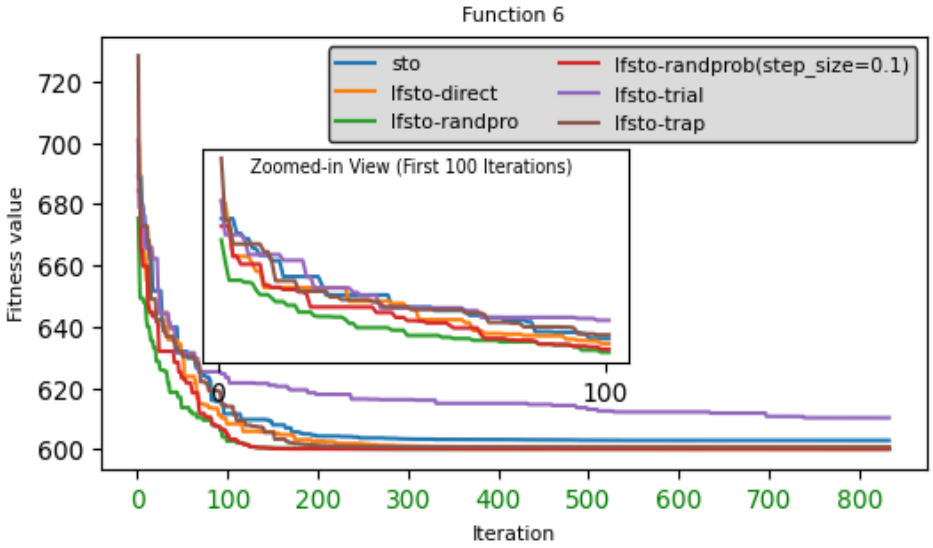


Figure 7.5. Convergence graph of LF integration strategies on CEC-2017 test functions F6 for dimension 10

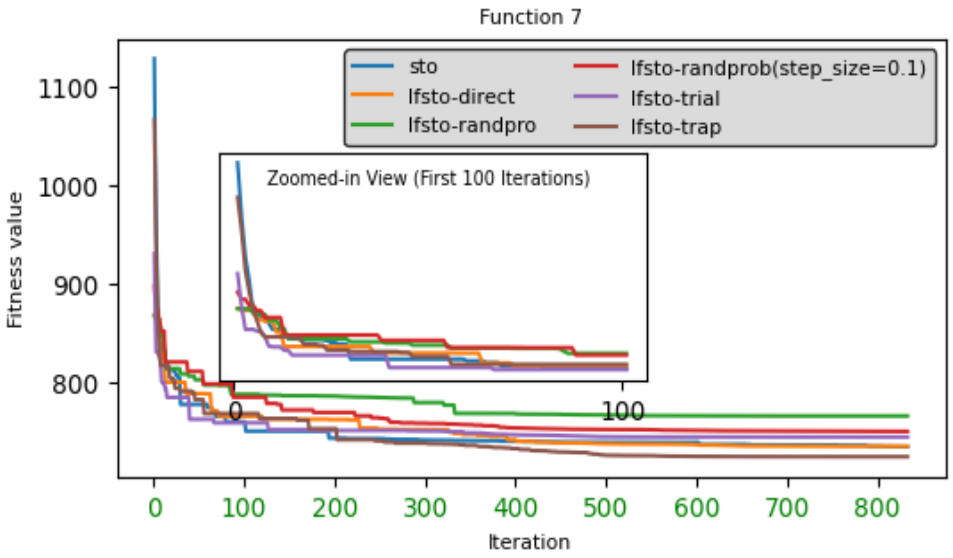


Figure 7.6. Convergence graph of LF integration strategies on CEC-2017 test functions F7 for dimension 10

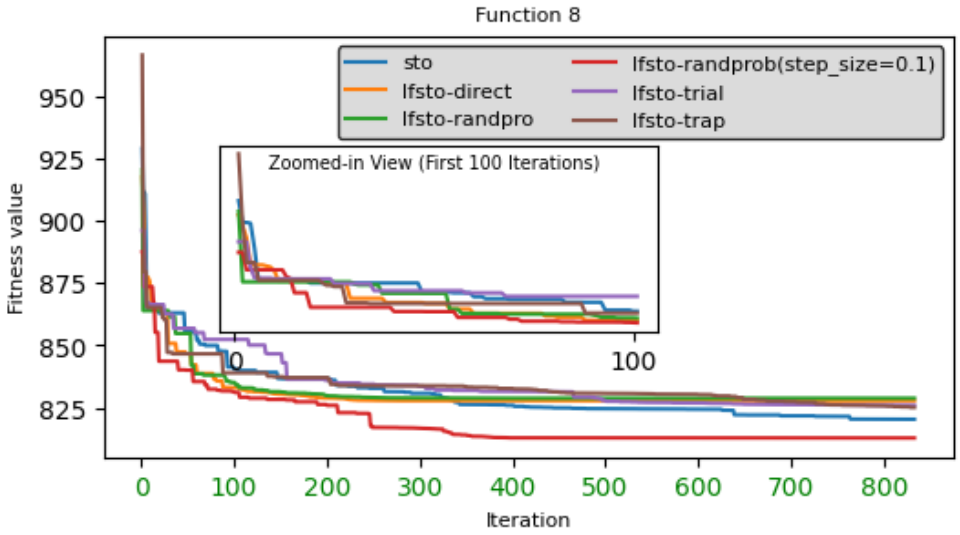


Figure 7.7. Convergence graph of LF integration strategies on CEC-2017 test functions F8 for dimension 10

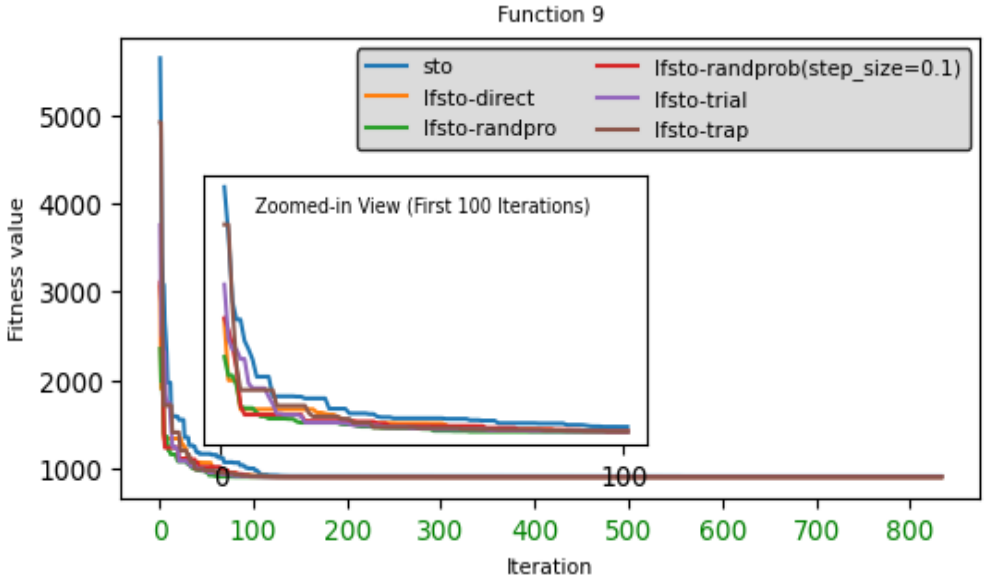


Figure 7.8. Convergence graph of LF integration strategies on CEC-2017 test functions F9 for dimension 10

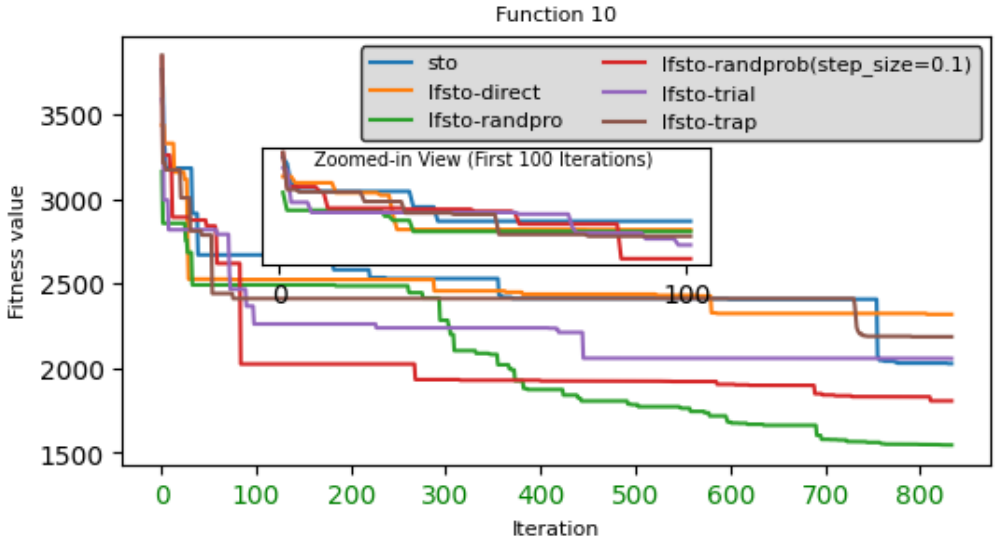


Figure 7.9. Convergence graph of LF integration strategies on CEC-2017 test functions F10 for dimension 10

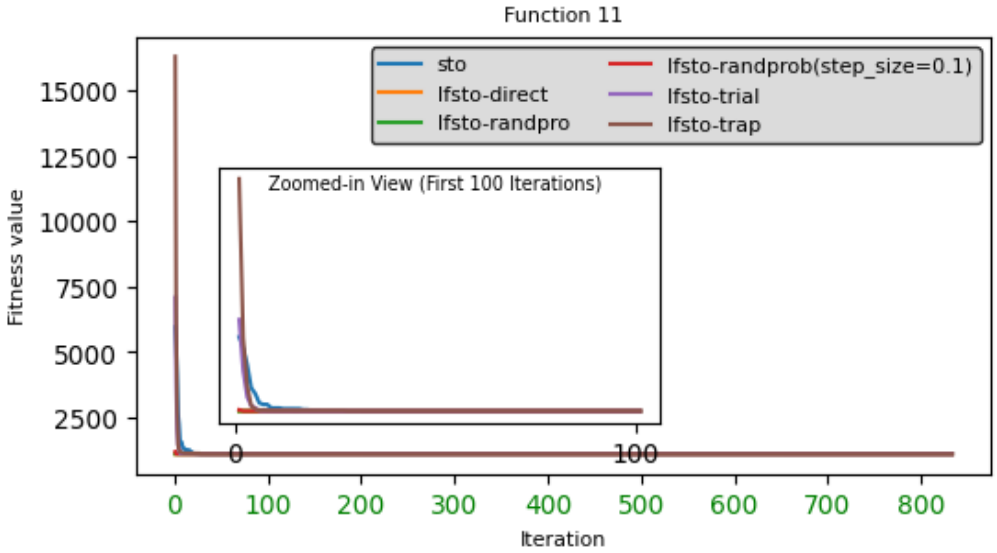


Figure 7.10. Convergence graph of LF integration strategies on CEC-2017 test functions F11 for dimension 10

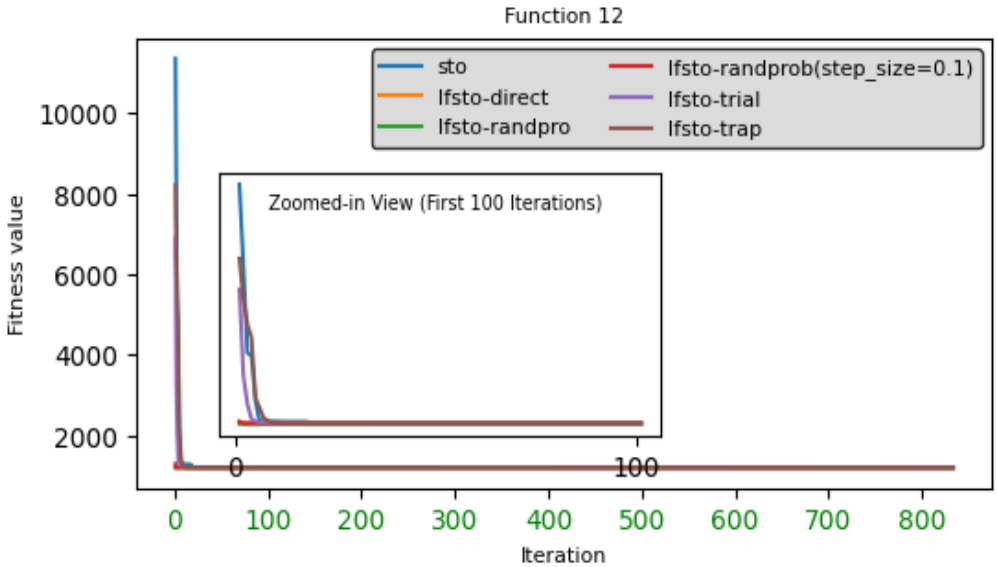


Figure 7.11. Convergence graph of LF integration strategies on CEC-2017 test functions F12 for dimension 10

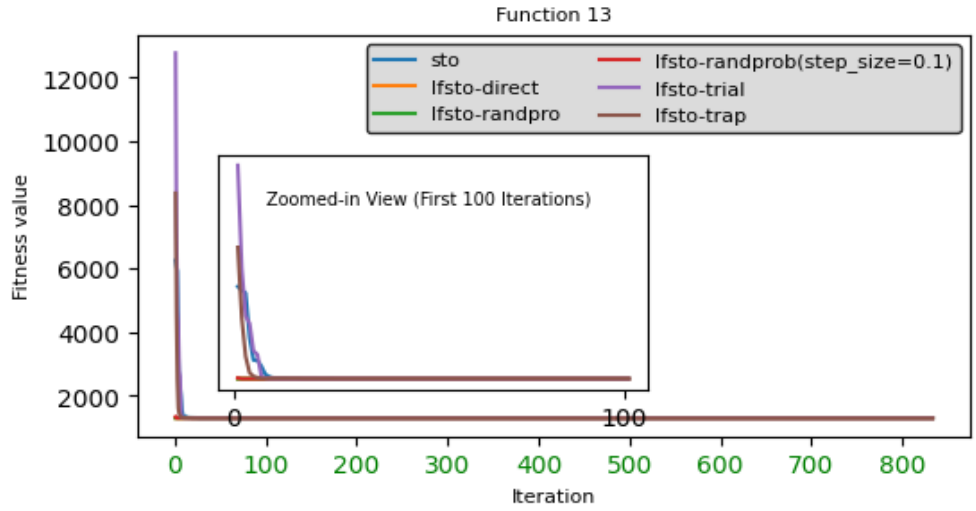


Figure 7.12. Convergence graph of LF integration strategies on CEC-2017 test functions F13 for dimension 10

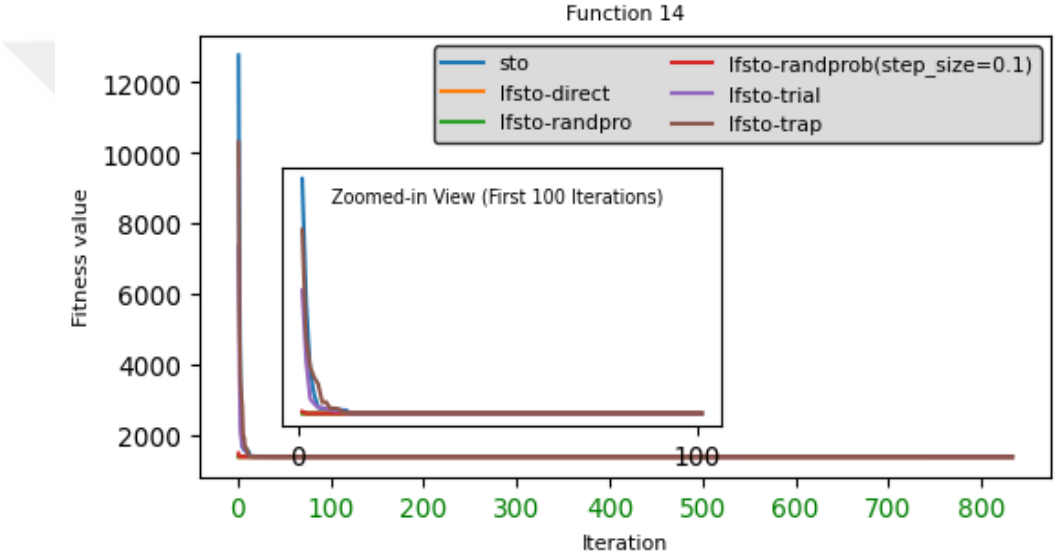


Figure 7.13. Convergence graph of LF integration strategies on CEC-2017 test functions F14 for dimension 10

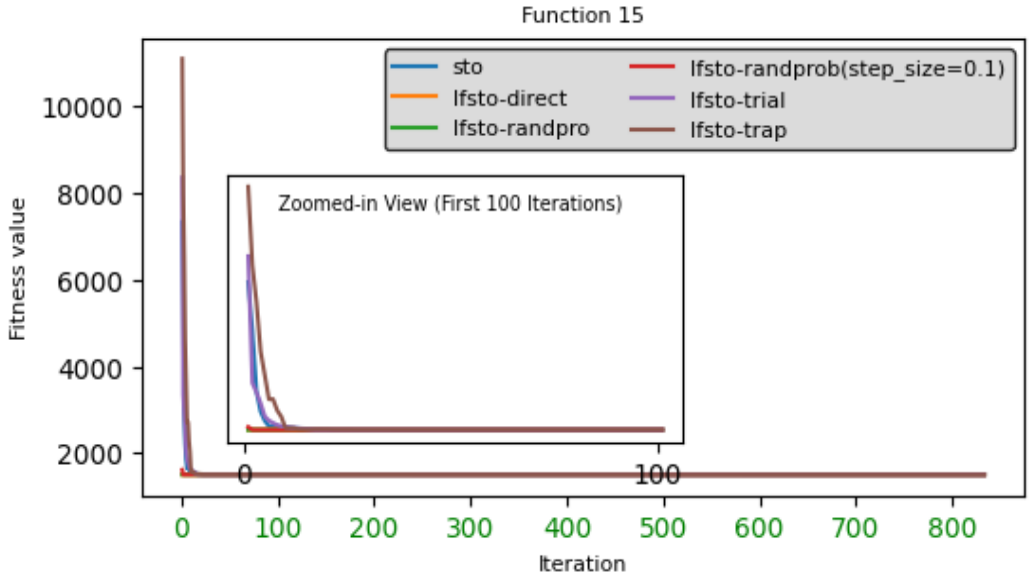


Figure 7.14. Convergence graph of LF integration strategies on CEC-2017 test functions F15 for dimension 10

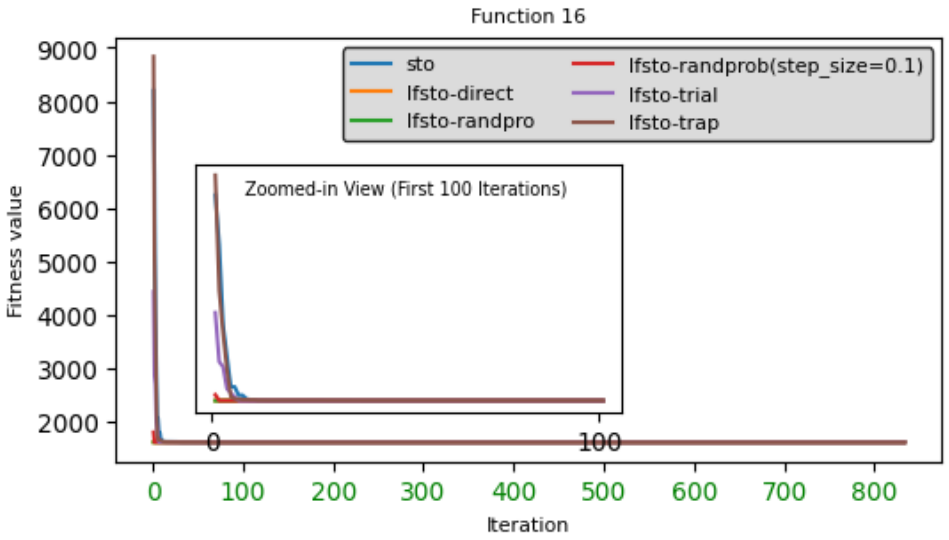


Figure 7.15. Convergence graph of LF integration strategies on CEC-2017 test functions F16 for dimension 10

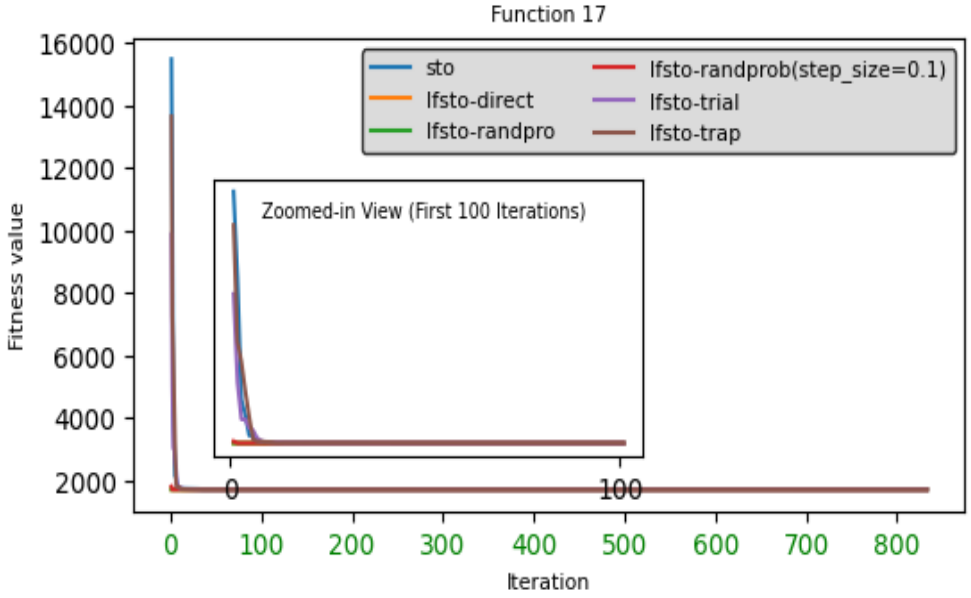


Figure 7.16. Convergence graph of LF integration strategies on CEC-2017 test functions F17 for dimension 10

