SAKARYA UNIVERSITY JOURNAL OF COMPUTER AND INFORMATION SCIENCES
http://saucis.sakarya.edu.tr/
e-ISSN: 2636-8129
Publisher: Sakarya University
Vol. 7, No. 2, 2024, 227-243
DOI: 10.35377/saucis...1474767
SAUCIS
SAKARYA UNIVERSITY

RESEARCH ARTICLE

Performance Assessment of Natural Survivor Method-Based Metaheuristic Optimizers in Global Optimization and Engineering Design Problems

Hüseyin Bakır[1]

[1] Department of Electronics and Automation, Vocational School, Dogus University, Istanbul, Türkiye

Corresponding author:
Hüseyin Bakır, Department of Electronics and Automation, Vocational School, Dogus University, Istanbul, Türkiye
hbakir@dogus.edu.tr

Article History:
Received: 28.04.2024
Accepted: 29.05.2024
Published Online: 26.08.2024

ABSTRACT
This study presents the comparative performance analysis of Natural Survivor Method (NSM)-based algorithms in solving the IEEE CEC 2022 test suite benchmark problems and four real-world engineering design problems. Three different variants (Case1, Case2, Case3) of the NSM-TLABC, NSM-SFS and NSM-LSHADE-SPACMA algorithms were used in the study. The data obtained from the experimental studies were statistically analyzed using Friedman and Wilcoxon signed-rank tests. Based on the Friedman test results, NSM-LSHADE-SPACMA_Case2 showed the best performance with an average Friedman score of 3.96. The Wilcoxon signed-rank test showed that NSM-LSHADE-SPACMA_Case2 outperformed its competitors in 13 out of 16 experiments, achieving a success rate of 81.25%. NSM-LSHADE-SPACMA_Case2, which was found to be the most powerful of the NSM-based algorithms, is used to solve cantilever beam design, tension/compression spring design, pressure vessel design and gear train design problems. The optimization results are also compared with eight state-of-the-art metaheuristics, including Rime Optimization Algorithm (RIME), Nonlinear Marine Predator Algorithm (NMPA), Northern Goshawk Optimization (NGO), Kepler Optimization Algorithm (KOA), Honey Badger Algorithm (HBA), Artificial Gorilla Troops Optimizer (GTO), Exponential Distribution Optimization (EDO) and Hunger Games Search (HGS). Given that all results are together, it is seen that NSM-LSHADE-SPACMA_Case2 algorithm consistently produced the best results for the global and engineering design problems studied.

Keywords: Natural survivor method-based algorithms, Global optimization, IEEE CEC 2022 test suite, Real-world engineering design problems

1. Introduction

Metaheuristic optimization algorithms (MOAs) have a great reputation for solving global and engineering problems [1, 2]. The ability of MOAs to solve non-convex optimization problems has encouraged researchers to conduct further research on metaheuristic algorithms [3, 4]. In this direction, they have been focused on two topics. The first is the development of new metaheuristics. The second is the performance improvement of existing MOAs [5].

Metaheuristic algorithm design is an active area of research. To date, numerous MOAs have been developed inspired by events, methods, and natural processes [6]. Some examples of MOAs introduced in the last four years include Black Widow Optimization (BWO, 2020) [7], Mayfly Algorithm (MA, 2020) [8] ,Group Teaching Optimization Algorithm (GTOA, 2020) [9], Gradient-Based Optimizer (GBO, 2020) [10], Transient Search Optimization (TSO, 2020) [11], Chameleon Swarm Algorithm (CSA, 2021) [12], Horse herd Optimization Algorithm (HOA, 2021) [13], Capuchin Search Algorithm (CapSA, 2021) [14], Archimedes Optimization Algorithm (AOA, 2021) [15], Dwarf Mongoose Optimization (DMO, 2022) [16], Gannet Optimization Algorithm (GOA, 2022) [17], Red Fox Optimization (RFO, 2021) [18], Tasmanian Devil Optimization (TDO, 2022) [19], War Strategy Optimization (WSO, 2022) [20], Wild Horse Optimizer (WHO, 2022) [21], Coati Optimization Algorithm (COA, 2023) [22], Nutcracker Optimization Algorithm (NOA, 2023) [23], Meerkat Optimization Algorithm(MOA, 2023) [24], Energy Valley Optimizer (EVO, 2023) [25], and Growth Optimizer (GO, 2023) [26]. In the above-mentioned literature works, authors usually verified the performance of the developed algorithms using global optimization and real-world engineering problems. Particularly, optimizing constrained engineering problems from various disciplines has played an important role in testing the algorithm's performance. Rolling Element Bearing Design (REBD), Three-Bar Truss Design (TBTD), Welded Beam Design (WBD), Cantilever Beam Design (CBD), Disc Clutch Brake Design (DCBD), Speed Reducer Design (SRD), Pressure Vessel Design (PVD), Gear Train Design (GTD), and Tension/Compression Spring Design (TCSD) are among the commonly studied real-world problems. Agushaka et al. [27] developed a novel optimizer named the Gazelle Optimization Algorithm (GOA) and applied it to solve WBD, TCSD, PVD,

and SRD problems. The findings showed that GOA can produce optimal solutions to the real-world engineering problems addressed in the study. In another work, Kaveh et al. [28] showed that the Orchard Algorithm (OA) found better solutions to PVD, WBD, TCSD, and TBTD problems compared to CapSA, ChOA, BWO, PSO, and GA methods. Han et al. [29] performed the optimization of GTD, CBD, TCSD, PVD, WBD and SRD problems with the Walrus Optimizer (WO). Numerical results showed that WO successfully converges to the global optimum in real-world engineering problems under study. Zhu et al. [30] successfully applied the Human Memory Optimization (HMO) algorithm to find optimal solutions to TBTD, TCSD, and WBD problems. Optimization results illustrated that HMO is superior to competitive optimizers about solution accuracy. El-kenawy et al. [31] introduced a nature-inspired Greylag Goose Optimization (GGO) method and tested the solution ability of the algorithm by solving constrained PVD and TCSD problems. Given that all results are together, it is seen that GGO achieves consistently optimal solutions to real-world problems.

In recent years, studies on improving the performance of existing MOAs have been increasing. The reason behind this can be that it is possible to increase the robustness, solution accuracy, and convergence speed of metaheuristic optimizers using well-known strategies such as chaos maps, Levy flights, fitness-distance balance selection, opposition-based learning, etc. For example, Aydemir [32] applied the chaotic maps strategy to overcome getting stuck in local traps and poor convergence problems of the Arithmetic Optimization Algorithm (AOA). The solution ability of the Chaotic AOA (CAOA) has been evaluated on benchmark functions. The results showed that the CAOA effectively scanned the search space and converged to the best solutions. Ekinci et al. [33] developed a powerful variant of the Reptile Search Algorithm (RSA) to obtain a better optimization structure for solving challenging power system problems. The authors redesigned the exploration operator of the optimizer with the Levy flight and the exploitation operator with the Nelder–Mead simplex search approach. The results obtained from parameter extraction of the Power System Stabilizer (PSS), and the design of the Automatic Voltage Regulator (AVR) system demonstrated that the developed method could obtain better solutions compared to the original RSA. Bakir et al. [34] used the Fitness Distance Balance (FDB) strategy to enhance the convergence rate of the Levy Flight Distribution (LFD) optimizer. The proposed FDB-LFD has been tested on 39 benchmark problems as well as AVR optimization. The results show that the FDB approach provides a notable enhancement in the exploration ability of the original LFD. Zhong et al. [35] focused on enhancing the convergence rate of the Equilibrium Optimizer (EO). In this direction, the authors equipped the EO algorithm with evolutionary population dynamics, opposition-based learning, and Levy flight operators. Considering the experimental studies performed in search spaces of 100 to 5000 dimensions, it is observed that modified EO exhibits outstanding optimization performance.

The search process life cycle of metaheuristic optimization algorithms is generally divided into three phases. These are selection, search, and update [36]. The task of the selection phase is to determine the guide solution candidate/s that will direct the search process. During the search phase, exploration and exploitation tasks are fulfilled. The update phase determines which solution candidates will survive and which will be killed. As can be seen from the above-mentioned literature review, many of the studies regarding improving the performance of existing algorithms have focused on the first two stages (selection and search). As far as the authors know, the latest study related to the design of an update strategy was conducted by Kahraman et al. [6] in 2023. In the reference work, the authors developed a new strategy for updating solution candidates called the Natural Survivor Method (NSM). The introduced update strategy was applied to the base version of the Teaching-Learning-Based Artificial Bee Colony (TLABC) [37], Stochastic Fractal Search (SFS) [38], and LSHADE-SPACMA [39] algorithms, and three new algorithms named NSM-TLABC, NSM-SFS, and NSM-LSHADE-SPACMA were developed. Although the success of NSM-based algorithms in optimizing CEC 2017 and CEC 2020 benchmark problems has been investigated, their performance in the challenging CEC 2022 benchmark functions is a matter of curiosity. In this regard, this study has focused on the comparative performance analysis of NSM-based algorithms in optimizing the CEC 2022 benchmark problems. In addition, the algorithm with the best performance among the NSM-based algorithms is applied to solve real-world engineering problems.

