Comparison of Metaheuristic Optimization Methods: Particle Swarm, Differential Evolution, Simulated Annealing, and Tabu Search

Zeliha Ilgın Güven

November 2025

1 Introduction

Optimization represents a fundamental concept that embodies the pursuit of the best possible outcome, both in natural laws and in human activities. In the broadest sense, it can be defined as the process of minimizing or maximizing a system's output in order to obtain the most suitable (optimal) solution in a given situation. In today's world, where resources are becoming increasingly scarce, the necessity of utilizing available resources—such as time, energy, labor, materials, or capital—in the most efficient way further emphasizes the importance of optimization. In this respect, optimization is not only a mathematical process but also a universal decision-making approach that aims to achieve maximum output from limited inputs. Furthermore, the literature highlights that optimization has a wide range of applications across various fields and that algorithms developed in this area have produced successful results in many problem domains (Yang, 2012).

Optimization problems arise in a broad spectrum of fields, from engineering to chemistry. For example, in analytical chemistry, determining the most appropriate experimental conditions or instrument settings to obtain the maximum amount of information with a limited number of experiments is a classical example of an optimization problem (Wehrens Buydens, 2000). In such cases, response surface models or simplex optimization techniques are frequently used; however, these methods yield effective results only when certain assumptions are satisfied (Wehrens Buydens, 2000). This demonstrates not only the strengths of classical optimization approaches but also their limitations in dealing with complex and nonlinear problems.

Nevertheless, classical optimization methods—such as linear programming, gradient-based, or deterministic algorithms—typically suffer from fundamental limitations, including getting trapped in local optima, failing to find the global optimum, and low efficiency in high-dimensional problems. In particular, gradient-based algorithms may become stuck in local minima when applied to complex, multimodal objective functions, thus failing to reach the global solution. (Wang Chen, 2013)Moreover, their dependency on derivative information restricts their applicability to discontinuous or noisy functions.

To overcome such limitations, metaheuristic approaches have been developed, featuring flexible search strategies that incorporate stochastic components. A metaheuristic algorithm can be broadly defined as an iterative generation processthat intelligently combines different concepts to explore and exploit the search space, thereby guiding subordinate heuristic methods (Said, Mahmoud, El-Horbaty, 2014). In this process, learning strategies organize information structures to enable the efficient discovery of near-optimal solutions . The iterative computational mechanism at the core of metaheuristic algorithms successively improves candidate solutions based on a given quality criterion(Wang Chen, 2013)

As a result, metaheuristic algorithms provide a powerful alternative to classical methods, as they can escape local minima through stochasticity, perform global searches, and produce near-optimal solutions efficiently in large and complex search spaces.

In this context, Particle Swarm Optimization (PSO), Differential Evolution (DE), Simulated Annealing (SA), and Tabu Search (TS) stand out as prominent metaheuristic methods used in fields such as engineering, geophysics (Balkaya, Göktürkler, Ekinci, Karaoğlan, 2014), and data science (Cheng et al., 2016). Inspired by natural processes or cognitive behaviors, these algorithms establish a balance between exploration and exploitation in complex solution spaces to generate effective solutions.

This report, based on a literature-driven approach, aims to analyze these four metaheuristic methods in detail in terms of their structural characteristics, search mechanisms, and application domains. The analysis seeks to reveal under what conditions, why, and how each algorithm is effective, explaining the appropriate usage scenarios and comparative advantages of these methods in solving optimization problems.

All code examples, algorithm simulations, and figure generation scripts used in this report are publicly available in the GitHub repository with the original version of the report : Github: https://github.com/zelihaguven/Optimization-Algorithms-Report

2 Approach

In this section, the four metaheuristic algorithms analyzed in the report—PSO, DE, SA, and TS—are discussed.

2.1 Particle Swarm Optimization (PSO)

Particle Swarm Optimization is a population-based metaheuristic optimization algorithm inspired by the social behavior of bird flocking or fish schooling (Naser Naser, 2024) (Bolufé-Röhler Tamayo-Vera, 2025). Proposed by Kennedy and

Eberhart in 1995, PSO leverages a collection of candidate solutions, known as a swarm, where each individual particle iteratively adjusts its trajectory based on its own best-found position and the global best position found by any particle in the swarm (Nasr et al., 2020). This collaborative search mechanism allows the swarm to collectively explore the search space and converge towards optimal solutions (Okechukwu et al., 2024). This approach distinguishes PSO from other metaheuristics by its reliance on emergent collective intelligence rather than complex genetic operators or simulated physical processes (Cuevas et al., 2024) (Martinez et al., 2020).

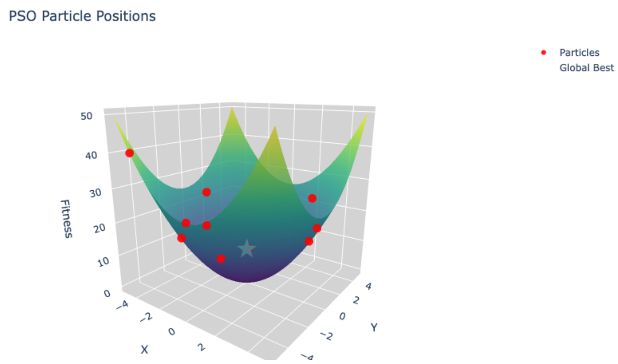

Figure 1: Conceptual visualization of PSO. Red dots represent particles exploring the fitness landscape, and the white star represents the global best solution. This visualization illustrates the main idea of particle movement and global best concept.

2.1.1 Operational Mechanism

The operational mechanism of PSO involves each particle maintaining its current position, velocity, and the best position it has ever achieved (Amini Fathian, 2020). At each iteration, the particle's velocity is updated, incorporating stochastic components that balance exploration and exploitation, and is based on its personal best and the swarm's global best positions (Sarhani et al., 2022; Tomar et al., 2024). This velocity update rule integrates three main components: the particle's previous velocity, a cognitive component guiding it back to its own best-found position, and a social component pulling it towards the swarm's global best position (Zambrano-Gutierrez et al., 2023). This dynamic adjustment of velocity enables particles to explore new regions of the search space while simultaneously converging towards promising areas identified by the swarm (Munirah et al., 2020). Subsequently, the position of each particle is updated by adding this modified velocity to its current position, moving the particle through the multi-dimensional search space (Dahal et al., 2024; Elshaboury, 2021). This iterative process continues until a predefined stopping criterion is met, such as a maximum number of iterations or a satisfactory solution quality (Hubálovský et al., 2023).