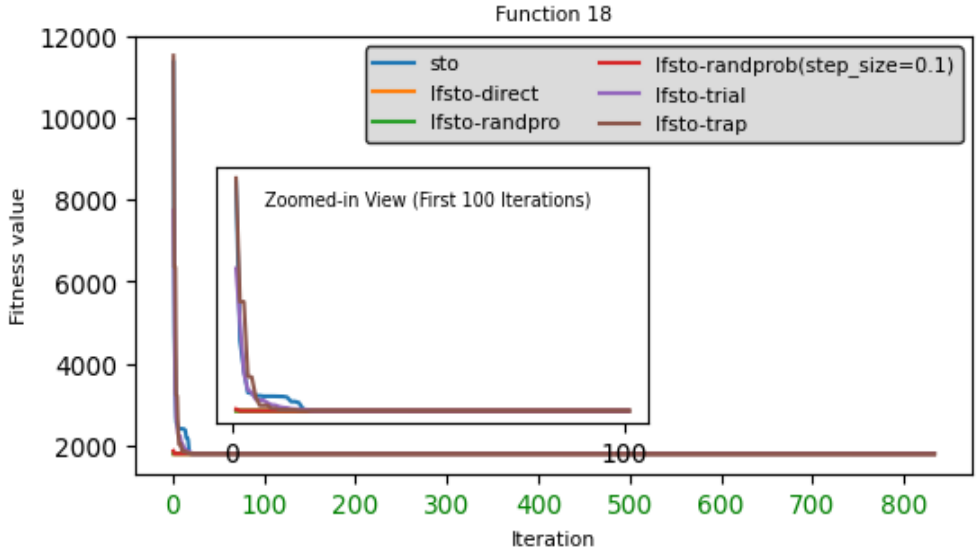


Figure 7.17. Convergence graph of LF integration strategies on CEC-2017 test functions F18 for dimension 10

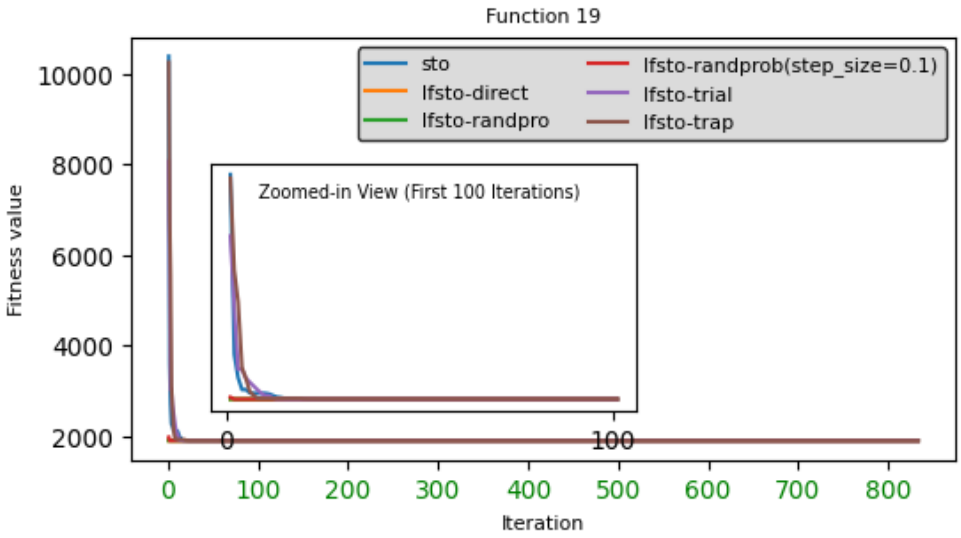


Figure 7.18. Convergence graph of LF integration strategies on CEC-2017 test functions F19 for dimension 10

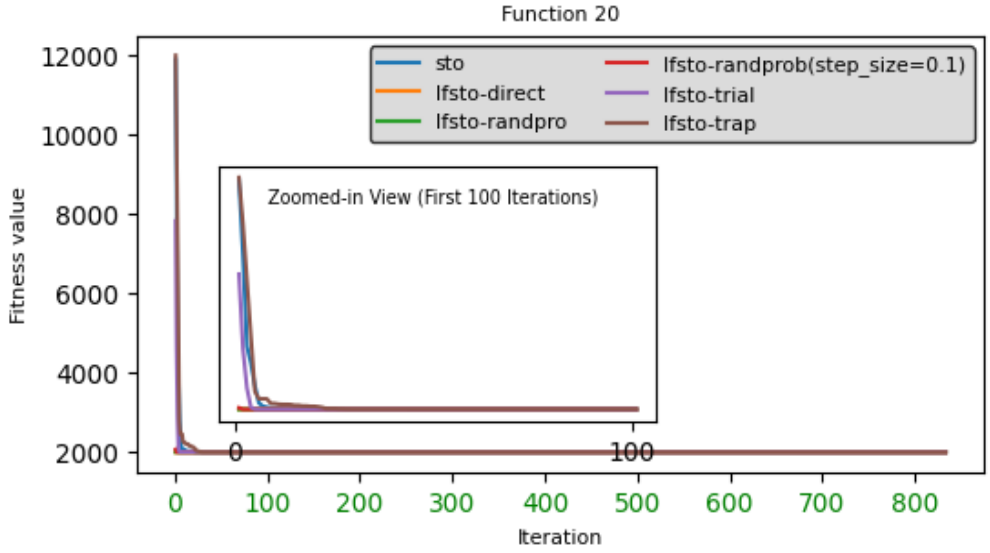


Figure 7.19. Convergence graph of LF integration strategies on CEC-2017 test functions F20 for dimension 10

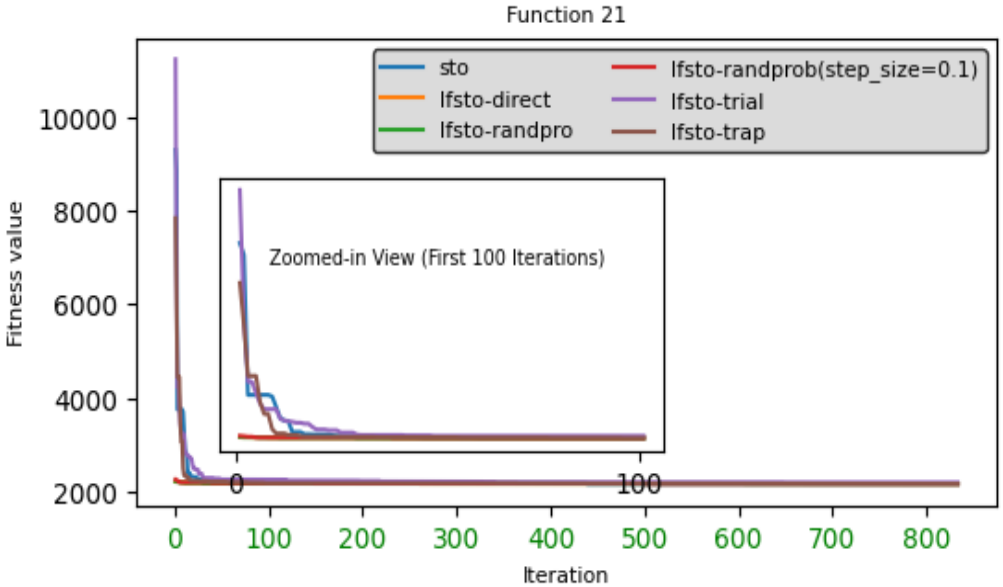


Figure 7.20. Convergence graph of LF integration strategies on CEC-2017 test functions F21 for dimension 10

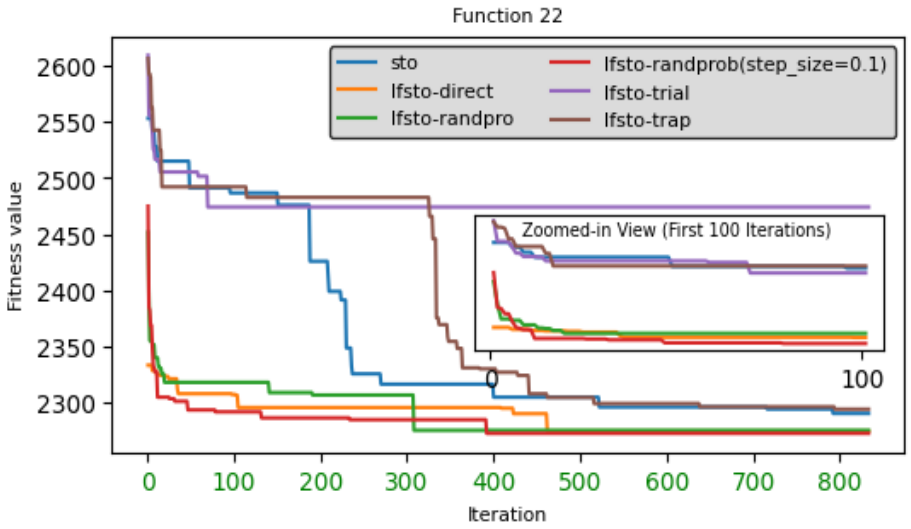


Figure 7.21. Convergence graph of LF integration strategies on CEC-2017 test functions F22 for dimension 10

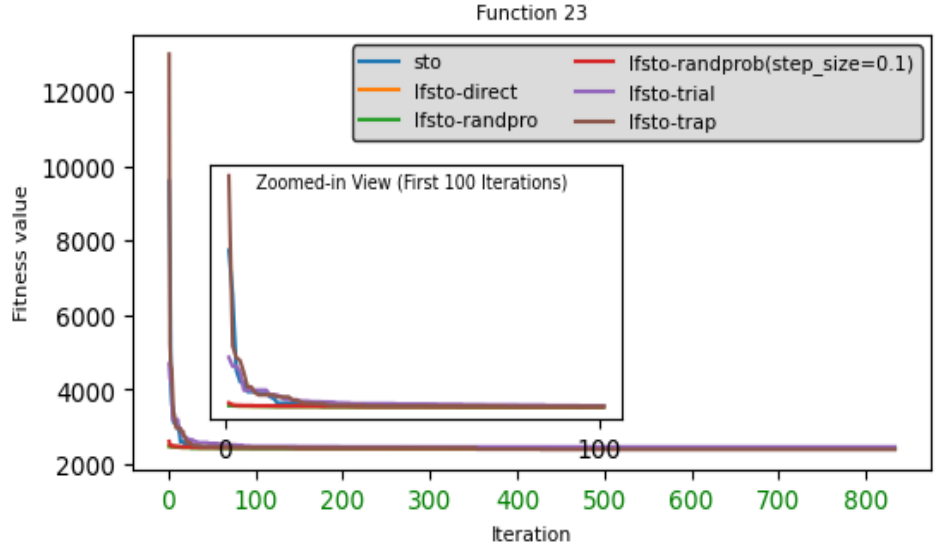


Figure 7.22. Convergence graph of LF integration strategies on CEC-2017 test functions F23 for dimension 10

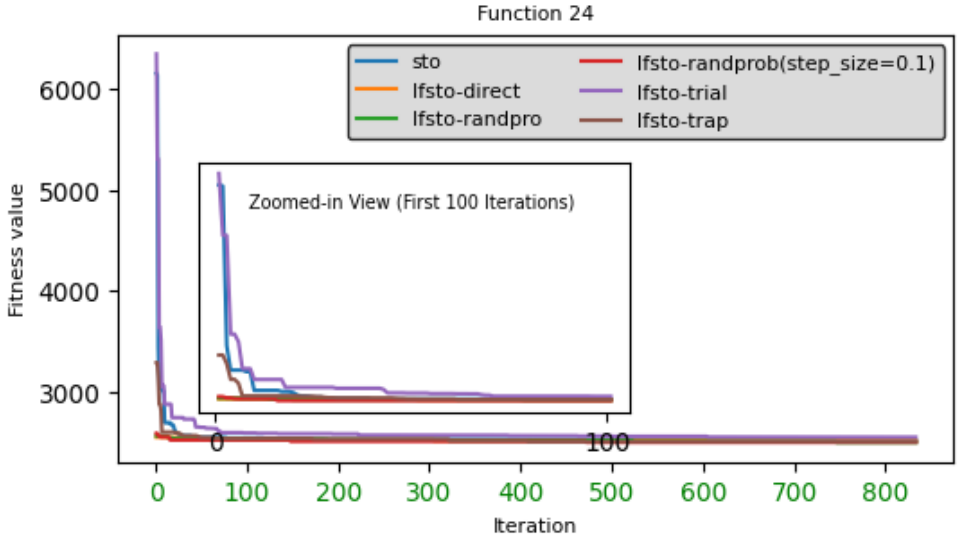


Figure 7.23. Convergence graph of LF integration strategies on CEC-2017 test functions F24 for dimension 10

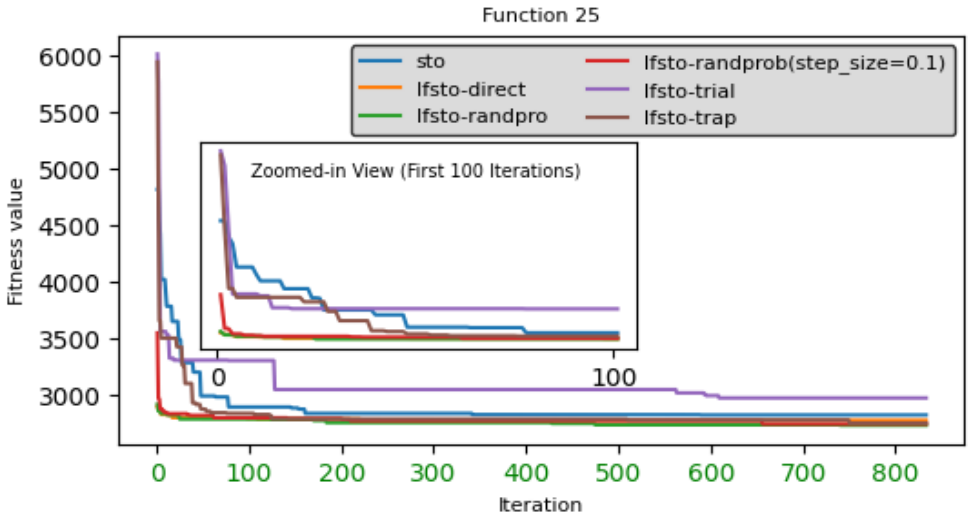


Figure 7.24. Convergence graph of LF integration strategies on CEC-2017 test functions F25 for dimension 10

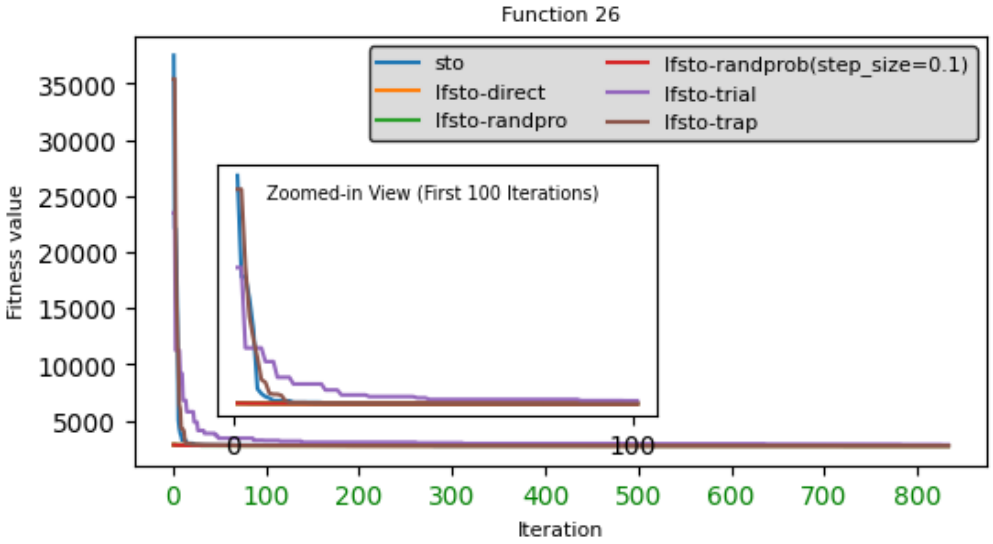


Figure 7.25. Convergence graph of LF integration strategies on CEC-2017 test functions F26 for dimension 10

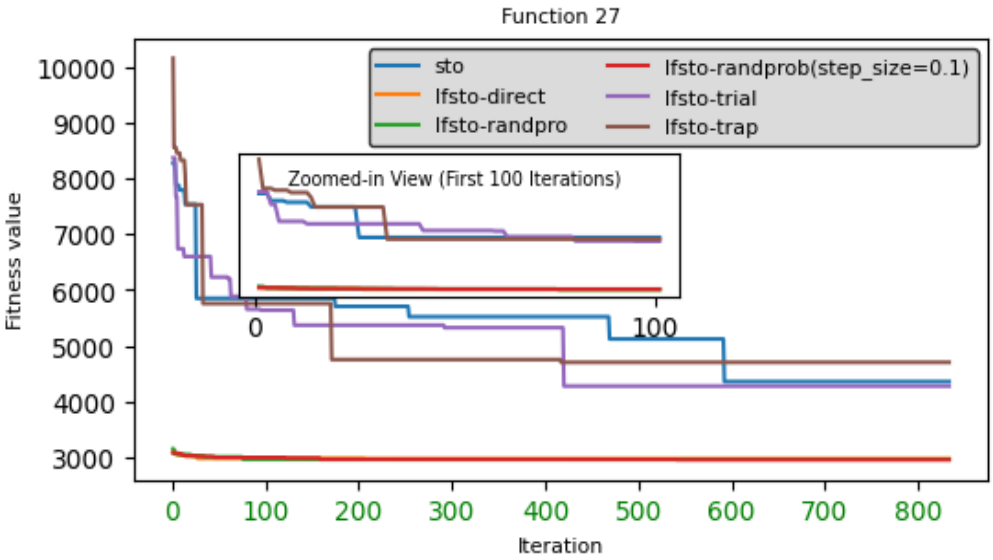


Figure 7.26. Convergence graph of LF integration strategies on CEC-2017 test functions F27 for dimension 10

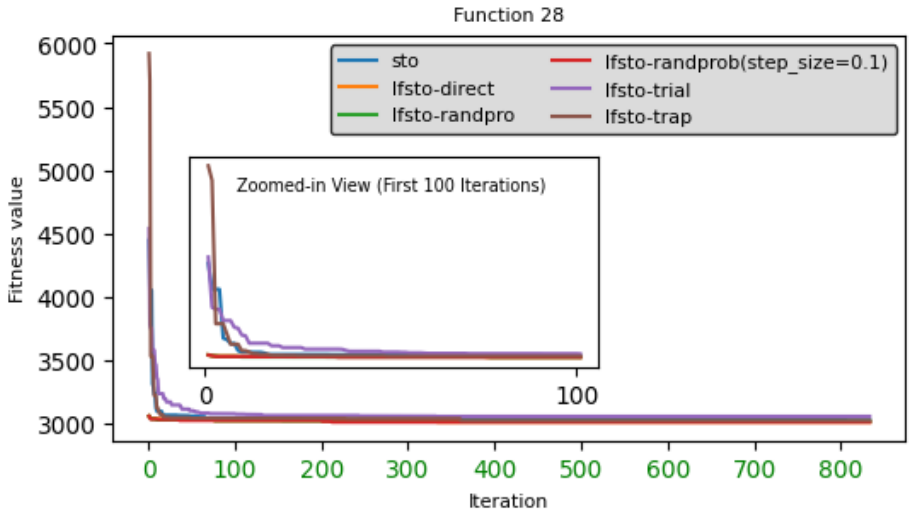


Figure 7.27. Convergence graph of LF integration strategies on CEC-2017 test functions F28 for dimension 10

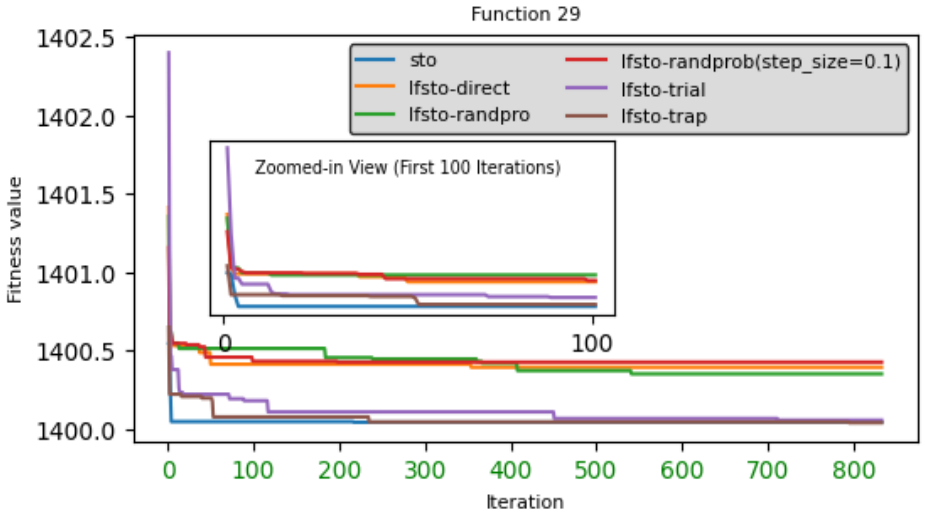


Figure 7.28. Convergence graph of LF integration strategies on CEC-2017 test functions F29 for dimension 10

