Metaheuristic optimization algorithms: An overview

Brahim Benaissa[1*], Masakazu Kobayashi[1], Musaddiq Al Ali[1],
Tawfiq Khatir[2], Mohamed El Amine Elaissaoui Elmeliani[3]

[1]Toyota Technological Institute, Nagoya, Japan
[2]University Centre Salhi Ahmed Naama, Naama, Algeria
[3]The University of Kitakyushu, Fukuoka, Japan
[*]Corresponding author: benaissa@toyota-ti.ac.jp

ARTICLE INFO

DOI:10.46223/HCMCOUJS.acs.en.14.1.200.2024

Received: November 06th, 2023

Revised: January 10th, 2024

Accepted: January 22nd, 2024

Keywords:

exploration; exploitation; optimization; metaheuristic; review

ABSTRACT

Metaheuristic optimization algorithms are versatile and adaptable tools that effectively solve various complex optimization problems. These algorithms are not restricted to specific types of problems or gradients. They can explore globally and handle multi-objective optimization efficiently. They strike a balance between exploration and exploitation, contributing to advancements in optimization. However, it's important to note their limitations, including the lack of a guaranteed global optimum, varying convergence rates, and their somewhat opaque functioning. In contrast, metaphor-based optimization algorithms, while intuitively appealing, have faced controversy due to potential oversimplification and unrealistic expectations. Despite these considerations, metaheuristic algorithms continue to be widely used for tackling complex problems. This research paper aims to explore the fundamental components and concepts that underlie optimization algorithms, focusing on the use of search references and the delicate balance between exploration and exploitation. Visual representations of the search behavior of selected metaheuristic algorithms will also be provided.

1. Introduction

Metaheuristic optimization algorithms represent a versatile class of algorithms employed to address intricate optimization problems spanning various domains (Abdel-Basset, Abdel-Fatah, & Sangaiah, 2018). They are particularly useful when conventional optimization techniques, including gradient-based methods or exact algorithms, prove unsuitable due to factors such as problem complexity, non-linearity, or extensive search spaces. In the realm of metaheuristic optimization, several essential characteristics and principles stand out. These algorithms systematically explore the solution space through iterative processes, continuously enhancing the initial solution or solution population over multiple iterations (Agrawal, Abutarboush, Ganesh, & Mohamed, 2021).

The infusion of randomness into the search process serves as a distinguishing characteristic of the majority of metaheuristic algorithms (Odili, 2018). This stochastic element plays a pivotal role in reducing the likelihood of converging to local optima while simultaneously promoting a more thorough exploration of the complete solution space. Most metaheuristic algorithms prioritize the discovery of global optima or top-quality solutions across a wide range of the solution space, as opposed to fixating solely on local optima (Yang, 2011a).

In contrast to gradient-based methods, metaheuristics do not depend on gradient information from the objective function (Wong & Ming, 2019). This characteristic renders them applicable to problems featuring non-differentiable, discontinuous, or noisy objective functions. They exhibit versatility in their application, accommodating a wide spectrum of optimization problems, including continuous, discrete, combinatorial, and mixed-integer optimization (Dokeroglu, Sevinc, Kucukyilmaz, & Cosar, 2019). They can be utilized for both single-objective and multi-objective optimization tasks. Among the array of metaheuristic optimization algorithms, most early algorithms include:

Genetic Algorithms (GAs), drawing inspiration from the fundamental principles of natural selection and genetics, are computational optimization techniques that leverage a population of potential solutions (Holland, 1992). These algorithms employ genetic operators, including crossover and mutation, in tandem with sophisticated selection mechanisms (Sivanandam & Deepa, 2008). The overarching goal of GAs is to iteratively improve and evolve increasingly optimal solutions as they progress through successive generations. By mimicking the mechanisms of biological evolution, GAs harness the power of genetic diversity and survival of the fittest to guide the search for high-quality solutions in complex optimization landscapes (Beasley, Bull, & Martin, 1993). Through a process of recombination, mutation, and natural selection, GAs continually refine and adapt the solution candidates, ultimately leading to the discovery of superior solutions over time.

Particle Swarm Optimization (PSO) is an optimization algorithm that takes inspiration from the coordinated behaviors observed in natural phenomena, specifically the collective movements of birds in flocks (Kennedy & Eberhart, 1995). In the realm of PSO, a population of particles dynamically explores the solution space. These particles undergo a continual process of adjustment in their positions, a process that encompasses not only their individual best-known solutions but also the incorporation of information gleaned from the best-known solutions of all particles (Wang, Tan, & Liu, 2018). This approach encapsulates the idea that each particle emulates the behavior of an individual entity within a larger group (Banks, Vincent, & Anyakoha, 2007). As they traverse the solution space, particles are not operating in isolation. Instead, they are influenced by the collective wisdom of their peers, just as birds in a flock synchronize their movements to optimize their group's overall trajectory. In essence, PSO seeks to connect the power of collaboration and information exchange among particles to steer them toward promising regions of the solution space, ultimately facilitating the discovery of optimal or near-optimal solutions to complex optimization problems (Blackwell & Kennedy, 2018).

Simulated Annealing (SA), which draws its inspiration from the annealing process in metallurgy (Bertsimas & Tsitsiklis, 1993), commences its search with a metaphorical "high temperature." As the algorithm iteratively proceeds, it gradually reduces this temperature, emulating the cooling of a material during the annealing process. This cooling process serves as a pivotal component in SA, enabling it to escape from the confines of local optima (Van Laarhoven & Aarts, 1987). Just as the gradual decrease in temperature in metallurgy allows a material's atomic structure to evolve towards a more stable and desirable state, the analogous reduction in "temperature" in Simulated Annealing facilitates the exploration of the solution space, leading to the discovery of superior and globally optimal solutions in complex optimization problems (Busetti, 2003).

Ant Colony Optimization (ACO) derives its fundamental principles from the intricate foraging behaviors exhibited by real ants (Dorigo & Stützle, 2003). This algorithm employs

artificial ants as agents that traverse a given graph or solution space in search of optimal solutions (Dorigo, Birattari, & Stutzle, 2006). The navigation strategy of these synthetic ants closely mimics the foraging patterns observed in nature, where real ants communicate through chemical signals known as pheromones to guide one another toward the most promising paths (Dorigo & Blum, 2005). Similarly, in ACO, the artificial ants leave virtual pheromone traces as they explore, and these traces play a pivotal role in influencing the decisions of subsequent ant agents. Through this pheromone-based guidance mechanism, ACO effectively directs its exploration efforts toward areas of the solution space that hold the potential for improved solutions, allowing for the discovery of optimal or high-quality solutions in complex optimization scenarios (Dorigo & Stützle, 2019).

Tabu Search (TS) relies on its short-term memory, which serves a dual purpose in the algorithm (Gendreau & Potvin, 2005). This memory maintains a record of previously explored 'tabu' solutions to prevent redundant searches and avoid stagnation in the search process (Pirim, Eksioglu, & Bayraktar, 2008). This unique feature of TS enhances the algorithm in multiple ways. It improves computational efficiency by preventing revisits to already evaluated solutions and enhances the overall robustness and versatility of TS. By avoiding redundant visits and promoting diversification, TS explores the solution space more comprehensively, increasing the chances of finding superior solutions (Hertz, Taillard, & De Werra, 1995). Differential Evolution (DE), a population-based algorithm, operates by creating new candidate solutions by synthesizing variations observed among individuals within the population (Lampinen, Price, & Storn, 2005). This collective wisdom of the population members plays a crucial role in searching for optimal solutions. The exchange of diverse perspectives within the population guides DE in exploring the solution space and discovering refined solutions (Das, Mullick, & Suganthan, 2016).

Harmony Search (HS) is an optimization algorithm inspired by the collaborative and iterative nature of music composition (Yang, 2009). In HS, a population of candidate solutions is analogous to musical elements, and they undergo iterative adjustments, mirroring the fine-tuning process in music composition (Geem, Kim, & Loganathan, 2001). This iterative approach allows HS to continuously explore the solution space, aiming to optimize the encountered solutions. The algorithm's primary objective is to identify and converge towards improved solutions, akin to musicians crafting harmonious compositions through creative improvisations. HS's effectiveness in optimizing complex problems lies in its ability to capture the essence of harmonization and refinement from the musical world, resulting in solutions that exhibit precision and artistry, akin to a well-composed musical piece (Geem, 2010).

The Arithmetic Optimization Algorithm (AOA) (Abualigah, Diabat, Mirjalili, Abd Elaziz, & Gandomi, 2021) combines exploration and exploitation strategies to seek optimal or near-optimal solutions. In the exploration phase, AOA employs high-dispersion mathematical operations, including division and multiplication, to allow solutions to dynamically adapt their positions based on random numbers and the Math Optimizer Accelerated (MOA) function. Conversely, in the exploitation phase, the algorithm focuses on fine-tuning solutions through low-dispersion operations like subtraction and addition, guided by random numbers and the MOA function. AOA operates through multiple iterations, continually tracking the best solution and terminating when specified conditions are met. This unique balance between exploration and exploitation, driven by randomization and mathematical operations under the guidance of the MOA function, distinguishes AOA as an effective method for optimizing complex problems.

The YUKI algorithm introduces a dynamic methodology for search space reduction, focusing on establishing a localized search region around the best-identified solution (Benaissa,

Hocine, Khatir, Riahi, & Mirjalili, 2021). It continuously adjusts the search area's dimensions using inter-point distance as a key metric. The first point corresponds to the global best solution with the lowest fitness value, while the second point, MeanBest, represents the centroid of the best solutions found by each population member (Benaissa, Kobayashi, Kinoshita, & Takenouchi, 2023). Local boundaries are computed based on these reference points. The exploration population extends its search beyond the local search area, with the population size iteratively adjusted based on a conditional expression (Amoura, Benaissa, Al Ali, & Khatir, 2023). This approach ensures a flexible adaptation of search space dimensions. Notably, the algorithm's mathematical formulation is straightforward and exhibits several key attributes, including a clear separation between exploration and exploitation efforts, independence of search area sizes across dimensions, dynamic fine-tuning of the local search area, and its contraction as solutions approach the optimum (Shirazi, Khatir, Benaissa, Mirjalili, & Wahab, 2023).

Some of the controversies related to recently developed optimization algorithms is due to the confusion in naming such as the Sinh cosh optimizer (Bai et al., 2023), the Sine cosine algorithm (Mirjalili, 2016b), and the Gravitational search algorithm (Yazdani, Nezamabadi-Pour, & Kamyab, 2014), Planet Optimization Algorithm (To, Le, Wahab, & Le, 2022), Solar system algorithm (Zitouni, Harous, & Maamri, 2020), also the unclear connection between the metaphor inspiration and the algorithm heuristics, within algorithms of similar names, such as Prey-predator algorithm (Tilahun & Ong, 2015) and the hunting of animals: Hunting search (Oftadeh, Mahjoob, & Shariatpanahi, 2010) and the recently developed algorithms, such as Hunter-prey optimization (Naruei, Keynia, & Molahosseini, 2022), crocodiles hunting search (Kareem, 2022), The archerfish hunting optimizer (Zitouni, Harous, Belkeram, & Hammou, 2022), and the Coati Optimization Algorithm (Dehghani, Montazeri, Trojovská, & Trojovský, 2023).

These algorithms possess a significant degree of adaptability, allowing them to be tailored to address specific optimization challenges. The choice of an appropriate metaheuristic is contingent upon various factors, including the intrinsic attributes of the problem, the computational resources at one's disposal, and the user's proficiency in customizing and fine-tuning the algorithm to align with the precise requirements of the optimization task in question (Eiben & Smit, 2011).

