e-Prime - Advances in Electrical Engineering, Electronics and Energy 5 (2023) 100243

Contents lists available at ScienceDirect

e-Prime - Advances in Electrical Engineering, Electronics and Energy

journal homepage: www.elsevier.com/locate/prime

Tyrannosaurus optimization algorithm: A new nature-inspired meta-heuristic algorithm for solving optimal control problems

Venkata Satya Durga Manohar Sahu, Padarbinda Samal *, Chinmoy Kumar Panigrahi

School of Electrical Engineering, KIIT Deemed to be University, India

ARTICLE INFO

Keywords:
Tyrannosaurus (T-Rex) optimization algorithm (TROA)
Optimization techniques
Optimal control problem (OCP)
Benchmark problems

ABSTRACT

Recently, the optimal control problem has gained much importance for solving practical problems. In this regard, the meta-heuristic algorithms are proven to be effective while solving these problems effectively and efficiently. However, these algorithms may not be effective for solving all the optimization problems as per the no free lunch theorem. Thus, there is always a scope of development of new meta-heuristic algorithms. This paper proposes a new hunting-based optimization algorithm called Tyrannosaurus (T-Rex) optimization algorithm (TROA). This algorithm is inspired by the hunting behavior of the T-Rex. This algorithm was tested on 12 benchmark problems and 4 practical optimal control problems. The performance of the TROA is compared with seven famous optimization techniques, i.e. Differential Evolution (DE) Algorithm, Particle Swarm Optimization (PSO), Grey Wolf Optimizer (GWO), White Shark Optimizer (WSO), Jellyfish Search (JS), Crow Search Algorithm (CSA), Golden Eagle Optimization (GEO). The results obtained for the proposed method have given better when compared to these methods.

1. Introduction

For the past few decades, optimization techniques have played a major role in obtaining/finding the optimal value for many engineering problems. This optimization deals with single or multi-objective functions with sets of decision/state and control variables. Moreover, this optimization is classified into two types (1) deterministic methods use the linear and non-linear search for finding the local minimum and optimal value, so it is simple to understand but scientifically not as sound as previously believed (2) Stochastic methods utilize the random variables to find the global best value and the advantages of these algorithms are easy to use, simplicity, gradient-free nature, independency of the problem, and flexibility but it is slow and not accurate enough.

On the other hand, the recent development of many metaheuristic algorithms made it easier to solve difficult/complex problems compared to traditional methods. Convergence analysis of this algorithm is hard and tricky in mathematical analysis because the relationships between the many parts of metaheuristic algorithms are nonlinear, complex, and random, and this problem is hard to solve. Also, these different ways have been used a lot to solve different kinds of problems. Therefore, we shall concentrate on these metaheuristic algorithms.

Even though classic and modern optimization algorithms work well, none of them can guarantee that the best global optimal solutions will be found for all optimization problems. This method gave us the idea to make a new optimization algorithm that can solve optimization problems more effectively. Because of this theory, we made a new optimization algorithm and tested it on several benchmark problems.

In this paper, a new method called the Tyrannosaurus algorithm (TROA) is introduced based on the social behavior of Tyrannosaurus. The paper [1,2] explained how the Tyrannosaurus hunts, lived, and existed. The prey-predictor hunting procedure inspires this method. Some researcher says that T-Rex hunt in single, and some say that T-Rex hunt in the group just like the apex predator. Here by considering a single T-Rex. Apart from this algorithm, the random position of the prey and T-Rex location is generated. Depending on the nearest position of the prey, i.e. Target prey is fixed. The T-Rex chases its prey while the

* Corresponding author.
E-mail address: padarbindasamal87@gmail.com (P. Samal).

https://doi.org/10.1016/j.prime.2023.100243
Received 2 February 2023; Received in revised form 19 May 2023; Accepted 9 August 2023
Available online 15 August 2023

Fig. 1. Tyrannosaurus [13].