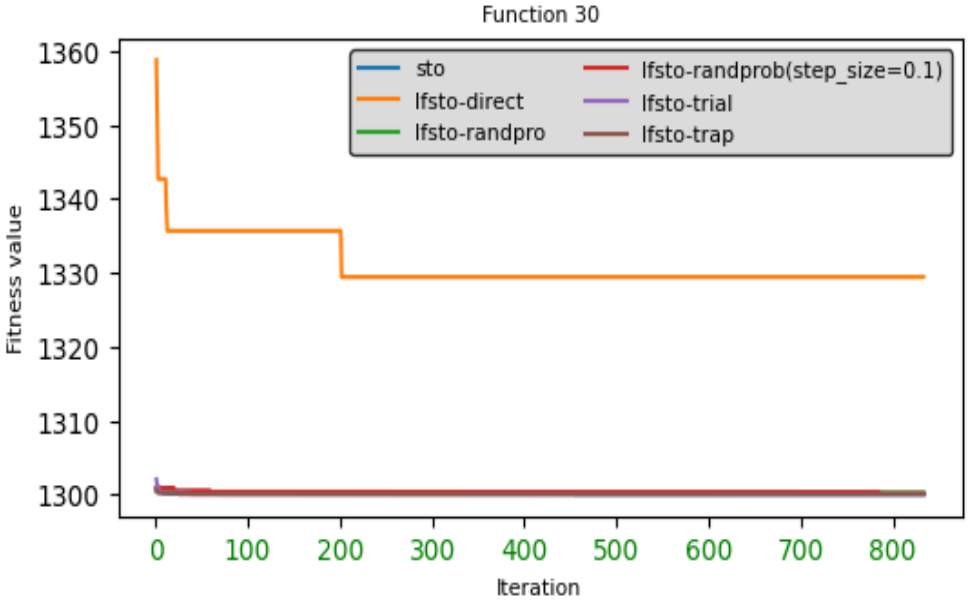


Figure 7.29. Convergence graph of LF integration strategies on CEC-2017 test functions F30 for dimension 10

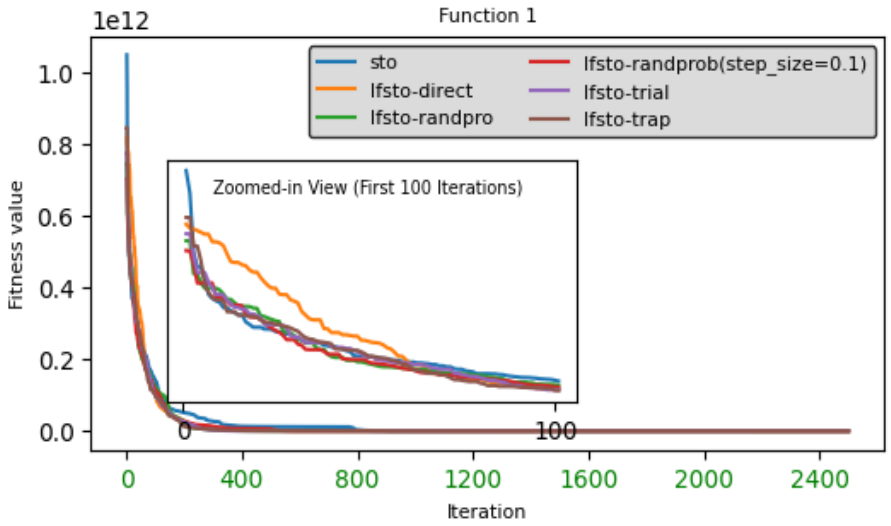


Figure 7.30. Convergence graph of LF integration strategies on CEC-2017 test functions F1 for dimension 30

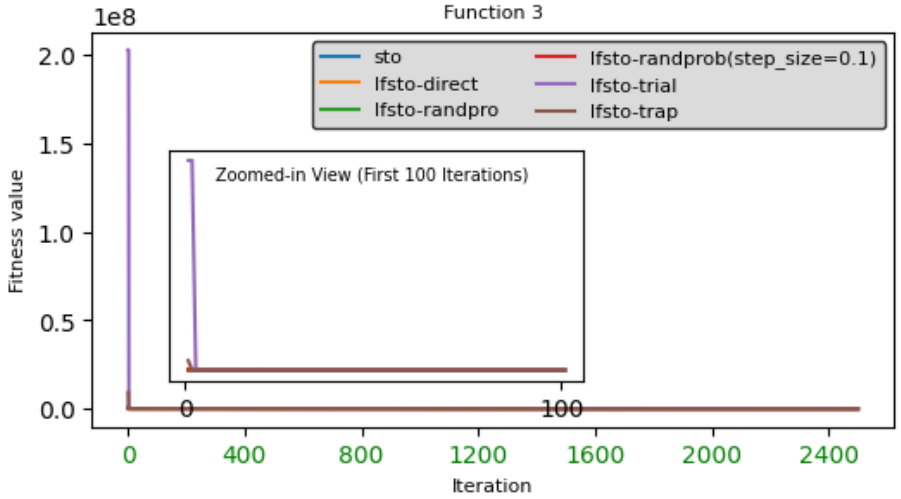


Figure 7.31. Convergence graph of LF integration strategies on CEC-2017 test functions F3 for dimension 30

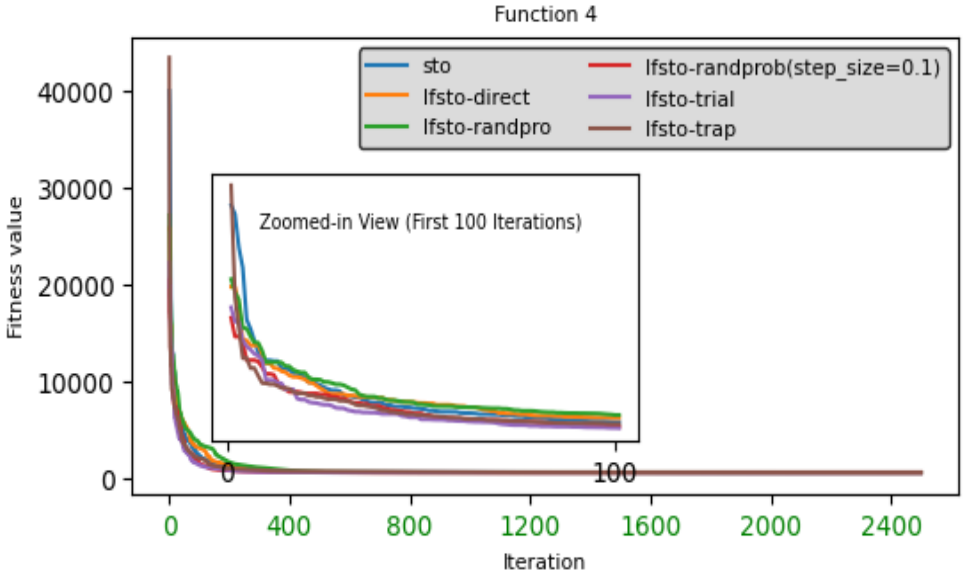


Figure 7.32. Convergence graph of LF integration strategies on CEC-2017 test functions F4 for dimension 30

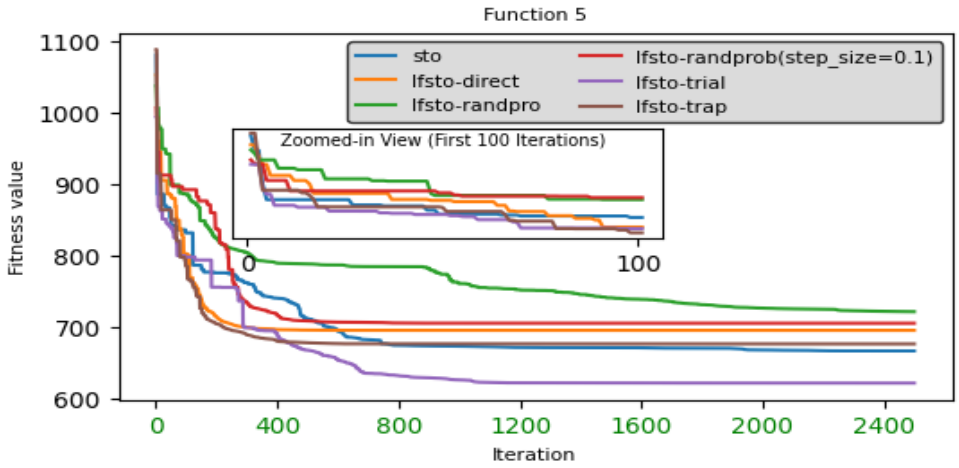


Figure 7.33. Convergence graph of LF integration strategies on CEC-2017 test functions F5 for dimension 30

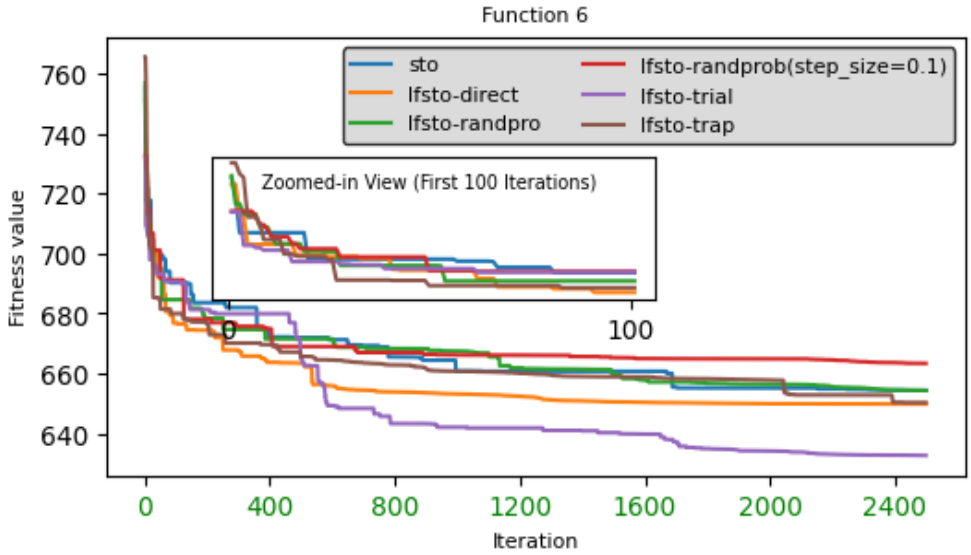


Figure 7.34. Convergence graph of LF integration strategies on CEC-2017 test functions F6 for dimension 30

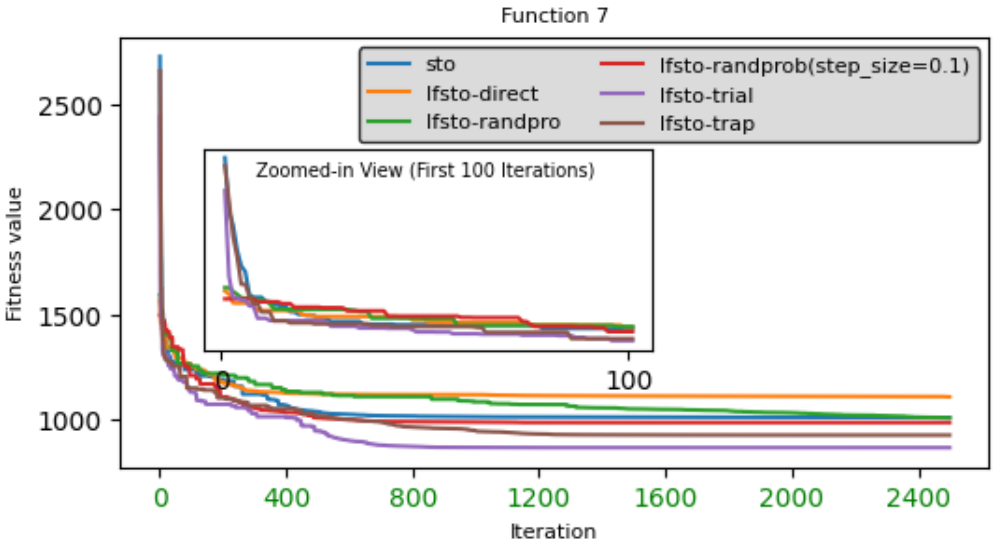


Figure 7.35. Convergence graph of LF integration strategies on CEC-2017 test functions F7 for dimension 30

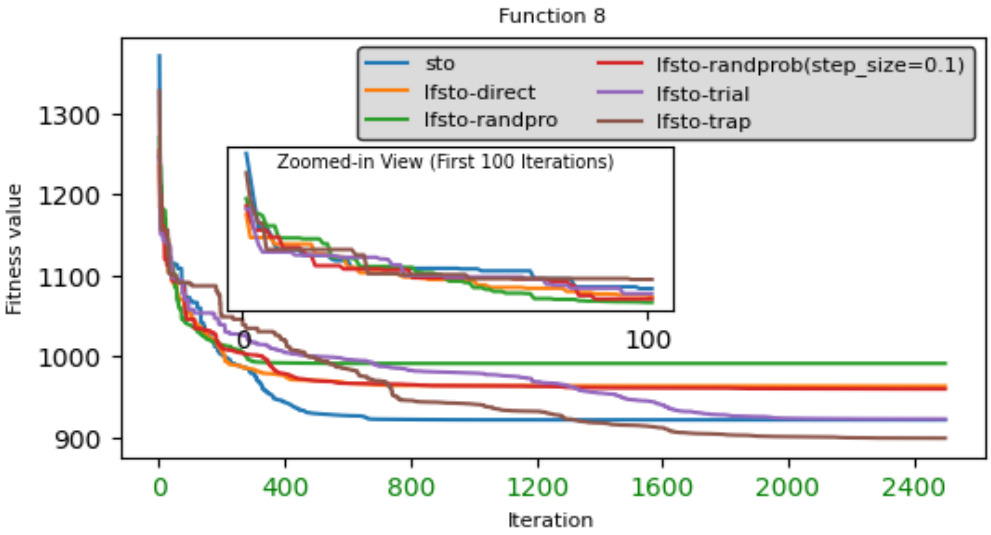


Figure 7.36. Convergence graph of LF integration strategies on CEC-2017 test functions F8 for dimension 30

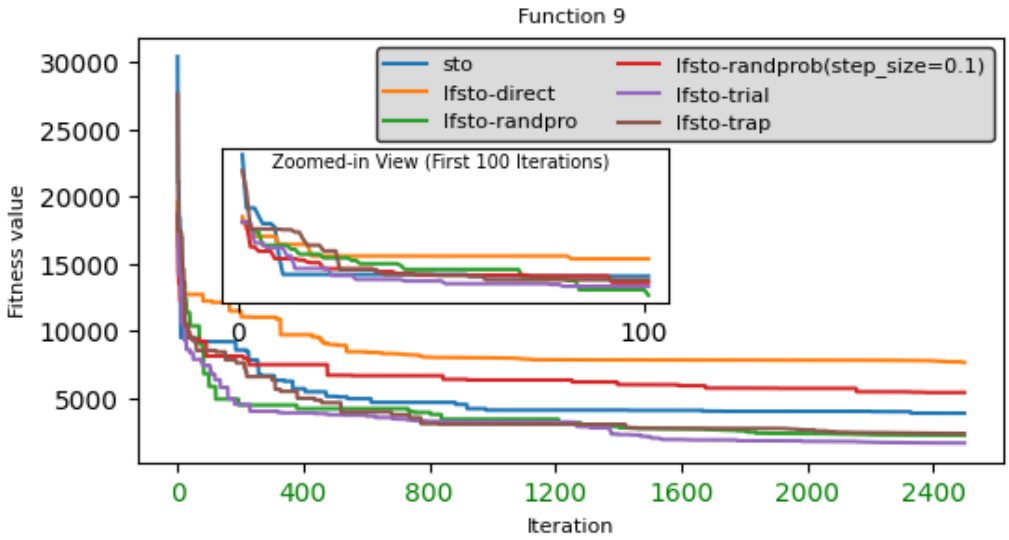


Figure 7.37. Convergence graph of LF integration strategies on CEC-2017 test functions F9 for dimension 30

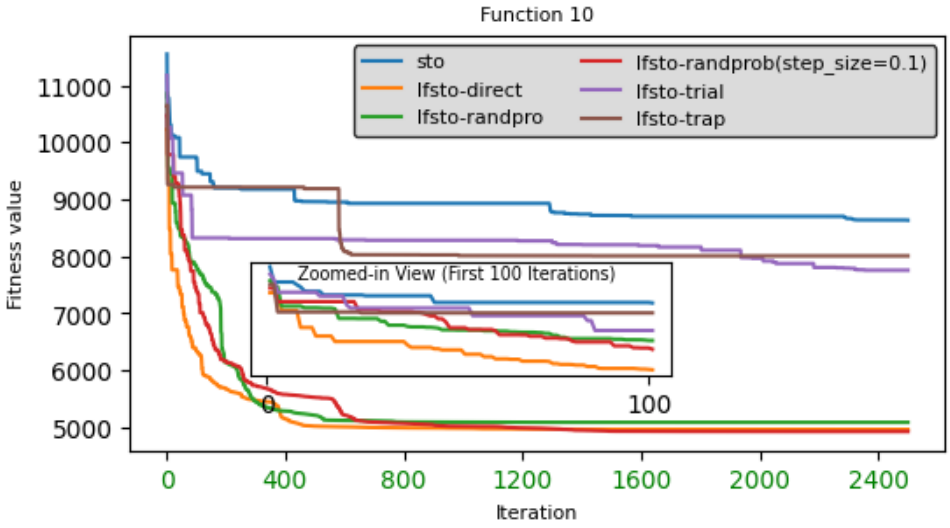


Figure 7.38. Convergence graph of LF integration strategies on CEC-2017 test functions F10 for dimension 30

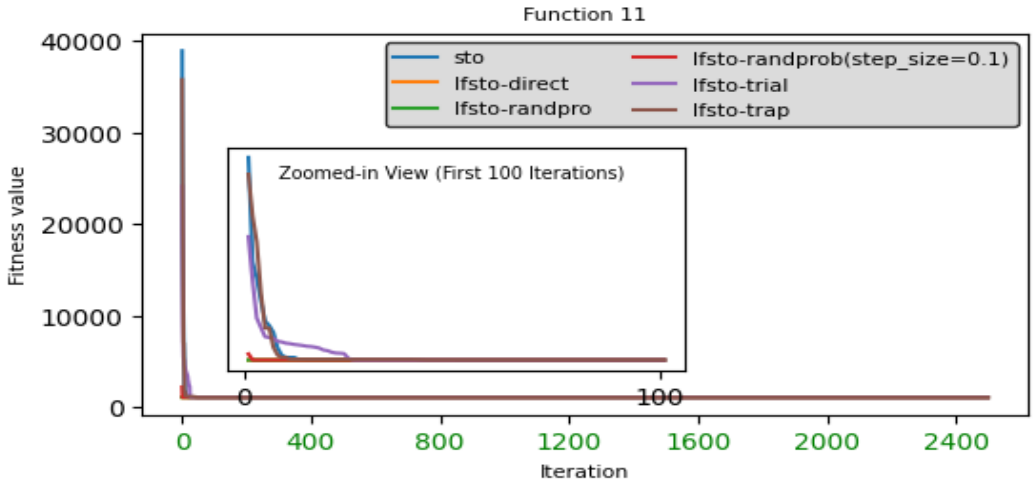


Figure 7.39. Convergence graph of LF integration strategies on CEC-2017 test functions F11 for dimension 30

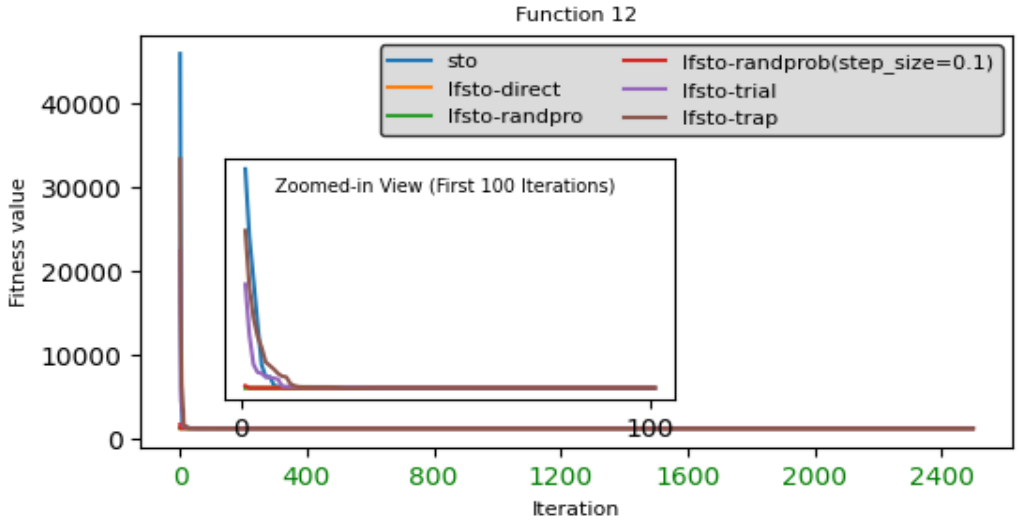


Figure 7.40. Convergence graph of LF integration strategies on CEC-2017 test functions F12 for dimension 30

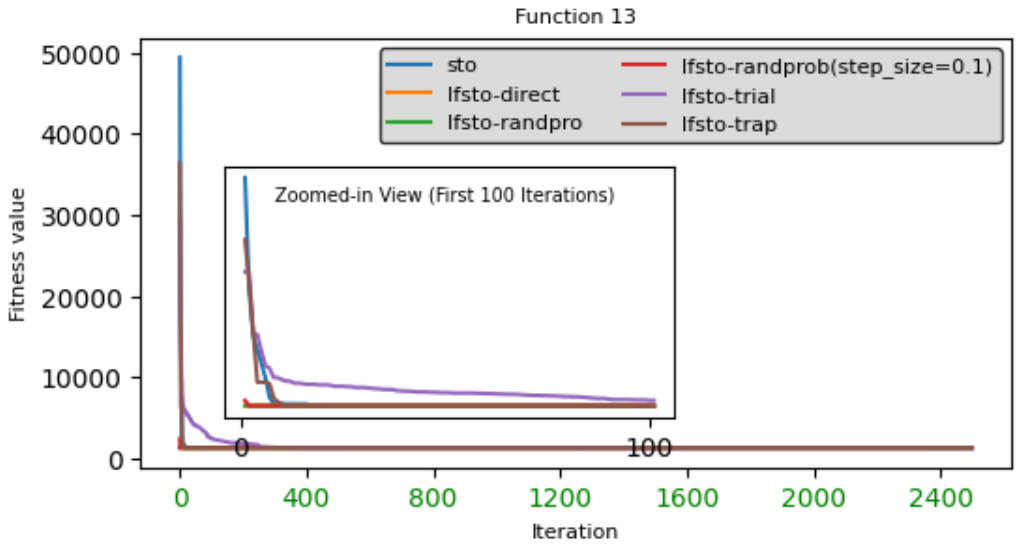


Figure 7.41. Convergence graph of LF integration strategies on CEC-2017 test functions F13 for dimension 30