2.1.2 Applications

Given its efficacy in navigating complex solution landscapes, PSO has found extensive application across diverse fields, including engineering, finance, and machine learning, particularly in problems requiring continuous optimization (Ajibade Ojeniyi, 2022). For instance, it is frequently employed in optimizing neural network weights, designing intelligent control systems, and scheduling complex tasks (Bhakhar Chhillar, 2024; MUKUNZI et al., 2024). Moreover, its robustness and adaptability make it a significant focus in evolutionary computation research, especially for global optimization and engineering challenges (Cai et al., 2024). The algorithm's population-based nature, which mimics collective intelligence through individual and social learning, enables it to efficiently search multi-dimensional spaces for optimal parameter values (Cheng et al., 2018; Sohail et al., 2014). PSO has been effectively applied to optimize resource allocation, including scenarios like scheduling manufacturing tasks and managing power distribution in wireless networks (Kumar et al., 2023). Its versatility further allows for effective use in various challenging optimization problems such as chemical systems, robotics, and image processing (Yuan Yin, 2015). Beyond these, PSO has demonstrated utility in fields such as energy systems, aerospace engineering, and artificial intelligence, showcasing its broad applicability to diverse research areas (Fang et al., 2023).

2.1.3 Advantages and Disadvantages

One of the primary advantages of PSO is its rapid convergence and computational efficiency, attributable to its minimal parameter requirements and derivative-free nature, which reduces computational costs (Liu et al., 2024). Its ability to effectively balance exploration and exploitation through the interplay of cognitive and social components further contributes to its widespread adoption across various domains (Utkarsh Jain, 2024) (Cai et al., 2024). Conversely, PSO can sometimes suffer from premature convergence in high-dimensional or multimodal landscapes, potentially leading to suboptimal solutions if the balance between exploration and exploitation is not carefully managed (Mogale et al., 2017). To mitigate this, advanced PSO variants introduce mechanisms such as inertia weights or constriction factors to refine the velocity update rule, aiming to prevent particles from converging too quickly to local optima (Weiel et al., 2021) (Hernández-Briones et al., 2024). The inertia weight, for instance, serves to balance global and local search capabilities, often decreasing over time to enhance local search in later iterations (Naderi et al., 2022). This adaptive mechanism, often alongside careful parameter tuning, significantly influences the algorithm's performance and the reliability of the solutions obtained (Innocente Sienz, 2006).

2.2 Differential Evolution(DE)

Differential Evolution is another population-based metaheuristic, distinguished by its unique mutation and crossover operations that leverage vector differences to efficiently explore the search space (Adediran Ameen, 2024). Developed by Storn and Price in 1997, DE operates on a population of candidate solutions, iteratively refining them through mutation, recombination, and selection processes tailored to generate new, improved solutions (Munirah et al., 2020; Zhang et al., 2024). It is particularly effective for continuous optimization problems and is considered an improvement over traditional genetic algorithms due to its self-adaptive nature and simplified operator structure (Wang et al., 2024). Similar to Particle Swarm Optimization, Differential Evolution emerged in the mid-1990s and quickly gained prominence as a robust population-based optimization method (Piotrowski et al., 2023). Its initial simplicity rapidly attracted widespread attention, fostering numerous variants and applications across diverse scientific and engineering disciplines over the past quarter-century (Munirah et al., 2020; Piotrowski et al., 2023).

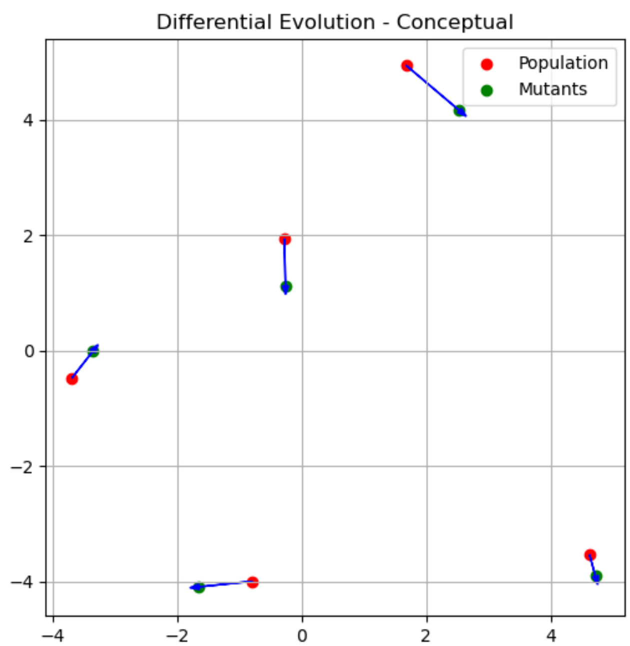

Figure 2: Conceptual visualization of Differential Evolution (DE). Red points represent the current population of candidate solutions, green points represent mutant vectors generated from differences between population members, and blue arrows illustrate mutation directions. This figure demonstrates the basic principle of creating new candidate solutions via vector differences.

2.2.1 Operational Mechanism

The operational mechanism of Differential Evolution involves a cyclical process where, for each individual in the current population, a mutant vector is generated by perturbing a base vector with the scaled difference of two other randomly selected vectors (Li et al., 2024). This newly generated mutant vector then undergoes a crossover operation with the target vector, leading to a trial vector (Schneider et al., 2019). Subsequently, a selection process determines whether the trial vector replaces the target vector in the next generation based on their fitness values, thereby driving the population towards better solutions. This iterative refinement process, emphasizing differential perturbations, allows DE to effectively navigate complex, non-linear search spaces and converge on optimal solutions (Brest Maučec, 2025). The algorithm's efficacy critically depends on the appropriate selection and tuning of hyperparameters, such as population size, mutation factor, and crossover rate, which profoundly influence its performance and convergence characteristics (Jiang et al., 2024; Zhang et al., 2024). These parameters dictate the balance between exploration and exploitation, ensuring the algorithm can thoroughly search the solution space while also rapidly converging towards optimal regions (Sharifi-Noghabi et al., 2016). The standard Differential Evolution algorithm, while robust, involves three key control parameters: population size, mutation scaling factor, and crossover rate, whose optimal values are problem-dependent (Khalfi et al., 2021). Tuning these parameters can be simplified through adaptive control mechanisms, allowing the algorithm to dynamically adjust itself during the search process for faster and more reliable convergence (Charles Parks, 2018).

2.2.2 Applications

Since its inception, Differential Evolution has proven highly valuable across various domains, successfully tackling intricate optimization problems in engineering, data science, and scientific research (Azzam et al., 2024; Bilal et al., 2020). Its broad applicability extends to fields such as signal processing, power systems optimization, and financial modeling (Bilal et al., 2020). Furthermore, DE's adaptability allows for its use in highly specialized domains, including bioinformatics for gene expression analysis and the design of intricate antenna systems (Kononova et al., 2021). This methodology utilizes three core operators—mutation, crossover, and selection—to iteratively generate progressively enhanced solutions across generations (Du et al., 2016).