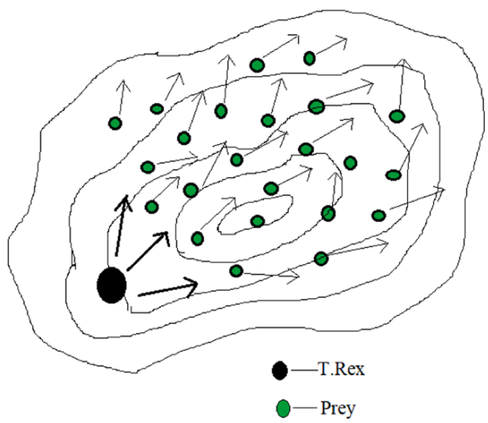

Fig. 2. Hunting region of prey and hunter.

prey tries to escape from the T-Rex; as a result, the chasing and hunting process goes on till the T-Rex hunting process succeeds .sometimes, the prey escapes from the T-Rex. The TROA is also compared with nine algorithms Differential Evolution (DE) [3], Particle Swarm Optimization (PSO) [4], Golden Eagle Optimizer(GEO) [5], Grey Wolf Optimizer [6], Jellyfish Search Optimizer (JS) [7], Crow Search Algorithm(CSA) [8], White Shark Optimizer (WSO) [9], on the 12 benchmark problems, and 4 practical control problems.

The main contributions of this research paper are:

- Usage of hunting speed and success rate of the T-Rex for catching the prey by varying the hunting speed *tr* and success rate *sr* operators during optimization.
- Utilization of prey running speed *pr* operator which enables the prey to protect itself. In other words, the hunting is failed and the prey has escaped which provides variations in solutions during the entire optimization process.
- Incorporation of estimation rate (*ER*) operator which decides whether the hunting is successful or not.
- Demonstration of superiority of the proposed approach by statistical comparison with other famous meta-heuristic algorithms while solving 12 benchmark test functions, and 4 practical control problems.

The rest of the paper comprised of as follows, Section 2 describes the proposed algorithm, Section 3 shows the effectiveness of the TROA for solving benchmark test functions, Section 4 shows the application of TROA for solving optimal control problems, the conclusion and future scope are provided in Section 5.

2. Tyrannosaurus optimization algorithm

In this section, the inspiration for the proposed method is first discussed. Then, the mathematical model is provided along with an algorithm and flowchart.

2.1. Inspiration

Over 66 million years ago, these dinosaurs used to live and especially the T-Rex lived in western North America. Tyrannosaurus is the king of all the dinosaurs. Tyrannosaurus rex (rex meaning "king" in Latin), often called T. rex or colloquially T-Rex, is one of the best-represented theropods. It is a genus of large theropod dinosaurs with a height of 3.66–3.96 m (12–13 ft), length of 12.3–12.4 m (40.4–40.7 ft), and an average weight of 8.4 metric tons (9.3 short tons) to 14 metric tons (15.4 short tons).

It is one of the biggest carnivore creatures in the world, and it is assumed that T-Rex is an apex predator like Lion, Wolf, etc. at the same time, some experts have suggested that the T-Rex was a scavenger [10]. Today Most of the experts accept that T-Rex was both an active predator and a scavenger. But there was a doubt about whether T-Rex is a scavenger or an apex predator [11,12]. It is known for its hunting behaviour as it has a strong bite force among all dinosaurs of its type (Fig. 1).

2.2. Proposed algorithm

In this section, we discuss the proposed algorithm's detailed procedure, algorithm, and flow chart. This algorithm consists of mainly three steps as follows:

2.1 Initialization
2.2 Hunting
2.3 Selection

Let us in see the steps in detail.

2.2.1. Initialization

The TROA is a population-based algorithm which randomly generate the number of prey in search space. Let us consider x x be the prey location or position, and it is randomly generated in a search space within the upper limit and lower limit, as shown in the Eq. (1).

$$X_i = rand(np, dim) * (ub - lb) + lb \tag{1}$$

where, $X_i = [x_1, x_2, \cdots x_n]$ is the prey location and $i = 1,2 \cdots n$, where, n is the dimension, np is the no of population, dim is the dimension of the search space, ub is the upper limit and lb is the lower limit.

In Fig. 2, green circles represent the prey position, and the black circle represents the T-Rex position.