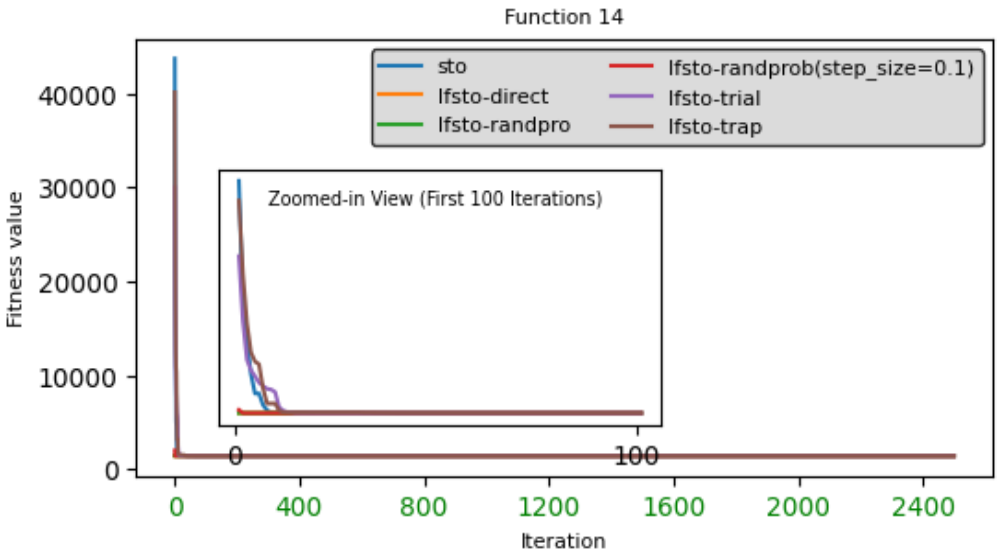


Figure 7.42. Convergence graph of LF integration strategies on CEC-2017 test functions F14 for dimension 30

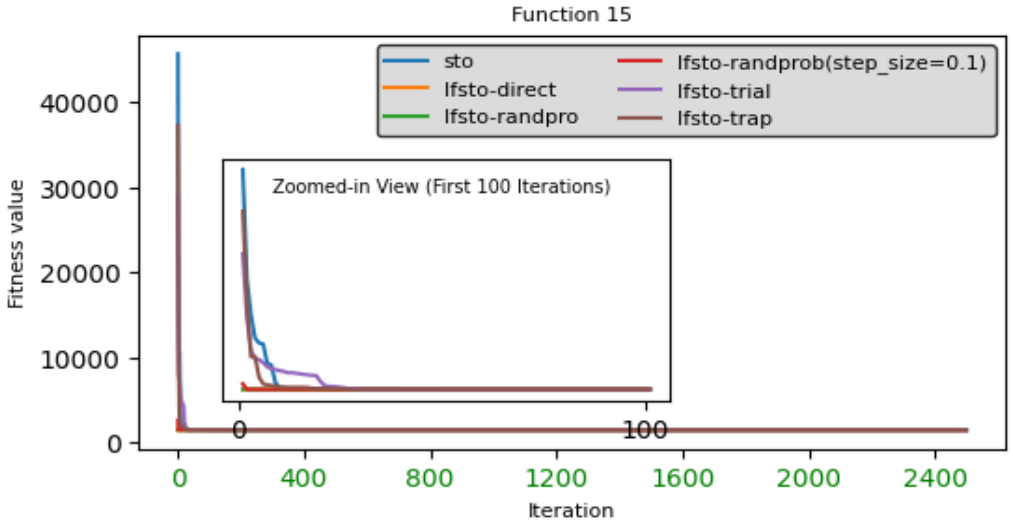


Figure 7.43. Convergence graph of LF integration strategies on CEC-2017 test functions F15 for dimension 30

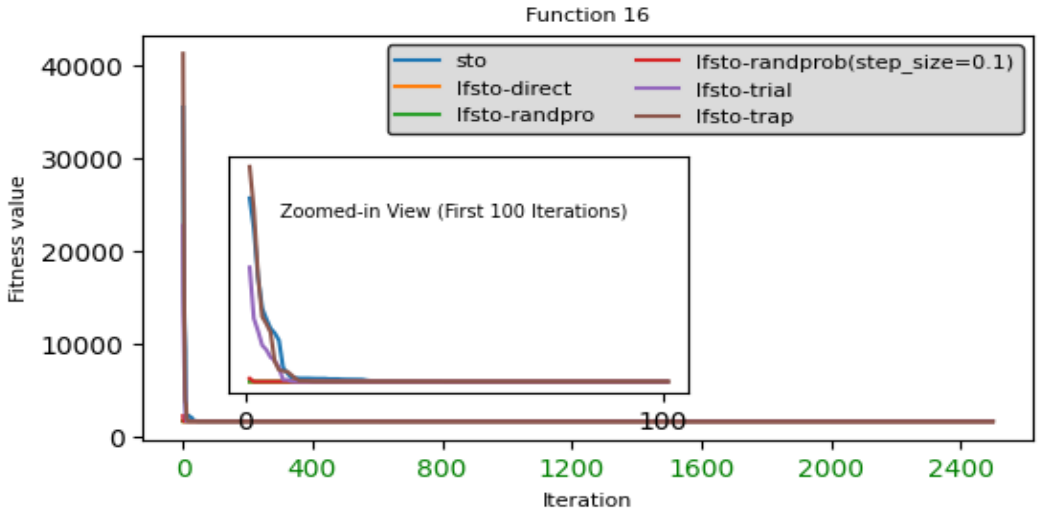


Figure 7.44. Convergence graph of LF integration strategies on CEC-2017 test functions F16 for dimension 30

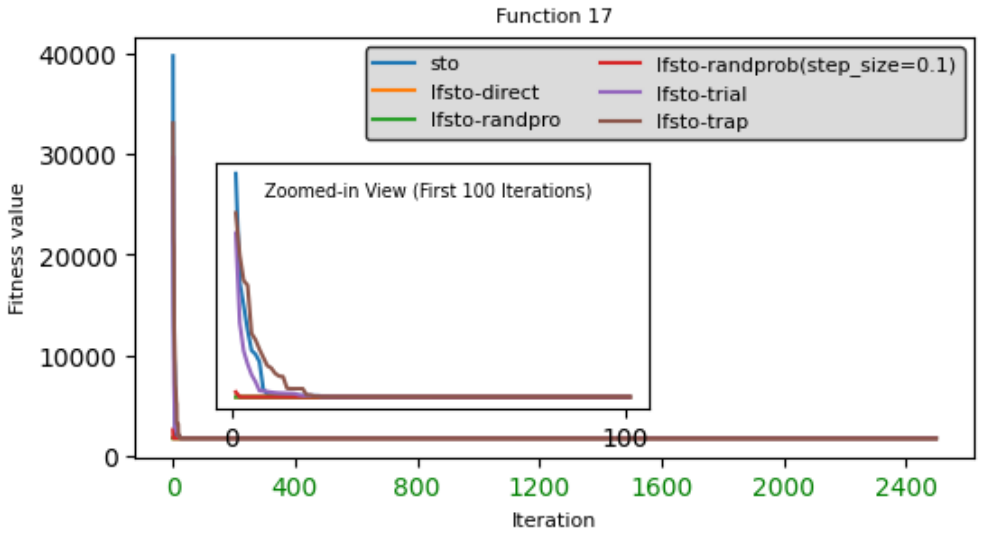


Figure 7.45. Convergence graph of LF integration strategies on CEC-2017 test functions F17 for dimension 30

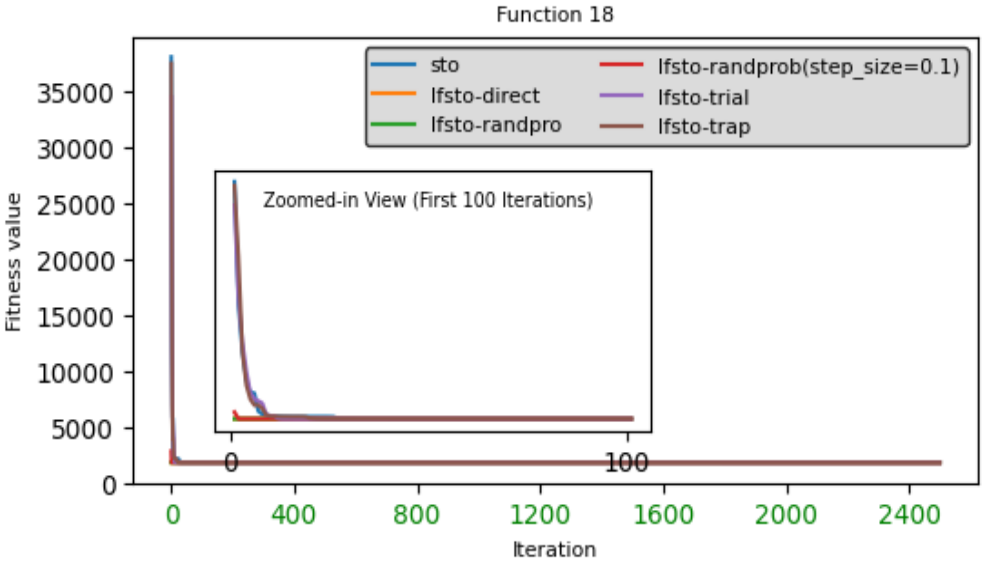


Figure 7.46. Convergence graph of LF integration strategies on CEC-2017 test functions F18 for dimension 30

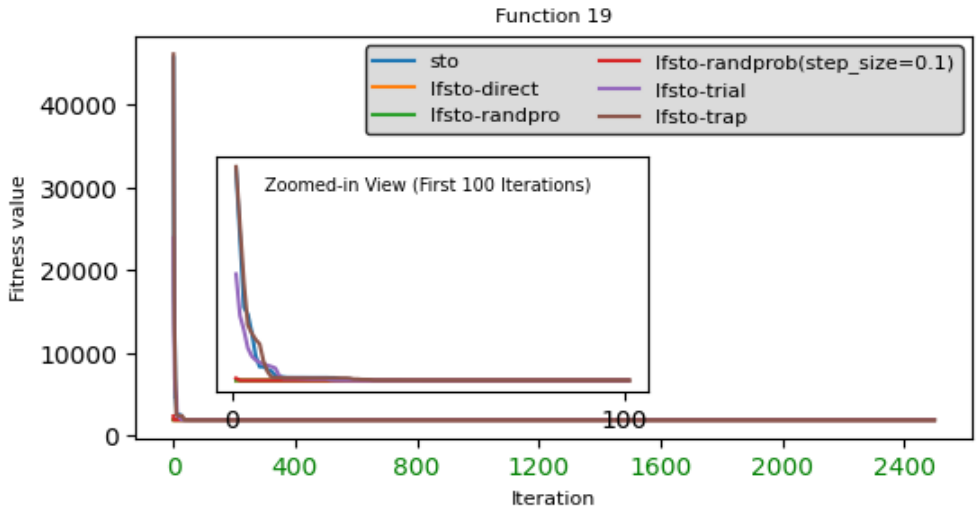


Figure 7.47. Convergence graph of LF integration strategies on CEC-2017 test functions F19 for dimension 30

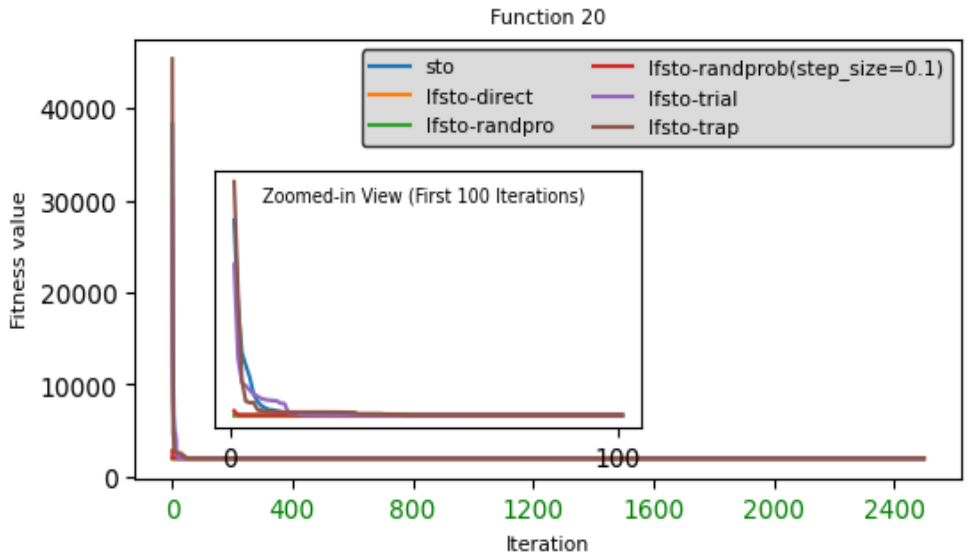


Figure 7.48. Convergence graph of LF integration strategies on CEC-2017 test functions F20 for dimension 30

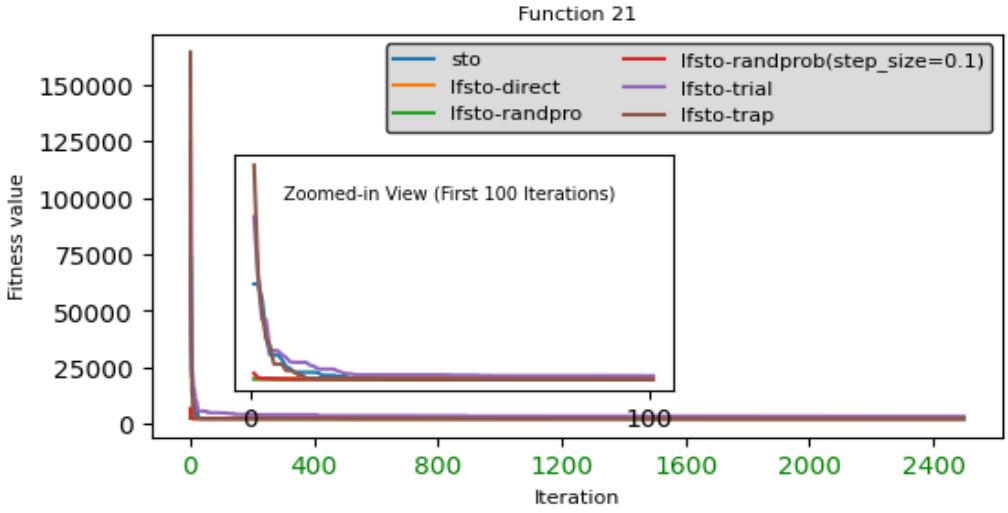


Figure 7.49. Convergence graph of LF integration strategies on CEC-2017 test functions F21 for dimension 30

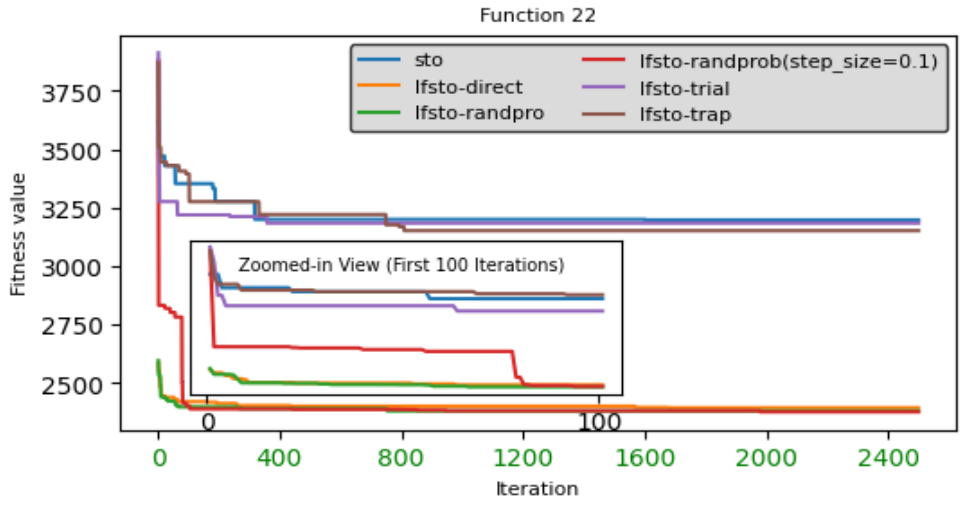


Figure 7.50. Convergence graph of LF integration strategies on CEC-2017 test functions F22 for dimension 30

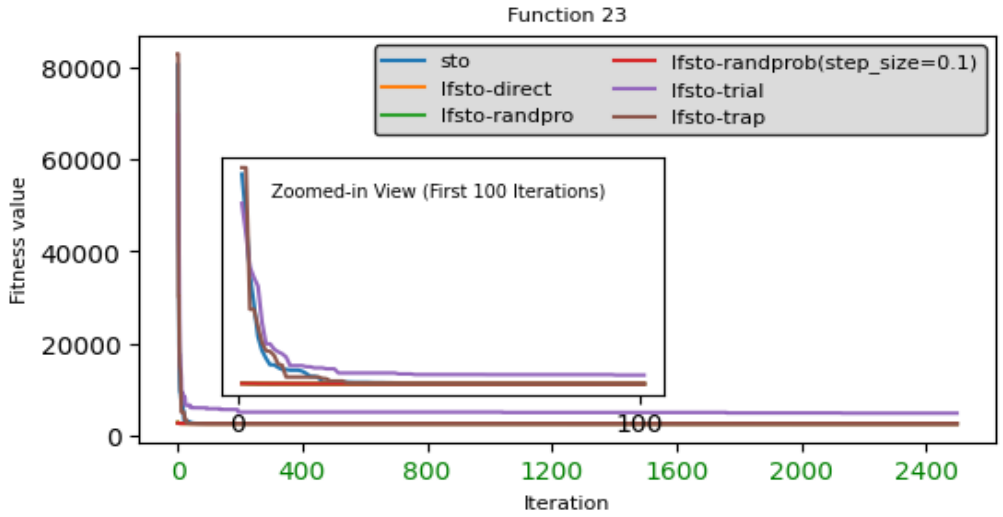


Figure 7.51. Convergence graph of LF integration strategies on CEC-2017 test functions F23 for dimension 30

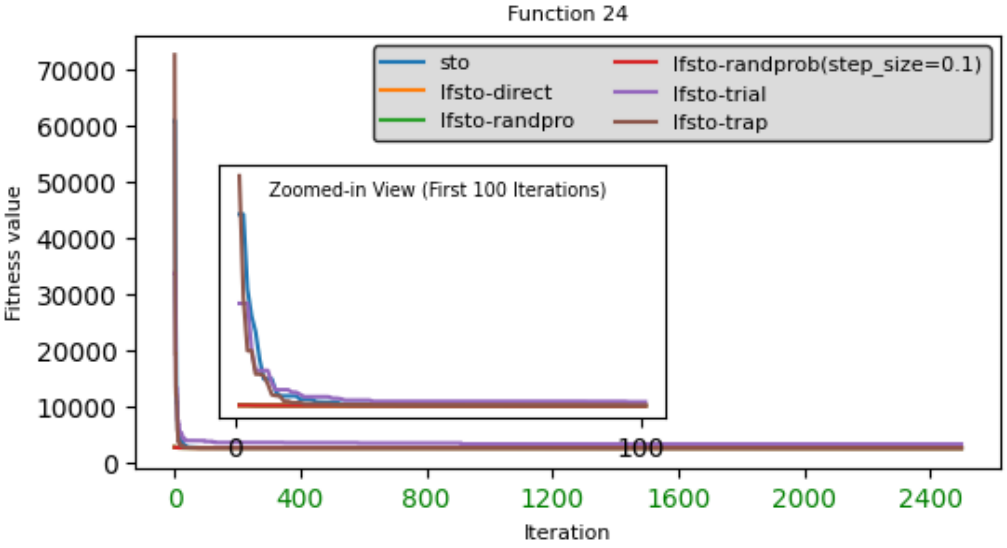


Figure 7.52. Convergence graph of LF integration strategies on CEC-2017 test functions F24 for dimension 30

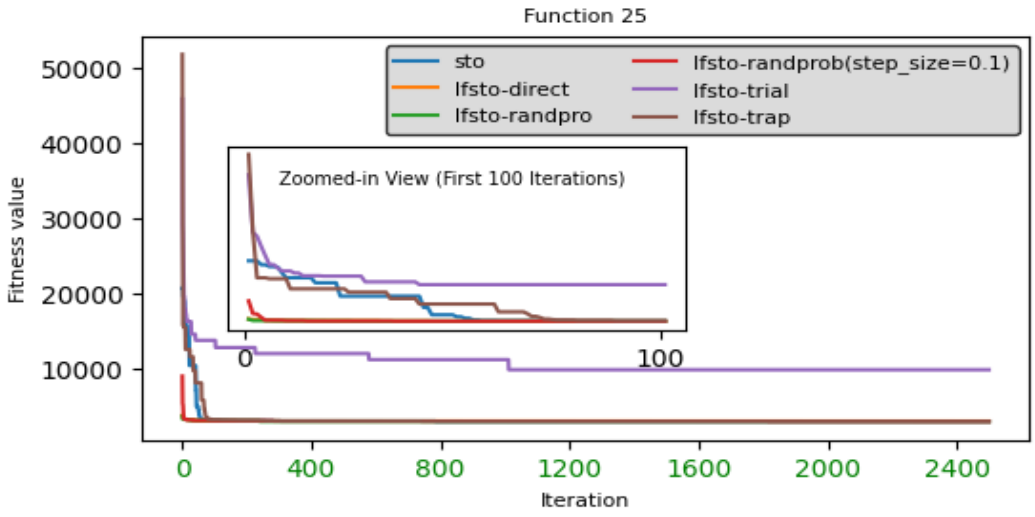


Figure 7.53. Convergence graph of LF integration strategies on CEC-2017 test functions F25 for dimension 30