The main contributions of the paper can be listed as follows:

- Optimization of twelve benchmark functions of the CEC 2022 test suite using NSM-based metaheuristics (NSM-TLABC, NSM-SFS, and NSM-LSHADE-SPACMA) for the first time.
- Non-parametric statistical analysis of NSM-based algorithms. Analysis results verified that the NSM-LSHADE-SPACMA_Case2 outperformed the compared ones.
- Application of NSM-LSHADE-SPACMA_Case2 algorithm to the solution of cantilever beam design, tension/compression spring design, pressure vessel design, and gear train design problems for the first time.
- Comparison of the results of NSM-LSHADE-SPACMA_Case2 on engineering optimization with eight optimization algorithms including RIME, NMPA, NGO, KOA, HBA, GTO, EDO, and HGS.

The remaining sections of the paper are designed in the following manner: Section 2 introduces the benchmark functions of the CEC 2022 test suite. Section 3 presents the optimization model of real-world engineering problems. In Section 4, the basics of NSM-based algorithms are presented. Section 5 evaluates the results of experimental studies. Finally, the findings of the present research are summarized in Section 6.

2. Benchmark Functions of the CEC 2022 Test Suite

The CEC 2022 [40] benchmark suite is an important resource for the optimization community. It provides a common platform to compare optimization algorithms on a set of challenging problems. The test suite consists of twelve benchmark functions in four categories. The summary of the CEC 2022 test suite is given in Table 1.

Table 1. Outline of CEC 2022 Test Suite [40]

Function Type	Function Number	Global Optimum	Function Type	Function Number	Global Optimum
Unimodal	F1	300	Hybrid	F6	1800
				F7	2000
				F8	2200
Multimodal	F2	400	Composition	F9	2300
	F3	600		F10	2400
	F4	800		F11	2600
	F5	900		F12	2700

As can be seen in Table 1, there is only one benchmark function with the unimodal type and that is F1. This problem measures the exploitation capability. There are multiple local optima solutions of the multimodal type F2, F3, F4, and F5 benchmark functions. Convergence to the global optimum in multimodal benchmark functions is a relatively challenging task. To overcome this, the algorithm must have strong exploration capability. In the CEC 2022 test suite, there are three hybrid (F6, F7, F8) and four composition (F9, F10, F11, F12) type benchmark functions. In both problem types, the convergence accuracy is directly related to successfully establishing the exploration-exploitation balance. Each benchmark function of the CEC 2022 is designed as a minimization problem. The search space boundaries for all benchmark functions are set to $[-100, 100]^D$. Detailed information on the CEC 2022 test suite problems can be found in Ref. [40].

3. Formulation of Engineering Problems

In the study, four engineering problems are optimized. The optimization model of engineering problems is introduced below.

3.1. Gear Train Design

In this problem, the objective is the minimization of gear ratio cost. Equation (1) gives the optimization model of the gear train design problem [41]. As shown in the equation there are four parameters to be optimized. These are teeth number of gearwheels: T_a, T_b, T_d, T_f. The pictorial representation of the problem is given in Figure 1 [41].

$$\begin{aligned} &minimize \quad F_{obj,1}(\vec{x}) = \left(\frac{1}{6.931} - \frac{T_b \times T_d}{T_a \times T_f}\right)^2 \\ &variable\ vector\ \ \vec{x} = [T_a, T_b, T_d, T_f] \\ &variable\ range\colon\ 0.01 \le T_a, T_b, T_d, T_f \le 60 \end{aligned} \tag{1}$$

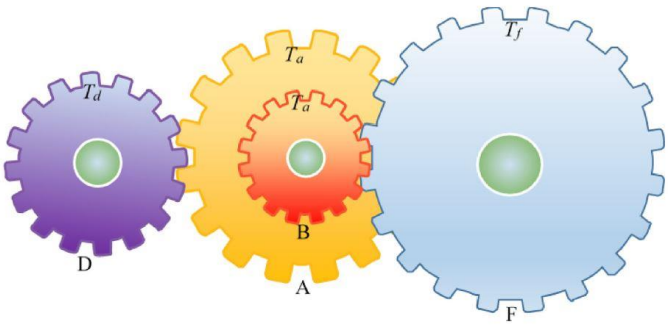


Figure 1. Gear Train Design [41]

3.2. Tension/Compression Spring Design

The primary task is to minimize the weight of the coil while satisfying various inequality constraints. As can be seen in the schematic diagram given in Figure 2, the problem includes three parameters to be optimized. These are the number of spring's active coils (N), the diameter of the winding (D), and the diameter of the wire (d). The optimization model of the problem can be written as follows [41]:

$$
\begin{aligned}
&minimize \quad F_{obj,2}(\vec{x}) = (x_3 + 2)\, x_2\, {x_1}^2 \\
&variable\ vector\ \ \vec{x} = [x_1, x_2, x_3] = [d, D, N] \\
&subject\ to\colon\ h_1(\vec{x}) = 1 - \frac{x_3\, {x_2}^3}{71785\, {x_1}^4} \le 0 \\
&subject\ to\colon\ h_2(\vec{x}) = \frac{4\, {x_2}^2 - x_1 x_2}{12566(\, x_2 {x_1}^3 - \, {x_1}^4)} + \frac{1}{5108\, {x_1}^2} - 1 \le 0 \\
&h_3(\vec{x}) = 1 - \frac{140.45\, x_1}{{x_2}^2\, x_3} \le 0, \quad h_4(\vec{x}) = \frac{x_1 + x_2}{1.5} - 1 \le 0 \\
&variable\ range\colon\ 0.05 \le x_1 \le 2, 0.25 \le x_2 \le 1.3, 2 \le x_3 \le 15
\end{aligned}
\tag{2}
$$

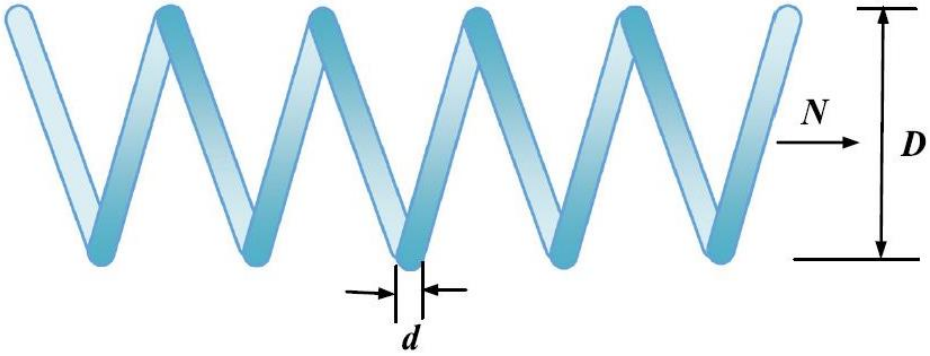


Figure 2. Tension/Compression Spring Design [41]

3.3. Pressure Vessel Design

Figure 3 illustrates the pictorial representation of the pressure vessel design. The objective of the problem is the minimization of fabrication cost through the optimal setting of the inner radius (R), the thickness of the head (T_h), length of the shell (L), and the thickness of the shell (T_s). The mathematical representation of the problem is given in Eq. (3) [41].

$$
\begin{aligned}
&minimize\ F_{obj,3}(\vec{x}) = 0.6224\, x_1 x_3 x_4 + 1.7781 x_2 {x_3}^2 + 3.1661\, {x_1}^2 x_4 + 19.84\, {x_1}^2 x_3 \\
&variable\ vector\ \ \vec{x} = [x_1, x_2, x_3, x_4] = [T_s, T_h, R, L] \\
&subject\ to\colon\ h_1(\vec{x}) = -x_1 + 0.0193 x_3 \le 0 \\
&h_2(\vec{x}) = -x_2 + 0.00954 x_3 \le 0 \\
&h_3(\vec{x}) = -\pi\, {x_3}^2 x_4 - \frac{4}{3}\pi\, {x_3}^3 + 1296000 \le 0 \\
&h_4(\vec{x}) = x_4 - 240 \le 0 \\
&variable\ range\colon\ 0 \le x_1, x_2 \le 99, 10 \le x_3, x_4 \le 200
\end{aligned}
\tag{3}
$$

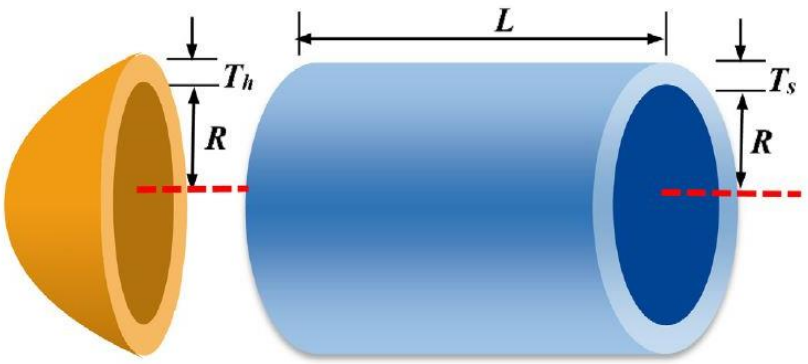


Figure 3. Pressure Vessel Design [41]