2.2.2. Hunting and chasing

T-Rex hunts its prey just like the apex predator like a lion, wolf etc. When T-Rex sees its nearest prey, it tries to hunt. Sometimes as prey defends itself from hunting, or it may escape. The T-Rex hunting involves juveniles chasing and catching prey, so while the T-Rex hunts, it hunts randomly

$$\mathrm{X}_{new} = \left\{ \begin{array}{ll} x_{new} & if\ rand() < Er \\ \text{Random} & else \end{array} \right\} \tag{2}$$

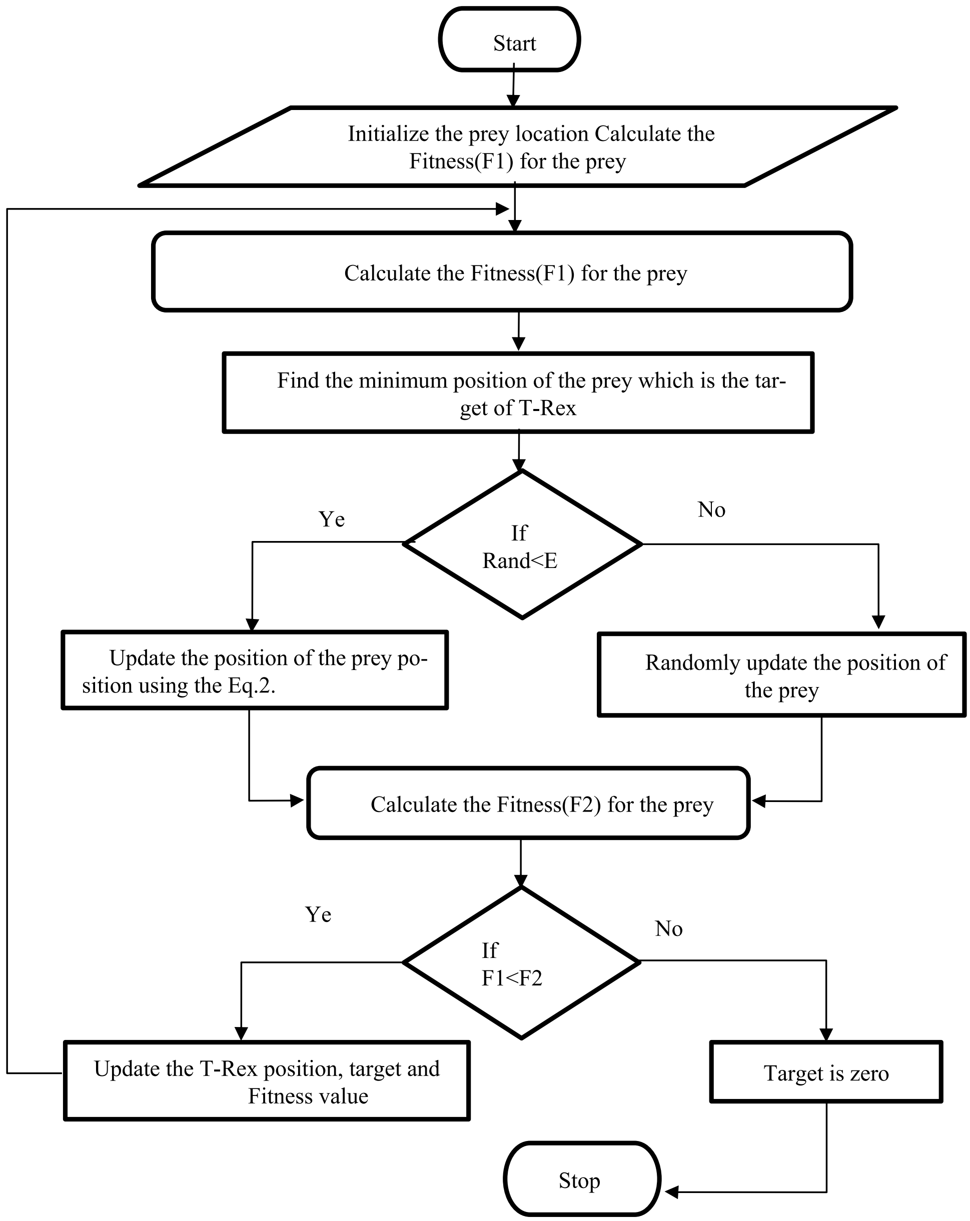

Fig. 3. Flow chart of the TROA.

Table 1
Benchmark problems utilized to test TROA.