2.2.3 Advantages and Disadvantages

Differential Evolution fundamentally distinguishes itself from algorithms like Particle Swarm Optimization through its search methodology. While PSO particles assess new solutions in each generation and only update their positions if the new solution is superior, DE adopts a different strategy: an individual evaluates a new solution per generation and updates its position only if the new solution is not worse (Napiórkowski et al., 2022). This characteristic significantly helps DE in avoiding local optima and reduces its susceptibility to initialization, which are common challenges in complex optimization landscapes (Mousavirad et al., 2024). A key benefit of Differential Evolution is its robust performance across a broad spectrum of optimization problems, marked by strong global search capabilities and fewer control parameters compared to other evolutionary algorithms (Farinati Vanneschi, 2024). DE's self-regulating nature and its capacity to manage non-linear, non-convex, and noisy objective functions establish it as a flexible tool for addressing real-world optimization challenges (Zhong et al., 2024). This population-based technique iteratively refines candidate solutions to converge towards an optimal solution for a given problem, setting itself apart with unique mutation, crossover, and selection mechanisms (Li Zhang, 2023). However, DE can sometimes suffer from instability in convergence (Munirah et al., 2020) and may be prone to losing diversity in the population, especially in greedy implementations (Charles Parks, 2018). The performance of DE is also highly dependent on the careful tuning of its control parameters, such as population size, mutation scaling factor, and crossover rate, which can be problem-dependent (Khalfi et al., 2021; Munirah et al., 2020).

2.3 Simulated Annealing (SA)

Simulated Annealing is a metaheuristic optimization algorithm that draws inspiration from the annealing process in metallurgy, where a material is heated and then slowly cooled to increase its crystal size and reduce defects (Das Suganthan, 2010). Applied to optimization problems, this analogy allows the algorithm to probabilistically accept worse solutions, especially during the initial phases of the search. This probabilistic acceptance, governed by a "temperature" parameter that gradually decreases over time, differentiates SA from purely greedy approaches, enabling it to escape local optima and thoroughly explore the solution space (Gift Deza, 2021). This mechanism facilitates a more exhaustive exploration of the solution landscape, gradually transitioning to a more exploitative search as the temperature cools and the probability of accepting suboptimal moves diminishes.

2.3.1 Operational Mechanism

The core mechanism of Simulated Annealing entails transitioning between solutions within the search space, evaluating the objective function at each stage, and making acceptance or rejection decisions for new solutions based on both their quality and the prevailing "temperature" (Tang Wang, 2024). Initially, at elevated temperatures, the algorithm exhibits a higher propensity to accept suboptimal solutions, thereby facilitating extensive exploration of the solution space (Odeyemi Zhang, 2025). As the temperature progressively diminishes in accordance with a predefined cooling schedule, the probability of accepting inferior solutions decreases, directing the search towards local optima and ultimately converging upon a global optimum (Ibrahim Younis, 2023; Pawłowski et al., 2022). This probability is derived from the Boltzmann distribution, which

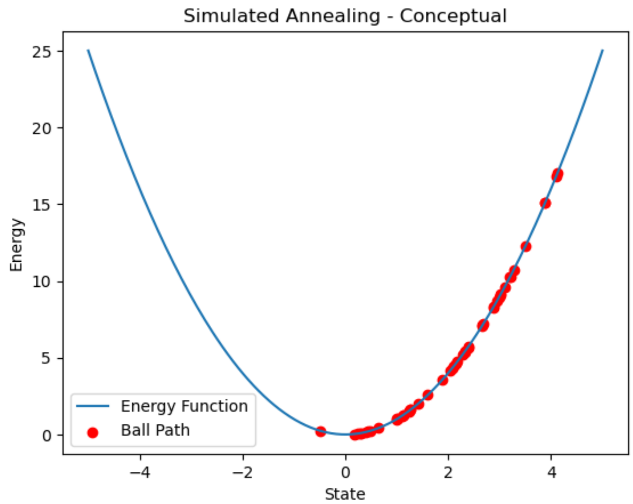

Figure 3: Conceptual visualization of Simulated Annealing (SA). Red points indicate the positions of a 'ball' on the energy landscape over time, while the curve shows the fitness or energy function. This figure illustrates how SA can probabilistically accept worse solutions at higher temperatures and gradually 'cool down' to converge toward a minimum.

quantifies the likelihood of accepting a suboptimal solution contingent on the alteration in the objective function and the current temperature (Ibrahim Younis, 2023). This annealing schedule is a pivotal component, as it dictates the rate of temperature reduction and significantly influences the algorithm's convergence characteristics and the caliber of the final solution (Suman Kumar, 2005). The systematic reduction of temperature through the cooling schedule in successive iterations manages the exploration of the search space (Yan et al., 2023).

2.3.2 Applications

The probabilistic characteristic of Simulated Annealing, enabling the acceptance of suboptimal solutions, is instrumental in circumventing local minima and facilitating comprehensive exploration of intricate solution spaces (Hunagund et al., 2017). SA finds application in diverse domains, such as the design of chiplet-based AI accelerators and the optimization of graphs and sequences (Liu et al., 2021; Mishty Sadi, 2024). Furthermore, SA has demonstrated efficacy in addressing complex combinatorial optimization problems across various disciplines, including economics and computational science (Naser Naser, 2024). It is particularly proficient in identifying pertinent genes within high-dimensional datasets during genomic feature selection processes (Sinha et al., 2024). Its extensive applicability encompasses scenarios where objective functions are non-differentiable, discontinuous, or highly multimodal, thereby limiting the effectiveness of gradient-based methodologies (Ibrahim Younis, 2023).

2.3.3 Advantages and Disadvantages

One of the primary advantages of Simulated Annealing is its theoretical guarantee of finding the global optimum with an infinitely long cooling schedule, though practical implementations employ finite schedules for computational feasibility (Liu et al., 2021). Despite this theoretical strength, a significant practical challenge lies in the algorithm's sensitivity to parameter tuning, particularly the cooling schedule and initial temperature, which directly impact its convergence speed and solution quality (Mishra, 2024; Vert et al., 2024). This parameter sensitivity, coupled with the computational cost of slow cooling rates necessary for thorough exploration, often presents a pragmatic constraint in real-world applications, making the selection of an appropriate cooling schedule crucial for balancing exploration and exploitation (Ball et al., 2018; Benfer et al., 2023). SA's capacity to escape local optima by probabilistically accepting non-improving moves is a key feature, especially useful in complex, multi-modal search spaces (Valadez-Vergara Szabó, 2024). However, this strength is counterbalanced by potential drawbacks, such as high system overhead and prolonged migration times (YANG, 2023). While SA can find good solutions, it does not guarantee optimality in a finite time, and its performance can be highly dependent on the initial solution and the specific annealing schedule implementation (Kunzmann et al., 2020; Liu et al., 2021).

2.4 Tabu Search (TS)

Tabu Search is a metaheuristic optimization algorithm designed to improve local search techniques by utilizing a short-term memory, referred to as the tabu list, which prevents the re-exploration of recently visited solutions (Abd et al., 2014). This sophisticated approach directs the search process away from suboptimal local solutions, thereby fostering a broader investigation of the entire solution landscape (Rostami et al., 2021). This is accomplished by systematically prohibiting specific movements that would return the algorithm to previously encountered solutions, which in turn encourages diversification and avoids repetitive search paths (Chen et al., 2024) (Amini Fathian, 2020). Consequently, by preventing the revisiting of prior solutions, Tabu Search effectively mitigates the common drawback of simple local search algorithms, which often become trapped in local optima (Okechukwu et al., 2024). Such a deliberate restriction enables the algorithm to more efficiently navigate and discover novel areas within the solution space (Caballero-Martin et al., 2024).