3.4. Cantilever Beam Design

This problem minimizes cantilever beam weight by optimizing five different block lengths. Figure (4) and Equation (4) give the schematic representation and mathematical model of the problem, respectively [41].

$$
\begin{aligned}
&minimize\ F_{obj,4}(\vec{x}) = 0.0624\ (x_1 + x_2 + x_3 + x_4 + x_5)\\
&variable\ vector\ \ \vec{x}\ = [x_1, x_2, x_3, x_4, x_5]\\
&subject\ to: h_1(\vec{x}) = \frac{61}{{x_1}^3} + \frac{37}{{x_2}^3} + \frac{19}{{x_3}^3} + \frac{7}{{x_4}^3} + \frac{1}{{x_5}^3} - 1 \le 0\\
&variable\ range:\ 0.01 \le x_1, x_2, x_3, x_4, x_5 \le 100
\end{aligned}
\tag{4}
$$

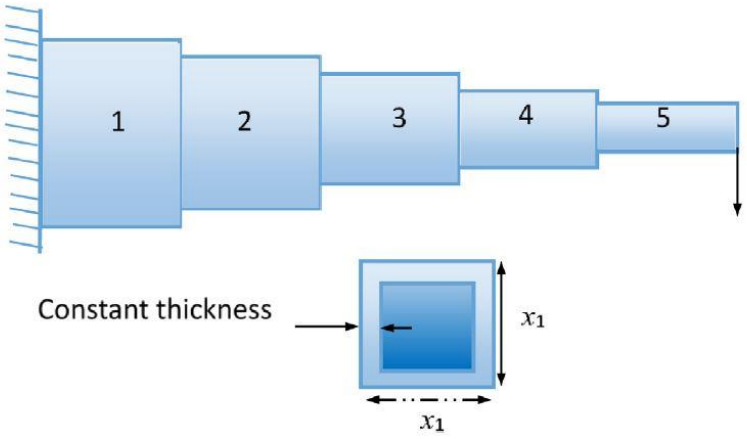


Figure 4. Cantilever Beam Design [41]

4. Natural Survivor Method-Based Metaheuristics

This section introduces the optimization framework of NSM-based algorithms. In this context, three subsections have been prepared. The following subsections provide detailed information about the NSM-TLABC, NSM-SFS and NSM-LSHADE-SPACMA algorithms, respectively.

4.1. NSM-TLABC

NSM-TLABC [6] is a powerful variant of the TLABC algorithm. The algorithm was created by redesigning the survivor selection of the original TLABC [37] using the NSM. NSM-TLABC initiates optimization by generating a random initial population. Then, it applies exploration and exploitation operators to find the global optimum solution. The pseudo-code of the NSM-TLABC algorithm is given in Algorithm 1.

Algorithm 1. NSM-TLABC Algorithm

1	**Initialization**
2	Create initial population with defaults (population size N and number of design variables D) $P \equiv \begin{bmatrix} p_{11} & \cdots & p_{1D} \\ \vdots & \ddots & \vdots \\ p_{N1} & \cdots & p_{ND} \end{bmatrix}_{N \times D}$
3	**for** i=1: N **do**
4	Calculate objective function $f(p_i)$ and fitness value $Fit(p_i)$ for *i*-th individual $Fit(p_i) = \begin{cases} \dfrac{1}{1+f(p_i)} & if\ f(p_i) \geq 0 \\ 1+\lvert f(p_i)\rvert & if\ f(p_i) < 0 \end{cases}$
5	Assign *trial(i)*=0 for each individual
6	**end**
7	**repeat** the following steps
8	***// Teaching-based employed bee //***
9	**for** i=1: N **do**
10	Create new solution using hybrid TLBO teaching strategy $z_{i,d} = \begin{cases} p_{i,d}^{old} + r_2\left(p_{teacher,d} - T_F \times p_{mean,d}\right) & if\ r_1 < 0.5 \\ p_{R1,d} + F\left(p_{R2,d} - p_{R3,d}\right) & else \end{cases}$
11	Calculate objective function $f(z_i)$ and fitness value $Fit(z_i)$
12	Apply NSM-score based survivor selection (NSM-TLABC_Case1) & (NSM-TLABC_Case3) $if\ z_{i\_NSM\_Score} > p_{i\_NSM\_Score}$ survivor is z_i else survivor is p_i end
13	if p_i does not enhance *trial(i)= trial(i)+1*, otherwise *trial(i)*=0
14	**end**
15	***// Learning-based onlooker bee //***
16	Calculate selection probability $sp_i = \dfrac{fit(p_i)}{\sum_{i=1}^{SN} fit(p_i)}$
17	**for** i=1: N **do**
18	Apply the roulette selection approach and select p_k based on sp_i
19	Generate new solution using TLBO learning strategy $z_k = \begin{cases} p_k + rand\ (p_k - p_j) & if\ f(p_k) \leq f(p_j) \\ p_k + rand\ (p_j - p_k) & if\ f(p_j) > f(p_k) \end{cases}$
20	Calculate objective function $f(z_k)$ and fitness value $Fit(z_k)$
21	Apply NSM-score based survivor selection-method (NSM-TLABC_Case2) **if** $z_{k\_NSM\_Score} > p_{k\_NSM\_Score}$ survivor is z_k **else** survivor is p_k **end**
22	if p_k does not enhance *trial(k)= trial(k)+1*, otherwise *trial(k)*=0 end
23	**end**
24	***// Generalized oppositional scout bee //***
25	**if** max (*trial (i)*) ≥ limit
26	Generate new solution p_i (randomly) and its generalized oppositional solution p_i^{op} Choose better one between p_i and p_i^{op}
27	$p_i = \begin{cases} p_i & if\ f(p_i) \leq f(p_i^{op}) \\ p_i^{op} & otherwise \end{cases}$
28	**end**
29	**until the** termination criterion is met
30	Display best solution (p_{best}) achieved so far

4.2. NSM-SFS

NSM-SFS [6] is a nature-inspired metaheuristic optimization technique developed by configuring of the original SFS [38] using the NSM update mechanism. The algorithm derives its power from updating the solutions in the population according to the NSM score. The NSM score is a more accurate measure of the fitness of a solution than the traditional fitness value. This allows the algorithm to focus on finding solutions that are both good and diverse, which can help to prevent premature convergence. NSM-SFS generates a random initial population and enhances the quality of existing solutions in the population with the diffusing and updating processes during the search process lifecycle. Algorithm 2 presents the pseudocode of the NSM-SFS algorithm.

Algorithm 2. NSM-SFS Algorithm

1	**Initialization**
2	Create an initial population and calculate the fitness value of each point $P \equiv \begin{bmatrix} p_{11} & \cdots & p_{1D} \\ \vdots & \ddots & \vdots \\ p_{N1} & \cdots & p_{ND} \end{bmatrix}_{N\times D}, f = \begin{bmatrix} f_1 \\ \vdots \\ f_N \end{bmatrix}_{N\times 1}$
3	Sort the population based on the fitness value and determine the best point in P (p_{best})
4	**repeat** the following steps
5	***// Diffusion process //***
6	**for** i=1: N **do**
7	Select two points from P randomly and select the i-th point with the ordinal-based method
8	**for** j=1: m (maximum diffusion number)
9	Create a new point using the Gaussian Walk
10	**if** rand < (walk=1)
11	$p_{inew,1} = Gaussian\left(\mu_{p_{best}}, \sigma\right) + (\varepsilon \times p_{best} - \varepsilon' p_i)$
12	**else**
13	$p_{inew,2} = Gaussian\left(\mu_p, \sigma\right),\ where\ \mu_p = \lvert p_i \rvert$
14	**end**
15	**end**
16	**end**
17	***// First updating process //***
18	Rank all points in P based on fitness function value $PBa_i = \frac{rank\ (p_i)}{N},\ \ i = 1, \ldots, N$
19	**for** i=1: N **do**
20	**for** j=1: D **do**
21	**if** $rand\ [0,1] \geq PBa_i$
22	$p_i'(j) = p_r(j) - \varepsilon\ (p_t(j) - p_i(j))$
23	**end**
24	**end**
25	**end**
26	Apply NSM-score based survivor selection (NSM-SFS_Case1, NSM-SFS_Case2, NSM-SFS_Case3 **if** ${p_i}'_{NSM\_Score} > p_{i\ NSM\_Score}$ survivor is p_i' **else** survivor is p_i **end**
27	***// Second updating process //***
28	Rank all points obtained from first update process based on the fitness value $PB_{ai}' = \frac{rank\ (p_i)}{N},\ \ i = 1, \ldots, N$
29	**for** i=1: N **do**
30	**if** $rand\ [0,1] \geq PBa_i$
31	$p_i'' = p_i' - \hat{\varepsilon} \times (p_t' - p_{best})\ \ if\ \varepsilon' \leq 0.5$
32	$p_i'' = p_i' + \hat{\varepsilon} \times (p_t' - p_r')\ \ \ if\ \varepsilon' > 0.5$
33	**end**
34	**end**
35	**until the** termination criterion is met
36	Display best solution (p_{best}) achieved so far

4.3. NSM-LSHADE-SPACMA

NSM-LSHADE-SPACMA [6] is an effective and powerful algorithm developed to increase the optimization performance of LSHADE-SPACMA [39]. The optimizer uses the NSM approach to perform population management. In other words, NSM-LSHADE-SPACMA determines which individuals in the population will survive and which will be killed with an NSM score value. Algorithm 3 presents the steps followed by the NSM-LSHADE-SPACMA algorithm in the optimization process.