Function number	Type of the fssunction	Function name	Range	Fmin
1	Unimodal	$f(z) = \sum_{i=1}^{n} z_i^2$	[−100,100]	0
2	Unimodal	$f(z) = \sum_{i=0}^{n} \lvert z_i\rvert + \prod_{i=0}^{n} \lvert z_i\rvert$	[−10,10]	0
3	Unimodal	$f(z) = \sum_{i=1}^{d} (\sum_{j=1}^{i} z_j)^2$	[−100,100]	0
4	Unimodal	$f(z) = \max\{\lvert z_i\rvert,\ 1 \le i \le n\}$	[−100,100]	0
5	Unimodal	$f(z) = \sum_{i=0}^{n} i z_i^4 + random[0,1)$	[−128,128]	0
6	Multimodal	$f(z) = \sum_{i=1}^{n} [z_i^2 - 10\cos(2\pi z_i) + 10]$	[−5.12,5.12]	0
7	Multimodal	$f(z) = -20exp\left(0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} z_i^2}\right) - exp\left(\frac{1}{n}\sum_{i=1}^{n} cos(2\pi z_i)\right) + 20 + e$	[−3.2,3.2]	0
8	Multimodal	$f(z) = 1 + \frac{1}{4000}\sum_{i=1}^{n} z_i^2 - \prod_{i=1}^{n} cos\left(\frac{z_i}{\sqrt{i}}\right)$	[−600,600]	0
9	Multimodal	$f(z) = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2}} (z_i - a_{ij})\right)^{-1}$	[−65,65]	1
10	Multimodal	$f(z) = 4z_1^2 - 2.1z_1^4 + \frac{1}{3}z_1^6 + z_1 z_2 - 4z_2^2 + 4z_2^4$	[−5,5]	−1.0316
11	Multimodal	$f(z) = \left(z_2 - \frac{5.1}{4\pi^2}z_1^2 + \frac{5}{\pi}z_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)cos\, z_1 + 10$	[−5,5]	0.398
12	Multimodal	$f(z) = \begin{bmatrix} 1 + (z_1 + z_2 + 1)^2 \\ \begin{pmatrix} 19 - 14z_1 + 3z_1^2 - 14z_2 + 6z_1 z_2 \\ +3z_2^2 \end{pmatrix} \end{bmatrix} \times \begin{bmatrix} 30 + (2z_1 - 3z_2)^2 \times \\ \begin{pmatrix} 18 - 32z_i + 12z_1^2 + 48z_2 - 36z_1 z_2 \\ +27z_2^2 \end{pmatrix} \end{bmatrix}$	[−2,2]	3

where, the *Er* is the Estimation of reaching the scattered prey, i.e., When the T-Rex starts hunting, then the prey starts to shatter and hunts the prey by updating its location as shown in the Eq. (2).

$$x_{new} = x + rand() * sr * (tpos * tr - \text{target} * \text{pr}) \tag{3}$$

where, *sr* is the success rate of hunting which lies in between [0.1,1]. If the success rate is 0, it means the prey has escaped, the hunting has failed, and the prey location has to be updated accordingly. Target is the minimum position of the prey to the T-Rex position. T-Rex running rate is *tr* But here, we consider only the prey has been hunted and only a single T-Rex. It is assumed that the T-Rex runs at an average speed of 30 miles/h and walks speed of 6.7 miles /h [14]. So, the T-Rex running rate consider between [0.067,0.3]. *pr* is the prey running rate, which lies in between [0,1] but here, we must consider that the prey running speed should be less than T-Rex speed (Fig. 3).

2.2.3. Selection

The selection process depends on the location of prey, i.e. present location of the target prey and the previous location. If the T-Rex fails to hunt, the prey location becomes zero if the prey runs away or protects itself from hunting. It is realized by comparing the fitness function.

$$X_i^{k+1} = \begin{cases} update\ the\ target\ position & if \quad f(X) < f(X_{new}) \\ target\ is\ zero & otherwise \end{cases} \tag{4}$$

where, $f(X)$ is the fitness function for the initial randomly prey location, and $f(X_{new})$ is the fitness function for the updated prey location.

2.3. Algorithm

1 Start
2 Initialize the position of the prey randomly using Eq. (1).
3 Calculate the fitness using Eq. (1).
4 Find the prey's nearest position and make it the target for the T-Rex.
5 Start the T-Rex hunting process using Eqs. (2) and (3).
6 Calculate the fitness for the new position of the prey.
7 If the $f(X) < f(X_{new})$update the prey position and target
8 If the condition fails, then the target is equal to zero
9 Stop

2.4. Pseudo code

Initialize the prey position using the Eq. (1).
Calculate the fitness of the prey and find the min position of the prey
Find the target prey
Start the while loop
Move the T-Rex randomly
If rand() < Er
Update the position of the prey using Eq. (2).
Else
Update the prey position randomly
End the if
Calculate the new fitness using Eq. (3)
If $f(X) < f(X_{new})$
Update the prey position and target
Else
Target is zero
End the if
Find the best value
$t = t + 1$
end while
return

3. Simulation results and analysis

In order to test the newly developed algorithm, we have tested on 12

Table 2
Parameter analysis of TROA.