2.4.1 Operational Mechanism

Tabu Search operates by iteratively moving from a current solution to an improved neighboring solution, centrally employing a "tabu list" to record recently

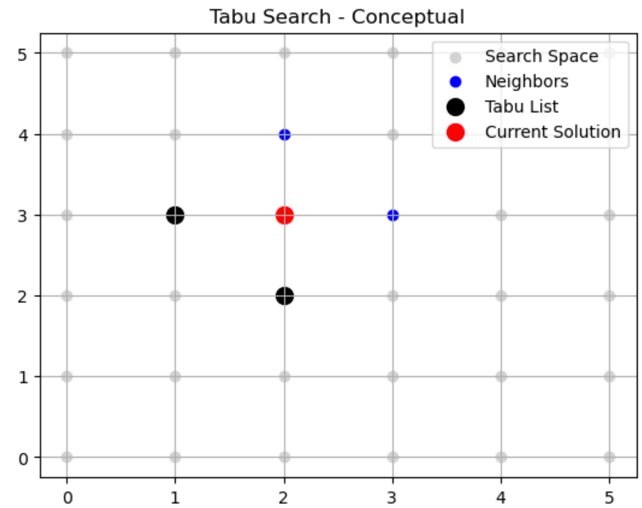

Figure 4: Conceptual illustration of Tabu Search. The red point represents the current solution, blue points show neighboring candidate solutions, and black points represent the tabu list of prohibited solutions. This visualization conveys the core idea of exploring a neighborhood while avoiding recently visited solutions.

visited solutions or moves (YANG, 2023). This list functions as a short-term memory, thereby preventing the algorithm from cycling and promoting the exploration of new regions within the search space (YANG, 2023). Moves that would lead to solutions present on the tabu list are prohibited, except when an aspiration criterion is satisfied, permitting a tabu move if it results in a solution superior to any previously discovered (Augusto et al., 2021). This iterative process persists until a termination condition is fulfilled, such as reaching a maximum number of iterations, exhausting a predefined computational budget, or experiencing a prolonged lack of improvement (Amini Fathian, 2020).

2.4.2 Applications

Tabu Search has been successfully applied to a broad spectrum of combinatorial optimization problems, including various scheduling problems, vehicle routing, and particularly structural optimization (Ghaemifard Ghannadiasl, 2024). Its adaptability to diverse scenarios makes it a powerful tool for tackling complex problems (Li Ai, 2023). For instance, it has been effectively utilized in resource-constrained project scheduling problems to minimize project duration (Golab et al., 2021), and for optimizing multi-return-to-depot petrol truck routing with time windows (Wang et al., 2019). Additionally, Tabu Search finds application in network design, telecommunications, and bioinformatics, demonstrating its versatility in contexts requiring efficient allocation and sequencing (Molina et al., 2020). Its ability to integrate with other heuristic approaches, such as Lagrangian-based methods, further enhances its efficacy in addressing complex interdependencies and constraints (Niroumandrad et al., 2024).

2.4.3 Advantages and Disadvantages

A significant advantage of Tabu Search is its capacity to escape local optima by utilizing a memory of recently visited solutions, which can involve short, intermediate, and long-term structures to ensure a more thorough exploration of the solution space and prevent repetitive exploration of suboptimal regions (Pelleau et al., 2009; Yang Burn, 2019). However, a key challenge is the careful selection of appropriate tabu tenure values, as an overly short tenure may lead to cycling, while an excessively long one might unduly constrain the search and slow down convergence (Niroumandrad et al., 2024; Shahmanzari Aksen, 2020). The effectiveness of Tabu Search is also highly dependent on the definition of the neighborhood structure and the aspiration criteria, which dictate the allowable moves and exceptions to the tabu rules, respectively (Díaz et al., 2014; Shahmanzari Aksen, 2020). In some implementations, the tabu list's size can be dynamic, varying between a lower and upper bound, and can even be cleared if a significantly better solution is discovered, akin to restarting the search (Gómez et al., 2023). Furthermore, the tabu status of a movement can be disregarded if it leads to a solution with a greater objective function value than previously visited solutions (Pacheco et al., 2023). While Tabu Search excels at finding high-quality solutions for computationally challenging problems, such as the quadratic assignment problem, its performance can be hindered by inefficient escape from local optima in scenarios with numerous such regions, potentially leading to slow convergence and suboptimal solutions (Abel Siraj, 2024; Ghnatios et al., 2019).

3 Comparative Analysis of Algorithms

Each metaheuristic algorithm exhibits distinct strengths and limitations, making their effectiveness highly dependent on problem type, dimensionality, and computational requirements.

Table 1: Performance Metrics and Characteristics of Algorithms

Metric / Criterion	**PSO**	**DE**	**SA**	**TS**
Search Type	Population-based	Population-based	Single-solution based	Single-solution based
Convergence Speed	Fast (risk of premature convergence)	Medium–fast (balanced)	Slow (depends on cooling schedule)	Medium (iterative progress)
Global Optimum Achievement	Medium–good, prone to early convergence	High — strong global optimum potential	High (with sufficient iterations)	Medium — tabu list prevents local traps
Exploration Ability	High initially, then decreases	High — mutation maintains diversity	Medium — good at high temperatures	Medium — diversity maintained via tabu mechanism
Exploitation (Intensification)	Medium	Strong — combination of individuals improves results	Strong — effective local search at low temperature	Very strong — deep neighborhood search
Computational Cost	Low–medium	Medium	Low	Medium
Parameter Sensitivity	Medium	Medium–high	High	Low–medium
Convergence Stability	Medium — depends on parameters	High	Medium	Medium
Suitability for Discrete / Continuous Problems	Continuous problems	Continuous optimization	Both types	Best for discrete
Parallelizability	High	High	Low	Low–medium
Premature Convergence Risk	High	Low	Low	Low–medium

Table 2: Suitability of Algorithms for Different Problem Types

Problem Type / Domain	**PSO**	**DE**	**SA**	**TS**
Continuous Optimization	Very suitable	Excellent	Suitable	Limited suitability
Discrete / Combinatorial	Limited — adaptation required	Limited — binary/integer version	Suitable	Excellent — planning, routing, scheduling
Multimodal Problems	Good, prone to early convergence	Excellent — strong diversity	Effective — stochastic escape	Medium — limited diversity control
Multi-objective Optimization	Easily extendable	Strong — MODE performs well	Requires additional strategies	Generally single-objective
Dynamic / Time-varying	Adaptable via population updates	May require restart	Adaptable via temperature change	Not ideal — static structure
Constrained Optimization	Can use penalty functions	Flexible	Works with simple constraints	Strong via tabu memory
Deterministic	Reliable	High accuracy	Variable due to randomness	Stable
Stochastic	Manages randomness well	Effective via mutation diversity	Suitable	Less flexible
Real-time / Speed	Excellent	Balanced	Slow	Medium

Table 3: Computational Complexity and Scalability

Metric / Criterion	**PSO**	**DE**	**SA**	**TS**
Basic Computational Complexity	$O(N_p \times D \times I)$	$O(N_p \times D \times I)$	$O(D \times I)$	$O(N_{neigh} \times I)$
Memory Usage	Low–medium	Medium	Very low	Medium
Time per Iteration	Low — can be parallelized	Medium — extra cost	Low	Medium — neighborhood and tabu checks
Scalability	Good — parallelization advantage	Good — may slow with large populations	Medium — convergence time increases with problem size	Limited — large neighborhood cost
Parallelizability	Very high	Very high	Low	Low–medium
High-dimensional Problems	Medium — risk of early convergence	High — stable	Medium–low — slow exploration	Low — high neighborhood cost
Computational Efficiency	High	Medium–high	High	Medium
Noisy / Complex Functions	Medium — diversity loss	High	Medium — stochastic inconsistencies	High — deterministic structure

4 Conclusion

Choosing the right metaheuristic algorithm requires careful consideration of problem type, dimensionality, and computational constraints, as each method offers distinct strengths and trade-offs.