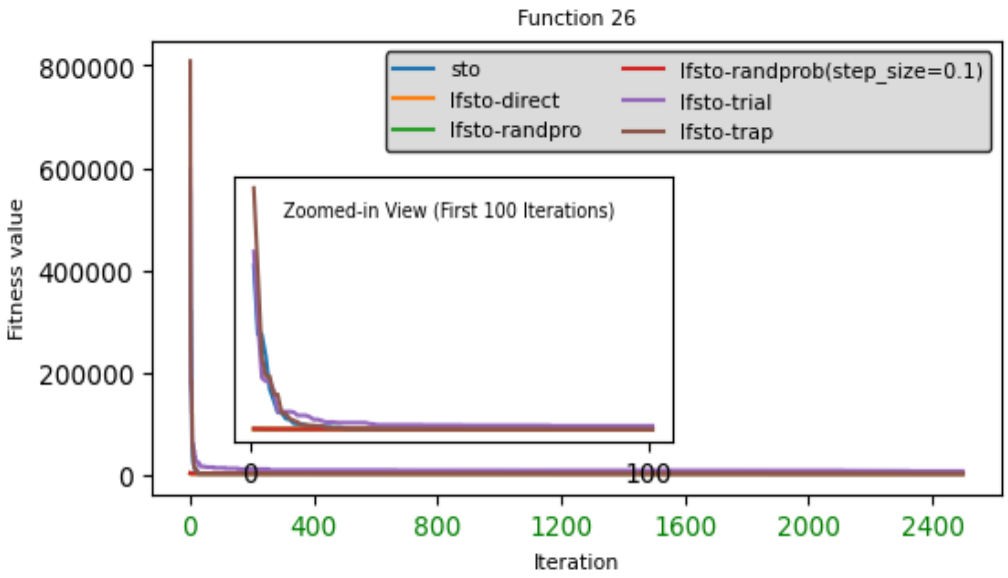


Figure 7.54. Convergence graph of LF integration strategies on CEC-2017 test functions F26 for dimension 30

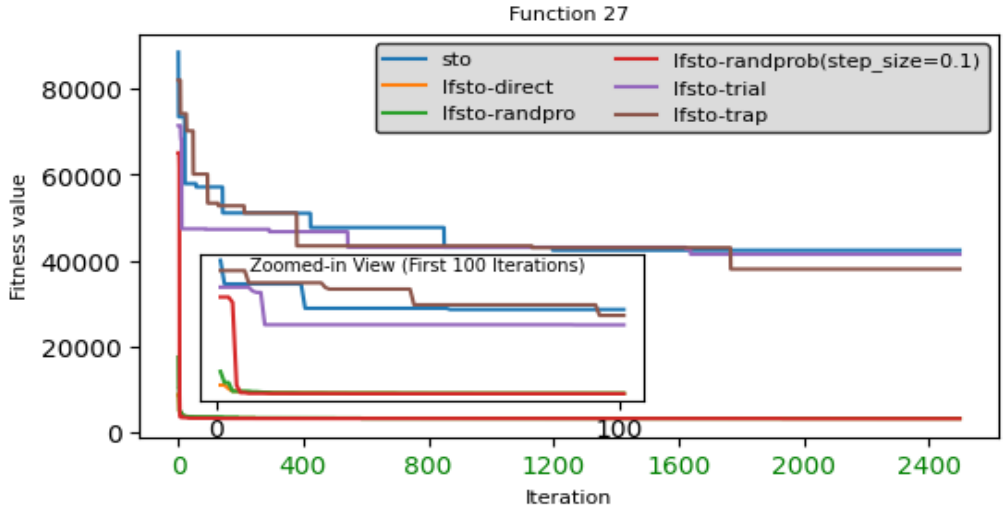


Figure 7.55. Convergence graph of LF integration strategies on CEC-2017 test functions F27 for dimension 30

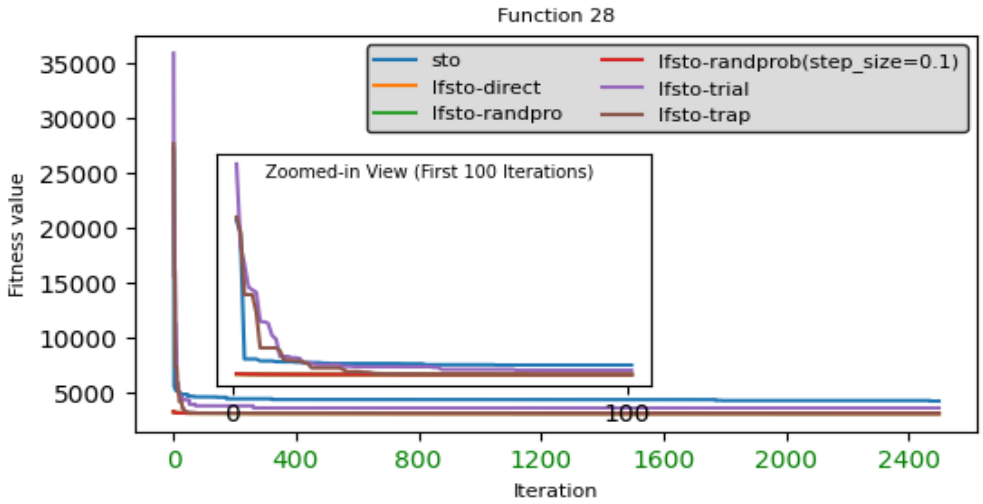


Figure 7.56. Convergence graph of LF integration strategies on CEC-2017 test functions F28 for dimension 30

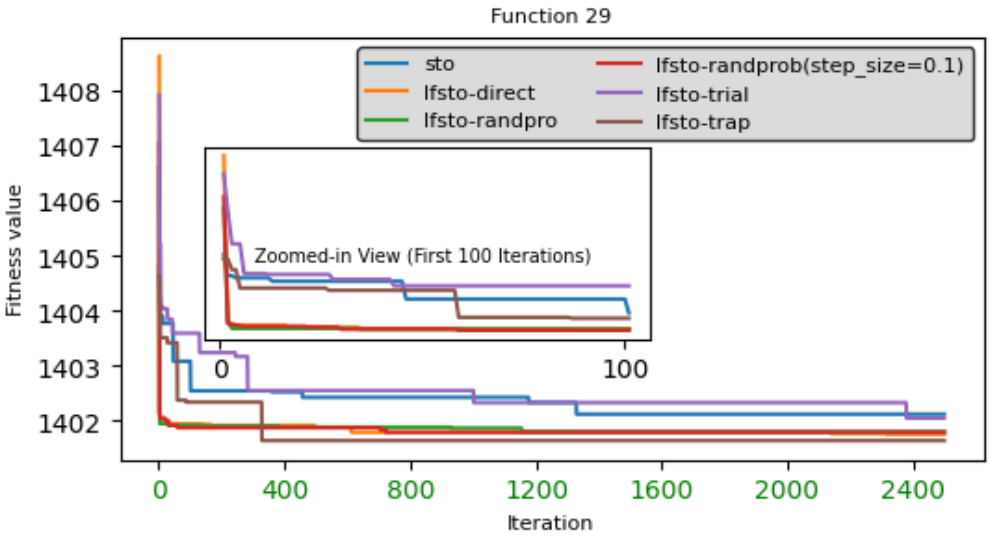


Figure 7.57. Convergence graph of LF integration strategies on CEC-2017 test functions F29 for dimension 30

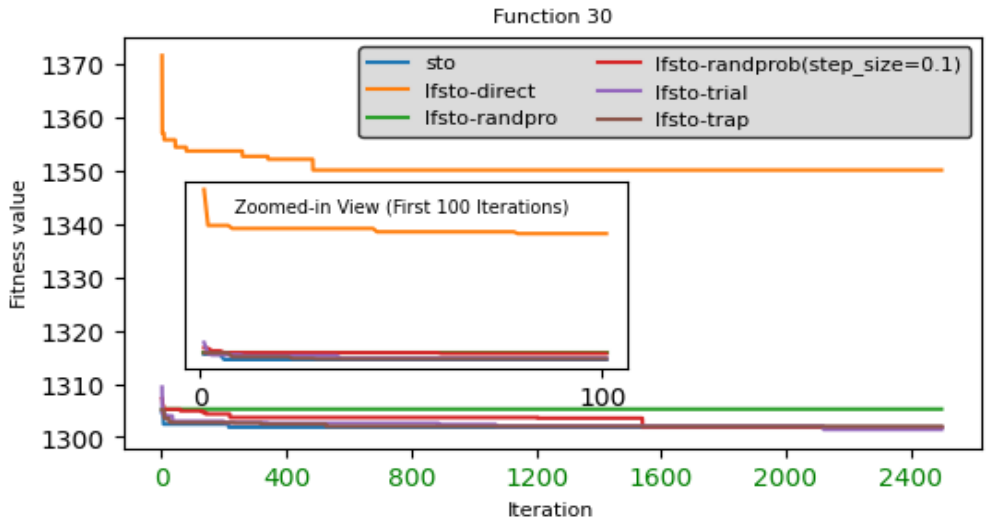


Figure 7.58. Convergence graph of LF integration strategies on CEC-2017 test functions F30 for dimension 30

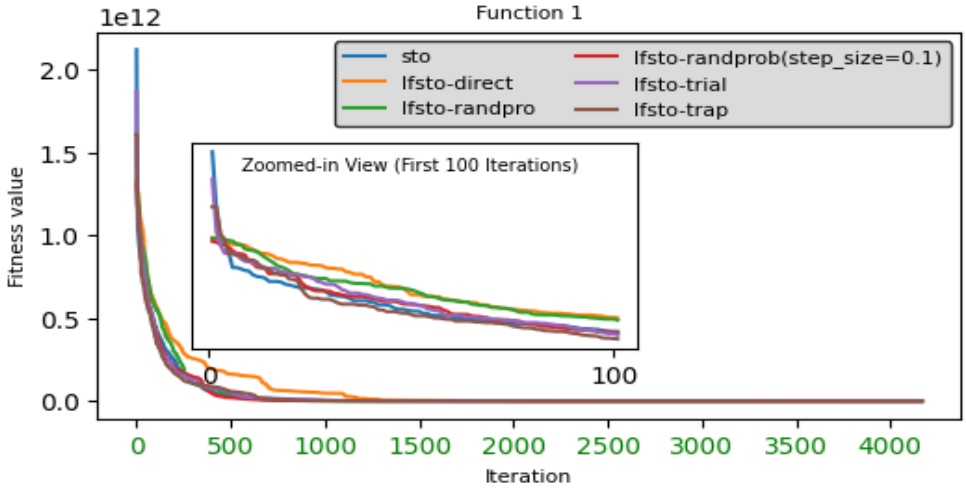


Figure 7.59. Convergence graph of LF integration strategies on CEC-2017 test functions F1 for dimension 50

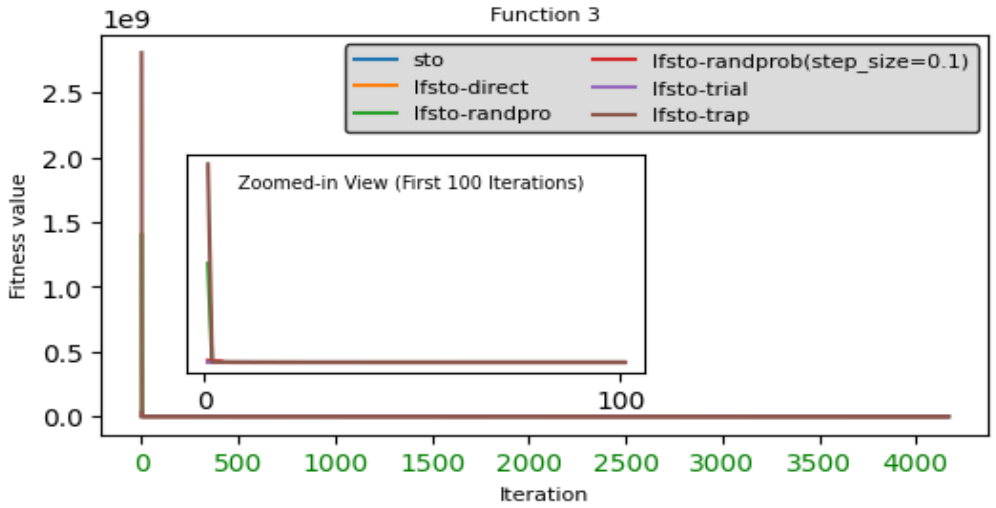


Figure 7.60. Convergence graph of LF integration strategies on CEC-2017 test functions F3 for dimension 50

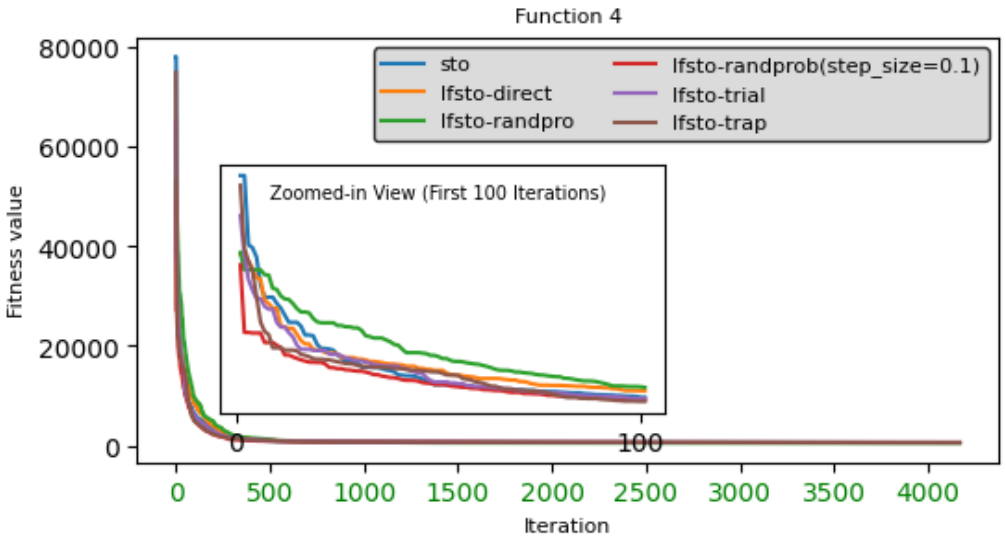


Figure 7.61. Convergence graph of LF integration strategies on CEC-2017 test functions F4 for dimension 50

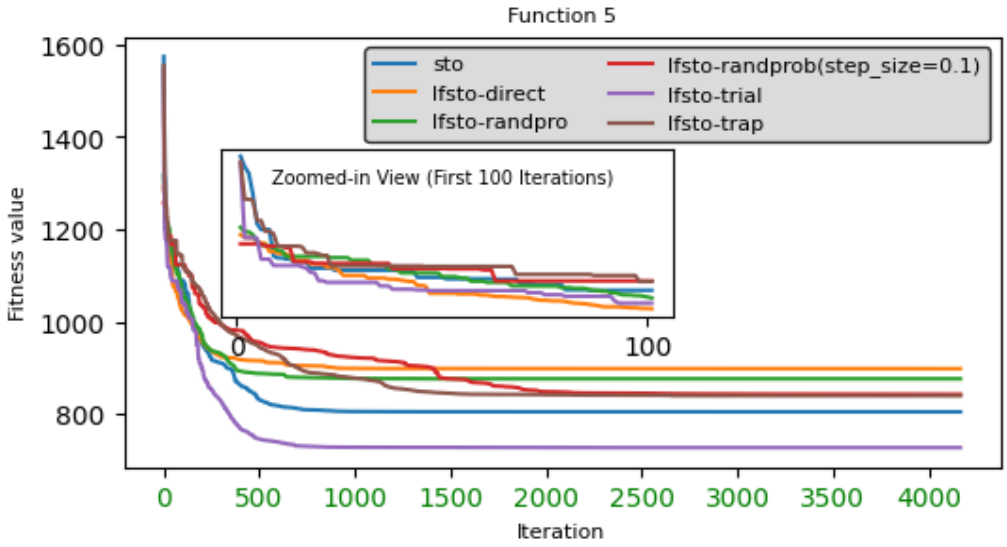


Figure 7.62. Convergence graph of LF integration strategies on CEC-2017 test functions F5 for dimension 50

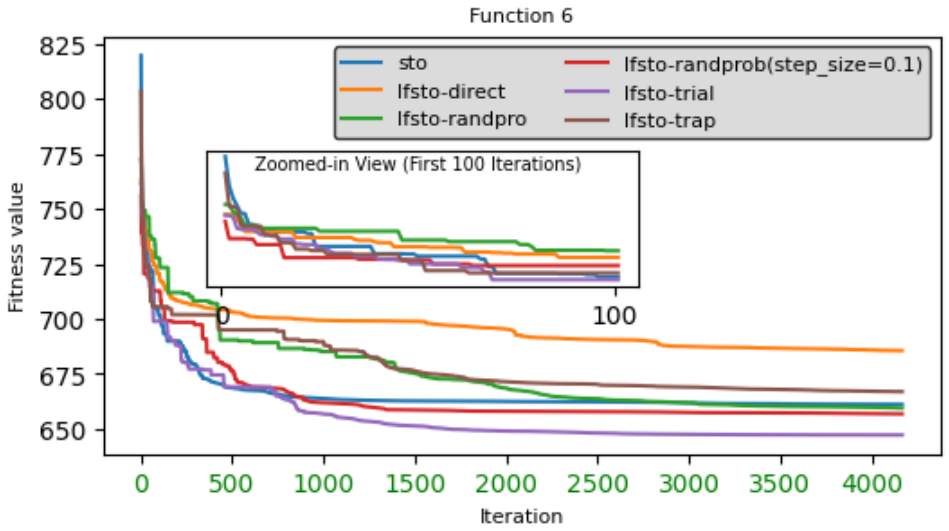


Figure 7.63. Convergence graph of LF integration strategies on CEC-2017 test functions F6 for dimension 50

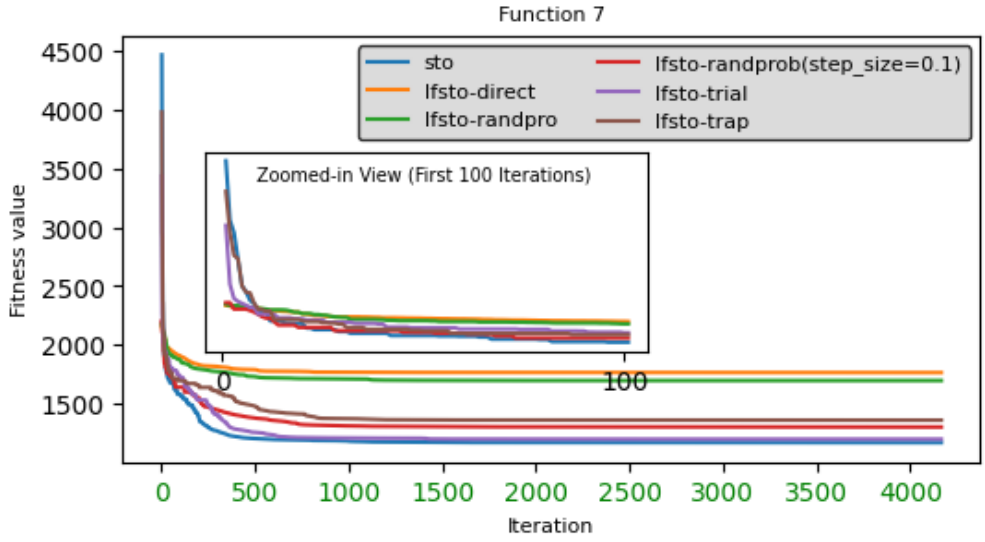


Figure 7.64. Convergence graph of LF integration strategies on CEC-2017 test functions F7 for dimension 50

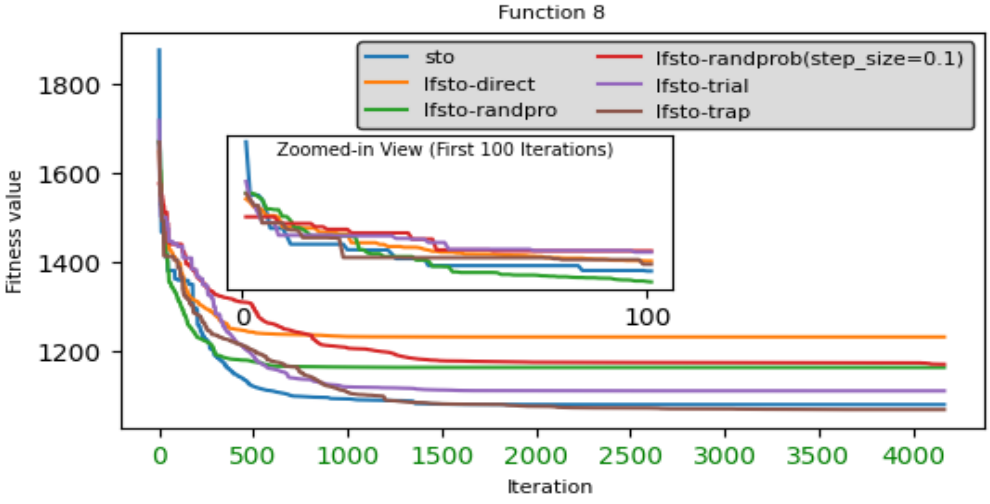


Figure 7.65. Convergence graph of LF integration strategies on CEC-2017 test functions F8 for dimension 50