Algorithm 3. NSM-LSHADE-SPACMA Algorithm

1	*Set function evaluation counter (FEs=0), generation counter (g=1), M_{CR}=0.5, M_F=0.5, and M_{FCP}=0.5*
2	Archive A=Ø, N_g=N_{init}
3	Initialize population P_g= ($x_{1,g}$, ..., $x_{N,g}$) and covariance matrix adaptation (CMA) parameters
4	**while** FEs<maxFEs
5	S_{CR}= Ø, S_F= Ø, and S_{FCP}= Ø
6	**for** i=1: *N* **do**
7	r_i =select randomly from [1, H]
8	$CR_{i,g}$= $randn_1$ (M_{CR_ri} , 0.1)
9	$FCP_{i,g}$= M_{FCP_ri}
10	**if** FEs < (maxFEs/2)
11	$F_{i,g}$= 0.45 +(0.1 × rand)
12	**else**
13	$F_{i,g}$= $randc_1$ + (M_{F_ri} , 0.1)
14	**end**
15	**end**
16	[$P_{LSHADE,g}$, $P_{CMA,g}$]=Split (P_g, FCP_g)
17	$V_{g,LSHADE}$ =Generate donor vectors using LSHADE
18	$V_{g,CMA}$ =Generate donor vectors using CMA
19	V_g =Concatenate ($V_{g,LSHADE}$, $V_{g,CMA}$)
20	U_g =Generate trial vectors (V_g, CR_g)
21	Evaluate U_g and update FEs
22	Update P_g using NSM approach
23	Store successful FCP_g, F_g, and CR_g
24	Update archive A
25	**if** (archive size)> |A|
26	delete randomly selected archive individuals
27	**end**
28	Update memory M_{CR}, M_{FCP}
29	**if** FEs > (maxFEs/2)
30	update memory M_F
31	**end**
32	$N_{g+1} = \text{round}\left[\left(\frac{N_{min} - N_{init}}{maxFEs}\right) \times FEs + N_{init}\right]$
33	**if** N_g < N_{g+1}
34	sort solutions based on fitness value and delete the lowest $N_g - N_{g+1}$ members
35	resize archive size |A| based on new |P|
36	**end**
37	Update CMA parameters
38	g++
39	**end**
40	Display the best solution achieved so far

5. Results and Discussions

This section summarizes the results of experimental studies conducted to evaluate the performance of NSM-based optimizers. In this direction, two subsections have been prepared. The first subsection presents the optimization results of the CEC 2022 global optimization problems. The second subsection elaborated on the results of four engineering problems.

5.1. Performance Analysis of NSM-based Metaheuristics on CEC 2022 Benchmark Functions

This subsection gives the optimization results of CEC 2022 benchmark functions obtained with NSM-TLABC, NSM-SFS, and NSM-LSHADE-SPACMA algorithms. The experimental studies used the source codes of the NSM-based algorithms, which were made available by the developers on the Matlab File Exchange platform (Click for source codes of NSM-based algorithms). Three versions of each algorithm are used and named as follows: NSM-TLABC_Case1, NSM-TLABC_Case2, NSM-TLABC_Case3, NSM-SFS_Case1, NSM-SFS_Case2, NSM-SFS_Case3, NSM-LSHADE-SPACMA_Case1, NSM-LSHADE-SPACMA_Case2, and NSM-LSHADE-SPACMA_Case3. The maximum number of fitness evaluations (maxFEs) was adopted as a termination criterion. The value of maxFEs is 200,000 and 1,000,000 for the 10 and 20 dimensional optimization of IEEE CEC 2022 benchmark problems, respectively. 6,480 (9 algorithms × 12 functions × 30 runs × 2 dimensions) data items are obtained from CEC 2022 experiments and used for statistical analysis.

This work uses the Friedman-rank test [42] to compare the performance of more than two algorithms. Friedman test results of NSM-based algorithms are given in Table 2. As can be seen in the table, the overall algorithm performance was determined based on the "*Mean rank*" value. In the table, a small Friedman score indicates the superior performance of the algorithm. Considering the 10-dimensional experiment, the NSM-LSHADE-SPACMA_Case2 algorithm exhibits the best result with a Friedman score of 4.30, while NSM-TLABC_Case1 gives the worst result with a Friedman score of 6.17. In 20-dimensional optimization, NSM-LSHADE-SPACMA versions performed better than all other compared algorithms. Among these versions, the NSM-LSHADE-SPACMA_Case1 comes to the fore with a Frieman score 3.58. Given that all experiments are together, it is observed that the NSM-LSHADE-SPACMA_Case2 algorithm ranked first with a mean Friedman score of 3.96.

Table 2. Friedman Test Results of NSM-Based Algorithms

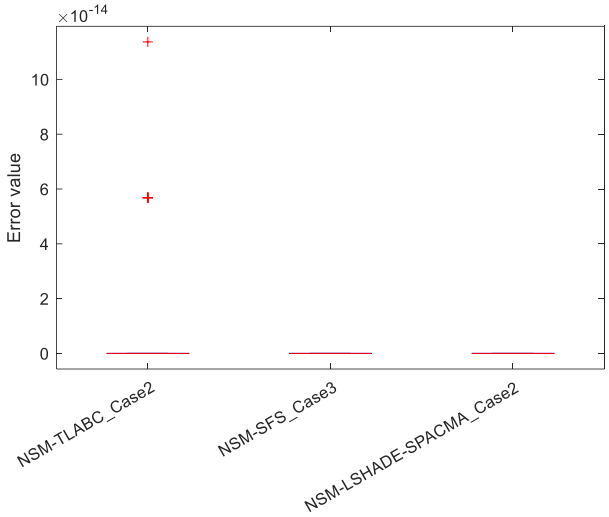
Algorithm	Dimension=10	Dimension=20	Mean rank	Overall rank
NSM-TLABC_Case1	6.17	6.73	6.45	9
NSM-TLABC_Case2	5.00	6.26	5.63	7
NSM-TLABC_Case3	5.72	6.62	6.17	8
NSM-SFS_Case1	5.34	4.57	4.95	6
NSM-SFS_Case2	4.86	4.68	4.77	5
NSM-SFS_Case3	4.44	5.08	4.76	4
NSM-LSHADE-SPACMA_Case1	4.46	**3.58**	4.02	2
NSM-LSHADE-SPACMA_Case2	**4.30**	3.62	**3.96**	1
NSM-LSHADE-SPACMA_Case3	4.66	3.82	4.24	3

Wilcoxon test [43] is applied for pairwise comparison between NSM-LSHADE-SPACMA_Case2 and competitive algorithms. Table 3 gives the Wilcoxon test results. The reason behind selecting the NSM-LSHADE-SPACMA_Case2 algorithm is that it is the winner of the Friedman test. As can be seen in Table 3, there are three scores in each cell. The first score shows the number of benchmark functions in which NSM-LSHADE-SPACMA_Case2 obtained better results than its rival. The second score indicates the number of problems for which the two algorithms produced similar results. The third score represents the number of benchmark functions where the competing algorithm is better than NSM-LSHADE-SPACMA_Case2. For example, in the 10-dimensional experiment between NSM-LSHADE-SPACMA_Case2 vs. NSM-TLABC_Case1, the Wilcoxon score is 8/3/1. In other words, the performance of NSM-LSHADE-SPACMA_Case2 is better in 8 test functions, while the result of the NSM-TLABC_Case1 algorithm is better in 1 function. In the 3 test functions, the algorithms could not outperform each other. In 13 out of 16 experiments, the NSM-LSHADE-SPACMA_Case2 outperformed the rival algorithm. The only exception is the 10-dimensional experiment between NSM-LSHADE-SPACMA_Case2 vs. NSM-SFS_Case3. To put it more clearly, the NSM-LSHADE-SPACMA_Case2 algorithm was only defeated by the NSM-SFS_Case3 algorithm with a score of 3/5/4.