This comparative analysis highlights the nuanced strengths and limitations of Particle Swarm Optimization, Differential Evolution, Simulated Annealing, and Tabu Search across various optimization problem characteristics. The evaluation demonstrates that no single metaheuristic algorithm universally outperforms the others across all problem domains.

As population-based methods, PSO and DE exhibit superior scalability and parallelization capabilities, making them particularly suitable for continuous and high-dimensional optimization. Among these, DE achieves a more balanced trade-off between exploration and exploitation, effectively reducing the risk of premature convergence often observed in PSO.

SA, while slower in convergence, provides robust performance in stochastic and multi-modal problems, effectively escaping local optima through its probabilistic acceptance mechanism. TS, in contrast, excels in discrete and combinatorial optimization, leveraging its memory-based search to avoid revisiting previously explored regions. However, its sequential and deterministic nature limits both scalability and adaptability to dynamic environments.

Overall, DE emerges as a highly versatile algorithm for continuous optimization problems, while TS dominates in combinatorial settings. SA offers robustness against randomness, and PSO remains a strong choice for scenarios demanding rapid convergence or computational efficiency.

4.1 Recommendations

Based on this analysis, the selection of a metaheuristic algorithm should be guided by specific problem types and computational considerations. DE is recommended for continuous, high-dimensional problems due to its robust global search capability. PSO is suitable when fast convergence or parallel computation is advantageous, though caution regarding premature convergence is advisable. SA is ideal for stochastic or multi-modal landscapes, providing reliable escape from local optima, albeit with slower convergence. TS is highly effective for discrete or combinatorial problems, such as scheduling, routing, or sequencing, thanks to its memory-based search and strategic avoidance of previously visited solutions. Ultimately, selecting an appropriate algorithm requires balancing convergence speed, solution quality, computational resources, and problem-specific characteristics.

5 GenAI Usage

Generative AI was utilized for grammar correction, LaTeX template generation, and translation. GitHub Copilot assisted with the development of readme and .md files and Plotly integration. The analytical insights and content presented in this report are the original work of the author, derived from a comprehensive review and synthesis of academic literature on optimization algorithms.

References

[1] Abd, G., Mahmoud, A. M., & El-Sayed, M. H. (2014). A Comparative Study of Meta-heuristic Algorithms for Solving Quadratic Assignment Problem. *International Journal of Advanced Computer Science and Applications*, *5*(1). https://doi.org/10.14569/ijacsa.2014.050101

[2] Abel, E. W., & Siraj, S. (2024). Investigating fairness in decisions involving ordinal classification: A COVID-19 case study. *Research Portal Denmark*. https://local.forskningsportal.dk/local/dki-cgi/ws/cris-link?src=sduid=sdu-5b3b1ec0-9758-4bc4-88d8-9eafa6dc1ab9

[3] Adediran, E. M., & Ameen, S. (2024). Machine Learning and Optimization Techniques for Solving Inverse Kinematics in a 7-DOF Robotic Arm. *arXiv preprint*. https://doi.org/10.48550/arxiv.2406.13064

[4] Ajibade, S.-S. M., & Ojeniyi, A. (2022). Bibliometric Survey on Particle Swarm Optimization Algorithms (2001–2021). *Journal of Electrical and Computer Engineering*, *2022*, 1–12. https://doi.org/10.1155/2022/3242949

[5] Amini, M. M., & Fathian, M. (2020). Optimizing bid search in large outcome spaces for automated multi-issue negotiations using meta-heuristic methods. *Decision Science Letters*, 1–20. https://doi.org/10.5267/j.dsl.2020.10.007

[6] Augusto, A., Dumas, M., Rosa, M. L., Leemans, S. J. J., & Broucke, S. K. L. M. vanden. (2021). Optimization framework for DFG-based automated process discovery approaches. *Software & Systems Modeling*, *20*(4), 1245–1270. https://doi.org/10.1007/s10270-020-00846-x

[7] Azzam, S. M., Emam, O. E., & Abolaban, A. S. (2024). An improved Differential evolution with Sailfish optimizer (DESFO) for handling feature selection problem. *Scientific Reports*, *14*(1). https://doi.org/10.1038/s41598-024-63328-w

[8] Balkaya, Ç., Göktürkler, G., Ekinci, Y. L., & Karaoğlan, S. T. (2014, March). Metaheuristics in applied geophysics. In *Proceedings of the 15th European Metaheuristics Community Workshop: Metaheuristics and Engineering*, Istanbul. https://doi.org/10.13140/2.1.4454.7526

[9] Ball, R. C., Branke, J., & Meisel, S. (2018). Optimal Sampling for Simulated Annealing Under Noise. *INFORMS Journal on Computing*, *30*(1), 200–215. https://doi.org/10.1287/ijoc.2017.0774

[10] Benfer, M., Heyer, V., Brützel, O., Liebrecht, C., Peukert, S., & Lanza, G. (2023). Analysis of metaheuristic optimisation techniques for simulated matrix production systems. *Production Engineering*, *18*(1), 159–168. https://doi.org/10.1007/s11740-023-01225-3

[11] Bhakhar, R., & Chhillar, R. S. (2024). Dynamic multi-criteria scheduling algorithm for smart home tasks in fog-cloud IoT systems. *Scientific Reports*, *14*(1). https://doi.org/10.1038/s41598-024-81055-0

[12] Bilal, Pant, M., Zaheer, H., García-Hernández, L., & Abraham, A. (2020). Differential Evolution: A review of more than two decades of research. *Engineering Applications of Artificial Intelligence*, *90*, 103479. https://doi.org/10.1016/j.engappai.2020.103479

[13] Bolufé-Röhler, A., & Tamayo-Vera, D. (2025). Machine Learning for Enhancing Metaheuristics in Global Optimization: A Comprehensive Review. https://doi.org/10.20944/preprints202507.2611.v1

[14] Brest, J., & Maučec, M. S. (2025). Comparative Study of Modern Differential Evolution Algorithms: Perspectives on Mechanisms and Performance. https://doi.org/10.20944/preprints202503.1551.v1

[15] Caballero-Martin, D., López-Guede, J. M., Estévez, J., & Graña, M. (2024). Artificial Intelligence Applied to Drone Control: A State of the Art. *Drones*, *8*(7), 296. https://doi.org/10.3390/drones8070296