2. The advantages of metaheuristic optimization algorithms

Metaheuristic optimization algorithms play a pivotal role in the realm of optimization due to their inherent adaptability and efficacy in addressing intricate optimization challenges across diverse domains. This discourse underscores their importance, elucidating the following salient factors:

2.1. Agnostic to the problem being solved

Metaheuristic algorithms possess an exceptional attribute that makes them highly adaptable and effective in a wide array of problem-solving scenarios (Yang, 2011a). They are characterized by their remarkable versatility as they do not discriminate based on the nature of the optimization problem (Yang, 2011b). This agnostic quality towards problem types is a valuable feature, as it allows metaheuristic algorithms to transcend traditional boundaries and find applications in various fields and domains.

For instance, in the realm of engineering (Kaveh, 2017), they can be used to optimize complex design parameters for structures (Kaveh, 2014), systems (Gavrilas, 2010), or processes (Al Thobiani et al., 2022). In logistics, these algorithms can help streamline supply chain operations (Griffis, Bell, & Closs, 2012) and route planning (Tarantilis, Ioannou, Kiranoudis, & Prastacos, 2005). Furthermore, they are equally at home in the world of finance (Soler-Dominguez,

Juan, & Kizys, 2017), where they can assist in portfolio optimization (Rahmani, Eraqi, & Nikoomaram, 2019), risk management (Azevedo, Vale, Oliveira, & Khodr, 2010), or algorithmic trading strategies (Kuo & Chou, 2021). Moreover, the data science domain benefits from metaheuristic algorithms for tasks such as feature selection (Agrawal et al., 2021), hyperparameter tuning (Ghandourah et al., 2023), or clustering analysis (Nanda & Panda, 2014).

2.2. Gradient independence

Unlike conventional methods, which heavily depend on the gradient information of the objective function, metaheuristics are particularly well-suited for situations where computing gradients is computationally expensive or even impossible (Gandomi, Yang, Talatahari, & Alavi, 2013). This attribute significantly broadens their applicability and makes them invaluable in various optimization scenarios. In traditional optimization, algorithms like gradient descent or conjugate gradient methods rely on the knowledge of the gradient, which provides the direction of the steepest ascent or descent for the objective function (Ruder, 2016). However, in many real-world problems, obtaining the gradient can be a challenging and resource-intensive task. This is especially true for problems with complex, non-differentiable, discontinuous (Yang, 2010), or noisy objective functions (Dang, Dardinier, Doerr, Izacard, & Nogneng, 2018).

Metaheuristics, on the other hand, are robust and versatile optimization approaches that do not require gradient information. Instead, they explore the search space by iteratively and intelligently sampling solutions (Hussain, Salleh, Cheng, & Shi, 2019). These methods often employ heuristics, randomization, and search strategies that make minimal assumptions about the problem's mathematical properties. As a result, they can effectively tackle optimization problems where gradients are unavailable, unreliable, or prohibitively expensive to compute.

2.3. Global search capability

These algorithms are specifically engineered to conduct a thorough exploration of the solution space, with the primary objective of finding global optima or high-quality solutions (Beheshti & Shamsuddin, 2013). This characteristic is of paramount importance, particularly when dealing with optimization problems that feature multiple local optima (Jaszkiewicz, 2001). In such problems, there are numerous points within the solution space where the objective function reaches local optima, which are solutions that are the best in their immediate vicinity but may not be the best overall. Traditional optimization methods, like gradient-based techniques, are susceptible to getting stuck at these local optima because they rely on local information to guide their search (Ruder, 2016).

Metaheuristic algorithms, on the other hand, employ a more versatile and global approach. They systematically explore the entire solution space, often by combining various search strategies, heuristics, or stochastic elements. This comprehensive exploration allows them to transcend the limitations of local convergence and increases the likelihood of discovering the global optima, which represents the best possible solution across the entire solution space (Hussain, Salleh, Cheng, & Naseem, 2017).

2.4. Multi-objective optimization

Multi-Objective Optimization (MOO) is a crucial field in the domain of optimization, where the primary focus is on simultaneously optimizing multiple, often conflicting objectives (Dong & Liu, 2021). This is in contrast to traditional single-objective optimization, where the goal is to find the optimal solution for a single objective. MOO is particularly pertinent in decision-making contexts where the pursuit of one objective may negatively impact another, requiring careful consideration of trade-offs (Sahali, Aini, Bouzit, Himed, & Benaissa, 2023).

The intrinsic adaptability of metaheuristics in handling MOO problems is instrumental in addressing complex real-world challenges, such as Kansei design (Kobayashi, 2019) and design optimization (Kobayashi, 2019). By providing a range of Pareto-optimal solutions that represent different compromise options, metaheuristics assist decision-makers in making informed choices that align with their preferences and priorities (Talbi, Basseur, Nebro, & Alba, 2012).

2.5. Exploration and exploitation

Exploration, as a fundamental component of metaheuristic algorithms, entails the systematic and purposeful search for new and uncharted regions within the solution space (Xu & Zhang, 2014). This phase aims to diversify the set of solutions under consideration, thereby increasing the chances of discovering novel and potentially superior solutions (Hussain et al., 2019). By venturing into unexplored territories, metaheuristics harness a degree of randomness and adaptability, allowing them to escape local optima - suboptimal solutions that are locally optimal but not globally so (Cuevas et al., 2021).

On the other hand, exploitation, the complementary facet of metaheuristics, involves intensively scrutinizing promising regions within the solution space, where known high-quality solutions have been identified. The goal here is to refine and enhance the existing solutions, leveraging the acquired knowledge to achieve further improvements. This aspect of metaheuristics is characterized by the ability to exploit historical information and focus the search toward the most promising areas, thereby gradually converging toward optimal or near-optimal solutions (Alorf, 2023).

The remarkable proficiency of metaheuristics lies in their ability to dynamically balance the two aspects of exploration and exploitation. This dynamic equilibrium is essential in avoiding premature convergence to suboptimal solutions, as a myopic emphasis on exploitation can result in stagnation at local optima. Conversely, an excessive focus on exploration may lead to an inefficient search that fails to capitalize on promising regions (Kobayashi, 2020). Metaheuristics, through sophisticated control mechanisms and adaptive strategies, achieve a harmonious synergy between these two facets, ensuring that the algorithm effectively explores the solution space, discovers valuable solutions, and exploits the knowledge acquired during the search to enhance the quality of solutions (Xu & Zhang, 2014).

2.6. Configurability and tuning

This high degree of configurability is instrumental in adapting metaheuristic algorithms to address the intricacies of distinct problem contexts. By fine-tuning algorithmic parameters, practitioners can optimize the behavior and performance of these heuristics to effectively tackle the idiosyncrasies of the problem at hand (Lessmann, Caserta, & Arango, 2011). This ability to adjust parameters can be crucial, as various problems exhibit diverse characteristics, such as search space dimensions, objective functions, and constraints.

Research has shown that the configurability of metaheuristics can enhance their adaptability (Huang, Li, & Yao, 2019). The judicious selection of parameter values allows practitioners to strike a balance between exploration and exploitation, optimizing the search process within the algorithm. Moreover, it permits the fine-grained adjustment of convergence speed, thereby tailoring the algorithm to meet specific performance requirements and computational resources. Some algorithms do not have tuning parameters (Syafruddin, Köppen, & Benaissa, 2018).

2.7. Practical problem solving

The significance of these algorithms also lies in their ability to offer viable solutions to problems that are typically characterized by high dimensionality, non-linearity, and a multitude of constraints (Singh & Choudhary, 2021). Traditional optimization techniques often struggle to provide optimal or near-optimal solutions within reasonable timeframes for such complex problem instances. Metaheuristics, on the other hand, exhibit remarkable adaptability and robustness, making them well-suited for scenarios where the search space is vast, and the objective function is not easily defined or computationally expensive to evaluate.

In machine learning, the hyperparameter tuning problem (Birattari & Kacprzyk, 2009), which involves configuring the parameters of machine learning algorithms to enhance predictive performance, has been notably addressed through the application of metaheuristics, providing valuable insights into the optimal hyperparameter configurations for diverse learning tasks.

2.8. Innovation

In the quest for improved metaheuristic algorithms, researchers embrace a multidisciplinary approach that draws from various fields such as computer science, mathematics, and biology (Velasco, Guerrero, & Hospitaler, 2023). This interdisciplinary synergy not only enriches the theoretical foundation of metaheuristic algorithms but also enhances their practical applicability across a broad spectrum of real-world problems.

The relentless pursuit of innovation in metaheuristic algorithms encompasses multiple dimensions, including the development of new algorithmic structures, exploration of diverse optimization landscapes (Chakraborty, Sharma, Saha, & Chakraborty, 2021), and the adaptation of metaheuristic principles to emerging technological paradigms. By doing so, researchers continually refine and expand the toolkit available to practitioners (Peres & Castelli, 2021).

Furthermore, the dynamic nature of this research field fosters a vibrant academic community, with scholars engaged in ongoing discourse, collaboration, and knowledge exchange. This ecosystem of intellectual exchange serves as a crucible for nurturing fresh ideas, refining existing techniques, and validating the practicality of novel approaches through rigorous experimentation and evaluation (Bolufé-Röhler & Chen, 2020).

3. The limitations of metaheuristic optimization algorithms

Metaheuristic algorithms stand as formidable and adaptable tools, designed to address intricate optimization challenges (Chopard & Tomassini, 2018). However, an exhaustive examination of their attributes reveals several limitations and constraints necessitating consideration. These constraints encompass the following aspects:

3.1. Absence of global optimality guarantee

Global optimality, in the context of an optimization problem, signifies the identification of the absolute best solution within the entire search space, as determined by the objective function (Adam, Alexandropoulos, Pardalos, & Vrahatis, 2019). The absence of a global optimality guarantee means that metaheuristic algorithms cannot ensure that the solution they converge upon is the global optimum, i.e., the best possible solution achievable for the given problem.

One of the primary reasons for this absence of a global optimality guarantee is the nature of metaheuristic algorithms themselves. These algorithms are often stochastic in nature and operate by iteratively exploring and exploiting the search space (Du & Swamy, 2016). They rely

on heuristics, which are problem-specific rules or guidelines, to guide their search. As a result, the solutions generated by these algorithms are contingent on the initial conditions, algorithm parameters, and the inherent randomness in their search strategies. Consequently, the solutions obtained may be influenced by these factors and may not consistently reach the ultimate pinnacle of optimization.

Furthermore, the challenge of global optimality becomes particularly pronounced in cases involving highly complex or multimodal search spaces. Complex search spaces exhibit a multitude of local optima, making it challenging for metaheuristic algorithms to differentiate between local and global optima (Kuyu & Vatansever, 2021). Multimodal search spaces feature multiple distinct optima, further complicating the task of locating the global optimum. Metaheuristic algorithms may inadvertently converge on a local optimum, which is a solution that is superior only within a limited neighborhood in the search space, and fail to explore other regions where the global optimum might reside (Singh & Singh, 2014).

3.2. Convergence speed

Convergence speed, a critical performance metric of metaheuristic algorithms, exhibits significant variability contingent upon the nature of the problem being addressed and the particular algorithm employed (Gutjahr, 2009). It is imperative to acknowledge that the convergence speed is not a constant parameter; rather, it is intricately intertwined with the intricacies of the optimization problem at hand and the chosen metaheuristic algorithm (Yang, 2011a).