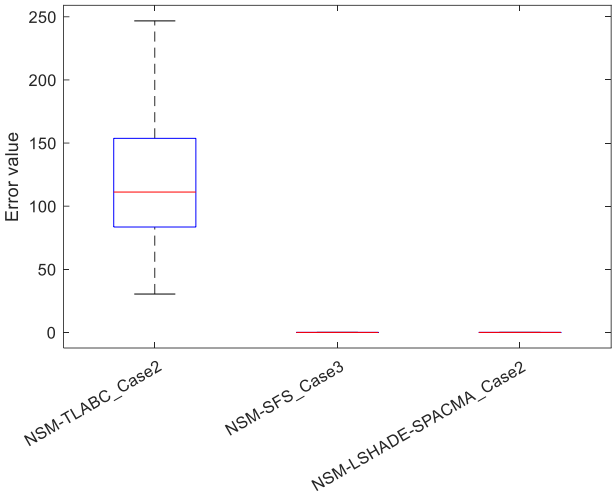
Table 3. Wilcoxon Pairwise Comparison Results

NSM-LSHADE-SPACMA_Case2 vs.	Dimension=10	Dimension=20
NSM-TLABC_Case1	8/3/1	11/1/0
NSM-TLABC_Case2	6/4/2	11/1/0
NSM-TLABC_Case3	8/3/1	11/1/0
NSM-SFS_Case1	6/3/3	5/5/2
NSM-SFS_Case2	5/4/3	6/4/2
NSM-SFS_Case3	3/5/4	6/4/2
NSM-LSHADE-SPACMA_Case1	1/10/1	3/7/2
NSM-LSHADE-SPACMA_Case3	3/8/1	1/10/1

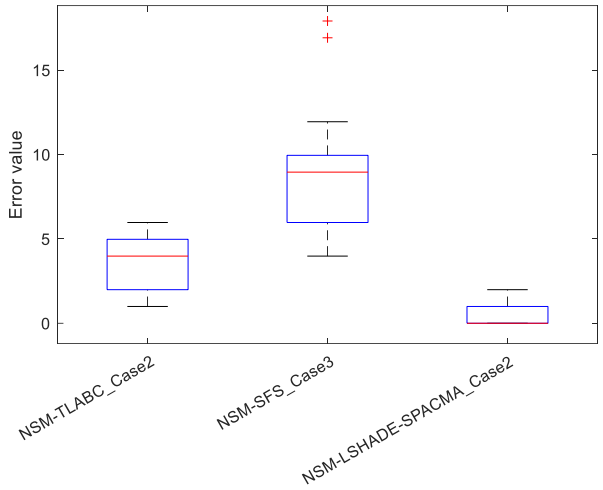
Box plots were prepared to observe the error statistics of the algorithms throughout 30 runs in 10- and 20-dimensional optimization of selected benchmark functions from CEC 2022. Figure 5 shows the box plots of F1, F4, F8, and F12 functions. In this figure, the algorithms NSM-TLABC_Case2, NSM-SFS_Case3 and NSM-LSHADE-SPACMA_Case2 represent the versions that give the best score according to the Friedman test (see Table 2).



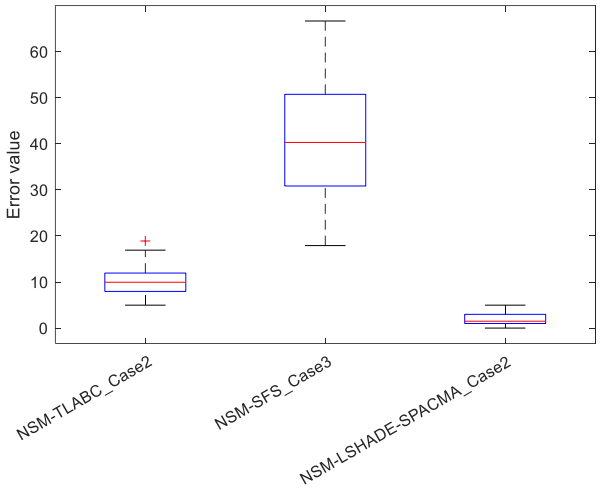
(a) F1 (Unimodal) D = 10



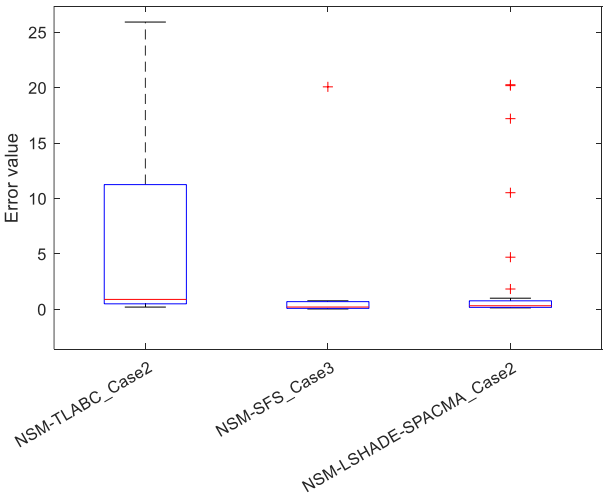
(b) F1 (Unimodal) D = 20



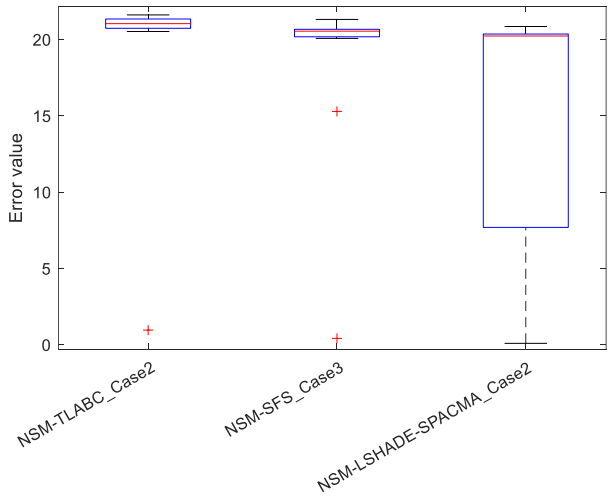
(c) F4 (Multimodal) D = 10



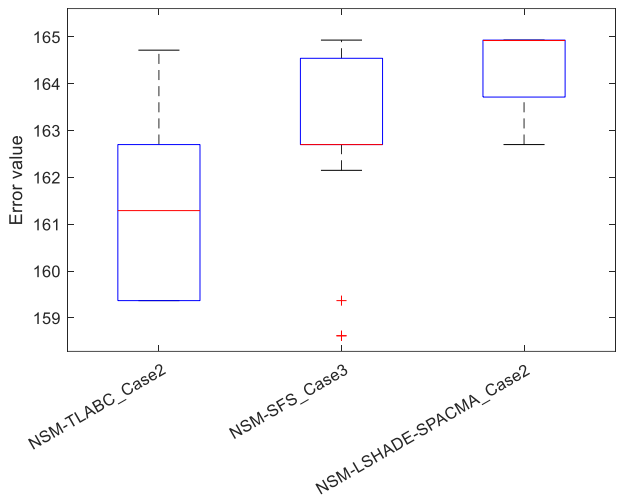
(d) F4 (Multimodal) D = 20



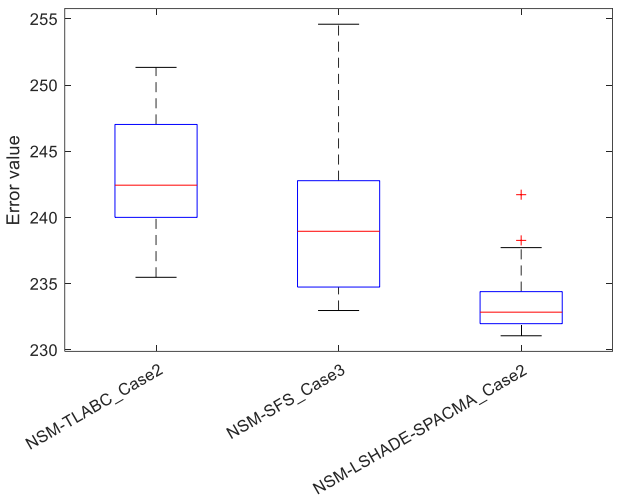
(e) F8 (Hybrid) D = 10



(f) F8 (Hybrid) D = 20



(g) F12 (Composition) D = 10



(h) F12 (Composition) D = 20

Figure 5. Box Plots of Best Versions of NSM-Based Algorithms on CEC 2022 Benchmark Functions

Upon examination of the box plots for the F1 unimodal problem, it is seen that NSM-SFS_Case3 and NSM-LSHADE-SPACMA_Case2 successfully converged to the global optimum in both problem sizes. Although NSM-TLABC_Case2 exhibits a remarkable search performance in 10 dimensions, the algorithm's performance is poor in 20 dimensions. The underlying reason behind it is that the algorithm cannot successfully fulfill the exploitation task in high-dimensional optimization. The box plots of the F4 multimodal problem show that the exploration ability of the NSM-LSHADE-SPACMA_Case2 is superior compared to the other two algorithms. On the other hand, NSM-SFS_Case3 gets stuck in local solution traps and converges prematurely. Upon examination of Figure 5e, it is seen that NSM-SFS_Case3 and NSM-LSHADE-SPACMA_Case2 algorithms have an overwhelming superiority over NSM-TLABC_Case2 in the 10-dimensional optimization of the hybrid F8 problem. In the 20-dimensional optimization of the same problem (Fig 5f), NSM-LSHADE-SPACMA_Case2 converges to a lower error value. From the box plots of the F12 composition problem (Figure 5g, h), it can be observed that NSM-TLABC_Case2 (in Dimension=10) and NSM-LSHADE-SPACMA_Case2 (in Dimension=20) balanced exploration and exploitation more successfully than competitors.

In a nutshell, this section presents the comparative performance analysis of NSM-based metaheuristic algorithms (NSM-TLABC, NSM-SFS, and NSM-LSHADE-SPACMA) in solving CEC 2022 benchmark problems. Given that the statistical analysis results and box plots are together, it is observed that the NSM-LSHADE-SPACMA_Case2 exhibits better optimization capability compared to other ones.