[16] Cai, T., Zhang, S., Ye, Z., Wen, Z., Wang, M., He, Q., Chen, Z., & Bai, W. (2024). Cooperative metaheuristic algorithm for global optimization and engineering problems inspired by heterosis theory. *Scientific Reports*, *14*(1). https://doi.org/10.1038/s41598-024-78761-0

[17] Charles, A., & Parks, G. T. (2018). Application of Differential Evolution algorithms to multi-objective optimization problems in mixed-oxide fuel assembly design. *Annals of Nuclear Energy*, *127*, 165–177. https://doi.org/10.1016/j.anucene.2018.12.002

[18] Chen, B., Cao, L., Chen, C., Chen, Y., & Yue, Y. (2024). A comprehensive survey on the chicken swarm optimization algorithm and its applications: state-of-the-art and research challenges. *Artificial Intelligence Review*, *57*(7). https://doi.org/10.1007/s10462-024-10786-3

[19] Cheng, S., Liu, B., Ting, T. O., Huang, K., et al. (2016, July). Survey on data science with population-based algorithms. *Big Data Analytics*, *1*(1), 1–20. https://doi.org/10.1186/s41044-016-0003-3

[20] Cheng, S., Lu, H., Lei, X., & Shi, Y. (2018). A quarter century of particle swarm optimization. *Complex & Intelligent Systems*, *4*(3), 227–239. https://doi.org/10.1007/s40747-018-0071-2

[21] Cuevas, E., Ascencio-Piña, C., Pérez, M., & Morales-Castañeda, B. (2024). Considering radial basis function neural network for effective solution generation in metaheuristic algorithms. *Scientific Reports*, *14*(1). https://doi.org/10.1038/s41598-024-67778-0

[22] Dahal, S., Hegglid, G. J., Nøland, J. K., Chhetri, B. B., & Øyvang, T. (2024). Integrating Multiple Slack Bus Operations and Metaheuristic Techniques for Power Flow Optimization. *Research Square*. https://doi.org/10.21203/rs.3.rs-5360850/v1

[23] Das, S., & Suganthan, P. N. (2010). Differential Evolution: A Survey of the State-of-the-Art. *IEEE Transactions on Evolutionary Computation*, *15*(1), 4–31. https://doi.org/10.1109/tevc.2010.2059031

[24] Díaz, I. M., Nasini, G., & Severín, D. (2014). A Tabu Search Heuristic for the Equitable Coloring Problem. In *Lecture notes in computer science* (p. 347–358). Springer. https://doi.org/10.1007/978-3-319-09174-7_30

[25] Du, W., Leung, S. Y. S., Tang, Y., & Vasilakos, A. V. (2016). Differential Evolution With Event-Triggered Impulsive Control. *IEEE Transactions on Cybernetics*, *47*(1), 244–257. https://doi.org/10.1109/tcyb.2015.2512942

[26] Elshaboury, N. (2021). Investigating the occupant existence to reduce energy consumption by using a hybrid artificial neural network with metaheuristic algorithms. *Decision Science Letters*, *11*(1), 91–104. https://doi.org/10.5267/j.dsl.2021.8.001

[27] Fang, J., Liu, W., Chen, L.-W., Lauria, S., Miron, A., & Liu, X.-H. (2023). A Survey of Algorithms, Applications and Trends for Particle Swarm Optimization. *International Journal of Network Dynamics and Intelligence*, 24–50. https://doi.org/10.53941/ijndi0201002

[28] Farinati, D., & Vanneschi, L. (2024). A survey on dynamic populations in bio-inspired algorithms. *Genetic Programming and Evolvable Machines*, *25*(2). https://doi.org/10.1007/s10710-024-09492-4

[29] Ghaemifard, S., & Ghannadiasl, A. (2024). A Comparison of Metaheuristic Algorithms for Structural Optimization: Performance and Efficiency Analysis. *Advances in Civil Engineering*, *2024*(1). https://doi.org/10.1155/2024/2054173

[30] Ghnatios, C., Hage, R.-M., & Hage, I. S. (2019). An efficient Tabu-search optimized regression for data-driven modeling. *Comptes Rendus Mécanique*, *347*(11), 806–816. https://doi.org/10.1016/j.crme.2019.11.006

[31] Gift, N., & Deza, A. (2021). *Practical MLOps Operationalizing Machine Learning Models*. O'Reilly Media.

[32] Golab, A., Gooya, E. S., Falou, A. A., & Cabon, M. (2021). Review of conventional metaheuristic techniques for resource-constrained project scheduling problem. *Journal of Project Management*, *7*(2), 95–110. https://doi.org/10.5267/j.jpm.2021.10.002

[33] Gómez, P. G., Vela, C. R., & González-Rodríguez, I. (2023). Neighbourhood search for energy minimisation in flexible job shops under fuzziness. *Natural Computing*, *22*(4), 685–704. https://doi.org/10.1007/s11047-023-09967-w

[34] Hernández-Briones, M. A., Reséndiz-Flores, E. O., & Serrano-Toledo, J. L. (2024). A novel intelligent prediction approach with optimal dimensional reduction on shrinkage defects in iron casting. *Research Square*. https://doi.org/10.21203/rs.3.rs-5182557/v1

[35] Hubálovský, Š., Hubálovská, M., & Matoušová, I. (2023). A New Hybrid Particle Swarm Optimization–Teaching–Learning-Based Optimization for Solving Optimization Problems. *Biomimetics*, *9*(1), 8. https://doi.org/10.3390/biomimetics9010008

[36] Hunagund, I. B., Pillai, V. M., & Kempaiah, U. N. (2017). A simulated annealing algorithm for unequal area dynamic facility layout problems with flexible bay structure. *International Journal of Industrial Engineering Computations*, 307–330. https://doi.org/10.5267/j.ijiec.2017.8.004

[37] Ibrahim, O. A. S., & Younis, E. M. G. (2023). A Hybrid-Box tool for Explainable and Interactive Simulated Annealing Strategy. *Research Square*. https://doi.org/10.21203/rs.3.rs-3578275/v1

[38] Innocente, M. S., & Sienz, J. (2006). Particle Swarm Optimization: Development of a General-Purpose Optimizer. *arXiv preprint*, 203–223. https://doi.org/10.48550/arXiv.2101.09835

[39] Jiang, R., Du, G., Yu, S., Guo, Y., Goh, S. K., & Tang, H.-K. (2024). CADE: Cosine Annealing Differential Evolution for Spiking Neural Network. *arXiv preprint*. https://doi.org/10.48550/arxiv.2406.02349

[40] Khalfi, S., Draa, A., & Iacca, G. (2021). A compact compound sinusoidal differential evolution algorithm for solving optimisation problems in memory-constrained environments. *Expert Systems with Applications*, *186*, 115705. https://doi.org/10.1016/j.eswa.2021.115705

[41] Kononova, A. V., Caraffini, F., & Bäck, T. (2021). Differential evolution outside the box. *Information Sciences*, *581*, 587–604. https://doi.org/10.1016/j.ins.2021.09.058