For numerous optimization problems, especially those characterized by high-dimensional or complex solution spaces, achieving convergence, wherein the algorithm reaches a satisfactory solution, can be a protracted process. In such scenarios, metaheuristic algorithms often require a substantial number of iterations before they are able to discover solutions that meet the predefined quality criteria (Blum & Roli, 2003). The requirement for an extensive iteration count can result in a considerable computational burden, entailing an extended computational runtime and increased resource utilization.

The computational expenses incurred due to slow convergence can manifest in various ways, including increased energy consumption, extended execution times, and greater computational resource utilization, such as CPU and memory. Therefore, understanding the convergence speed and its implications on computational costs is paramount for practitioners and researchers when selecting and configuring metaheuristic algorithms for specific optimization tasks (Chopard & Tomassini, 2018).

3.3. Parameter tuning

The process of identifying optimal parameter settings for metaheuristics can be characterized as a challenging and time-intensive endeavor. It involves a systematic exploration of the parameter space, aiming to strike a balance between exploration and exploitation. This balance is essential for achieving efficient convergence and high-quality solutions. The search for suitable parameter values often necessitates extensive experimentation, simulation, and empirical evaluation.

Researchers and practitioners frequently employ various optimization techniques, including running the metaheuristic with different combinations of parameter values, evaluating the performance of each configuration, and selecting the one that yields the best results according to predefined criteria (Osaba et al., 2021). Furthermore, the quest for optimal parameter settings is

compounded by the fact that the choice of parameters can depend on the specific problem instance or dataset at hand. Consequently, it may be necessary to perform parameter tuning for each unique problem to achieve optimal performance, making it a laborious and resource-intensive process (Huang et al., 2019).

3.4. Black-box nature

The term "Black-Box" in the context of metaheuristic optimization algorithms implies that these methods operate without a comprehensive understanding of the problem being solved. Unlike traditional mathematical optimization methods, such as linear programming or integer programming, which rely on a detailed problem formulation, metaheuristics approach problems without requiring a priori knowledge of the problem structure (Sala & Müller, 2020). This feature is particularly advantageous in scenarios where problem formulations are complex or unknown, and in cases where the objective function might be non-differentiable, discontinuous, or computationally expensive to evaluate. Researchers and practitioners may find it challenging to interpret the results, make informed adjustments to the optimization process, or validate the quality of solutions generated by the metaheuristic algorithms (Omidvar, Li, & Yao, 2021).

Despite these inherent limitations, the enduring utilization of metaheuristic optimization algorithms persists due to their aptitude for tackling complex problems, for which precise or specialized methods remain elusive. Researchers and practitioners commonly employ an empirical approach, experimenting with diverse metaheuristics and parameter configurations to discern optimal solutions specific to their individual optimization challenges (Gallagher, 2016).

4. The controversy of metaphor-based optimization algorithms

Metaphor-based metaheuristic algorithms leverage metaphorical concepts and analogies from the natural world to solve complex problems. By drawing parallels between real-world phenomena and optimization processes, metaphor-based metaheuristics provide a unique and intuitive approach to tackling intricate, multi-dimensional, and often non-deterministic optimization challenges (Sörensen, 2015). This innovative paradigm offers a bridge between human intuition and algorithmic problem-solving, enabling more effective and holistic problem-solving strategies. In this section, we delve into the controversy surrounding the use of metaphors in the context of metaheuristic algorithms.

4.1. Oversimplification

While metaphors undeniably enhance the accessibility of algorithms, rendering them more approachable and comprehensible to a wider audience, this very accessibility may inadvertently lead to the oversimplification of the underlying algorithmic intricacies. In so doing, metaphors risk fostering misunderstandings and misconceptions among practitioners and researchers, who may be enticed to embrace a superficial understanding of the algorithm's functionality (Chica, Pérez, Cordon, & Kelton, 2017).

The challenge here lies in striking a delicate balance between the accessibility facilitated by metaphors and the preservation of algorithmic intricacies and nuances. The risk is that the metaphorical framework may emphasize the overarching principles and analogies, potentially obscuring the intricate details that are critical for precise comprehension of algorithmic operations. Consequently, this approach may impart an incomplete or erroneous understanding of the algorithm, which could hinder its effective application and limit its potential for addressing complex optimization challenges (Tovey, 2018).

4.2. Misleading expectations

Metaphors, which are often employed to conceptualize and communicate complex optimization processes, can inadvertently create an expectation that these optimization algorithms will closely emulate the behavior of their metaphorical counterparts in real-world scenarios (Camacho-Villalón, Dorigo, & Stützle, 2023). This is a reasonable presumption given that metaphors are designed to bridge the gap between abstract mathematical concepts and concrete, real-world phenomena. However, the crux of the issue lies in the fact that this presumption frequently does not align with the reality of algorithmic performance (Aranha et al., 2022).

4.3. Metaphor algorithm names

One of the primary concerns is that the naming of these algorithms doesn't always reflect the underlying principles or strategies they employ. Researchers sometimes choose names that are catchy or trendy, but these names may not adequately convey the uniqueness or innovation of the algorithm (Camacho-Villalón, Dorigo, & Stützle, 2022). As a result, it becomes challenging to discern what sets one algorithm apart from another, hindering the efficient selection of an appropriate algorithm for a specific problem. Furthermore, the similarity in names can lead to misunderstandings and misattribution of ideas. It can create a situation where algorithms with similar-sounding names are assumed to be closely related or even identical when, in fact, they may have distinct design philosophies, parameters, or performance characteristics, and the opposite can be true (Du & Swamy, 2016).

5. Metaheuristic algorithmic framework

The fundamental structural components governing the operation of these algorithms are succinctly outlined as follows:

5.1. Initialization

At the outset of the algorithm, it begins with the creation of an initial solution or a population of potential solutions. These starting points can be generated in one of two ways:

In this step, the algorithm randomly generates the initial solutions. This randomness can help explore a wide range of possible solutions, making it especially useful in situations where there is little prior knowledge about the problem. Alternatively, the algorithm may utilize problem-specific heuristics to create the initial solutions, leveraging domain expertise and problem-specific insights to guide the initial solution-generation process.

Once the initial solutions are generated, the algorithm then proceeds to evaluate their quality using an objective function. The objective function quantifies how well each solution performs with respect to the problem's goals and constraints. The algorithm aims to optimize this objective function, typically by adjusting the solutions iteratively (Osaba et al., 2021).

In the context of optimization algorithms like Particle Swarm Optimization (PSO), the algorithm uses strategies that rely on search references to update and refine the solutions over time. The Global Best: In PSO, each particle maintains a "global best" reference, which represents the best solution found by any particle in the entire swarm. This global best serves as a guide for the entire swarm, influencing the movement of individual particles towards a potentially better solution. And Personal Best: In addition to the global best, each particle maintains its "personal best" reference, which is the best solution it has found during its individual journey. This personal best reference guides the particle's movement, helping it explore and exploit the solution space efficiently (Engelbrecht, 2013).

These references are different from other algorithms, such as: Tabu Search maintains a tabu list of recently visited solutions. Solutions on this list are considered "tabu," and the algorithm avoids revisiting them. The tabu list serves as a search reference to prevent cycling (Pirim et al., 2008). In Differential Evolution, a parent vector is chosen as a reference for creating new candidate solutions. The mutation and crossover operators are applied to the parent vectors to generate new solutions (Price, Storn, & Lampinen, 2006). In Harmony Search, a memory matrix stores the best solutions found so far. During the search, the algorithm creates new solutions by improvising based on the values in the memory matrix. The memory matrix serves as a reference for creating new harmonies that aim to improve upon past solutions (Alia & Mandava, 2011).

In Ant Colony Optimization, pheromone levels on paths represent references. Ants deposit pheromones on paths they explore, and other ants are more likely to follow paths with higher pheromone levels. This reinforces the exploration of promising paths and leads to the discovery of better solutions over time (Ribeiro, Hansen, Maniezzo, & Carbonaro, 2002). YUKI algorithm identifies the best solution found so far, defined as the point with the minimum fitness value. This solution is crucial as it represents the center of the local search area (Al Ali, Shimoda, Benaissa, & Kobayashi, 2023). And the MeanBest is calculated as the center of the best solutions found so far by each member of the population. It acts as a reference point to determine the size and location of the local search area (Khatir et al., 2023).

By continuously updating solutions based on these references and following specific algorithms' strategies, the optimization process refines the solutions, gradually converging towards optimal or near-optimal solutions to the problem.

5.2. Objective function assessment

Objective Function Assessment is a crucial step in various optimization algorithms and problem-solving approaches. It involves evaluating the quality of initial solutions or those present in a population, primarily through the use of an objective function. This objective function serves as a critical metric that quantifies how well each solution aligns with the primary optimization goal. This goal could involve either maximizing or minimizing a specific criterion, such as a cost or fitness function, depending on the nature of the problem being addressed (Halim, Ismail, & Das, 2021).

In essence, the objective function acts as a guiding compass for the algorithm, helping it discern the direction in which it should steer the search for better solutions. By quantifying the degree of alignment between a solution and the desired objective, it provides a means of ranking and comparing different solutions. This allows the algorithm to prioritize and select those solutions that show the most promise in achieving the optimization goal. The algorithm relies on this function to make informed decisions at each iteration, iteratively refining the solutions in pursuit of the overarching optimization objective.

5.3. Iterative refinement

The core essence of a metaheuristic optimization algorithm revolves around a cyclic progression characterized by the following steps (Dokeroglu et al., 2019):

1. Solution Selection: In the initial phase of an optimization algorithm, denoted as Solution Selection, the algorithm makes critical decisions concerning the choice of solutions from the current population for further refinement or evolution. The selection strategy adopted in this phase is contingent upon the particular optimization algorithm under consideration. Various methods

have been proposed in the literature, encompassing techniques such as roulette wheel selection, tournament selection, and rank-based selection. The chosen solutions are deemed as potential candidates for the subsequent optimization iterations.

2. Solution Modification: Following the Solution Selection phase, selected solutions undergo a series of transformative operations known as Solution Modification. These operations emulate natural or problem-specific processes, aiming to generate new candidate solutions with potentially improved qualities. Notable instances of such operations include crossover and mutation in Genetic Algorithms, as well as particle movement in Particle Swarm Optimization. Crossover involves combining genetic material from two or more parent solutions to create offspring, while mutation introduces random changes to a solution. The selection and configuration of these operators play a pivotal role in the algorithm's efficacy.

3. Evaluation: Subsequent to Solution Modification, the algorithm proceeds to the Evaluation phase. During this stage, the newly generated or modified solutions are subjected to a rigorous assessment using a predefined objective function. The primary objective is to ascertain the quality and fitness of these solutions in the context of the optimization problem. The objective function encapsulates the optimization problem's goals, constraints, and requirements, and it provides a quantitative measure of how well a solution aligns with the problem's objectives.

4. Solution Replacement: Upon completing the Evaluation phase, the updated solutions are integrated back into the existing population in the Solution Replacement step. This integration may involve the replacement or repositioning of some of the pre-existing solutions based on various criteria, such as fitness, diversity enhancement, or algorithm-specific rules. This phase is pivotal for maintaining population diversity and promoting the convergence of the algorithm toward optimal or near-optimal solutions.

5. Termination Criteria: Throughout the optimization process, the algorithm continually monitors predefined Termination Criteria to determine whether the search process should be concluded. Commonly employed termination criteria encompass reaching a maximum number of iterations, attaining a predefined target quality threshold, or encountering stagnation in the optimization process. It is imperative to balance computational resources with the search for better solutions, and the careful selection of termination criteria plays a vital role in achieving this balance.