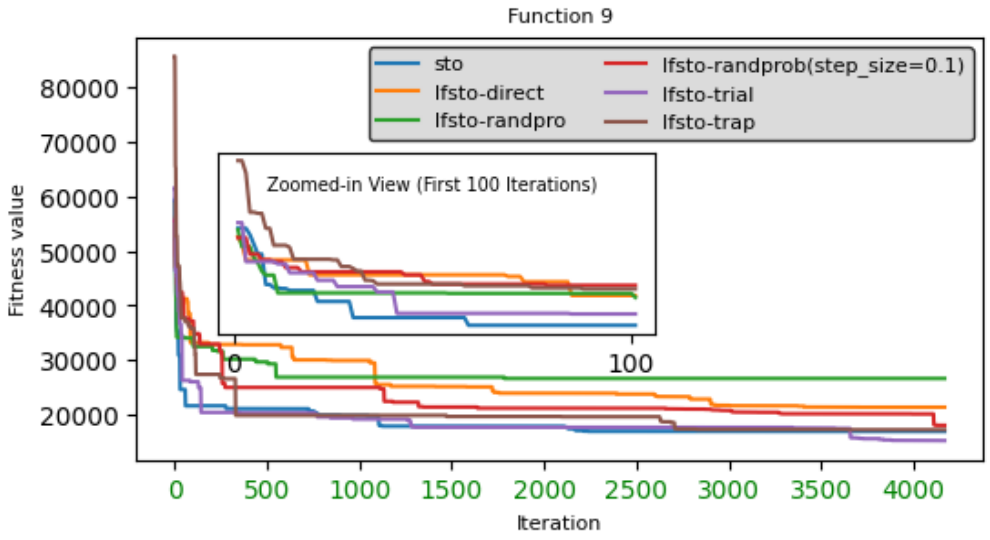


Figure 7.66. Convergence graph of LF integration strategies on CEC-2017 test functions F9 for dimension 50

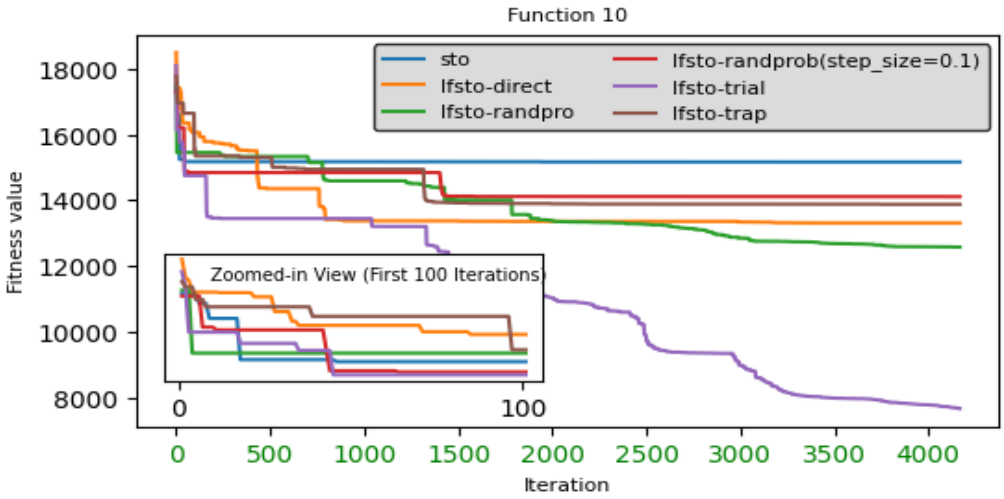


Figure 7.67. Convergence graph of LF integration strategies on CEC-2017 test functions F10 for dimension 50

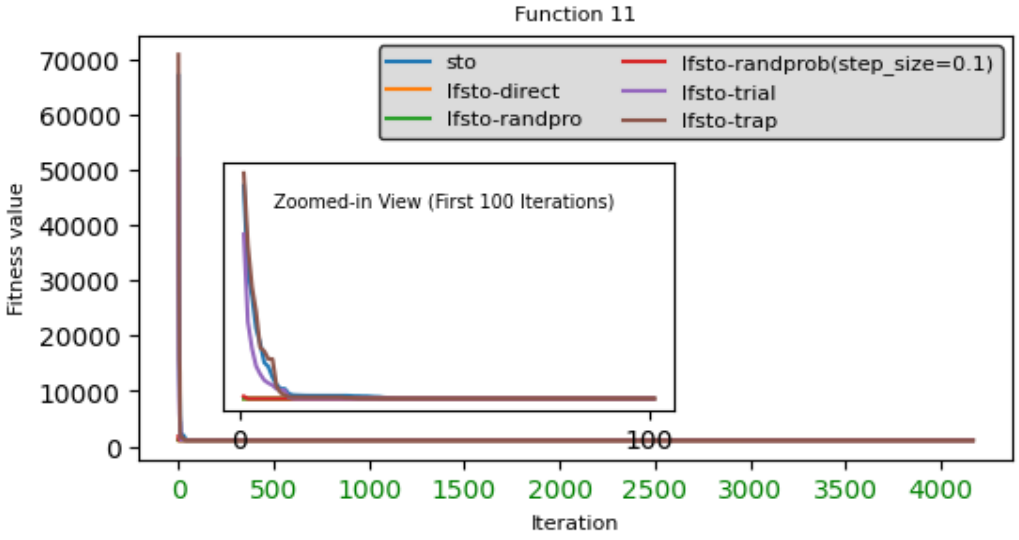


Figure 7.68. Convergence graph of LF integration strategies on CEC-2017 test functions F11 for dimension 50

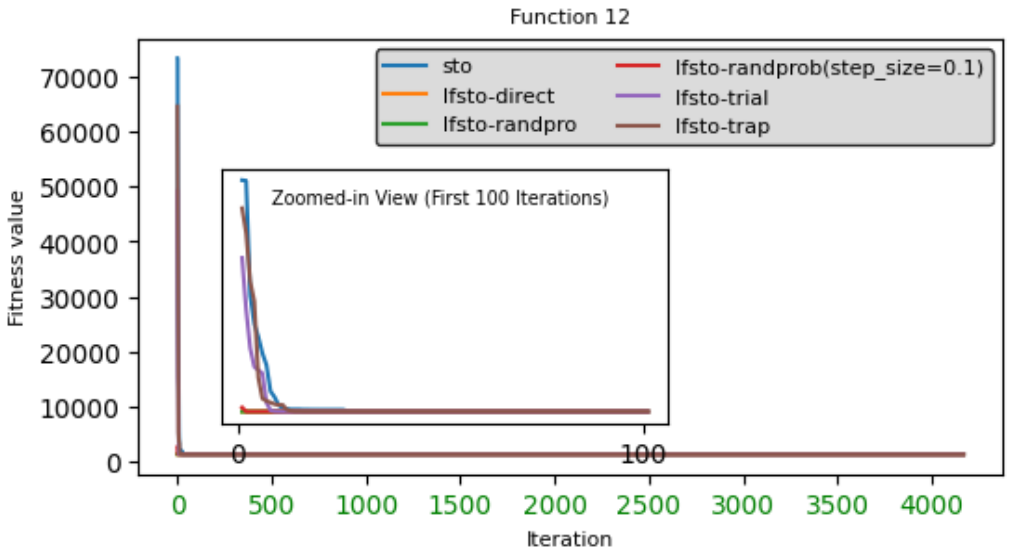


Figure 7.69. Convergence graph of LF integration strategies on CEC-2017 test functions F12 for dimension 50

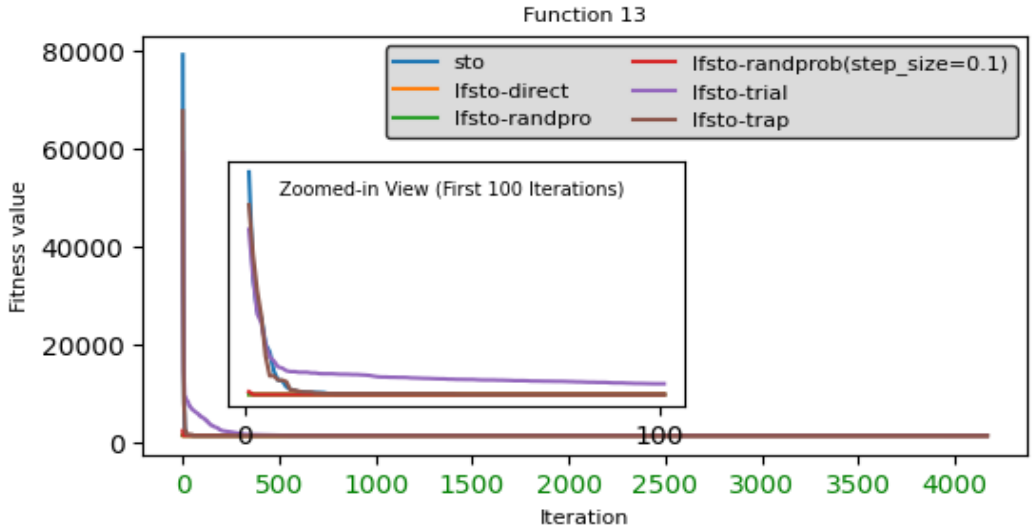


Figure 7.70. Convergence graph of LF integration strategies on CEC-2017 test functions F13 for dimension 50

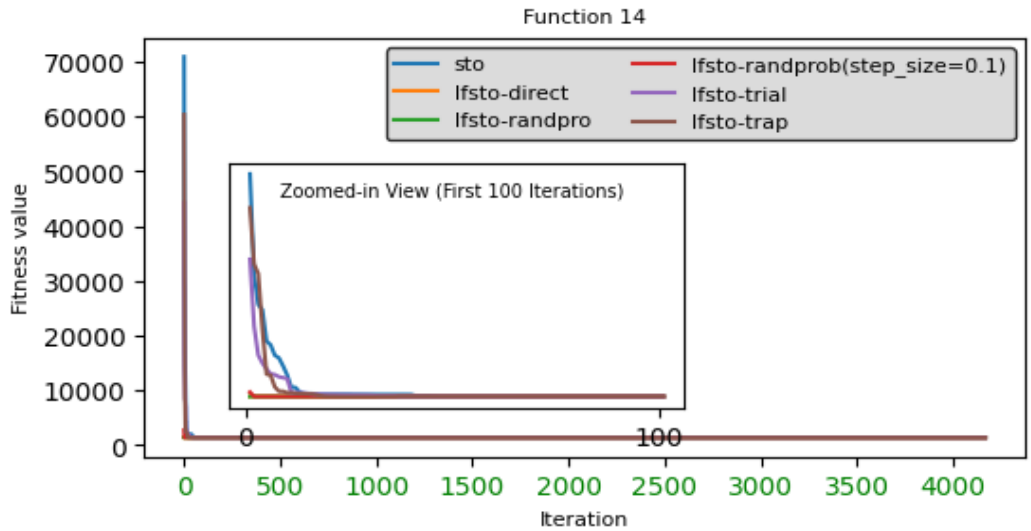


Figure 7.71. Convergence graph of LF integration strategies on CEC-2017 test functions F14 for dimension 50

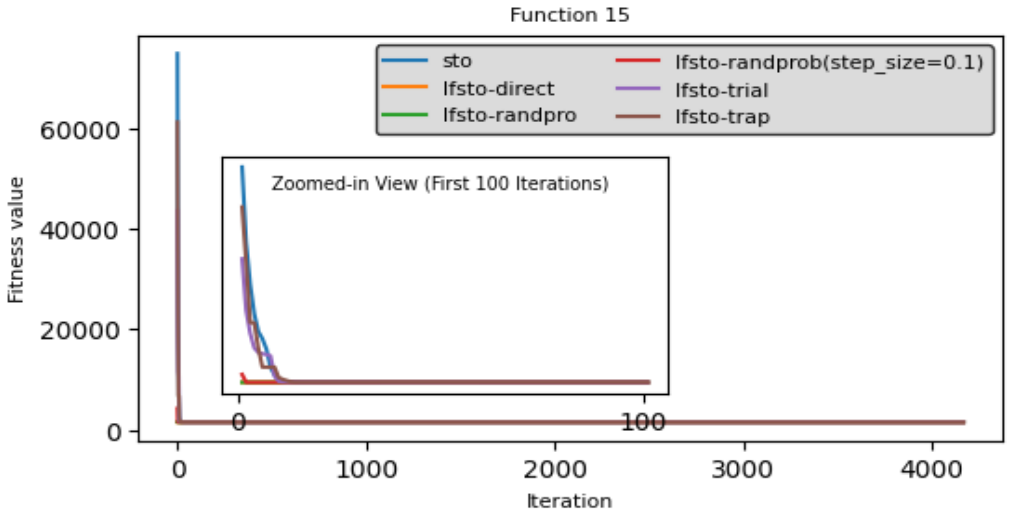


Figure 7.72. Convergence graph of LF integration strategies on CEC-2017 test functions F15 for dimension 50

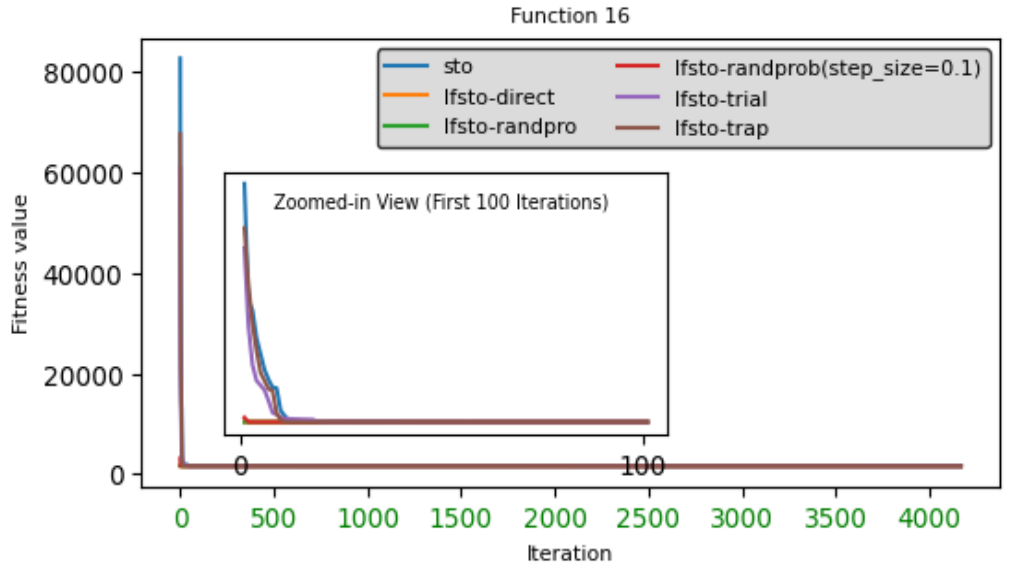


Figure 7.73. Convergence graph of LF integration strategies on CEC-2017 test functions F16 for dimension 50

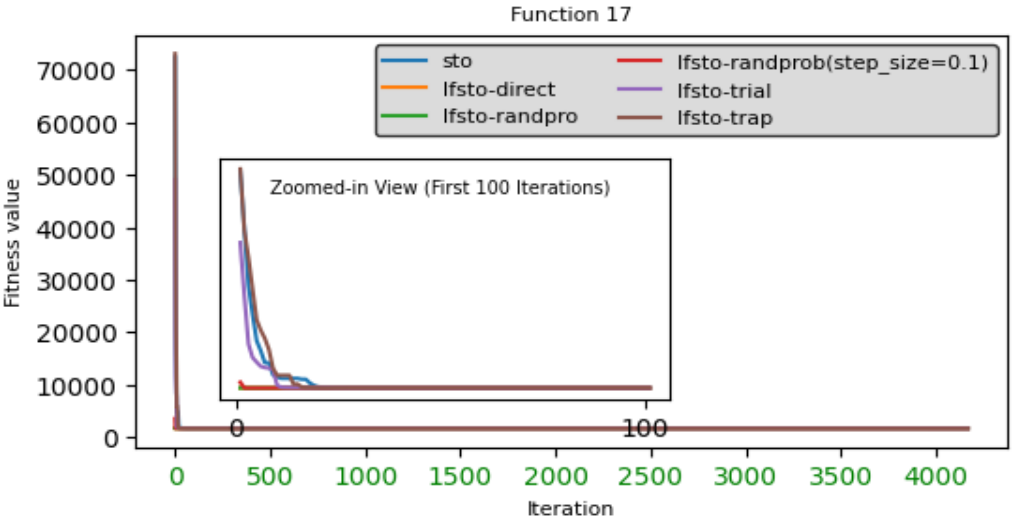


Figure 7.74. Convergence graph of LF integration strategies on CEC-2017 test functions F17 for dimension 50

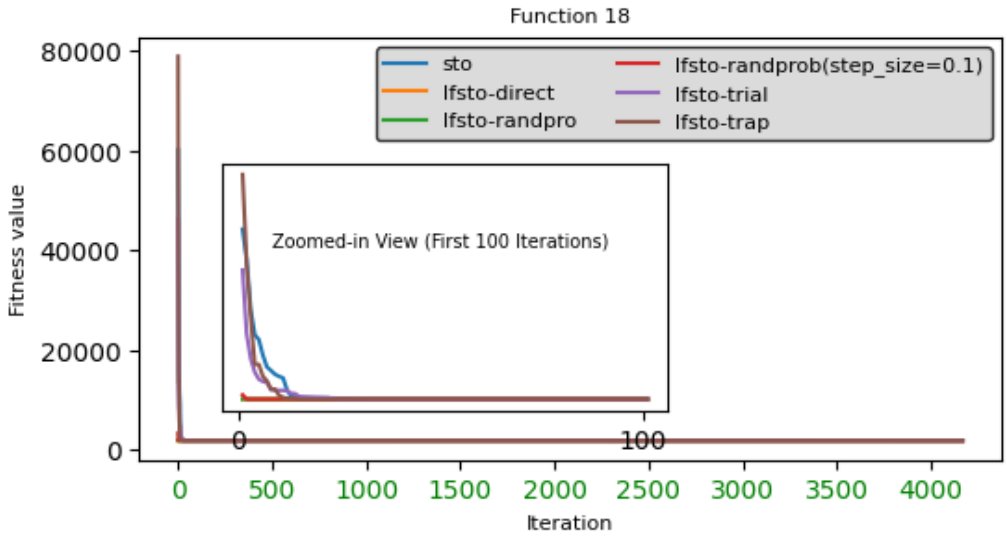


Figure 7.75. Convergence graph of LF integration strategies on CEC-2017 test functions F18 for dimension 50

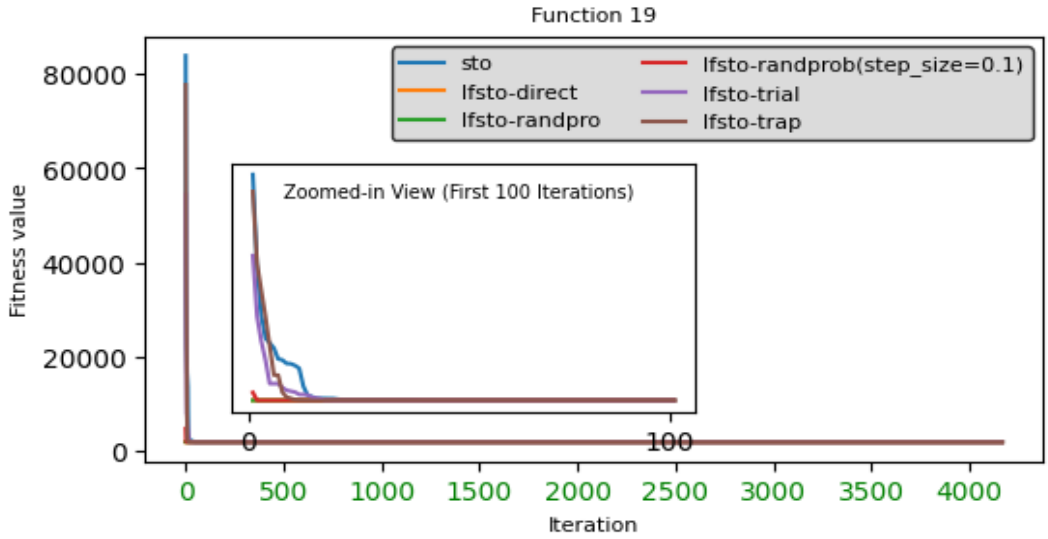


Figure 7.76. Convergence graph of LF integration strategies on CEC-2017 test functions F19 for dimension 50

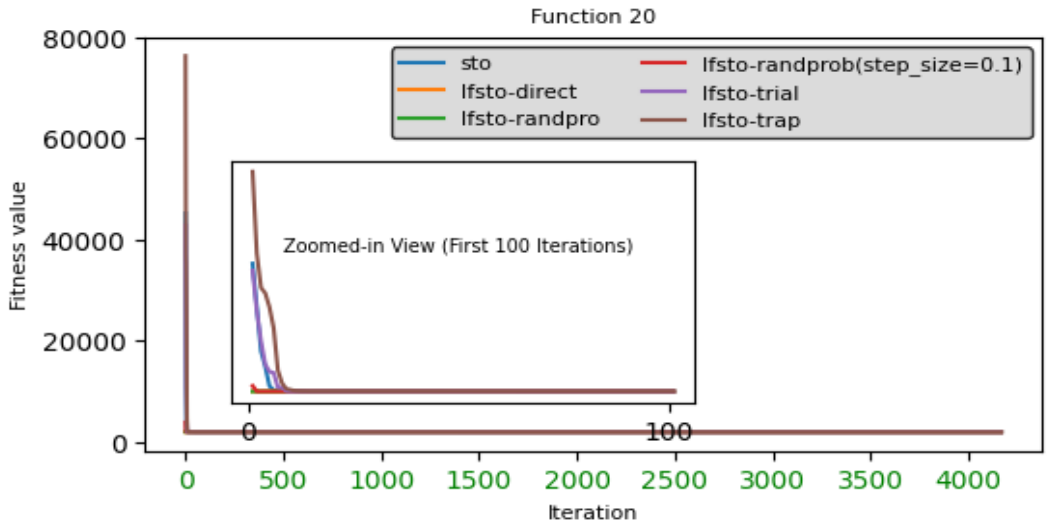


Figure 7.77. Convergence graph of LF integration strategies on CEC-2017 test functions F20 for dimension 50