5.2. Application of NSM-LSHADE-SPACMA to Engineering Design Problems

In this subsection, NSM-LSHADE-SPACMA_Case2, which is determined as the most powerful of NSM-based algorithms, was applied to the solution of gear train design, tension/compression spring design, pressure vessel design, and cantilever beam design problems. Also, the performance of NSM-LSHADE-SPACMA_Case2 is compared with Rime Optimization Algorithm (RIME) [44], Nonlinear Marine Predator Algorithm (NMPA) [45], Northern Goshawk Optimization (NGO) [46], Kepler Optimization Algorithm (KOA) [47], Honey Badger Algorithm (HBA) [5], Artificial Gorilla Troops Optimizer (GTO) [48], Exponential Distribution Optimization (EDO) [49], and Hunger Games Search (HGS) [50]. The algorithms were run with the settings given in their original articles. The iteration number is 1000 for all algorithms.

- Gear train design

The optimized teeth numbers of gearwheels and the corresponding gear ratio cost are presented in Table 4. As per the results in the table, NSM-LSHADE-SPACMA_Case2, NMPA, NGO, HBA, GTO, and EDO algorithms obtained the best cost result with a value of 2.7009E-12. They are followed by RIME and KOA algorithms. The HGS algorithm gives the worst result (8.8876E-10). Figure 6 illustrates the convergence curves of optimizers in the gear train design problem. As is evident from the figure, NSM-LSHADE-SPACMA_Case2 reached the global optimum faster than the other algorithms.

Table 4. Optimization Results of Gear Train Design Problem

Algorithms	Optimized parameters				Optimal cost
	T_a	T_b	T_d	T_f	
NSM-LSHADE-SPACMA_Case2	49	19	16	43	**2.7009E-12**
RIME	34	13	20	53	2.3078E-11
NMPA	43	19	16	49	**2.7009E-12**
NGO	43	16	19	49	**2.7009E-12**
KOA	34	20	13	53	2.3078E-11
HBA	43	16	19	49	**2.7009E-12**
GTO	43	16	19	49	**2.7009E-12**
EDO	49	19	16	43	**2.7009E-12**
HGS	57	37	12	54	8.8876E-10

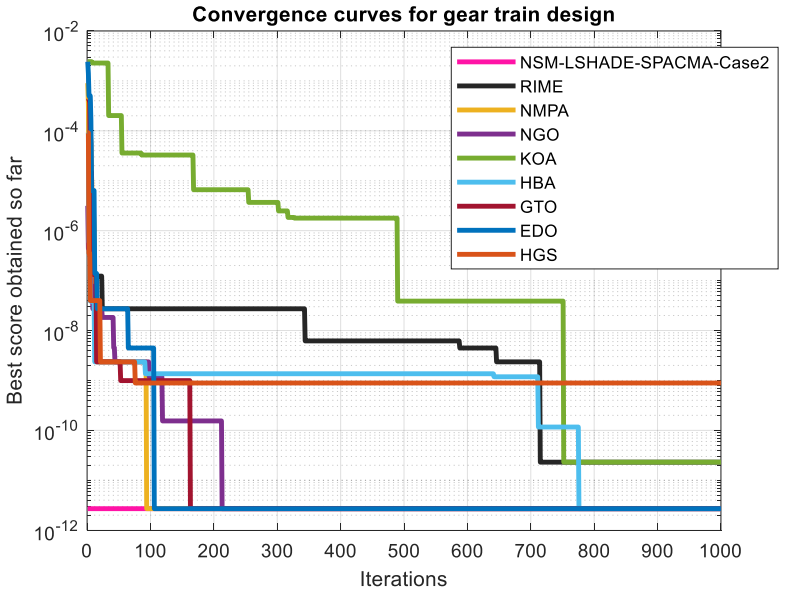


Figure 6. Convergence Curves of Metaheuristic Optimizers in Gear Train Design Problem

- Tension/compression spring design

The best settings of control variables and corresponding objective function results are depicted in Table 5. As per the results in the table, NSM-LSHADE-SPACMA_Case2, NMPA, NGO, HBA, and GTO algorithms obtained minimum coil weight with a value of 0.01266602. KOA gives the highest coil weight. The convergence behaviour of the competitive optimizer is depicted in Figure 7. From the convergence curves, it is observed that the NSM-LSHADE-SPACMA_Case2 successfully converged the best solution and reduced the iteration number. On the other hand, RIME and KOA methods get caught in local solution traps and converge prematurely.

Table 5. Optimization Results of Tension/Compression Spring Design Problem

Algorithms	Optimized parameters			Optimal weight
	d	D	N	
NSM-LSHADE-SPACMA_Case2	0.051897	0.361748	11.135056	**0.01266602**
RIME	0.051937	0.362229	11.313641	0.01270275
NMPA	0.051897	0.361748	10.729495	**0.01266602**
NGO	0.051897	0.361748	10.754378	**0.01266602**
KOA	0.050000	0.311346	15.000000	0.01323221
HBA	0.051897	0.361748	10.500000	**0.01266602**
GTO	0.051897	0.361748	11.415919	**0.01266602**
EDO	0.051897	0.361755	11.225818	0.01266653
HGS	0.051204	0.345163	11.685449	0.01266962

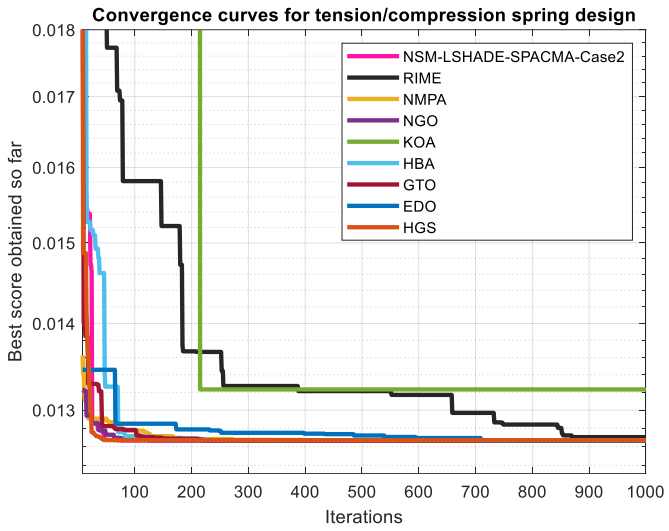


Figure 7. Convergence Curves of Metaheuristic Optimizers in Tension/Compression Spring Design Problem

- Pressure vessel design

The optimum parameters of the pressure vessel design problem and corresponding fabrication costs are given in Table 6. What is clear from the simulation results is that the best cost result (5885,332773) is achieved with NSM-LSHADE-SPACMA_Case2, HBA, and HGS algorithms. The respective decision variables are found to be 0.778168 for T_s, 0.384649 for T_h, 40.319618 for R, and 200 for L. The numeric results of the NMPA, NGO, and GTO algorithms are remarkable and close to each other. Figure 8 depicts the comparative convergence curves of optimizers. The illustrated charts in the figure show that the HGS algorithm exhibits a better convergence speed. Admittedly, the convergence ability of the HBA, NGO, and NSM-LSHADE-SPACMA_Case2 algorithms was also impressive.

Table 6. Optimization Results of Pressure Vessel Design Problem

Algorithms	Optimized parameters				Optimal cost
	T_s	T_h	R	L	
NSM-LSHADE-SPACMA_Case2	0.778168	0.384649	40.319618	200	**5885,332773**
RIME	0.888095	0.445929	45.981860	133.819439	6131,386806
NMPA	0.778168	0.384649	40.319618	199.999996	5885,332786
NGO	0.778168	0.384649	40.319618	199.999999	5885,332775
KOA	1.157765	0.573756	47.995036	145.907363	9291,834612
HBA	0.778168	0.384649	40.319618	200	**5885,332773**
GTO	0.778168	0.384649	40.319619	199.999985	5885,332808
EDO	0.778720	0.385479	40.333025	199.893895	5891,657726
HGS	0.778168	0.384649	40.319618	200	**5885,332773**

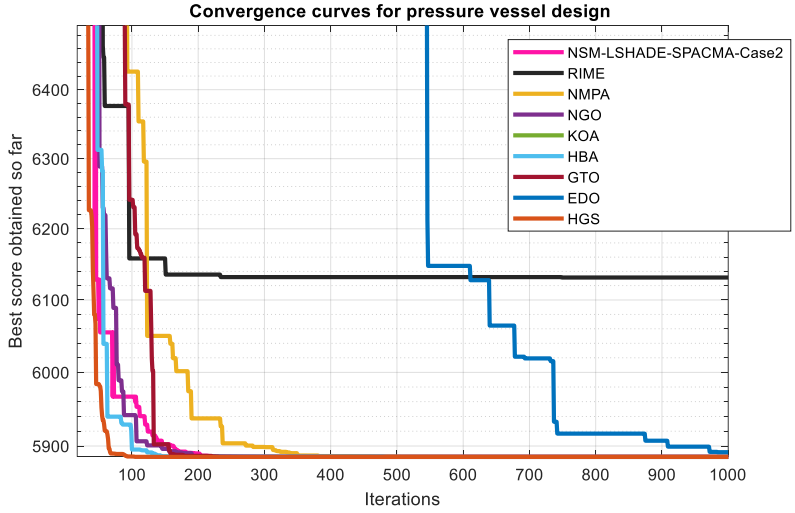


Figure 8. Convergence Curves of Metaheuristic Optimizers in Pressure Vessel Design Problem

- Cantilever beam design

Table 7 gives the optimized block lengths and corresponding cantilever beam weights. As per the results in the table, NSM-LSHADE-SPACMA_Case2, NMPA, and NGO algorithms obtained the optimal weight, and then followed by HBA and GTO algorithms. KOA algorithm gives the worst result. The reason behind the poor performance of the KOA can be that it gets stuck in local solution traps and convergence prematurely. Figure 9 depicts the change in the objective function over the iterations. From the figure, it can be seen that NSM-LSHADE-SPACMA_Case2 rapidly converged to the optimal weight result after 100 iterations.