5.4. Balancing exploration and exploitation

Exploration refers to the systematic diversification of search space exploration in order to discover novel and potentially superior solutions, while exploitation emphasizes the focused refinement and optimization of promising solutions (Morales-Castañeda, Zaldivar, Cuevas, Fausto, & Rodríguez, 2020). Achieving an optimal trade-off between these two strategies is imperative for the success of iterative algorithms in various applications (Halim et al., 2021).

Conversely, the exploitation involves the concentrated refinement and optimization of solutions that have exhibited promise in terms of their quality or effectiveness. This entails the allocation of resources and effort towards fine-tuning and maximizing the utility of these solutions. Exploitation aims to capitalize on the known strengths of existing solutions, leveraging their attributes to their fullest potential.

The challenge of balancing exploration and exploitation arises from the inherent trade-off between these strategies. Overemphasis on exploration can lead to a lack of focus and scattered efforts, potentially delaying the convergence to optimal solutions. On the other hand, an excessive

bias towards exploitation may result in premature convergence to suboptimal solutions, limiting the algorithm's ability to discover superior alternatives (Hussain et al., 2019).

5.5. Memory and diversity maintenance

Memory retention within metaheuristics pertains to the preservation and utilization of information related to previously explored solutions. The concept of memory is preserved in various ways, including the storage of promising solutions, historical search trajectories, or knowledge about the problem structure. This retention of past knowledge serves the purpose of facilitating informed decisions during the search process, allowing the algorithm to capitalize on insights gained from prior explorations. These insights may manifest as adaptive parameters, guiding operators, or informed perturbation strategies, which collectively enhance the algorithm's ability to exploit promising regions of the solution space (Akay, Karaboga, & Akay, 2022).

On the other hand, diversity maintenance encompasses strategies employed to ensure that the solution population generated and manipulated by the metaheuristic remains sufficiently diverse (Castillo & Segura, 2020). The diversity of solutions is a vital aspect in the context of optimization, as it guards against premature convergence to local optima. Diverse populations provide the algorithm with a broader exploration capability, as they offer a wider array of perspectives on the solution space. To this end, various mechanisms such as diversification operators, population diversity measures, and selection schemes designed to preserve and enhance diversity, are integrated into metaheuristics (Parouha & Verma, 2021).

6. Gradient-based algorithms vs. Metaheuristic algorithms in optimization

Gradient-Based Optimization methods present several merits in optimization. They exhibit notable advantages, particularly their rapid convergence, especially when dealing with smooth and convex objective functions (Daoud et al., 2023).

Moreover, their suitability for high-dimensional problems, coupled with their amenable parallelization, renders them highly efficient in resource-rich computational settings. Nonetheless, these techniques suffer from noteworthy limitations, including susceptibility to local minima entrapment, thereby rendering them less amenable for non-convex functions. Furthermore, they mandatorily necessitate the availability of gradient information, a requirement that may not always be met in various problem domains (Dalla, da Silva, Dutra, & Colaço, 2021).

Conversely, Metaheuristic Algorithms provide a distinct set of advantages. They excel in solving intricate, non-convex problems characterized by discontinuous or noisy objective functions. Metaheuristic algorithms, characterized by their versatility, do not rely on gradient information, extending their applicability to a broader spectrum of optimization challenges. Notably, these algorithms exhibit proficiency in exploring diverse solutions, albeit at the cost of slower convergence rates for simpler functions, often necessitating substantial computational resources (Khanduja & Bhushan, 2021). Table 1 provides a Comparative Analysis of Optimization Algorithms: Gradient-Based vs. Metaheuristic Approaches.

Applications well-suited for Gradient-Based Optimization encompass diverse tasks such as training machine learning models, deep learning parameter tuning (Zhang, 2019), and select scientific simulations (Issa & Mostafa, 2022). In contrast, as discussed previously, Metaheuristic Algorithms prove particularly adept in addressing an array of complex objective functions and extensive search spaces (Agrawal et al., 2021).

Table 1

A comparative analysis of optimization algorithms: Gradient-based vs. Metaheuristic approaches

Algorithm aspect	Gradient-based algorithms	Metaheuristic algorithms
Objective Function	Typically differentiable	Can be non-differentiable, complex, or black-box
Search Space Exploration	Systematic and focused on local optima	Exploratory and aim to escape local optima
Initialization	Reliant on initial parameter guess	Often uses random or heuristic initial solutions
Update Rules	Gradient-based with an explicit formula	Diverse strategies and operations specific to the algorithm
Termination Criteria	Convergence-based (e.g., gradient norm or target value)	Diverse termination conditions
Deterministic vs. Stochastic	Deterministic	Often stochastic with random or heuristic elements
Problem Applicability	Well-suited for differentiable functions	More versatile and applicable to a broader range of problems
Use of Gradient Information	Requires gradient information	Does not rely on gradient information

Table 2

Gradient-based vs. Metaheuristic - advantages and disadvantages

	Gradient-based algorithms	Metaheuristic algorithms
Advantages	- Converge faster for smooth, convex functions	- Suitable for complex, non-convex problems
	- Well-suited for high-dimensional problems	- No need for analytical derivatives
	- Good for local optimization	- Can handle discontinuous or noisy functions
	- Easily parallelizable	- Exploration of diverse solutions
Disadvantages	- Can get stuck in local minima	- Slower convergence on simple functions
	- Sensitive to initial conditions	- Difficulty in fine-tuning parameters
	- Not suitable for discrete optimization	- Lack of theoretical convergence guarantees
	- Assumes differentiability of the objective	- Computationally intensive

The selection between these two approaches predominantly hinges on the inherent nature of the optimization problem at hand and the availability of computational resources (Dalla et al., 2021). Table 2, highlights the advantages and disadvantages of Gradient-Based Optimization and Metaheuristic-based Optimization.

7. Metaheuristic techniques for solution generation

Metaheuristic algorithms encompass versatile optimization approaches applicable across diverse problem domains. These algorithms employ a variety of solution-generation techniques to traverse the solution space in pursuit of optimal or near-optimal solutions. Table 3 showcases a selection of the used solution-generation techniques in metaheuristic algorithms.

The table illustrates a wide array of approaches employed in optimization algorithms. Some, such as the Genetic Algorithm and Particle Swarm Optimization, are based on population-based methodologies, while others, such as Simulated Annealing and Tabu Search, adopt neighborhood search strategies.

Additionally, the table underscores the prevalence of nature-inspired algorithms, with examples like Ant Colony Optimization and Firefly Algorithm emulating natural behaviors like ant foraging and firefly attraction, indicating a burgeoning interest in bio-inspired and evolutionary techniques for optimization challenges. Moreover, the inclusion of the YUKI algorithm, which employs a problem-specific approach involving exploration and exploitation, suggests that certain algorithms may be tailored to specific problem domains. Lastly, the algorithms vary in complexity, with Genetic Algorithm and Differential Evolution incorporating intricate processes like crossover and mutation, in contrast to simpler operations like perturbation found in Simulated Annealing.

8. Metaheuristic techniques for balancing exploration and exploitation

In each of these algorithms, the trade-off between exploration and exploitation is achieved through unique mechanisms, such as probabilistic models, dynamic parameters, and population-based behaviors. The specific strategies employed by these algorithms determine their efficiency in solving different optimization problems. Balancing these two aspects effectively is a key challenge, and researchers often customize these algorithms to address the requirements of specific problem domains. The flexibility of metaheuristic algorithms in striking this balance makes them valuable tools in optimization tasks across various fields. Table 4 shows some distinctive Exploration and Exploitation behaviors that characterize each algorithm’s performance.

The Table 4 encompasses examples of popular optimization algorithms. This assortment underscores the extensive spectrum of techniques utilized in solving optimization challenges. The table underscores the central trade-off between exploration, the quest for novel solutions, and exploitation, the refinement of known, promising solutions. Each algorithm employs distinctive strategies to strike a balance between these aspects.

Additionally, the table delineates the specific techniques or attributes of each algorithm that contribute to their exploration and exploitation methods. For instance, Genetic Algorithms employ crossover and mutation to explore diverse populations and favor individuals with superior fitness, while Simulated Annealing accepts suboptimal solutions to shift towards exploiting the best ones. This adaptability allows their exploration and exploitation strategies to solve various problems. Such as, Ant Colony Optimization and Bee Colony Optimization, leverage both exploration and exploitation by permitting individuals to explore new paths while reinforcing exploitation through mechanisms like pheromone deposition or information sharing.

Table 3

Optimization algorithms and their solution-generation techniques

Algorithm	**Solution-generation technique**
Genetic Algorithm (Holland, 1992)	Uses genetic operators like Crossover and Mutation to evolve solutions
Particle Swarm Optimization (Kennedy & Eberhart, 1995)	Generates solutions by adjusting particle movement and updating
YUKI algorithm (Benaissa et al., 2021)	Create a random distribution of points inside the local search area, and allocate part of it to exploration and the other part to exploitation
Simulated Annealing (Van Laarhoven & Aarts, 1987)	Perturbs solutions based on a temperature schedule
Ant Colony Optimization (Ribeiro et al., 2002)	Constructs solutions probabilistically by simulating ant behavior
Tabu Search (Hertz et al., 1995)	Explores neighborhoods guided by a tabu list
Harmony Search (Geem et al., 2001)	Generates solutions based on harmony memory and composition
Differential Evolution (Lampinen et al., 2005)	Creates solutions using mutation and crossover operations
Firefly Algorithm (Yang & Slowik, 2020)	Generates solutions based on firefly attraction and movement
Gravitational Search Algorithm (Rashedi, Nezamabadi-Pour, & Saryazdi, 2009)	Simulates particle movement in a gravitational field to find solutions
Cuckoo Search (Gandomi, Yang, & Alavi, 2013)	Generates solutions using Lévy flights and nest replacement
Bat Algorithm (Yang & Gandomi, 2012)	Updates solutions based on frequency and loudness
Grey Wolf Optimizer (Mirjalili, Mirjalili, & Lewis, 2014)	Finds solutions by encircling and attacking prey
Artificial Bee Colony (Karaboga, 2010)	Utilizes employed bees, onlooker bees, and scout bees to find solutions
Teaching-Learning-Based Optimization (Rao, Savsani, & Vakharia, 2011)	Involves two phases, teacher and learner phases for solution generation
Sinh Cosh Algorithm (Bai et al., 2023)	Represents solutions using sinusoidal waves
The arithmetic optimization algorithm (Abualigah et al., 2021)	Modify existing solutions using arithmetic operators Division, Multiplication, Subtraction, and Addition

Table 4

Exploration and exploitation strategies in optimization algorithms

Algorithm Name	Exploration	Exploitation
Genetic Algorithms (Holland, 1992)	- Creation of diverse populations	- Favoring individuals with better fitness
	- Crossover and mutation operations	- Concentrating search efforts on promising solutions
Particle Swarm Optimization (Kennedy & Eberhart, 1995)	- Using social and cognitive learning	- Converging toward best-known solutions
	- Guiding particles to explore different areas	- Adjusting positions based on experiences
YUKI algorithm (Benaissa et al., 2021)	- Using historical personal best locations	- Exploitation toward best-known solutions
	- Allocate part of the population to explore outside the local search area	- Adjusting the size of the local search area based on the progress of the search
Simulated Annealing (Van Laarhoven & Aarts, 1987)	- Allowing acceptance of worse solutions	- Shifting toward exploiting the best solutions
	- With decreasing probability	- Converging toward a global optimum
Ant Colony Optimization (Ribeiro et al., 2002)	- Allowing ants to explore new paths	- Reinforcing exploitation through pheromone deposition
	- Based on pheromone evaporation and choices	- Guiding ants toward successful solutions
Tabu Search (Hertz et al., 1995)	- Preventing revisiting explored solutions	- Considering previously visited solutions if showing potential
	- Promoting exploration	- For improvement, leading to exploitation
Differential Evolution (Lampinen et al., 2005)	- Generating diverse trial solutions	- Preferring individuals with better fitness
	- Using differential operators	- Promoting exploitation of promising solutions
Harmony Search (Geem et al., 2001)	- Introducing randomness in note generation	- Leading to exploitation by memory and pitch adjustments
	- Emphasizing exploration	- Emphasizing promising harmonies
Firefly Algorithm (Yang & Slowik, 2020)	- Random movement of fireflies	- Converging toward the brightest fireflies
	- Exploring different areas	- Promoting exploitation through light intensity