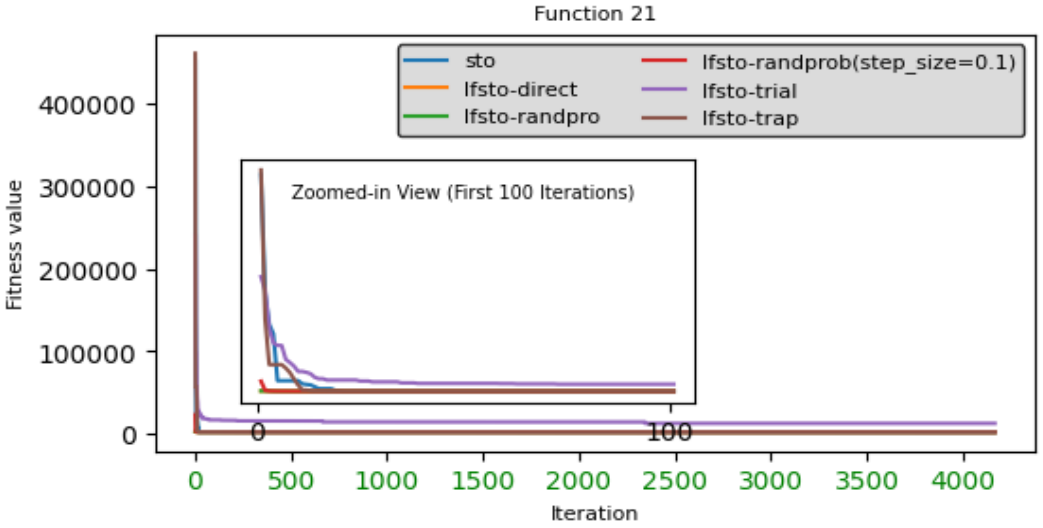


Figure 7.78. Convergence graph of LF integration strategies on CEC-2017 test functions F21 for dimension 50

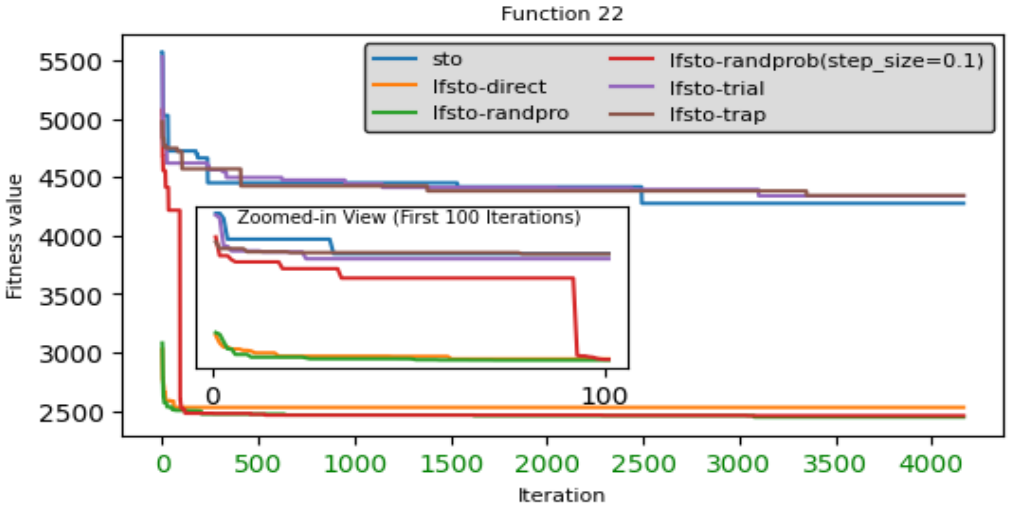


Figure 7.79. Convergence graph of LF integration strategies on CEC-2017 test functions F22 for dimension 50

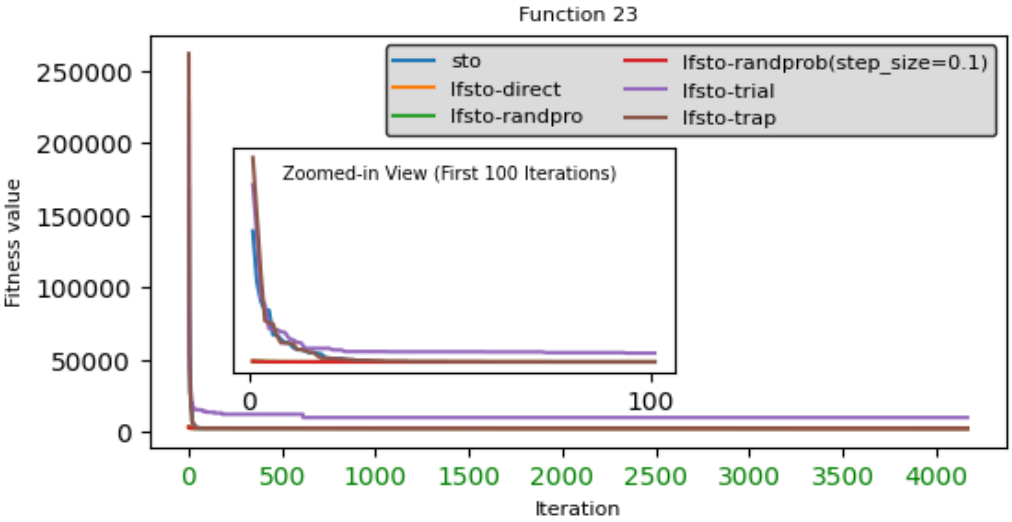


Figure 7.80. Convergence graph of LF integration strategies on CEC-2017 test functions F23 for dimension 50

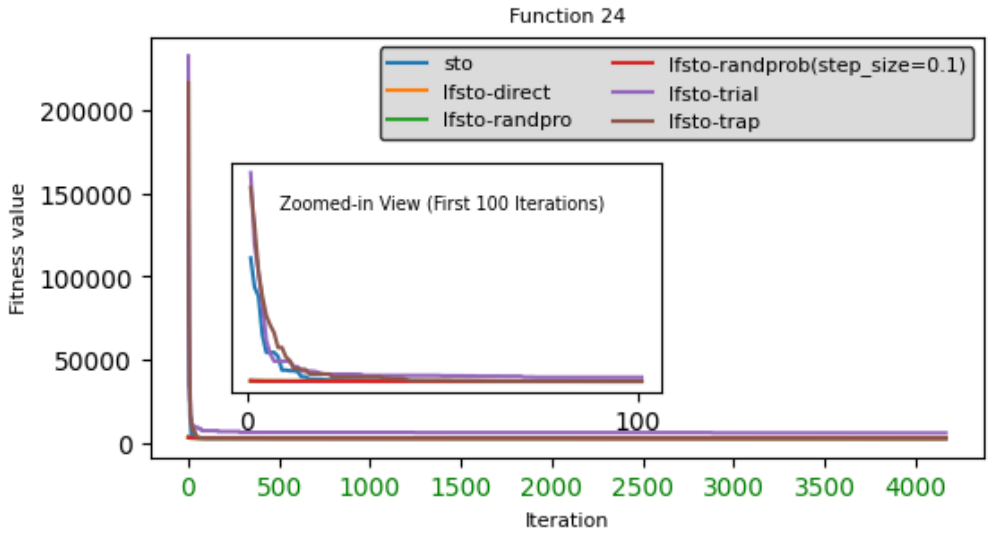


Figure 7.81. Convergence graph of LF integration strategies on CEC-2017 test functions F24 for dimension 50

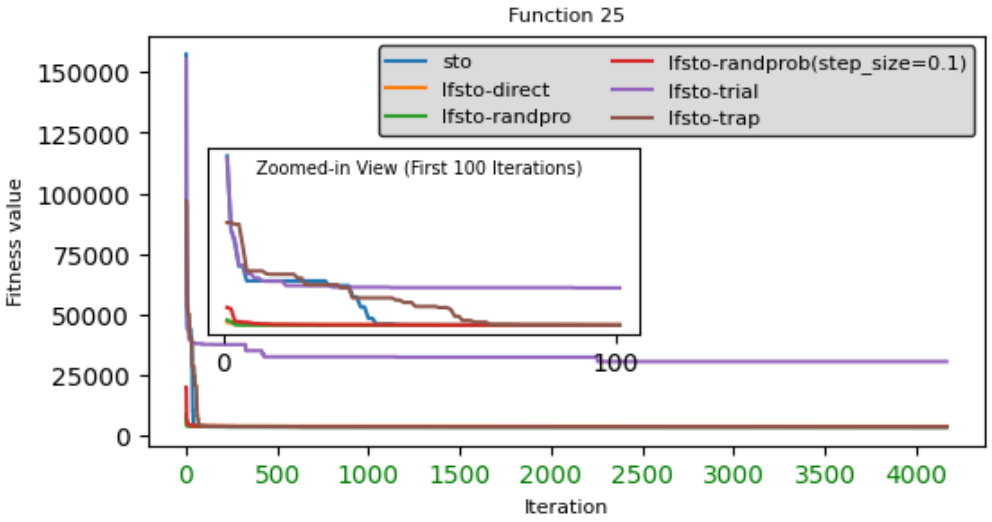


Figure 7.82. Convergence graph of LF integration strategies on CEC-2017 test functions F25 for dimension 50

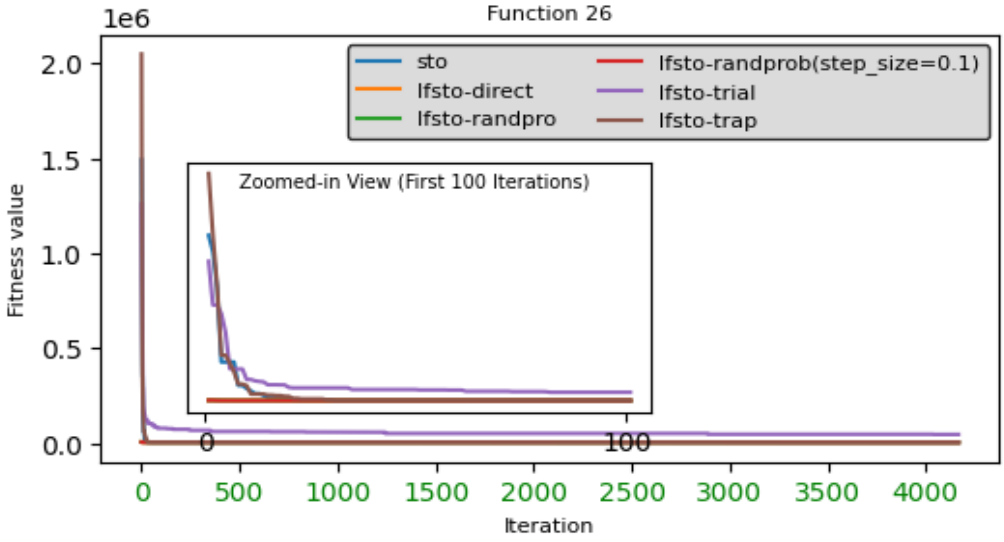


Figure 7.83. Convergence graph of LF integration strategies on CEC-2017 test functions F26 for dimension 50

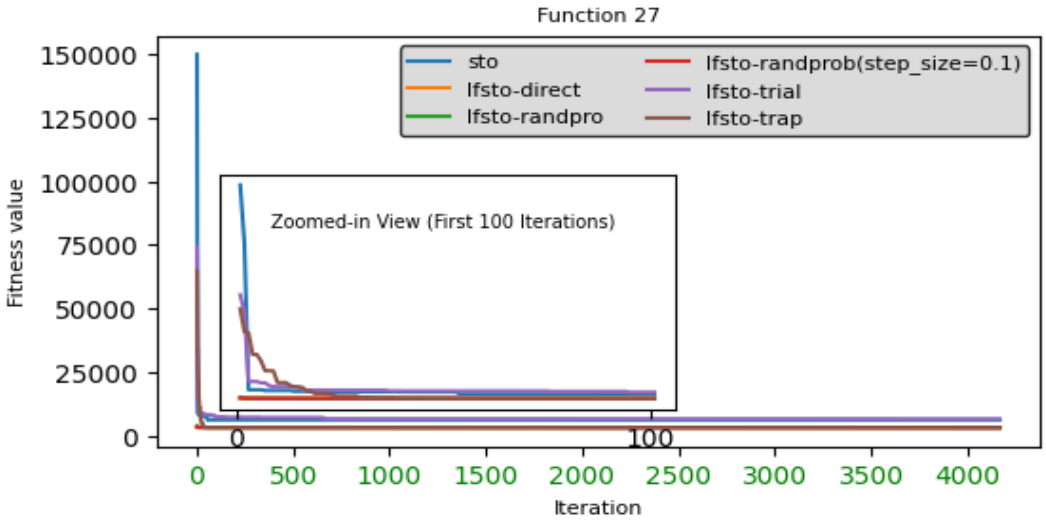


Figure 7.85. Convergence graph of LF integration strategies on CEC-2017 test functions F27 for dimension 50

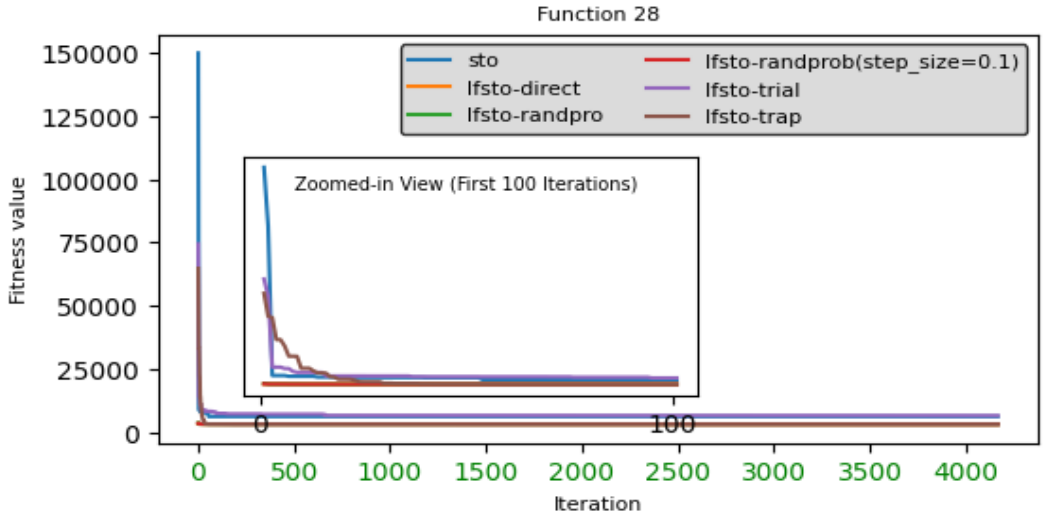


Figure 7.86. Convergence graph of LF integration strategies on CEC-2017 test functions F28 for dimension 50

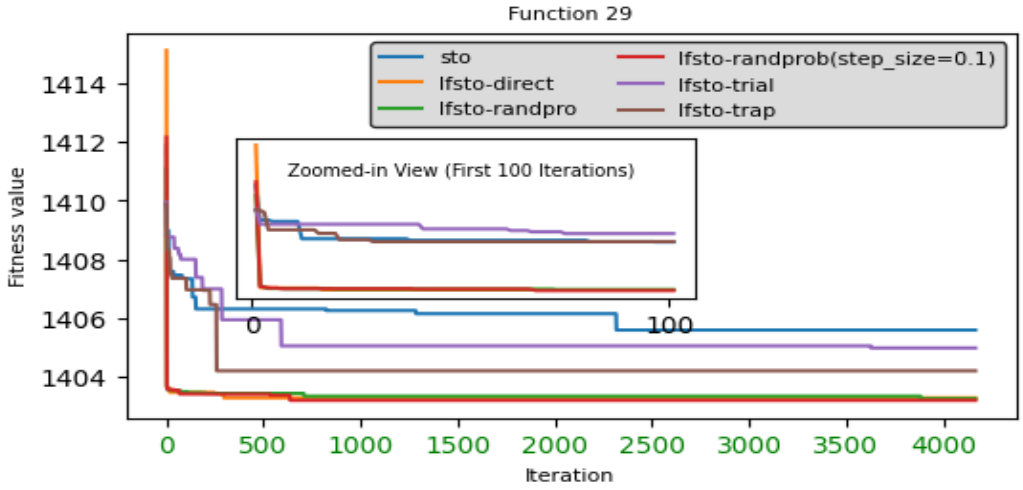


Figure 7.87. Convergence graph of LF integration strategies on CEC-2017 test functions F29 for dimension 50

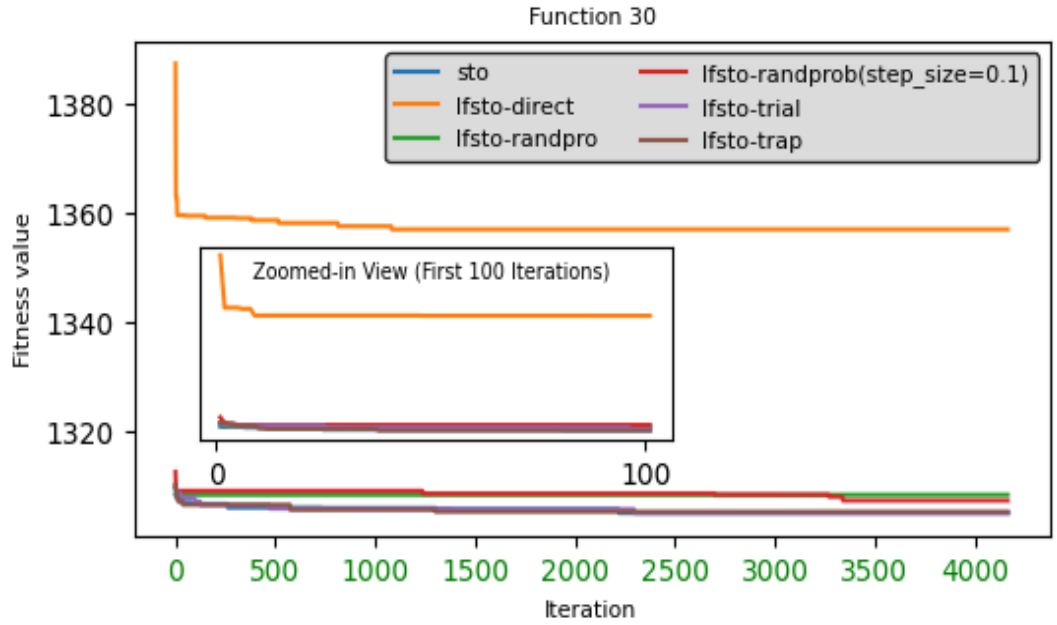


Figure 7.88. Convergence graph of LF integration strategies on CEC-2017 test functions F30 for dimension 50

8. CONCLUSIONS AND SUGGETIONS

8.1. Concluding Remarks

This thesis developed and evaluated novel LFSTO integration strategies utilizing the LF distribution, providing valuable insights into their efficacy in optimization contexts. Extensive testing and analysis demonstrated that LFSTO strategies can enhance convergence rates and solution quality compared to other optimization algorithms. The LFSTO approaches are adept at efficiently exploring the search space and adapting to solve various complex problems. The research assessed the performance of LFSTO strategies using twenty-nine benchmark functions from the CEC-2017 benchmark suite, as well as 10, 30, and 50-dimensional scenarios. Results indicate that LFSTO strategies outperform the standard STO algorithm across all three dimensions.

The LFSTO-randprob strategy outperforms the standard STO and the other three strategies across all dimensions. A non-parametric statistical test reveals significant differences in several comparisons between LFSTO strategies and the traditional STO. Convergence graphs further demonstrate that LFSTO strategies achieve the optimal solution more effectively than STO. The improved STO versions robustly express all findings, showing greater efficacy in overcoming local optima and delivering precise results for benchmark functions. Therefore, it is evident that the LF distribution is a powerful optimization tool, significantly enhancing both the traditional STO algorithm and various MH algorithms. This dissertation establishes the proposed LFSTO-randprob strategy as the most effective method for integrating LF to improve algorithm performance.

8.2. Suggestions

Apart from providing enlightening information on how the strategies perform in a variety of problem settings, this research has the potential to provide an enlarged comparison with other optimization approaches that are considered to be state-of-the-art. Furthermore, future researchers can leverage the well-known performance of the LFSTO-randprob strategy to integrate it into various nature-inspired algorithms. In different optimization settings, Levy Flight (LF) can be utilized, and research can be conducted on a variety of different tactics. Proposed LFSTO strategies are capable of addressing real-world issues like data clustering, feature selection (FS), vehicle routing, and traveling salesman problems (TSP), among others. This is because these techniques have been

demonstrated to be viable for a variety of benchmark functions. To be more precise, FS, the routing problem, and the TSP are all issues that are essentially binary or discrete. To properly apply LF to these problems, discrete and binary variations of the algorithm can be built. This allows for a more in-depth evaluation of the influence that LF has.

One intriguing approach to improving algorithm performance is to hybridize LF and other enhancement algorithms. This presents a promising path for improving algorithm performance. It is possible that doing research into the impact of LF distribution on multi-objective and many-objective optimization algorithms could result in the discovery of useful insights regarding the optimization of difficulties. Additionally, new methods can be created through the use of chaotic mapping by methodically analyzing the impacts of LF on convergence in the process of solving problems with low, medium, high, and very high dimensions. In sum, investigators may fully utilize the LFSTO strategy perspective and drive the discipline of optimization to novel heights of creativity and consequence by following these investigation lines and continually improving the algorithm's construction and execution.

REFERENCES

Abdulwahab, H. A., Noraziah, A., Alsewari, A. A. and Salih, S. Q., 2019, An Enhanced Version of Black Hole Algorithm via Levy Flight for Optimization and Data Clustering Problems, *IEEE Access*, 7, 142085-142096.

Abualigah, L., Diabat, A. and Geem, Z. W., 2020, A Comprehensive Survey of the Harmony Search Algorithm in Clustering Applications, *Applied Sciences*, 10 (11).

Abualigah, L., Yousri, D., Abd Elaziz, M., Ewees, A. A., Al-qaness, M. A. A. and Gandomi, A. H., 2021, Aquila Optimizer: A novel meta-heuristic optimization algorithm, *Computers & Industrial Engineering*, 157.

Aghaee, R., Momeni, M. and Moallem, P., 2022, Semisupervised Band Selection From Hyperspectral Images Using Levy Flight-Based Genetic Algorithm, *IEEE Geoscience and Remote Sensing Letters*, 19, 1-5.

Alweshah, M., 2020, Solving Feature Selection Problems by Combining Mutation And Crossover Operations with The Monarch Butterfly Optimization Algorithm, *Applied Intelligence*, 51 (6), 4058-4081.

Awad, N. H., Ali, M. Z., Suganthan, P. N., Liang, J. J. and B. Y., Q., 2016, Problem Definitions and Evaluation Criteria for the CEC 2017 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization, *Nanyang Technological University, Singapore, Jordan University of Science and Technology, Jordan and Zhengzhou University, Zhengzhou, China*.