[42] Kumar, P., Kumar, K., Bohre, A. K., Adhikary, N., & Tesfaye, E. L. (2023). Intelligent planning of controllers for improved resilience in multi-area system involving nuclear power. *Scientific Reports*, *13*(1). https://doi.org/10.1038/s41598-023-42155-5

[43] Kunzmann, P., Mayer, B. E., & Hamacher, K. (2020). Substitution matrix based color schemes for sequence alignment visualization. *BMC Bioinformatics*, *21*(1). https://doi.org/10.1186/s12859-020-3526-6

[44] Li, J., Majeed, A. P. P. A., & Lefevre, P. (2024). Halfway Escape Optimization: A Quantum-Inspired Solution for General Optimization Problems. *Research Square*. https://doi.org/10.21203/rs.3.rs-5285753/v1

[45] Li, M., & Ai, J. (2023). Energy-Aware Clustering in the Internet of Things using Tabu Search and Ant Colony Optimization Algorithms. *International Journal of Advanced Computer Science and Applications*, *14*(12). https://doi.org/10.14569/ijacsa.2023.0141238

[46] Li, M., & Zhang, Y. (2023). Integrating Social Media Data and Historical Stock Prices for Predictive Analysis: A Reinforcement Learning Approach. *International Journal of Advanced Computer Science and Applications*, *14*(12). https://doi.org/10.14569/ijacsa.2023.0141203

[47] Liu, R., Yang, Z., Su, J., & Yu, C. (2024). A Hybrid Framework for Evaluating Financial Market Price: An Analysis of the Hang Seng Index Case Study. *International Journal of Advanced Computer Science and Applications*, *15*(6). https://doi.org/10.14569/ijacsa.2024.01506111

[48] Liu, X., Li, P., Meng, F., Zhou, H., Zhong, H., Zhou, J., Mou, L., & Song, S. (2021). Simulated annealing for optimization of graphs and sequences. *Neurocomputing*, *465*, 310–324. https://doi.org/10.1016/j.neucom.2021.09.003

[49] Martinez, A. D., Ser, J. D., Villar-Rodríguez, E., Osaba, E., Poyatos, J., Tabik, S., Molina, D., & Herrera, F. (2020). Lights and shadows in Evolutionary Deep Learning: Taxonomy, critical methodological analysis, cases of study, learned lessons, recommendations and challenges. *Information Fusion*, *67*, 161–194. https://doi.org/10.1016/j.inffus.2020.10.014

[50] Mishra, A. (2024). Machine Learning-Driven Optimization of TPMS Architected Materials Using Simulated Annealing. *arXiv preprint*. https://doi.org/10.48550/arxiv.2406.05142

[51] Mishty, K., & Sadi, M. (2024). Chiplet-Gym: Optimizing Chiplet-based AI Accelerator Design with Reinforcement Learning. *arXiv preprint*. https://doi.org/10.48550/arxiv.2406.00858

[52] Mogale, D. G., Kumar, S. K., & Tiwari, M. K. (2017). An MINLP model to support the movement and storage decisions of the Indian food grain supply chain. *Control Engineering Practice*, *70*, 98–113. https://doi.org/10.1016/j.conengprac.2017.09.017

[53] Molina, C. A., Salmerón, J. L., Eguía, I., & Racero, J. (2020). The heterogeneous vehicle routing problem with time windows and a limited number of resources. *Engineering Applications of Artificial Intelligence*, *94*, 103745. https://doi.org/10.1016/j.engappai.2020.103745

[54] Mousavirad, S. J., Schaefer, G., Rezaee, K., Oliva, D., Zabihzadeh, D., Chakrabortty, R. K., Mohammadigheymasi, H., & Pedram, M.

(2024). A novel metaheuristic population algorithm for optimising the connection weights of neural networks. *Evolving Systems*, *16*(1). https://doi.org/10.1007/s12530-024-09641-1

[55] MUKUNZI, W. K., Muganda, B. W., & Kasamani, B. S. (2024). A Machine Learning Model for Algorithmic Optimization of Superannuation Schemes. *Research Square*. https://doi.org/10.21203/rs.3.rs-5280421/v1

[56] Munirah, N. A., Remli, M. A., Mohd, N., Wen, H., Saberi, M., & Nizar, K. (2020). The Development of Parameter Estimation Method for Chinese Hamster Ovary Model using Black Widow Optimization Algorithm. *International Journal of Advanced Computer Science and Applications*, *11*(11). https://doi.org/10.14569/ijacsa.2020.0111126

[57] Naderi, S., Blondin, M. J., & Rezaie, B. (2022). Optimizing an adaptive fuzzy logic controller of a 3-DOF helicopter with a modified PSO algorithm. *International Journal of Dynamics and Control*, *11*(4), 1895–1913. https://doi.org/10.1007/s40435-022-01091-4

[58] Napiórkowski, J. J., Piotrowski, A., Karamuz, E., & Senbeta, T. B. (2022). Calibration of conceptual rainfall-runoff models by selected differential evolution and particle swarm optimization variants. *Acta Geophysica*, *71*(5), 2325–2338. https://doi.org/10.1007/s11600-022-00988-0

[59] Naser, M. Z., & Naser, A. (2024). The Firefighter Algorithm: A Hybrid Metaheuristic for Optimization Problems. *arXiv preprint*. https://doi.org/10.48550/arxiv.2406.00528

[60] Nasr, M., Farouk, O., Mohamedeen, A., Elrafie, A., Bedeir, M., & Khaled, A. (2020). Benchmarking Meta-heuristic Optimization. *International Journal of Advanced Networking and Applications*, *11*(6). https://doi.org/10.35444/ijana.2020.11063

[61] Niroumandrad, N., Lahrichi, N., & Lodi, A. (2024). Learning tabu search algorithms: A scheduling application. *Computers & Operations Research*, *170*, 106751. https://doi.org/10.1016/j.cor.2024.106751

[62] Odeyemi, J., & Zhang, W. (2025). Benchmarking Randomized Optimization Algorithms on Binary, Permutation, and Combinatorial Problem Landscapes. https://doi.org/10.48550/ARXIV.2501.17170

[63] Okechukwu, C., Šilhavý, R., Oyelere, S. S., & Šilhavý, P. (2024). Comparative Review of Single-Criteria and Multi-Criteria Optimisation Problems using Meta-heuristic Algorithms. *Research Square*. https://doi.org/10.21203/rs.3.rs-4973948/v1

[64] Pacheco, J., Saiz-Vázquez, O., Casado, S., & Landa, S. U. (2023). A multistart tabu search-based method for feature selection in medical applications. *Scientific Reports*, *13*(1). https://doi.org/10.1038/s41598-023-44437-4

[65] Pawłowski, J., Kutorasiński, K., & Szewczyk, M. (2022). Multifrequency nonlinear model of magnetic material with artificial intelligence optimization. *Scientific Reports*, *12*(1). https://doi.org/10.1038/s41598-022-23810-9

[66] Pelleau, M., Hentenryck, P. V., & Truchet, C. (2009). Sonet Network Design Problems. *arXiv preprint*, *5*, 81–95. https://doi.org/10.4204/eptcs.5.7