Algorithm Name	Exploration	Exploitation
Bat Algorithm (Yang & Gandomi, 2012)	- Using echolocation for random exploration	- Locating and exploiting the best solutions as the algorithm progresses
	- Emitting frequency-tuned pulses	- Adjusting emission intensity and loudness
Glowworm Swarm Optimization (Krishnanand & Ghose, 2009)	- Modeling the behavior of glowworms and their spatial awareness for exploration	- Promoting the exploitation of brighter neighbors
Krill Herd Algorithm (Gandomi & Alavi, 2012)	Employing random movements of krill to explore different areas	Attracting krill toward promising solutions through interactions
Bee Colony Optimization (Karaboga, 2010)	- Random selection of bees for exploration	- Sharing information and exploiting best solutions with other bees
	- Visiting different solutions	- Promoting exploitation through information sharing
Dragonfly Algorithm (Mirjalili, 2016a)	- Mimicking the hunting behavior of dragonflies for exploration	- Concentrating on capturing the most promising prey
Jaya Algorithm (Rao, 2016)	Focusing the search away from the best-known solution	Focusing on improving the best-known solutions
Gravitational Search Algorithm (Rashedi et al., 2009)	- Modeling gravitational attraction of masses	- Converging toward promising solutions
	- Moving masses randomly to explore	- Promoting exploitation through gravitational attraction
The arithmetic optimization algorithm (Abualigah et al., 2021)	- Diversify the search by exploring different regions	- Choice of which operator to use is based on random values
	- Move away from the current best solution	- Intensively searches for the near-optimal solution
Monkey Search (Mucherino & Seref, 2007)	- Mimicking the climbing and exploration behavior of monkeys	- Concentrating on the best-known solutions found during climbing
Vortex Search Algorithm (Doğan & Ölmez, 2015)	Applying a vortex concept to generate diversity in the search	Concentrating on improving solutions within the vortex
Termite life cycle optimizer (Le, To, Theraulaz, Wahab, & Le, 2023)	- Create reproductive termites when workers are not successful	- Soldiers are responsible for exploiting known promising solutions
	- Randomness to the movement and decision-making of termites	- Focusing on the best-known solution

Figures 1 to 5 depict the search behaviors of five distinct optimization algorithms, namely Particle Swarm Optimization (PSO), Grey Wolf Optimizer (GWO), Differential Evolution (DE), Teaching-Learning-Based Optimization (TLBO), and the YUKI algorithm. Two distinct test functions are employed in this analysis: the unimodal sphere function (denoted as Function (a)), which is particularly amenable to exploitation-oriented algorithms, and the multimodal Schwefel function (denoted as Function (b)), which is more suitable for exploration-oriented algorithms.

In each algorithmic evaluation, 500 particles are utilized, and the search process is iterated for a duration of 10 iterations. The primary objective is to visualize and assess the progressive evolution of the search process. These figures provide a comprehensive representation of the spatial distribution of the population of solutions at each iteration, offering insights into the extent of the search space explored by the population throughout the optimization process.

In PSO, exploration occurs as particles move randomly through the solution space, and exploitation is promoted through movement towards better solutions (Figure 1). Similarly, in GWO algorithm, wolves within the pack take on the role of explorers, venturing into uncharted solution spaces, while others act as exploiters, refining and improving upon the best solutions identified so far (Figure 2). In DE, however, individuals in the population are subjected to a mutation operation that creates a new candidate solution by taking the difference between two existing solutions and adding it to a third solution. Additionally, DE also uses strategies for recombination and selection to reinforce the exploitation of promising solutions (Figure 3).

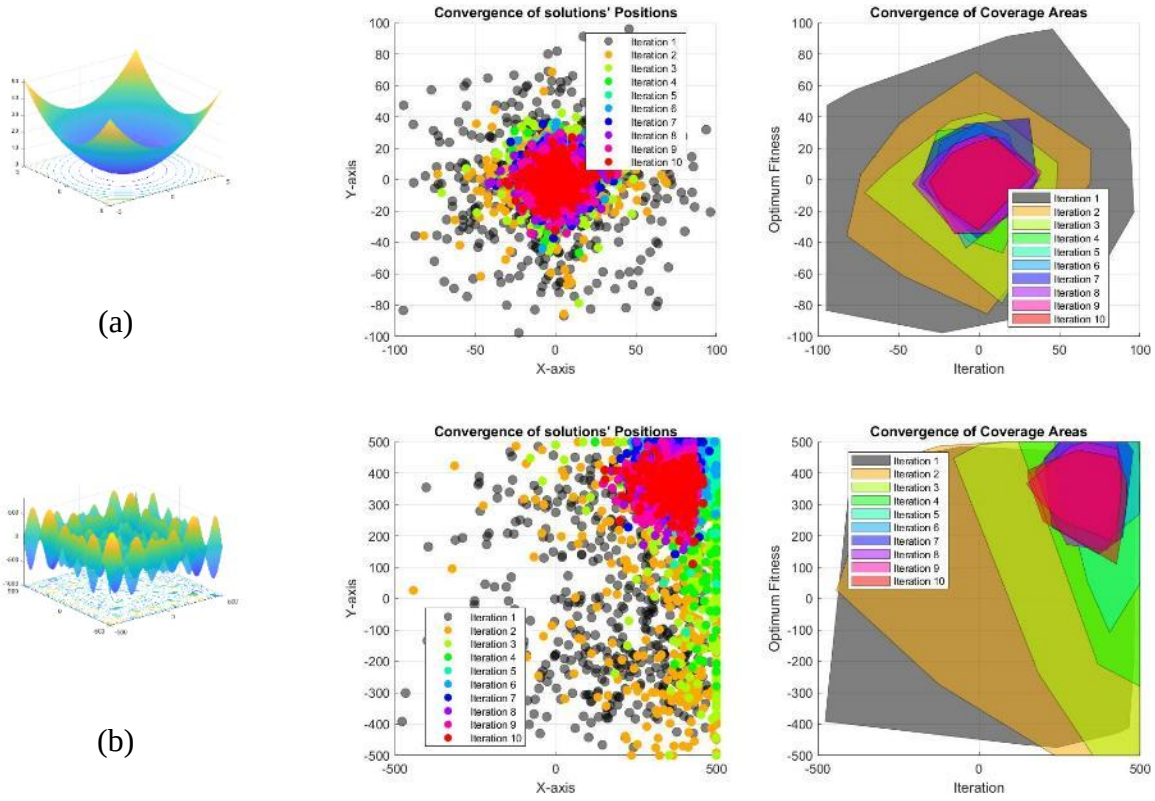

Figure 1. Particle Swarm Optimization search behavior, average computational time 0.092s