Ba, A. F., Huang, H., Wang, M., Ye, X., Gu, Z., Chen, H. and Cai, X., 2020, Levy-based Antlion-Inspired Optimizers with Orthogonal Learning Scheme, *Engineering with Computers*, 38 (1), 397-418.

Bandopadhyay, J. and Roy, P. K., 2020, Application of Hybrid Multi-Objective Moth Flame Optimization Technique for Optimal Performance of Hybrid Micro-Grid System, *Applied Soft Computing*, 95.

Barthelemy, P., Bertolotti, J. and Wiersma, D. S., 2008, A Lévy flight for Light, *Nature*, 453 (7194), 495–498

Bashath, S., Ismail, A. R., Alwan, A. A. and Hussin, A. A. A., 2022, An Improved Particle Swarm Optimization Based on Lévy Flight and Simulated Annealing for High Dimensional Optimization Problem, *International Journal of Advances in Intelligent Informatics*, 8 (1).

Bejarbaneh, E. Y., Masoumnezhad, M., Armaghani, D. J. and Pham, B. T., 2020, Design of Robust Control Based on Linear Matrix Inequality and a Novel Hybrid PSO Search Technique for Autonomous Underwater Vehicle, *Applied Ocean Research*, 101.

Blum, C., Puchinger, J., Raidl, G. R. and Roli, A., 2011, Hybrid Metaheuristics In Combinatorial Optimization: A Survey, *Applied Soft Computing*, 11 (6), 4135-4151.

Boudjemaa, R., Oliva, D. and Ouaar, F., 2020, Fractional Levy flight Bat Algorithm For Global Optimisation, *International Journal of Bio-Inspired Computation*,, 15 (2), 100-112.

Brownlee, J., 2021, A Gentle Introduction to Stochastic Optimization Algorithms, Online, https://machinelearningmastery.com/stochastic-optimization-for-machine-learning/: [15 August 2023].

Črepinšek, M., Liu, S.-H. and Mernik, M., 2013, Exploration and Exploitation in Evolutionary Algorithms, *ACM Computing Surveys*, 45 (3), 1-33.

Cui, Z., Hou, X., Zhou, H., Lian, W. and Wu, J., 2020, Modified Slime Mould Algorithm via Levy Flight, *2020 13th International Congress On Image And Signal Processing, Biomedical Engineering And Informatics (Cisp-Bmei 2020)*, Electr Network.

Dale G., M., John M., G., Evgeny N., S., Phillip A., S., Olga Yu, Z., Guillame, C., Linda, K., Andre A., M., Maurice G. , H. and Howard B. , Q., 2010, The Amur tiger: a case study of living on the edge *Biology and conservation of wild felids* 325-339.

Dang, D.-C., Friedrich, T., Kötzing, T., Krejca, M. S., Lehre, P. K., Oliveto, P. S., Sudholt, D. and Sutton, A. M., 2016, Escaping Local Optima with Diversity Mechanisms and Crossover. Proceedings of the Genetic and Evolutionary Computation Conference 2016**:** 645-652.

Dang, D.-C., Friedrich, T., Kotzing, T., Krejca, M. S., Lehre, P. K., Oliveto, P. S., Sudholt, D. and Sutton, A. M., 2018, Escaping Local Optima Using Crossover With Emergent Diversity, *IEEE Transactions on Evolutionary Computation*, 22 (3), 484-497.

Deepa, R. and Venkataraman, R., 2021, Enhancing Whale Optimization Algorithm with Levy Flight for Coverage Optimization in Wireless Sensor Networks, *Computers & Electrical Engineering*, 94.

Desale, S., Rasool, A., Andhale, S. and Rane, P., 2015, Heuristic and Meta-Heuristic Algorithms and Their Relevance to the Real World: A Survey, *International Journal of Computer Engineering In Research Trends*, 2 (5), 296-304.

Dorigo, M. M., V. Colorni, A., 1996, Ant System: Optimization by A Colony of Cooperating Agents, *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 26 (1), 29-41.

Emary, E., Zawbaa, H. M. and Sharawi, M., 2019, Impact of Lèvy flight on Modern Meta-Heuristic Optimizers, *Applied Soft Computing*, 75, 775-789.

Ewees, A. A., Mostafa, R. R., Ghoniem, R. M. and Gaheen, M. A., 2022, Improved Seagull Optimization Algorithm Using Lévy Flight and Mutation Operator for Feature Selection, *Neural Computing and Applications*, 34 (10), 7437-7472.

Feng, Y.-H. and Wang, G.-G., 2018, Binary Moth Search Algorithm for Discounted {0-1} Knapsack Problem, *IEEE Access*, 6, 10708-10719.

Francisco, M., Revollar, S., Vega, P. and Lamanna, R., 2005, A Comparative Study of Deterministic and Stochastic Optimization Methods for Integrated Design of Processes, *IFAC Proceedings Volumes*, 38 (1), 335-340.

Fulber-Garcia, V., 2020, Deterministic and Stochastic Optimization Methods, online, https://www.baeldung.com/cs/deterministic-stochastic-optimization: [15 August].

Gao, S., Yu, Y., Wang, Y., Wang, J., Cheng, J. and Zhou, M., 2021, Chaotic Local Search-Based Differential Evolution Algorithms for Optimization, *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 51 (6), 3954-3967.

Guerrero, M., Castillo, O. and García, M., 2015, Cuckoo Search via Lévy Flights and a Comparison with Genetic Algorithms, In: Fuzzy Logic Augmentation of Nature-Inspired Optimization Metaheuristics, Eds, p. 91-103.

Guo, L., Wang, G.-G., H. Gandomi, A., H. Alavi, A. and Duan, H., 2014, A New Improved Krill Herd Algorithm for Global Numerical Optimization, *Neurocomputing*, 138, 392-402.

Haklı, H. and Uğuz, H., 2014, A Novel Particle Swarm Optimization Algorithm with Levy Flight, *Applied Soft Computing*, 23, 333-345.

Hashim, F. A., Houssein, E. H., Mabrouk, M. S., Al-Atabany, W. and Mirjalili, S., 2019, Henry Gas Solubility Optimization: A Novel Physics-Based Algorithm, *Future Generation Computer Systems*, 101, 646-667.

Heidari, A. A. and Pahlavani, P., 2017, An Efficient Modified Grey Wolf Optimizer with Lévy Flight for Optimization Tasks, *Applied Soft Computing*, 60, 115-134.

Holland, J. H., 1992, Adaptation in Natural And Artificial Systems: An Introductory Analysis with Applications to Biology, Control, And Artificial Intelligence., *MIT press*.

Houssein, E. H., Saad, M. R., Hashim, F. A., Shaban, H. and Hassaballah, M., 2020, Lévy flight distribution: A New Metaheuristic Algorithm for Solving Engineering Optimization Problems, *Engineering Applications of Artificial Intelligence*, 94.

Hussain, K., Salleh, M. N. M., Cheng, S. and Shi, Y., 2018, On the Exploration and Exploitation in Popular Swarm-Based Metaheuristic Algorithms, *Neural Computing and Applications*, 31 (11), 7665-7683.

Hussien, A. G. and Amin, M., 2021, A Self-Adaptive Harris Hawks Optimization Algorithm with Opposition-Based Learning and Chaotic Local Search Strategy for Global Optimization and Feature Selection, *International Journal of Machine Learning and Cybernetics*, 13 (2), 309-336.

Iacca, G., dos Santos Junior, V. C. and Veloso de Melo, V., 2021, An Improved Jaya Optimization Algorithm with Lévy Flight, *Expert Systems with Applications*, 165.

Ingle, K. K. and Jatoth, D. R. K., 2020, An Efficient JAYA Algorithm with Lévy Flight for Non-linear Channel Equalization, *Expert Systems with Applications*, 145.

Jensi, R. and Jiji, G. W., 2016, An Enhanced Particle Swarm Optimization with Levy Flight for Global Optimization, *Applied Soft Computing*, 43, 248-261.

Jeter W, E., 2018, Mathematical Programming: An Introduction to Optimization, *Newyork*, Routledge, p.

Jie, C., Bin, X., Zhihong, P., Lihua, D. and Juan, Z., 2009, Optimal Contraction Theorem for Exploration–Exploitation Tradeoff in Search and Optimization, *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 39 (3), 680-691.

Joshi, S. K., 2023, Levy Flight Incorporated Hybrid Learning Model for Gravitational Search Algorithm, *Knowledge-Based Systems*, 265.

Kamaruzaman, A. F., Zain, A. M., Yusuf, S. M. and Udin, A., 2013, Levy Flight Algorithm for Optimization Problems - A Literature Review, *Applied Mechanics and Materials*, 421, 496-501.

Kammoun, A., Slama, R., Tabia, H., Ouni, T. and Abid, M., 2022, Generative Adversarial Networks for Face Generation: A Survey, *ACM Computing Surveys*.

Karaboga, D. and Basturk, B., 2007, A Powerful and Efficient Algorithm for Numerical Function Optimization: Artificial Bee Colony (ABC) Algorithm, *Journal of Global Optimization*, 39 (3), 459-471.

Katkovnik, V., Egiazarian, K. and Astola, J., 2002, Adaptive Window Size Image Denoising Based on Intersection of Confidence Intervals (ICI) Rule, *Journal of Mathematical Imaging and Vision*, 16, 223-235.

Keelawat, P., Thammasan, N., Numao, M. and Kijsirikul, B., 2021, A Comparative Study of Window Size and Channel Arrangement on EEG-Emotion Recognition Using Deep CNN, *Sensors (Basel)*, 21 (5).

Kennedy, J. E., Russell 1995, Particle Swarm Optimization, *International Conference on Neural Networks*, 1942-1948.

Khurma, R. A., Aljarah, I. and Sharieh, A., 2021, A Simultaneous Moth Flame Optimizer Feature Selection Approach Based on Levy Flight and Selection Operators for Medical Diagnosis, *Arabian Journal for Science and Engineering*, 46 (9), 8415-8440.

Kreischer, V. M., Thiago Tavares and Barbosa, H. J. K., Eduardo 2017, Evaluation of Bound Constraints Handling Methods in Differential Evolution Using the cec2017 Benchmark. XIII Brazilian Congress on Computational Intelligence.

Krishnanand, K. N. G., D., 2005, Detection of Multiple Source Locations using a Glowworm Metaphor with Applications to Collective Robotics, *IEEE Swarm Intelligence Symposium, doi:10.1109/sis.2005.1501606* CA, USA, 84-91.

Kumar, K. P., Singarapu, S., Singarapu, M. and Karra, S. R., 2023, Balancing Exploration and Exploitation in Nature Inspired Computing Algorithm Springer, Cham., p. 163--172.

Kumar, V. and Yadav, S. M., 2022, A State-of-the-Art Review of Heuristic And Metaheuristic Optimization Techniques for The Management of Water Resources, *Water Supply*, 22 (4), 3702-3728.

Li, J., An, Q., Lei, H., Deng, Q. and Wang, G.-G., 2022, Survey of Lévy Flight-Based Metaheuristics for Optimization, *Mathematics*, 10 (15).

Li, X., Epitropakis, M. G., Deb, K. and Engelbrecht, A., 2017, Seeking Multiple Solutions: An Updated Survey on Niching Methods and Their Applications, *IEEE Transactions on Evolutionary Computation*, 21 (4), 518-538.

Liang, J.-J. and Suganthan, P. N. D., Kalyanmoy, 2005, Novel Composition Test Functions for Numerical Global Optimization, *Proceedings 2005 IEEE Swarm Intelligence Symposium*, 68-75.

Ling, Y., Zhou, Y. and Luo, Q., 2017, Lévy Flight Trajectory-Based Whale Optimization Algorithm for Global Optimization, *IEEE Access*, 5, 6168-6186.

Liu, Y. and Cao, B., 2020, A Novel Ant Colony Optimization Algorithm with Levy Flight, *IEEE Access*, 8, 67205-67213.

Liu, Y., Cao, B. and Li, H., 2020, Improving Ant Colony Optimization Algorithm with Epsilon Greedy and Levy Flight, *Complex & Intelligent Systems*, 7 (4), 1711-1722.

Long, W., Jiao, J., Liang, X. and Tang, M., 2018, An Exploration-Enhanced Grey Wolf Optimizer to Solve High-Dimensional Numerical Optimization, *Engineering Applications of Artificial Intelligence*, 68, 63-80.

Lu, H., Sun, S., Cheng, S. and Shi, Y., 2021, An Adaptive Niching Method Based on Multi-Strategy Fusion for Multimodal Optimization, *Memetic Computing*, 13 (3), 341-357.

Ma, L., Cheng, S. and Shi, Y., 2021, Enhancing Learning Efficiency of Brain Storm Optimization via Orthogonal Learning Design, *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 51 (11), 6723-6742.

Mahdavi, S., Rahnamayan, S. and Deb, K., 2018, Opposition based learning: A literature review, *Swarm and Evolutionary Computation*, 39, 1-23.

Maier, H. R., Razavi, S., Kapelan, Z., Matott, L. S., Kasprzyk, J. and Tolson, B. A., 2019, Introductory Overview: Optimization Using Evolutionary Algorithms and Other Metaheuristics, *Environmental Modelling & Software*, 114, 195-213.

Mirjalili, S., Mirjalili, S. M. and Lewis, A., 2014, Grey Wolf Optimizer, *Advances in Engineering Software*, 69, 46-61.

Mirjalili, S., 2015, Dragonfly Algorithm: A New Meta-Heuristic Optimization Technique for Solving Single-Objective, Discrete, and Multi-Objective Problems, *Neural Computing and Applications*, 27 (4), 1053-1073.

Mirjalili, S., Mirjalili, S. M. and Hatamlou, A., 2015, Multi-Verse Optimizer: A Nature-Inspired Algorithm for Global Optimization, *Neural Computing and Applications*, 27 (2), 495-513.

Mujawib Alashjaee, A., 2023, An Efficient Approach Based on Remora Optimization Algorithm and Levy Flight for Intrusion Detection, *Intelligent Automation & Soft Computing*, 37 (1), 235-254.

Nautiyal, B., Prakash, R., Vimal, V., Liang, G. and Chen, H., 2021, Improved Salp Swarm Algorithm with Mutation Schemes for Solving Global Optimization and Engineering Problems, *Engineering with Computers*, 38 (S5), 3927-3949.

Nicholson, T. A. J., 2017, Optimization Techniques, In: Optimization in industry, Eds, *New York*: Routledge, p.

Pavlyukevich, I., 2007, Lévy Flights, Non-Local Search and Simulated Annealing, *Journal of Computational Physics*, 226 (2), 1830-1844.

Peng, L. and Zhang, D., 2022, An adaptive Lévy Flight Firefly Algorithm for Multilevel Image Thresholding Based on Rényi Entropy, *Journal of Supercomputing* 78 (5), 6875-6896.

Piotrowski, A. P., Napiorkowski, J. J. and Piotrowska, A. E., 2023, Choice of Benchmark Optimization Problems Does Matter, *Swarm and Evolutionary Computation*, 83.

Sağ, T., 2022, PVS: A New Population-Based Vortex Search Algorithm with Boosted Exploration Capability Using Polynomial Mutation, *Neural Computing and Applications*, 34 (20), 18211-18287.

Saji, Y. and Barkatou, M., 2021, A Discrete Bat Algorithm Based on Lévy Flights for Euclidean Traveling Salesman Problem, *Expert Systems with Applications*, 172.

Salcedo-Sanz, S., 2016, Modern Meta-Heuristics Based on Nonlinear Physics Processes: A Review of Models and Design Procedures, *Physics Reports*, 655, 1-70.

Salgotra, R., Singh, U., Singh, S., Singh, G. and Mittal, N., 2021, Self-adaptive Salp Swarm Algorithm for Engineering Optimization Problems, *Applied Mathematical Modelling*, 89 (1), 188-207.

Serbet, F. and Kaya, T., 2023, Optimization Approach in Window Function Design for Real-Time Filter Applications, *Journal of Circuits, Systems and Computers*, 32 (09), 2350143.

Seyyedabbasi, A., Aliyev, R., Kiani, F., Gulle, M. U., Basyildiz, H. and Shah, M. A., 2021, Hybrid Algorithms Based on Combining Reinforcement Learning and Metaheuristic Methods to Solve Global Optimization Problems, *Knowledge-Based Systems*, 223.

Sharma, H., Bansal, J. C., Arya, K. V. and Yang, X.-S., 2015, Lévy Flight Artificial Bee Colony Algorithm, *International Journal of Systems Science*, 47 (11), 2652-2670.

Sharma, P. and Raju, S., 2023, Metaheuristic Optimization Algorithms: A Comprehensive Overview and Classification Of Benchmark Test Functions, *Soft Computing*, 28 (4), 3123-3186.

Singh, H., Singh, B. and Kaur, M., 2021, An Improved Elephant Herding Optimization for Global Optimization Problems, *Engineering with Computers*, 38 (S4), 3489-3521.

Spall, J. C., 2012, Stochastic Optimization, In: Handbook of Computational Statistics, Eds, p. 173-201.

Stork, J., Eiben, A. E. and Bartz-Beielstein, T., 2020, A New Taxonomy of Global Optimization Algorithms, *Natural Computing*, 21 (2), 219-242.

Tan, K. C., Chiam, S. C., Mamun, A. A. and Goh, C. K., 2009, Balancing Exploration and Exploitation with Adaptive Variation for Evolutionary Multi-Objective Optimization, *European Journal of Operational Research*, 197 (2), 701-713.

Tizhoosh, H. R., 2005, Opposition-based learning: A new scheme for machine intelligence, *Proceedings - International Conference on Computational Intelligence for Modelling, Control and Automation, CIMCA 2005 and International Conference on Intelligent Agents, Web Technologies and Internet*, Vienna, Austria, 695-701.

Trojovsky, P., Dehghani, M. and Hanus, P., 2022, Siberian Tiger Optimization: A New Bio-Inspired Metaheuristic Algorithm for Solving Engineering Optimization Problems, *IEEE Access*, 10, 132396-132431.

Verma, S., Sahu, S. P. and Sahu, T. P., 2023, MCSO: Levy's Flight Guided Modified Chicken Swarm Optimization, *IETE Journal of Research*, 1-15.

Wang, G., Guo, L., Gandomi, A. H., Cao, L., Alavi, A. H., Duan, H. and Li, J., 2013, Lévy-Flight Krill Herd Algorithm, *Mathematical Problems in Engineering*, 2013, 1-14.

Wang, Z., Chen, Y., Ding, S., Liang, D. and He, H., 2022, A Novel Particle Swarm Optimization Algorithm with Lévy Flight and Orthogonal Learning, *Swarm and Evolutionary Computation*, 75.

Wilcoxon, F., 1947, Probability Tables for Individual Comparisons by Ranking Methods, *Biometrics*, 3 (3), 119-122.

Wilcoxon, F., 1992, Individual Comparisons by Ranking Methods, In: Breakthroughs in Statistics: Methodology and Distribution, Eds: Kotz, S. and Johnson, N. L., *New York*: Springer, New York, NY, p. 196-202.

Wu, G., Mallipeddi, R. and Suganthan, P. N., 2017, Problem Definitions and Evaluation Criteria for the CEC 2017 Competition on Constrained RealParameter Optimization, *National University of Defense Technology, Changsha, Hunan, PR China and Kyungpook National University, Daegu, South Korea and Nanyang Technological University, Singapore, Technical Report*.

Wu, H., Wu, P., Xu, K. and Li, F., 2020, Finite Element Model Updating Using Crow Search Algorithm with Levy Flight, *International Journal for Numerical Methods in Engineering*, 121 (13), 2916-2928.

Wu, L., Wu, J. and Wang, T., 2023, Enhancing Grasshopper Optimization Algorithm (GOA) with Levy Flight for Engineering Applications, *Sci Rep*, 13 (1), 124.

Yang, H., Dou, H., Baniya, R. K., Han, S., Guan, Y., Xie, B., Zhao, G., Wang, T., Mou, P., Feng, L. and Ge, J., 2018, Seasonal Food Habits and Prey Selection of Amur Tigers and Amur Leopards in Northeast China, *Sci Rep*, 8 (1), 6930.

Yang, J. a. B., Abdesselam and Phung, S. L., 2010, A Particle Swarm Optimization Algorithm Based on Orthogonal Design. IEEE Congress on Evolutionary Computation. Barcelona, Spain, IEEE**:** 1-7.

Yang, X.-S. and Deb, S., 2009, Cuckoo Search via L´evy Flights. *World Congress on Nature & Biologically Inspired Computing (NaBIC). Coimbatore, India, IEEE*.

Yang, X.-S., 2010a, Firefly Algorithm, Lévy Flights and Global Optimization, In: Research and Development in Intelligent Systems XXVI, Eds, p. 209-218.

Yang, X.-S., 2010b, Firefly Algorithm, Stochastic Test Functions and Design Optimisation, *International Journal of Bio-Inspired Computation*, 2 (2), 78-84.

Yang, X.-S. and Deb, S., 2013, Multiobjective Cuckoo Search for Design Optimization, *Computers & Operations Research*, 40 (6), 1616-1624.

Yi, H., Duan, Q. and Liao, T. W., 2013, Three Improved Hybrid Metaheuristic Algorithms for Engineering Design Optimization, *Applied Soft Computing*, 13 (5), 2433-2444.

Yonar, A. and Yapici Pehlivan, N., 2020, Artificial Bee Colony with Levy Flights for Parameter Estimation of 3-p Weibull Distribution, *Iranian Journal of Science and Technology, Transactions A: Science*, 44 (3), 851-864.

Zhang, Y. and Chen, H., 2023, Bio-Inspired Optimization in Engineering and Sciences, *Computer Modeling in Engineering & Sciences*, 137 (2), 1065-1067.

Zhao, R., Wang, Y., Liu, C., Hu, P., Li, Y., Li, H. and Yuan, C., 2020, Selfish Herd Optimizer with Levy-Flight Distribution Strategy for Global Optimization Problem, *Physica A: Statistical Mechanics and its Applications*, 538.

Zhou, X., Ma, H., Gu, J., Chen, H. and Deng, W., 2022, Parameter Adaptation-Based Ant Colony Optimization with Dynamic Hybrid Mechanism, *Engineering Applications of Artificial Intelligence*, 114, 105139.