S. No	Function		Condition1 (SR=0.2, TR=0.3 , PR=0.25)	Condition2 (SR=0.6, TR=0.3 , PR=0.25)	Condition3 (SR=0.8, TR=0.3, PR=0.25)	Condition4 (SR=0.2, TR=0.3 , PR=0.2)	Condition5 (SR=0.6, TR=0.3 , PR=0.2)	Condition6 (SR=0.8, TR=0.3 , PR=0.2)	Condition7 (SR=0.2 TR=0.2 , PR=0.1)	Condition8 (SR=0.6, TR=0.2 , PR=0.1)	Condition9 (SR=0.8, TR=0.2 , PR=0.1)
	F1	Best	2.1971e-246	1.3520e-240	4.1056e-248	1.7461e-236	8.7193e-227	4.9998e-227	1.8875e-238	2.6110e-243	2.9078e-233
		Maximum	1.8497e-213	3.7758e-212	4.1233e-211	2.9189e-210	2.4523e-206	1.7093e-208	2.6272e-217	4.4696e-218	4.4696e-218
		Mean	1.8772e-214	3.7764e-213	4.1268e-212	3.6159e-211	2.4523e-207	1.7093e-209	2.8345e-218	8.1939e-219	8.1939e-219
		Standard deviation	0	0	0	0	0	0	0	0	0
		Rank	2	4	1	6	9	8	5	3	7
	F2	Best	5.8641e-119	2.6744e-123	2.8430e-127	4.5482e-123	2.8889e-122	2.4883e-124	6.0953e-127	1.6373e-124	3.6842e-125
		Maximum	1.4922e-108	5.5145e-108	5.9051e-105	9.6777e-107	5.1482e-105	6.9576e-101	5.8467e-103	9.4243e-104	1.5950e-98
		Mean	2.3316e-109	6.0392e-109	5.9094e-106	9.6781e-108	5.2740e-106	6.9576e-102	1.1698e-104	2.7718e-105	3.1901e-100
		Standard deviation	4.8435e-109	1.7313e-108	1.8672e-105	3.0603e-107	1.6241e-105	2.2002e-101	8.2684e-104	1.4295e-104	2.2557e-99
		Rank	9	6	1	7	8	5	2	4	3
	F3	Best	1.7297e-227	6.0349e-237	7.0165e-231	2.3293e-234	2.4721e-230	1.9992e-238	4.4729e-249	1.0100e-233	4.0345e-239
		Maximum	1.6515e-204	4.1222e-208	8.7430e-201	3.9178e-204	3.0346e-203	1.2045e-200	7.2576e-197	3.1763e-198	1.0317e-202
		Mean	1.6515e-205	4.1277e-209	8.7433e-202	3.9178e-205	3.0346e-204	1.2053e-201	1.4515e-198	8.5806e-200	2.0647e-204
		Standard deviation	0	0	0	0	0	0	0	0	0
		Rank	9	4	8	5	7	3	2	6	1
	F4	Best	3.8423e-116	5.4044e-120	2.2915e-120	1.3824e-116	1.1516e-117	1.3769e-118	8.5118e-117	1.2187e-116	3.5152e-119
		Maximum	3.5202e-105	8.5642e-105	8.9035e-104	1.6357e-107	1.9222e-106	4.8235e-108	2.5401e-105	5.1239e-104	2.4331e-105
		Mean	3.7345e-106	8.5642e-106	1.0826e-104	2.6488e-108	1.9396e-107	5.2808e-109	2.5401e-106	5.1243e-105	2.4349e-106
		Standard deviation	1.1066e-105	2.7082e-105	2.8126e-104	5.4998e-108	6.0726e-107	1.5159e-108	8.0326e-106	1.6203e-104	7.6933e-106
		Rank	9	2	1	8	5	4	6	7	3
	F5	Best	1.1017e-05	8.7239e-06	4.3031e-06	2.5950e-06	1.8453e-06	3.1376e-05	5.0696e-06	3.0460e-05	8.3497e-06
		Maximum	2.8516e-04	2.3453e-04	2.3997e-04	3.8766e-04	9.0262e-05	3.3549e-04	4.0403e-04	2.5088e-04	2.3443e-04
		Mean	1.0564e-04	1.2575e-04	7.6608e-05	8.7793e-05	3.7024e-05	1.1422e-04	1.0805e-04	1.0221e-04	7.5255e-05
		Standard deviation	8.3437e-05	8.0042e-05	8.1710e-05	1.1731e-04	2.3012e-05	9.8612e-05	1.1785e-04	6.5249e-05	7.0596e-05
		Rank	7	6	3	2	1	9	4	8	5
	F6	Best	0	0	0	0	0	0	0	0	0
		Maximum	0	0	0	0	0	0	0	0	0
		Mean	0	0	0	0	0	0	0	0	0
		Standard deviation	0	0	0	0	0	0	0	0	0
		Rank	1	1	1	1	1	1	1	1	1
	F7	Best	8.8818e-16	8.8818e-16	8.8818e-16	8.8818e-16	8.8818e-16	8.8818e-16	8.8818e-16	8.8818e-16	8.8818e-16
		Maximum	8.8818e-16	8.8818e-16	8.8818e-16	8.8818e-16	8.8818e-16	8.8818e-16	8.8818e-16	8.8818e-16	8.8818e-16
		Mean	8.8818e-16	8.8818e-16	8.8818e-16	8.8818e-16	8.8818e-16	8.8818e-16	8.8818e-16	8.8818e-16	8.8818e-16
		Standard deviation	0	0	0	0	0	0	0	0	0
		Rank	1	1	1	1	1	1	1	1	1
	F8	Best	0	0	0	0	0	0	0	0	0
		Maximum	0	0	0	0	0	0	0	0	0
		Mean	0	0	0	0	0	0	0	0	0
		Standard deviation	0	0	0	0	0	0	0	0	0
		Rank	1	1	1	1	1	1	1	1	1
	F9	Best	0.9982	2.9821	2.8847	1.9920	1.8608	2.5186	1.0368	1.1021	2.9821
		Maximum	10.7536	8.0759	12.6705	8.0130	7.8951	7.9022	7.8740	7.9155	10.7632
		Mean	5.2721	6.1564	4.5238	5.7844	5.4517	5.8965	3.7142	3.9913	6.5786
		Standard deviation	3.5430	2.1807	3.1178	2.2306	2.6469	2.3081	2.1253	2.3333	2.6579
		Rank	1	8	7	5	4	6	2	3	9
	F10	Best	−1.0316	−1.0309	−1.0315	−1.0315	−1.0315	−1.0310	−1.0315	−1.0315	−1.0313
		Maximum	−1.0228	−1.0235	−1.0012	−1.0253	−1.0050	−1.0046	−1.0045	−1.0259	−1.0096
		Mean	−1.0289	−1.0274	−1.0254	−1.0289	−1.0251	−1.0207	−1.0257	−1.0294	−1.0269
		Standard deviation	0.0032	0.0027	0.0101	0.0022	0.0083	0.0080	0.0088	0.0018	0.0064
		Rank	1	9	5	3	6	8	4	2	7