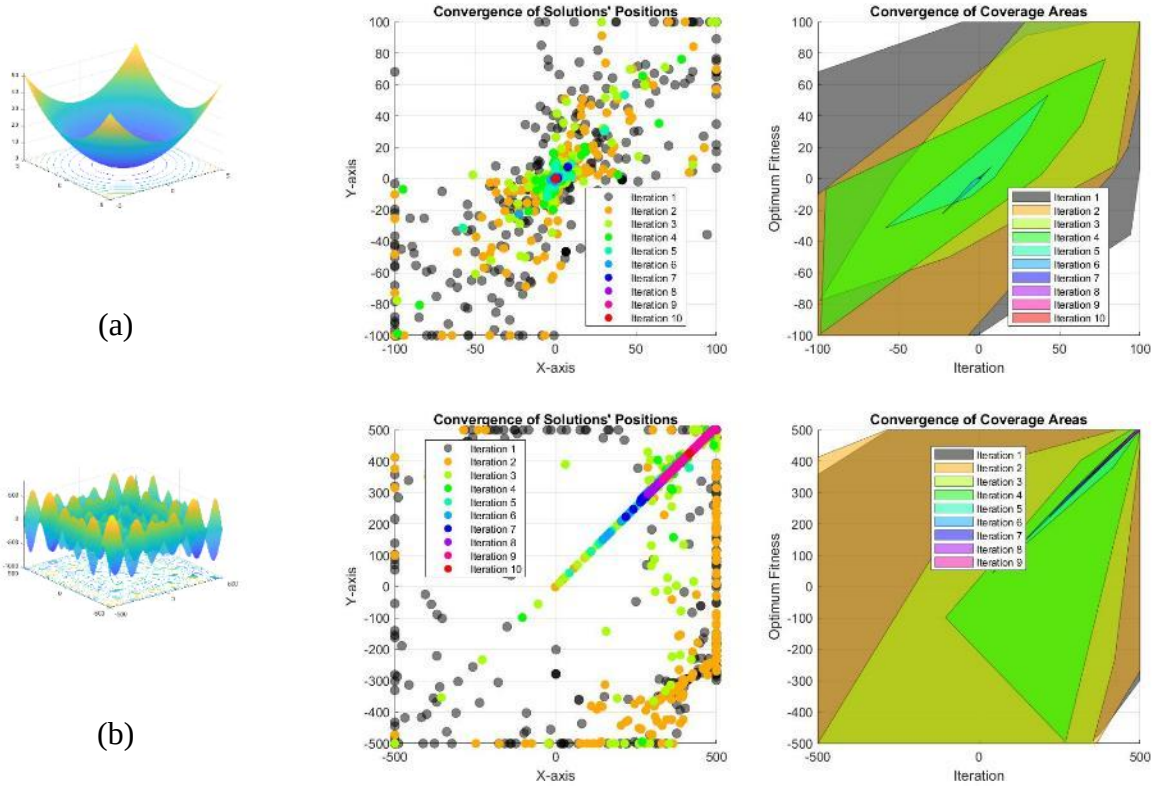

Figure 2. Grey Wolf Optimizer search behavior, average computational time 0.081s

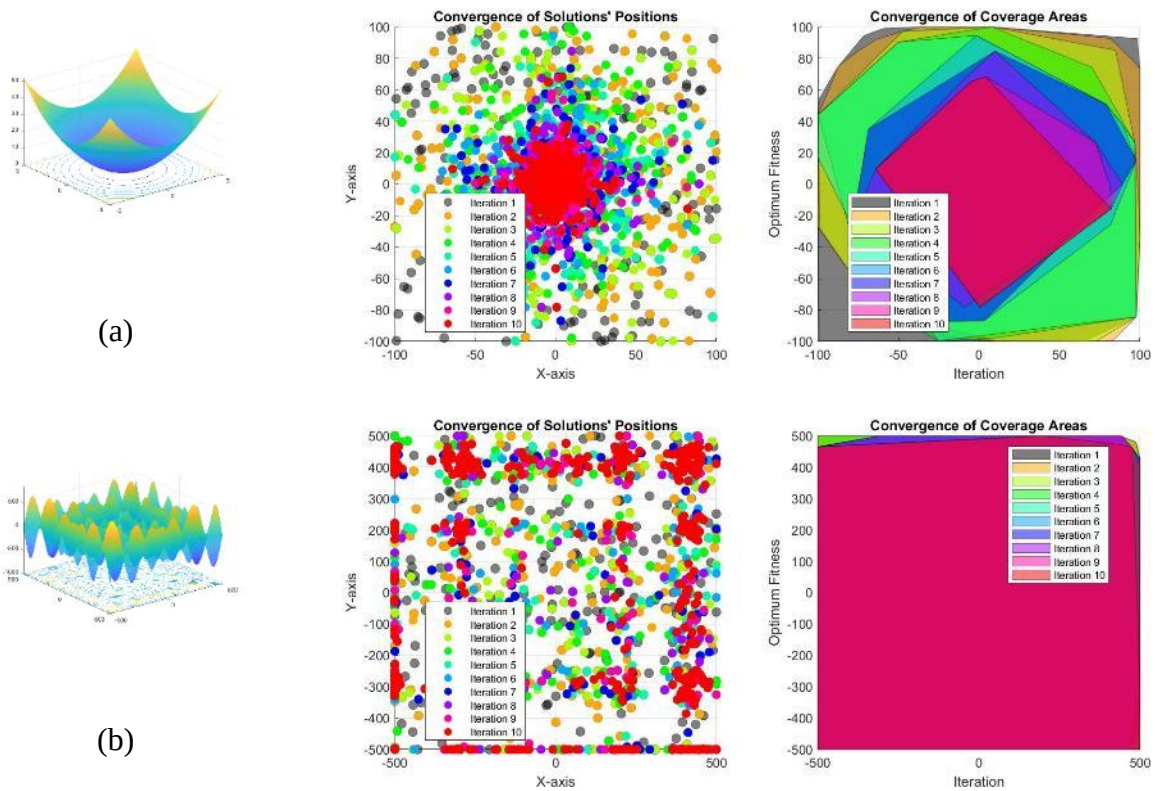

Figure 3. Differential Evolution search behavior, average computational time 0.085s

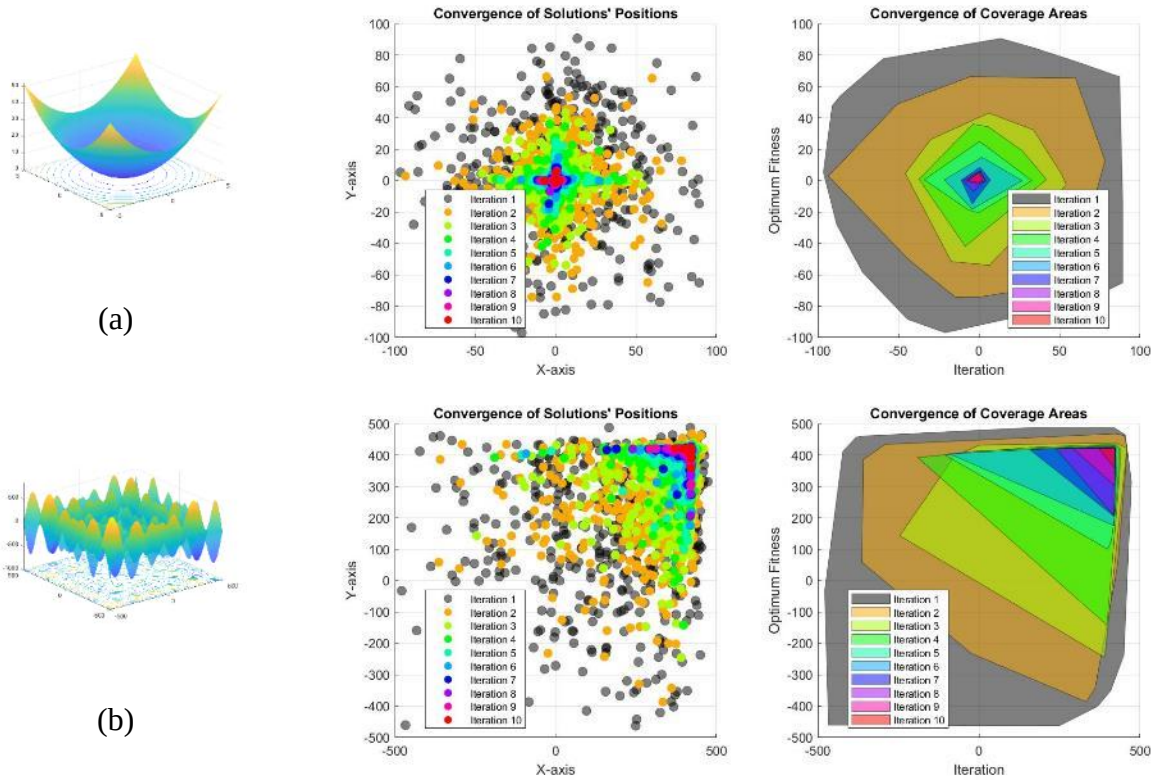

Figure 4. Teaching-Learning-Based Optimization search behavior, average computational time 0.181s

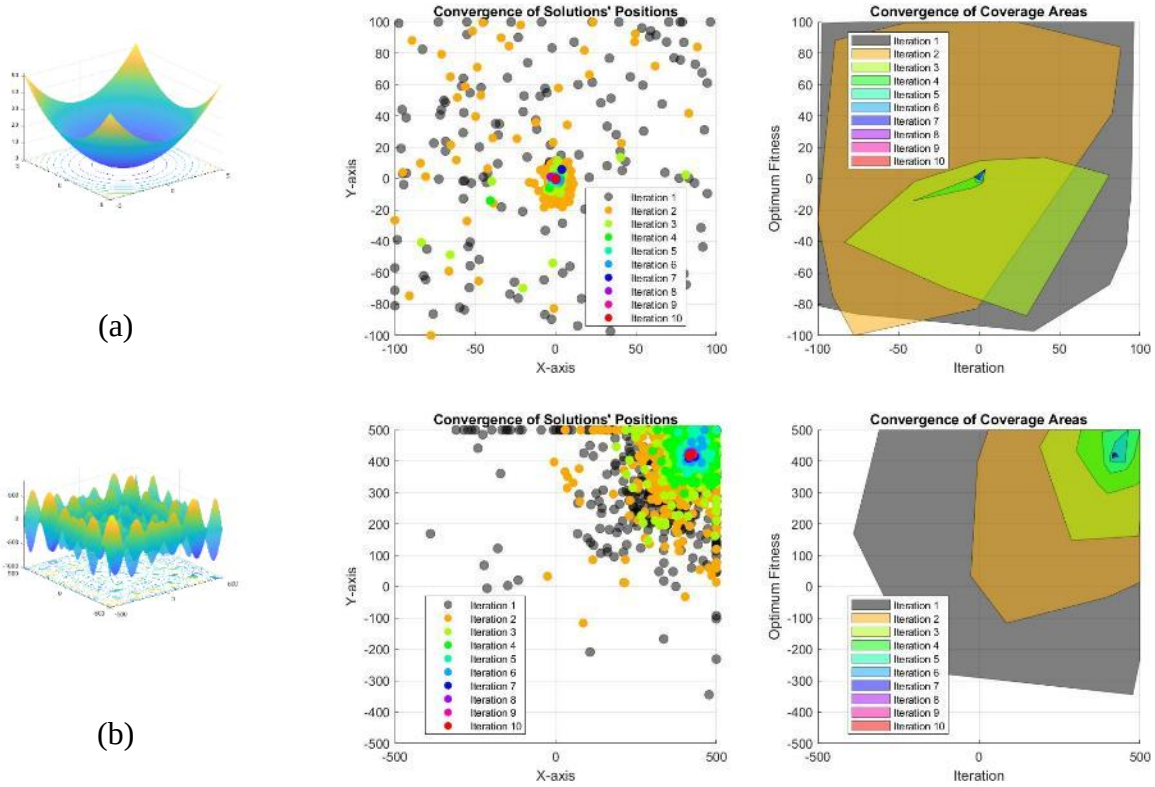

Figure 5. YUKI Algorithm search behavior, average computational time 0.083s

During the teaching phase of the TLBO algorithm, individuals with better solutions "teach" or share their knowledge with those who have poorer solutions. This knowledge transfer encourages exploitation, as it helps the population converge towards better solutions. In the learning phase, individuals learn from their peers and adjust their solutions based on the information received. This phase fosters exploration, as individuals incorporate new ideas and approaches from others into their solutions, potentially leading to the discovery of novel and improved solutions, these rules encourage the search behavior shown in Figure 4. YUKI algorithm performs both tasks simultaneously by focusing part of the population to search around the best solution found so far and the rest to explore outside the local search area, which develops into a powerful search behavior as seen in Figure 5.

9. Conclusion

This paper provides a comprehensive exploration of the core components, techniques, and strategies inherent to metaheuristic algorithms. It underscores the pivotal role played by the delicate equilibrium between exploration and exploitation in optimizing solutions. Moreover, it conducts a discerning comparison between the merits and limitations of metaheuristic algorithms and gradient-based optimization methods, offering insights into their applicability within diverse problem domains and under varying resource constraints.

Examining a diverse range of solution-generation techniques across various algorithms emphasizes their remarkable versatility and adaptability. These methodologies prove highly effective in tackling a broad spectrum of optimization challenges. Additionally, the paper examines the specific strategies employed by these algorithms. Adding visual representations of how certain metaheuristic algorithms behave enhances our grasp of how they work in both research and real-world applications. These visuals offer a practical insight into the processes involved.

In the future direction of metaheuristic optimization algorithms, it is imperative to address and overcome the identified limitations to enhance their efficacy. Encouraging a shift towards simplicity and a clear theoretical foundation may prove beneficial in mitigating challenges. By emphasizing a more straightforward and theoretically grounded approach, future developments in metaheuristic algorithms can strive to provide solutions that are not only effective in addressing complex optimization challenges but also more interpretable and easier to configure.

References

Abdel-Basset, M., Abdel-Fatah, L., & Sangaiah, A. K. (2018). Metaheuristic algorithms: A comprehensive review. *Computational Intelligence for Multimedia Big Data on the Cloud with Engineering Applications*, 185-231.

Abualigah, L., Diabat, A., Mirjalili, S., Abd Elaziz, M., & Gandomi, A. H. (2021). The arithmetic optimization algorithm. *Computer Methods in Applied Mechanics and Engineering*, *376*, Article 113609.

Adam, S. P., Alexandropoulos, S.-A. N., Pardalos, P. M., & Vrahatis, M. N. (2019). No free lunch theorem: A review. *Approximation and Optimization: Algorithms, Complexity and Applications*, 57-82.

Agrawal, P., Abutarboush, H. F., Ganesh, T., & Mohamed, A. W. (2021). Metaheuristic algorithms on feature selection: A survey of one decade of research (2009-2019). *Ieee Access*, *9*, 26766-26791.

Akay, B., Karaboga, D., & Akay, R. (2022). A comprehensive survey on optimizing deep learning models by metaheuristics. *Artificial Intelligence Review*, 1-66.

Al Ali, M., Shimoda, M., Benaissa, B., & Kobayashi, M. (2023). Non-parametric optimization for lightweight and high heat conductive structures under convection using metaheuristic structure binary-distribution method. *Applied Thermal Engineering*, Article 121124.

Al Thobiani, F., Khatir, S., Benaissa, B., Ghandourah, E., Mirjalili, S., & Wahab, M. A. (2022). A hybrid PSO and Grey Wolf Optimization algorithm for static and dynamic crack identification. *Theoretical and Applied Fracture Mechanics*, *118*, Article 103213.

Alia, O. M., & Mandava, R. (2011). The variants of the harmony search algorithm: An overview. *Artificial Intelligence Review*, *36*, 49-68.