[67] Piotrowski, A., Napiórkowski, J. J., & Piotrowska, A. E. (2023). Particle Swarm Optimization or Differential Evolution—A comparison. *Engineering Applications of Artificial Intelligence*, *121*, 106008. https://doi.org/10.1016/j.engappai.2023.106008

[68] Rostami, M., Berahmand, K., & Forouzandeh, S. (2021). A novel community detection based genetic algorithm for feature selection. *Journal Of Big Data*, *8*(1). https://doi.org/10.1186/s40537-020-00398-3

[69] Sarhani, M., Voß, S., & Jovanović, R. (2022). Initialization of metaheuristics: comprehensive review, critical analysis, and research directions. *International Transactions in Operational Research*, *30*(6), 3361–3397. https://doi.org/10.1111/itor.13237

[70] Schneider, P., Santiago, X. G., Soltwisch, V., Hammerschmidt, M., Burger, S., & Rockstuhl, C. (2019). Benchmarking Five Global Optimization Approaches for Nano-optical Shape Optimization and Parameter Reconstruction. *ACS Photonics*, *6*(11), 2726–2733. https://doi.org/10.1021/acsphotonics.9b00706

[71] Shahmanzari, M., & Aksen, D. (2020). A Multi-Start Granular Skewed Variable Neighborhood Tabu Search for the Roaming Salesman Problem. *Applied Soft Computing*, *102*, 107024. https://doi.org/10.1016/j.asoc.2020.107024

[72] Sharifi-Noghabi, H., Mashhadi, H. R., & Shojaee, K. (2016). A novel mutation operator based on the union of fitness and design spaces information for Differential Evolution. *Soft Computing*, *21*(22), 6555–6562. https://doi.org/10.1007/s00500-016-2359-8

[73] Sinha, K., Chakraborty, S., Bardhan, A., Saha, R., Chakraborty, S., & Biswas, S. (2024). A New Differential Gene Expression Based Simulated Annealing for Solving Gene Selection Problem: A Case Study on Eosinophilic Esophagitis and Few Other Gastro-Intestinal Diseases. *Research Square*. https://doi.org/10.21203/rs.3.rs-4997447/v1

[74] Sohail, M. S., Saeed, M. O. B., Rizvi, S. Z., Shoaib, M., & Sheikh, A. U. H. (2014). Low-Complexity Particle Swarm Optimization for Time-Critical Applications. *arXiv preprint*. https://doi.org/10.48550/arXiv.1401.0546

[75] Suman, B., & Kumar, P. (2005). A survey of simulated annealing as a tool for single and multiobjective optimization. *Journal of the Operational Research Society*, *57*(10), 1143–1160. https://doi.org/10.1057/palgrave.jors.2602068

[76] Tang, J., & Wang, L. (2024). A whale optimization algorithm based on atom-like structure differential evolution for solving engineering design problems. *Scientific Reports*, *14*(1). https://doi.org/10.1038/s41598-023-51135-8

[77] Tomar, V., Bansal, M., & Singh, P. (2024). Metaheuristic Algorithms for Optimization: A Brief Review. 238. https://doi.org/10.3390/engproc2023059238

[78] Utkarsh, & Jain, P. (2024). Predicting bentonite swelling pressure: optimized XGBoost versus neural networks. *Scientific Reports*, *14*(1). https://doi.org/10.1038/s41598-024-68038-x

[79] Valadez-Vergara, R., & Szabó, N. P. (2024). Level of thermal maturity estimation in unconventional reservoirs using interval inversion and simulating annealing method. *Acta Geophysica*.

[80] Vert, D., Willsch, M., Yenilen, B., Sirdey, R., Louise, S., & Michielsen, K. (2024). *Benchmarking quantum annealing with maximum cardinality matching problems*. Frontiers in Computer Science, 6. https://doi.org/10.3389/fcomp.2024.1286057

[81] Wang, X., & Chen, Y. (2013). *Metaheuristic Optimization: Nature-Inspired Algorithms and Applications.*

[82] Wang, B., Liang, Y., Yuan, M., Zhang, H., & Liao, Q. (2019). A metaheuristic method for the multireturn-to-depot petrol truck routing problem with time windows. *Petroleum Science*, 16(3), 701–710. https://doi.org/10.1007/s12182-019-0316-8

[83] Wang, H., Deng, Y., Huangshui, H., Liu, K., & Xinji, F. (2024). Particle Swarm Optimization Based Mobile Sink Routing Protocol in Wireless Sensor Networks. *Research Square*. https://doi.org/10.21203/rs.3.rs-3974420/v1

[84] Wehrens, R., & Buydens, L. M. C. (2000). Classical and nonclassical optimization methods. In R. A. Meyers (Ed.), *Encyclopedia of Analytical Chemistry* (pp. 9678–9689). John Wiley & Sons Ltd.

[85] Weiel, M., Götz, M., Klein, A., Coquelin, D., Floca, R., & Schug, A. (2021). Dynamic particle swarm optimization of biomolecular simulation parameters with flexible objective functions. *Nature Machine Intelligence*, 3(8), 727–735. https://doi.org/10.1038/s42256-021-00366-3

[86] Yan, W., Li, N., & Zhang, X. (2023). Enhancing supply chain management in the physical internet: a hybrid SAGA approach. *Scientific Reports*, 13(1). https://doi.org/10.1038/s41598-023-48384-y

[87] Yang, X. (2023). Nature-Inspired Optimization for Virtual Machine Allocation in Cloud Computing: Current Methods and Future Directions. *International Journal of Advanced Computer Science and Applications*, 14(11). https://doi.org/10.14569/ijacsa.2023.01411121

[88] Yang, Z., & Burn, D. H. (2019). Automatic feature selection and weighting for the formation of homogeneous groups for regional IDF estimation. *Journal of Hydrology*, 575, 292–302. https://doi.org/10.1016/j.jhydrol.2019.05.015

[89] Yuan, Q., & Yin, G. (2015). Analyzing convergence and rates of convergence of particle swarm optimization algorithms using stochastic approximation methods. *IEEE Transactions on Automatic Control*, 60(7), 1760–1773. https://doi.org/10.1109/tac.2014.2381454

[90] Zambrano-Gutierrez, D. F., Cruz-Duarte, J. M., Avina–Cervantes, J. G., Ortíz-Bayliss, J. C., Yanez-Borjas, J. J., & Amaya, I. (2023). Automatic design of metaheuristics for practical engineering applications. *IEEE Access*, 11, 7262–7275. https://doi.org/10.1109/access.2023.3236836

[91] Zhang, Z., Zhu, J., & Nie, F. (2024). A novel hybrid adaptive differential evolution for global optimization. *Scientific Reports*, 14(1). https://doi.org/10.1038/s41598-024-70731-w

[92] Zhong, R., Cao, Y., Zhang, E., & Munetomo, M. (2024). Introducing Competitive Mechanism to Differential Evolution for Numerical Optimization. *arXiv preprint*, arXiv:2406.05436. https://doi.org/10.48550/arxiv.2406.05436