(*continued on next page*)

Table 2 (*continued*)

S. No	Function		Condition1 (SR=0.2, TR=0.3 , PR=0.25)	Condition2 (SR=0.6, TR=0.3 , PR=0.25)	Condition3 (SR=0.8, TR=0.3, PR=0.25)	Condition4 (SR=0.2, TR=0.3 , PR=0.2)	Condition5 (SR=0.6, TR=0.3 , PR=0.2)	Condition6 (SR=0.8, TR=0.3 , PR=0.2)	Condition7 (SR=0.2 TR=0.2 , PR=0.1)	Condition8 (SR=0.6, TR=0.2 , PR=0.1)	Condition9 (SR=0.8, TR=0.2 , PR=0.1)
	F11	Best	0.3979	0.3980	0.3979	0.3980	0.3979	0.3979	0.3979	0.3979	0.3979
		Maximum	0.3985	0.4002	0.4008	0.3986	0.3995	0.4013	0.3983	0.3988	0.4011
		Mean	0.3981	0.3988	0.3996	0.3983	0.3984	0.3986	0.3981	0.3983	0.3988
		Standard deviation	1.9365e-04	8.2611e-04	9.4133e-04	2.2001e-04	5.6479e-04	0.0010	1.5513e-04	2.6875e-04	9.2336e-04
		Rank	2	9	7	8	4	5	1	3	6
	F12	Best	3.0366	3.0004	3.0000	3.0161	3.0390	3.0053	3.0063	3.0000	3.0013
		Maximum	7.6898	7.9540	10.7367	19.7104	9.2071	4.8598	7.6164	4.9948	7.7947
		Mean	4.5718	4.9211	4.1353	6.3205	4.2449	4.0419	3.8201	3.4595	4.0133
		Standard deviation	1.9217	1.7968	2.4133	5.8517	1.9534	0.6352	1.3910	0.6472	1.5228
		Rank	8	3	2	7	9	5	6	1	4
	Avg Rank		4.25	4.5	3.166667	4.5	4.666667	4.666667	2.916667	3.333333	4