Alorf, A. (2023). A survey of recently developed metaheuristics and their comparative analysis. *Engineering Applications of Artificial Intelligence*, *117*, Article 105622.

Amoura, N., Benaissa, B., Al Ali, M., & Khatir, S. (2023). *Deep neural network and YUKI algorithm for inner damage characterization based on elastic boundary displacement BT - Proceedings of the International Conference of Steel and Composite for Engineering Structures*. Cham, Switzerland: Springer International Publishing.

Aranha, C., Camacho Villalón, C. L., Campelo, F., Dorigo, M., Ruiz, R., Sevaux, M., … Stützle, T. (2022). Metaphor-based metaheuristics, a call for action: The elephant in the room. *Swarm Intelligence*, *16*(1), 1-6.

Azevedo, F., Vale, Z. A., Oliveira, P. B. M., & Khodr, H. M. (2010). A long-term risk management tool for electricity markets using swarm intelligence. *Electric Power Systems Research*, *80*(4), 380-389.

Bai, J., Li, Y., Zheng, M., Khatir, S., Benaisa, B., Abualigah, L., & Wahab, M. A. (2023). A sinh cosh optimizer. *Knowledge-Based Systems*, Article 111081.

Banks, A., Vincent, J., & Anyakoha, C. (2007). A review of particle swarm optimization. Part I: Background and development. *Natural Computing*, *6*, 467-484.

Beasley, D., Bull, D. R., & Martin, R. R. (1993). An overview of genetic algorithms: Part 1, fundamentals. *University Computing*, *15*(2), 56-69.

Beheshti, Z., & Shamsuddin, S. M. H. (2013). A review of population-based meta-heuristic algorithms. *International Journal of Advances in Soft Computing and its Applications*, *5*(1), 1-35.

Benaissa, B., Hocine, N. A., Khatir, S., Riahi, M. K., & Mirjalili, S. (2021). YUKI Algorithm and POD-RBF for elastostatic and dynamic crack identification. *Journal of Computational Science*, *55*, Article 101451.

Benaissa, B., Kobayashi, M., Kinoshita, K., & Takenouchi, H. (2023). A novel approach for individual design perception based on fuzzy inference system training with YUKI algorithm. *Axioms*, *12*(10), Article 904.

Bertsimas, D., & Tsitsiklis, J. (1993). Simulated annealing. *Statistical Science*, *8*(1), 10-15.

Birattari, M., & Kacprzyk, J. (2009). *Tuning metaheuristics: A machine learning perspective* (Vol. 197). Heidelberg, Berlin: Springer.

Blackwell, T., & Kennedy, J. (2018). Impact of communication topology in particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, *23*(4), 689-702.

Blum, C., & Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys (CSUR)*, *35*(3), 268-308.

Bolufé-Röhler, A., & Chen, S. (2020). A multi-population exploration-only exploitation-only hybrid on CEC-2020 single objective bound constrained problems. *2020 IEEE Congress on Evolutionary Computation (CEC)*, 1-8.

Busetti, F. (2003). *Simulated annealing overview*. Retrieved October 10, 2022, from www.Geocities.Com/Francorbusetti/Saweb.Pdf,4

Camacho-Villalón, C. L., Dorigo, M., & Stützle, T. (2022). An analysis of why cuckoo search does not bring any novel ideas to optimization. *Computers & Operations Research*, *142*, Article 105747.

Camacho-Villalón, C. L., Dorigo, M., & Stützle, T. (2023). Exposing the grey wolf, moth-flame, whale, firefly, bat, and antlion algorithms: Six misleading optimization techniques inspired by bestial metaphors. *International Transactions in Operational Research*, *30*(6), 2945-2971.

Castillo, J. C., & Segura, C. (2020). Differential evolution with enhanced diversity maintenance. *Optimization Letters*, *14*, 1471-1490.

Chakraborty, S., Sharma, S., Saha, A. K., & Chakraborty, S. (2021). SHADE-WOA: A metaheuristic algorithm for global optimization. *Applied Soft Computing*, *113*, Article 107866.

Chica, M., Pérez, A. A. J., Cordon, O., & Kelton, D. (2017). *Why simheuristics? Benefits, limitations, and best practices when combining metaheuristics with simulation*. Retrieved October 10, 2022, from http://dx.doi.org/10.2139/ssrn.2919208

Chopard, B., & Tomassini, M. (2018). Performance and limitations of metaheuristics. *An Introduction to Metaheuristics for Optimization*, 191-203.

Cuevas, E., Diaz, P., Camarena, O., Cuevas, E., Diaz, P., & Camarena, O. (2021). Experimental analysis between exploration and exploitation. *Metaheuristic Computation: A Performance Perspective*, 249-269.

Dalla, C. E. R., da Silva, W. B., Dutra, J. C. S., & Colaço, M. J. (2021). A comparative study of gradient-based and meta heuristic optimization methods using Griewank benchmark function. *Brazilian Journal of Development*, *7*(6), 55341-55350.

Dang, N. R., Dardinier, T., Doerr, B., Izacard, G., & Nogneng, D. (2018). A new analysis method for evolutionary optimization of dynamic and noisy objective functions. *Proceedings of the Genetic and Evolutionary Computation Conference*, 1467-1474.

Daoud, M. S., Shehab, M., Al-Mimi, H. M., Abualigah, L., Zitar, R. A., & Shambour, M. K. Y. (2023). Gradient-Based Optimizer (GBO): A review, theory, variants, and applications. *Archives of Computational Methods in Engineering*, *30*(4), 2431-2449.

Das, S., Mullick, S. S., & Suganthan, P. N. (2016). Recent advances in differential evolution - An updated survey. *Swarm and Evolutionary Computation*, *27*, 1-30.

Dehghani, M., Montazeri, Z., Trojovská, E., & Trojovský, P. (2023). Coati Optimization Algorithm: A new bio-inspired metaheuristic algorithm for solving optimization problems. *Knowledge-Based Systems*, *259*, Article 110011.

Doğan, B., & Ölmez, T. (2015). A new metaheuristic for numerical function optimization: Vortex Search algorithm. *Information Sciences*, *293*, 125-145.

Dokeroglu, T., Sevinc, E., Kucukyilmaz, T., & Cosar, A. (2019). A survey on new generation metaheuristic algorithms. *Computers & Industrial Engineering*, *137*, Article 106040.

Dong, X., & Liu, X. (2021). Multi-objective optimization of heat transfer in microchannel for non-Newtonian fluid. *Chemical Engineering Journal*, *412*, Article 128594.

Dorigo, M., & Blum, C. (2005). Ant colony optimization theory: A survey. *Theoretical Computer Science*, *344*(2/3), 243-278.

Dorigo, M., & Stützle, T. (2003). The ant colony optimization metaheuristic: Algorithms, applications, and advances. *Handbook of Metaheuristics*, 250-285.

Dorigo, M., & Stützle, T. (2019). *Ant colony optimization: Overview and recent advances*. Heidelberg, Berlin: Springer.

Dorigo, M., Birattari, M., & Stutzle, T. (2006). Ant colony optimization. *IEEE Computational Intelligence Magazine*, *1*(4), 28-39.

Du, K.-L., & Swamy, M. N. S. (2016). Search and optimization by metaheuristics. *Techniques and Algorithms Inspired by Nature*, 1-10.

Eiben, A. E., & Smit, S. K. (2011). Parameter tuning for configuring and analyzing evolutionary algorithms. *Swarm and Evolutionary Computation*, *1*(1), 19-31.

Engelbrecht, A. P. (2013). Particle swarm optimization: Global best or local best? *2013 BRICS Congress on Computational Intelligence and 11th Brazilian Congress on Computational Intelligence*, 124-135.

Gallagher, M. (2016). Towards improved benchmarking of black-box optimization algorithms using clustering problems. *Soft Computing*, *20*, 3835-3849.

Gandomi, A. H., & Alavi, A. H. (2012). Krill herd: A new bio-inspired optimization algorithm. *Communications in Nonlinear Science and Numerical Simulation*, *17*(12), 4831-4845.

Gandomi, A. H., Yang, X.-S., & Alavi, A. H. (2013). Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems. *Engineering with Computers*, *29*, 17-35.

Gandomi, A. H., Yang, X.-S., Talatahari, S., & Alavi, A. H. (2013). Metaheuristic algorithms in modeling and optimization. *Metaheuristic Applications in Structures and Infrastructures*, *1*, 1-24.

Gavrilas, M. (2010). Heuristic and metaheuristic optimization techniques with application to power systems. *Proceedings of the 12th WSEAS International Conference on Mathematical Methods and Computational Techniques in Electrical Engineering*, 95-103.

Geem, Z. W. (2010). *Recent advances in harmony search algorithm*. doi:10.1007/978-3-642-04317-8

Geem, Z. W., Kim, J. H., & Loganathan, G. V. (2001). A new heuristic optimization algorithm: Harmony search. *Simulation*, *76*(2), 60-68.

Gendreau, M., & Potvin, J.-Y. (2005). Tabu search. *Search methodologies: Introductory tutorials in optimization and decision support techniques*, 165-186.

Ghandourah, E., Khatir, S., Banoqitah, E. M., Alhawsawi, A. M., Benaissa, B., & Wahab, M. A. (2023). Enhanced ANN predictive model for composite pipes subjected to low-velocity impact loads. *Buildings*, *13*(4), Article 973.

Griffis, S. E., Bell, J. E., & Closs, D. J. (2012). Metaheuristics in logistics and supply chain management. *Journal of Business Logistics*, *33*(2), 90-106.

Gutjahr, W. J. (2009). Convergence analysis of metaheuristics. In *Matheuristics: Hybridizing metaheuristics and mathematical programming* (pp. 159-187). Heidelberg, Berlin: Springer.

Halim, A. H., Ismail, I., & Das, S. (2021). Performance assessment of the metaheuristic optimization algorithms: An exhaustive review. *Artificial Intelligence Review*, *54*, 2323-2409.

Hertz, A., Taillard, E., & De Werra, D. (1995). A tutorial on tabu search. *Proceeding of Giornate Di Lavoro AIRO*, *95*, 13-24.

Holland, J. H. (1992). Genetic algorithms. *Scientific American*, *267*(1), 66-73.

Huang, C., Li, Y., & Yao, X. (2019). A survey of automatic parameter tuning methods for metaheuristics. *IEEE Transactions on Evolutionary Computation*, *24*(2), 201-216.

Hussain, K., Salleh, M. N. M., Cheng, S., & Naseem, R. (2017). Common benchmark functions for metaheuristic evaluation: A review. *JOIV: International Journal on Informatics Visualization*, *1*(4/2), 218-223.

Hussain, K., Salleh, M. N. M., Cheng, S., & Shi, Y. (2019). On the exploration and exploitation in popular swarm-based metaheuristic algorithms. *Neural Computing and Applications*, *31*, 7665-7683.

Issa, M., & Mostafa, Y. (2022). Gradient-based optimizer for structural optimization problems. In *Integrating meta-heuristics and machine learning for real-world optimization problems* (pp. 461-480). Heidelberg, Berlin: Springer.

Jaszkiewicz, A. (2001). *Multiple objective metaheuristic algorithms for combinatorial optimization*. Poland: Politechniki PoznaĘąnskiej.

Karaboga, D. (2010). Artificial bee colony algorithm. *Scholarpedia*, *5*(3), Article 6915.

Kareem, S. W. (2022). A nature-inspired metaheuristic optimization algorithm based on Crocodiles Hunting Search (CHS). *International Journal of Swarm Intelligence Research (IJSIR)*, *13*(1), 1-23.