Table 7. Optimization Results of Cantilever Beam Design Problem

Algorithms	Optimized parameters					Optimal weight
	x_1	x_2	x_3	x_4	x_5	
NSM-LSHADE-SPACMA_Case2	6.016015	5.309173	4.494329	3.501474	2.152665	**1.339956**
RIME	6.194038	5.423264	4.313638	3.502088	2.076832	1.342215
NMPA	6.016014	5.309174	4.494329	3.501474	2.152665	**1.339956**
NGO	6.016679	5.310302	4.493161	3.502687	2.150836	**1.339956**
KOA	6.990157	15.187510	5.099891	3.831937	1.850069	2.056676
HBA	6.018072	5.305518	4.497259	3.499427	2.153394	1.339957
GTO	6.015852	5.308207	4.494019	3.505487	2.150108	1.339957
EDO	5.984261	5.335545	4.493889	3.450257	2.219157	1.340546
HGS	6.028996	5.297238	4.480894	3.506450	2.160339	1.339972

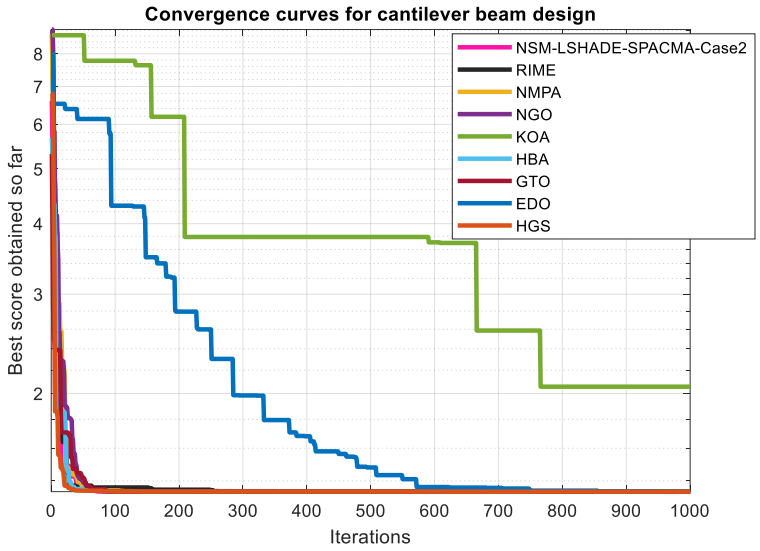


Figure 9. Convergence Curves of Metaheuristic Optimizers in Cantilever Beam Design Problem

6. Conclusions

This paper assesses the optimization ability of NSM-based metaheuristic algorithms in solving global optimization and constrained real-world engineering design problems. Firstly, NSM-TLABC, NSM-SFS, and NSM-LSHADE-SPACMA methods are applied to the optimization of CEC 2022 benchmark functions. The NSM-based algorithm that performs best in the global optimization is then used to find optimal solutions for the gear train design, tension/compression spring design, pressure vessel design and cantilever beam design problems. The results are compared with RIME, NMPA, NGO, KOA, HBA, GTO, EDO, and HGS algorithms. Based on the extensive experimental studies, the conclusions of the present research can be summarized as follows:

- Considering the Friedman test results, it is noticed that the best-performing variants of NSM-based algorithms are NSM-TLABC_Case2, NSM-SFS_Case3, and NSM-LSHADE-SPACMA_Case2 in the IEEE CEC 2022 global optimization problems.
- Among all NSM-based algorithms, NSM-LSHADE-SPACMA_Case2 stands out as the best optimiser with an average rank of 3.96 Friedman score.
- The winner of the Wilcoxon test is NSM-LSHADE-SPACMA_Case2. The algorithm gave statistically better results in 13 of 16 experiments. In other words, the NSM-LSHADE-SPACMA_Case2 showed a success rate of 81.25% against its competitors.

- Box plot analysis shows that the NSM-LSHADE-SPACMA algorithm exhibits better exploration and exploitation-exploration balance compared to other ones.
- The best objective function results are calculated with the NSM-LSHADE-SPACMA algorithm for gear train design, tension/compression spring design, pressure vessel design, and cantilever beam design problems as 2.7009E-12, 0.01266602, 5885,332773 and 1.339956, respectively.

In conclusion, this study strongly reports that NSM-LSHADE-SPACMA_Case2 performs better for global and engineering design problems. The strength of the NSM-based algorithm is that it performs population management using the NSM strategy. Based on this, it can be said that it is possible to increase the optimisation capacity of other population-based metaheuristic algorithms by hybridising them with NSM. In this way, high-quality solutions can be achieved for non-convex real-world engineering design problems.

References

[1] L. Abualigah, D. Yousri, M. Abd Elaziz, A. A. Ewees, M. A. Al-Qaness, and A. H. Gandomi, "Aquila optimizer: a novel meta-heuristic optimization algorithm," *Computers & Industrial Engineering*, vol. 157, 107250, 2021.

[2] O. Altay, "Chaotic slime mould optimization algorithm for global optimization," *Artificial Intelligence Review*, vol. 55, no. 5, pp. 3979-4040, 2022.

[3] P. Civicioglu and E. Besdok, "Bezier Search Differential evolution algorithm for numerical function optimization: A comparative study with CRMLSP, MVO, WA, SHADE and LSHADE," *Expert Systems with Applications*, vol. 165, 113875, 2021.

[4] A. Özkış and A. Babalık, "Solving constrained engineering design problems with multi-objective artificial algae algorithm," *Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi*, vol. 29, no. 2, pp. 183-193, 2023.

[5] F. A. Hashim, E. H. Houssein, K. Hussain, M. S. Mabrouk, and W. Al-Atabany, "Honey Badger Algorithm: New metaheuristic algorithm for solving optimization problems," *Mathematics and Computers in Simulation*, vol. 192, pp. 84-110, 2022.

[6] H. T. Kahraman, M. Katı, S. Aras, and D. A. Taşci, "Development of the Natural Survivor Method (NSM) for designing an updating mechanism in metaheuristic search algorithms," *Engineering Applications of Artificial Intelligence*, vol. 122, 106121, 2023.

[7] V. Hayyolalam and A. A. P. Kazem, "Black widow optimization algorithm: a novel meta-heuristic approach for solving engineering optimization problems," *Engineering Applications of Artificial Intelligence*, vol. 87, 103249, 2020.

[8] K. Zervoudakis and S. Tsafarakis, "A mayfly optimization algorithm," *Computers & Industrial Engineering*, vol. 145, 106559, 2020.

[9] Y. Zhang and Z. Jin, "Group teaching optimization algorithm: A novel metaheuristic method for solving global optimization problems," *Expert Systems with Applications*, vol. 148, 113246, 2020.

[10] I. Ahmadianfar, O. Bozorg-Haddad, and X. Chu, "Gradient-based optimizer: A new metaheuristic optimization algorithm," *Information Sciences*, vol. 540, pp. 131-159, 2020.

[11] M. H. Qais, H. M. Hasanien, and S. Alghuwainem, "Transient search optimization: a new meta-heuristic optimization algorithm," *Applied Intelligence*, vol. 50, 3926-3941, 2020.

[12] M. S. Braik, "Chameleon Swarm Algorithm: A bio-inspired optimizer for solving engineering design problems," *Expert Systems with Applications*, vol. 174, 114685, 2021.

[13] F. MiarNaeimi, G. Azizyan, and M. Rashki, "Horse herd optimization algorithm: A nature-inspired algorithm for high-dimensional optimization problems," *Knowledge-Based Systems*, vol. 213, 106711, 2021.

[14] M. Braik, A. Sheta, and H. Al-Hiary, "A novel meta-heuristic search algorithm for solving optimization problems: capuchin search algorithm," *Neural Computing and Applications*, vol. 33, no.7, pp. 2515-2547, 2021.

[15] F. A. Hashim, K. Hussain, E. H. Houssein, M. S. Mabrouk, and W. Al-Atabany, "Archimedes optimization algorithm: a new metaheuristic algorithm for solving optimization problems," *Applied Intelligence*, vol. 51, pp. 1531-1551, 2021.

[16] J. O.Agushaka, A. E. Ezugwu, and L. Abualigah, "Dwarf mongoose optimization algorithm". *Computer Methods in Applied Mechanics and Engineering*, vol. 391, 114570, 2022.