Kaveh, A. (2014). *Advances in metaheuristic algorithms for optimal design of structures*. Heidelberg, Berlin: Springer.

Kaveh, A. (2017). *Applications of metaheuristic optimization algorithms in civil engineering*. Heidelberg, Berlin: Springer.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *Proceedings of ICNN'95-International Conference on Neural Networks*, *4*, 1942-1948.

Khanduja, N., & Bhushan, B. (2021). Recent advances and application of metaheuristic algorithms: A survey (2014-2020). In *Metaheuristic and evolutionary computation: Algorithms and applications* (pp. 207-228).

Khatir, A., Capozucca, R., Khatir, S., Magagnini, E., Benaissa, B., Le, C. T., & Wahab, M. A. (2023). A new hybrid PSO-YUKI for double crack identification in CFRP cantilever beam. *Composite Structures*, Article 116803.

Kobayashi, M. (2019). Multi-objective aesthetic design optimization for minimizing the effect of variation in customer Kansei. *Computer-Aided Design and Applications*, *17*(4), 690-698.

Kobayashi, M. (2020). *Multi-objective aesthetic design optimization for minimizing the effect of variation in customer Kansei*. Abingdon, UK: Taylor and Francis.

Krishnanand, K. N., & Ghose, D. (2009). Glowworm swarm optimisation: A new method for optimising multi-modal functions. *International Journal of Computational Intelligence Studies*, *1*(1), 93-119.

Kuo, S.-Y., & Chou, Y.-H. (2021). Building intelligent moving average-based stock trading system using metaheuristic algorithms. *IEEE Access*, *9*, 140383-140396.

Kuyu, Y. Ç., & Vatansever, F. (2021). Advanced metaheuristic algorithms on solving multimodal functions: Experimental analyses and performance evaluations. *Archives of Computational Methods in Engineering*, 1-13.

Lampinen, J. A., Price, K. V., & Storn, R. M. (2005). *Differential evolution.* Heidelberg, Berlin: Springer.

Le, M. H., To, S. T., Theraulaz, G., Wahab, M. A., & Le, C. T. (2023). Termite life cycle optimizer. *Expert Systems with Applications*, *213*, Article 119211.

Lessmann, S., Caserta, M., & Arango, I. M. (2011). Tuning metaheuristics: A data mining based approach for particle swarm optimization. *Expert Systems with Applications*, *38*(10), 12826-12838.

Mirjalili, S. (2016a). Dragonfly algorithm: A new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. *Neural Computing and Applications*, *27*, 1053-1073.

Mirjalili, S. (2016b). SCA: A sine cosine algorithm for solving optimization problems. *Knowledge-Based Systems*, *96*, 120-133.

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software*, *69*, 46-61.

Morales-Castañeda, B., Zaldivar, D., Cuevas, E., Fausto, F., & Rodríguez, A. (2020). A better balance in metaheuristic algorithms: Does it exist? *Swarm and Evolutionary Computation*, *54*, Article 100671.

Mucherino, A., & Seref, O. (2007). Monkey search: A novel metaheuristic search for global optimization. *AIP Conference Proceedings*, *953*(1), 162-173.

Nanda, S. J., & Panda, G. (2014). A survey on nature inspired metaheuristic algorithms for partitional clustering. *Swarm and Evolutionary Computation*, *16*, 1-18.

Naruei, I., Keynia, F., & Molahosseini, A. S. (2022). Hunter-prey optimization: Algorithm and applications. *Soft Computing*, *26*(3), 1279-1314.

Odili, J. B. (2018). The dawn of metaheuristic algorithms. *International Journal of Software Engineering and Computer Systems*, *4*(2), 49-61.

Oftadeh, R., Mahjoob, M. J., & Shariatpanahi, M. (2010). A novel meta-heuristic optimization algorithm inspired by group hunting of animals: Hunting search. *Computers & Mathematics with Applications*, *60*(7), 2087-2098.

Omidvar, M. N., Li, X., & Yao, X. (2021). A review of population-based metaheuristics for large-scale black-box global optimization - Part I. *IEEE Transactions on Evolutionary Computation*, *26*(5), 802-822.

Osaba, E., Villar-Rodriguez, E., Del Ser, J., Nebro, A. J., Molina, D., LaTorre, A., … Herrera, F. (2021). A tutorial on the design, experimentation and application of metaheuristic algorithms to real-world optimization problems. *Swarm and Evolutionary Computation*, *64*, Article 100888.

Parouha, R. P., & Verma, P. (2021). State-of-the-art reviews of meta-heuristic algorithms with their novel proposal for unconstrained optimization and applications. *Archives of Computational Methods in Engineering*, *28*, 4049-4115.

Peres, F., & Castelli, M. (2021). Combinatorial optimization problems and metaheuristics: Review, challenges, design, and development. *Applied Sciences*, *11*(14), Article 6449.

Pirim, H., Eksioglu, B., & Bayraktar, E. (2008). *Tabu search: A comparative study*. In W. Jaziri (Ed.), *Tabu search.* doi:10.5772/5637

Price, K., Storn, R. M., & Lampinen, J. A. (2006). *Differential evolution: A practical approach to global optimization*. Heidelberg, Berlin: Springer Science & Business Media.

Rahmani, M., Eraqi, M. K., & Nikoomaram, H. (2019). Portfolio optimization by means of Meta heuristic algorithms. *Advances in Mathematical Finance and Applications*, *4*(4), 83-97.

Rao, R. (2016). Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems. *International Journal of Industrial Engineering Computations*, *7*(1), 19-34.

Rao, R. V., Savsani, V. J., & Vakharia, D. P. (2011). Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems. *Computer-Aided Design*, *43*(3), 303-315.

Rashedi, E., Nezamabadi-Pour, H., & Saryazdi, S. (2009). GSA: A gravitational search algorithm. *Information Sciences*, *179*(13), 2232-2248.

Ribeiro, C. C., Hansen, P., Maniezzo, V., & Carbonaro, A. (2002). Ant colony optimization: An overview. *Essays and Surveys in Metaheuristics*, 469-492.

Ruder, S. (2016). *An overview of gradient descent optimization algorithms*. Retrieved October 10, 2022, from https://arxiv.org/pdf/1609.04747.pdf

Sahali, M. A., Aini, A., Bouzit, L., Himed, L., & Benaissa, B. (2023). Experimental modeling and multi-objective optimization of friction stir welding parameters of AA 3004 aluminum alloy. *The International Journal of Advanced Manufacturing Technology*, *124*(3), 1229-1244.

Sala, R., & Müller, R. (2020). Benchmarking for metaheuristic black-box optimization: Perspectives and open challenges. *2020 IEEE Congress on Evolutionary Computation (CEC)*, 1-8.

Shirazi, M. I., Khatir, S., Benaissa, B., Mirjalili, S., & Wahab, M. A. (2023). Damage assessment in laminated composite plates using modal Strain Energy and YUKI-ANN algorithm. *Composite Structures*, *303*, Article 116272. doi:10.1016/j.compstruct.2022.116272

Singh, G., & Singh, A. (2014). Comparative study of krill herd, firefly and cuckoo search algorithms for unimodal and multimodal optimization. *International Journal of Intelligent Systems and Applications in Engineering*, *2*(3), 26-37.

Singh, P., & Choudhary, S. K. (2021). Introduction: Optimization and metaheuristics algorithms. *Metaheuristic and Evolutionary Computation: Algorithms and Applications*, 3-33.

Sivanandam, S. N., & Deepa, S. N. (2008). *Genetic algorithms*. Heidelberg, Berlin: Springer.

Soler-Dominguez, A., Juan, A. A., & Kizys, R. (2017). A survey on financial applications of metaheuristics. *ACM Computing Surveys (CSUR)*, *50*(1), 1-23.

Sörensen, K. (2015). Metaheuristics - The metaphor exposed. *International Transactions in Operational Research*, *22*(1), 3-18.

Syafruddin, W. A., Köppen, M., & Benaissa, B. (2018). Does the Jaya algorithm really need no parameters? *10th International Joint Conference on Computational Intelligence*, 264-268.

Talbi, E., Basseur, M., Nebro, A. J., & Alba, E. (2012). Multi-objective optimization using metaheuristics: Non-standard algorithms. *International Transactions in Operational Research*, *19*(1/2), 283-305.

Tarantilis, C. D., Ioannou, G., Kiranoudis, C. T., & Prastacos, G. P. (2005). Solving the open vehicle routeing problem via a single parameter metaheuristic algorithm. *Journal of the Operational Research Society*, *56*, 588-596.

Tilahun, S. L., & Ong, H. C. (2015). Prey-predator algorithm: A new metaheuristic algorithm for optimization problems. *International Journal of Information Technology & Decision Making*, *14*(6), 1331-1352.

To, S. T., Le, H. M., Wahab, M. A., & Le, C. T. (2022). An efficient planet optimization algorithm for solving engineering problems. *Scientific Reports*, *12*(1), Article 8362.

Tovey, C. A. (2018). Nature-inspired heuristics: Overview and critique. *Recent Advances in Optimization and Modeling of Contemporary Problems*, 158-192.

Van Laarhoven, P. J. M., & Aarts, E. H. L. (1987). *Simulated annealing*. Heidelberg, Berlin: Springer.

Velasco, L., Guerrero, H., & Hospitaler, A. (2023). A literature review and critical analysis of metaheuristics recently developed. *Archives of Computational Methods in Engineering*, 1-22.

Wang, D., Tan, D., & Liu, L. (2018). Particle swarm optimization algorithm: An overview. *Soft Computing*, *22*, 387-408.

Wong, W. K., & Ming, C. I. (2019). A review on metaheuristic algorithms: Recent trends, benchmarking and applications. *2019 7th International Conference on Smart Computing & Communications (ICSCC)*, 1-5.

Xu, J., & Zhang, J. (2014). Exploration-exploitation tradeoffs in metaheuristics: Survey and analysis. *Proceedings of the 33rd Chinese Control Conference*, 8633-8638.

Yang, X., & Gandomi, A. H. (2012). Bat algorithm: A novel approach for global engineering optimization. *Engineering Computations*, *29*(5), 464-483.

Yang, X.-S. (2009). Harmony search as a metaheuristic algorithm. *Music-Inspired Harmony Search Algorithm: Theory and Applications*, 1-14.

Yang, X.-S. (2010). *Engineering optimization: An introduction with metaheuristic applications*. Hoboken, NJ: John Wiley & Sons.

Yang, X.-S. (2011a). Metaheuristic optimization: Algorithm analysis and open problems. *International Symposium on Experimental Algorithms*, 21-32.

Yang, X.-S. (2011b). Metaheuristic optimization. *Scholarpedia*, *6*(8), Article 11472.

Yang, X.-S., & Slowik, A. (2020). Firefly algorithm. In *Swarm intelligence algorithms* (pp. 163-174). Boca Raton, FL: CRC Press.

Yazdani, S., Nezamabadi-Pour, H., & Kamyab, S. (2014). A gravitational search algorithm for multimodal optimization. *Swarm and Evolutionary Computation*, *14*, 1-14.

Zhang, J. (2019). *Gradient descent based optimization algorithms for deep learning models training.* Retrieved October 10, 2022, from https://arxiv.org/pdf/1903.03614.pdf

Zitouni, F., Harous, S., & Maamri, R. (2020). The solar system algorithm: A novel metaheuristic method for global optimization. *IEEE Access*, *9*, 4542-4565.

Zitouni, F., Harous, S., Belkeram, A., & Hammou, L. E. B. (2022). The archerfish hunting optimizer: A novel metaheuristic algorithm for global optimization. *Arabian Journal for Science and Engineering*, *47*(2), 2513-2553.