[17] J. S. Pan, L. G. Zhang, R. B. Wang, V. Snášel, and S. C. Chu, "Gannet optimization algorithm: A new metaheuristic algorithm for solving engineering optimization problems," *Mathematics and Computers in Simulation*, vol. 202, pp. 343-373, 2022.

[18] D. Połap and M. Woźniak, "Red fox optimization algorithm," *Expert Systems with Applications*, vol. 166, 114107, 2021.

[19] M. Dehghani, Š. Hubálovský, and P. Trojovský, "Tasmanian devil optimization: a new bio-inspired optimization algorithm for solving optimization algorithm," *IEEE Access*, vol. 10, 19599-19620, 2022.

[20] T. S. Ayyarao, N. S. S. Ramakrishna, R. M. Elavarasan, N. Polumahanthi, M. Rambabu, G. Saini, B. Khan, and B. Alatas, "War strategy optimization algorithm: a new effective metaheuristic algorithm for global optimization," *IEEE Access*, vol. 10, pp. 25073-25105, 2022.

[21] I. Naruei and F. Keynia, "Wild horse optimizer: A new meta-heuristic algorithm for solving engineering optimization problems," *Engineering with Computers*, vol. 38, pp. 3025-3056, 2022.

[22] M. Dehghani, Z. Montazeri, E. Trojovská, and P. Trojovský, "Coati Optimization Algorithm: A new bio-inspired metaheuristic algorithm for solving optimization problems," *Knowledge-Based Systems*, vol. 259, 110011, 2023.
[23] M. Abdel-Basset, R. Mohamed, M. Jameel, and M. Abouhawwash, "Nutcracker optimizer: A novel nature-inspired metaheuristic algorithm for global optimization and engineering design problems," *Knowledge-Based Systems*, vol. 262, 110248, 2023.
[24] S. Xian and X. Feng, "Meerkat optimization algorithm: A new meta-heuristic optimization algorithm for solving constrained engineering problems," *Expert Systems with Applications*, vol. 231, 120482, 2023.
[25] M. Azizi, U. Aickelin, H. A. Khorshidi, and M. B. Shishehgarkhane, "Energy valley optimizer: a novel metaheuristic algorithm for global and engineering optimization," *Scientific Reports*, vol. 13, no. 1, 2023.
[26] Q. Zhang, H. Gao, Z. H. Zhan, J. Li, and H. Zhang, "Growth Optimizer: A powerful metaheuristic algorithm for solving continuous and discrete global optimization problems," *Knowledge-Based Systems*, vol. 261, 110206, 2023.
[27] J. O. Agushaka, A. E. Ezugwu, and L. Abualigah, "Gazelle optimization algorithm: a novel nature-inspired metaheuristic optimizer," *Neural Computing and Applications*, vol. 35, no. 5, pp. 4099-4131, 2023.
[28] M. Kaveh, M. S. Mesgari, and B. Saeidian, "Orchard Algorithm (OA): A new meta-heuristic algorithm for solving discrete and continuous optimization problems," *Mathematics and Computers in Simulation*, vol. 208, pp. 95-135, 2023.
[29] M. Han, Z. Du, K. Yuen, H. Zhu, Y. Li, and Q. Yuan, "Walrus Optimizer: A novel nature-inspired metaheuristic algorithm," *Expert Systems with Applications*, vol. 239, 122413, 2023.
[30] D. Zhu, S. Wang, C. Zhou, S. Yan, and J. Xue "Human memory optimization algorithm: A memory-inspired optimizer for global optimization problems," *Expert Systems with Applications*, vol. 237, 121597, 2024.
[31] E. S. M. El-kenawy, N. Khodadadi, S. Mirjalili, A. A. Abdelhamid, M. M. Eid, and A. Ibrahim, "Greylag Goose Optimization: Nature-inspired optimization algorithm," *Expert Systems with Applications*, vol. 238, 122147, 2023.
[32] S. B. Aydemir, "A novel arithmetic optimization algorithm based on chaotic maps for global optimization," *Evolutionary Intelligence*, vol. 16, no. 3, pp. 981-996, 2023.
[33] S. Ekinci, D. Izci, R. A. Zitar, A. R. Alsoud, and L. Abualigah, "Development of Lévy flight-based reptile search algorithm with local search ability for power systems engineering design problems," *Neural Computing and Applications*, vol. 34, no. 22, pp. 20263-20283, 2022.
[34] H. Bakir, U. Guvenc, H. T. Kahraman, and S. Duman, "Improved Lévy flight distribution algorithm with FDB-based guiding mechanism for AVR system optimal design," *Computers & Industrial Engineering*, vol. 168, 108032, 2022.
[35] C. Zhong, G. Li, Z. Meng, and W. He, "Opposition-based learning equilibrium optimizer with Levy flight and evolutionary population dynamics for high-dimensional global optimization problems," *Expert Systems with Applications*, vol. 215, 119303, 2023.
[36] H. Bakır, S. Duman, U. Guvenc, and H. T. Kahraman, "Improved adaptive gaining-sharing knowledge algorithm with FDB-based guiding mechanism for optimization of optimal reactive power flow problem," *Electrical Engineering*, vol. 105, no. 5, pp. 3121-3160, 2023.
[37] X. Chen, B. Xu, C. Mei, Y. Ding, and K. Li, "Teaching–learning–based artificial bee colony for solar photovoltaic parameter estimation," *Applied Energy*, vol. 212, pp. 1578-1588, 2018.
[38] H. Salimi, "Stochastic fractal search: a powerful metaheuristic algorithm," *Knowledge-Based Systems*, vol. 75, pp. 1-18, 2015.
[39] A. W. Mohamed, A. A. Hadi, A. M. Fattouh, and K. M. Jambi, "LSHADE with semi-parameter adaptation hybrid with CMA-ES for solving CEC 2017 benchmark problems," *In 2017 IEEE Congress on Evolutionary Computation (CEC)*, pp. 145-152. IEEE, 2017.
[40] A. Kumar, K. V. Price, A. W. Mohamed, A. A. Hadi, and P. N. Suganthan, "Problem Definitions and Evaluation Criteria for the 2022 Special Session and Competition on Single Objective Bound Constrained Numerical Optimization Nanyang Technological University," *Tech. Rep*, 2022.
[41] L. Wang, Q. Cao, Z. Zhang, S. Mirjalili, and W. Zhao, "Artificial rabbits optimization: A new bio-inspired meta-heuristic algorithm for solving engineering optimization problems," *Engineering Applications of Artificial Intelligence*, vol. 114, 105082, 2022.
[42] S. García, A. Fernández, J. Luengo, and F. Herrera, "Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power," *Information Sciences*, vol. 180, no. 10, pp. 2044-2064, 2010.
[43] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 3-18, 2011.
[44] H. Su, D. Zhao, A. A. Heidari, L. Liu, X. Zhang, M. Mafarja, and H. Chen, "RIME: A physics-based optimization". *Neurocomputing*, vol. 532, pp. 183-214, 2023.
[45] A. S. Sadiq, A. A. Dehkordi, S. Mirjalili, and Q. V. Pham, "Nonlinear marine predator algorithm: A cost-effective optimizer for fair power allocation in NOMA-VLC-B5G networks," *Expert Systems with Applications*, vol. 203, 117395, 2022.
[46] M. Dehghani, Š. Hubálovský, and P. Trojovský, "Northern goshawk optimization: a new swarm-based algorithm for solving optimization problems," *IEEE Access*, vol. 9, pp. 162059-162080, 2021.

[47] M. Abdel-Basset, R. Mohamed, S. A. A. Azeem, M. Jameel, and M. Abouhawwash, "Kepler optimization algorithm: A new metaheuristic algorithm inspired by Kepler's laws of planetary motion," *Knowledge-Based Systems*, vol. 268, 110454, 2023.
[48] B. Abdollahzadeh, F. S. Gharehchopogh, and S. Mirjalili "Artificial gorilla troops optimizer: a new nature-inspired metaheuristic algorithm for global optimization problems," *International Journal of Intelligent* Systems, vol. 36, no. 10, pp. 5887-5958, 2021.
[49] M. Abdel-Basset, D. El-Shahat, M. Jameel, and M. Abouhawwash, "Exponential distribution optimizer (EDO): a novel math-inspired algorithm for global optimization and engineering problems," *Artificial Intelligence Review*, vol. 56, no. 9, pp. 9329-9400, 2023.
[50] Y. Yang, H. Chen, A. A. Heidari, and A. H. Gandomi, "Hunger games search: Visions, conception, implementation, deep analysis, perspectives, and towards performance shifts," *Expert Systems with Applications*, vol. 177, 114864, 2021.

Author Contribution
Hüseyin Bakır: Conceptualization, Methodology, Investigation, Software, Formal analysis, Data curation, Writing - original draft.

Conflict of Interest
The author declares that there is no conflict of interest regarding the publication of this paper.

Ethical Approval and Informed Consent
It is declared that during the preparation process of this study, scientific and ethical principles were followed, and all the studies benefited from are stated in the bibliography.

Availability of data and material
Not applicable.

Plagiarism Statement
This article has been scanned by iThenticate ™.