benchmark problems, shown in Table 1.

To find the best parameters to be utilized in the TROA, we have performed the parameter analysis on the 12 benchmark problems by considering the population size as 50, the number of iterations as 500, and 50 runs. The values obtained for different SR, TR, and PR values are shown in Table 2. The average best value obtained for the 50 runs optimal values, the best minimum value for obtained 50 values of the 50 runs, the maximum value for obtained 50 values of the 50 runs, and the standard deviation for obtained 50 values of the 50 runs are represented in Table 2.

From Table 2, Condition 7 (SR=0.2, TR=0.2, PR=0.1) obtained the best values among the considered nine considerations as it provides the best or second best fitness value in most of the cases. Fig. 4 displays the parameter space, search history, and convergence curve of the 12 mentioned functions. The parameter space indicates the dimensions of the initialized population. The search history illustrates the search space for the test functions which is represented in the blue circle. Whereas, the red circle indicates the nearest prey location or target to the T-Rex. The convergence curve represents the variation of fitness value with respect to the number of iterations for a sample simulation run.

Table 3 shows the result obtained with the Wilcoxon rank sum test [24] for pairwise comparison with proposed technique and other algorithms. The result shows that the p values are less than 0.05 for all test functions except F8. This indicates that TROA performs better than other algorithms except for F8.

3.1. Comparative analysis of the TROA

In this section, we have compared the proposed algorithm with seven different optimization algorithms such as; Differential Evolution (DE) [3], Particle Swarm Optimization (PSO) [4], Golden Eagle Optimizer (GEO) [5], Grey Wolf Optimizer [6], Jellyfish Search Optimizer (JS) [7], Crow Search Algorithm(CSA) [8], White Shark Optimizer (WSO) [9], on the 12 benchmark test problems.

Table 4 shows the comparison of the fitness values of the proposed algorithm with other seven existing algorithms. From Table 4, it is clear that TROA has given the best result among the other techniques due to better exploration and exploitation while solving the benchmark test problems. It is observed that the standard deviation value is less in comparison with other techniques. So, TROA is more stable.

From Fig. 5, it can be observed that the TROA converges faster than other algorithms. The y-axis represents the fitness value on a logarithmic scale. This logarithmic scale helps in identifying the difference between the fitness curve values of the different algorithms. The proposed algorithm shows a large difference in the fitness value in comparison to other approaches. This is due to the dynamic position updation strategy of the target prey. The TROA shows better results in comparison to other techniques as it covers the entire search space in very less iterations.

3.2. Comparison of TROA with other algorithms

TROA is simple to program depending on the natural hunting behavior of the T-Rex. According to the parameter setting rule, if there are few numbers of parameters then it makes it easier to adjust. The TROA contains only three parameters in comparison to other meta-heuristic algorithms such as Differential Evolution (DE) [3], Particle Swarm Optimization (PSO) [4], Golden Eagle Optimizer (GEO) [5], Grey Wolf Optimizer [6], Jellyfish Search (JS) Optimizer [7], Crow Search Algorithm(CSA) [8], and White Shark Optimizer (WSO) [9]. The parameters are *sr, tr*, and *pr*. So, TROA is easier to adjust.

The TROA provides the best solution compared to GWO, WSO, GEO, and other hunting algorithms due to faster updation of the position of prey and T-Rex. In this algorithm, each position of the prey is attracted towards the best position in the group of prey like PSO and CSA. So, it convergences much faster than the other algorithms. The best position of TROA is computed for each iteration, as the T.Rex moves much nearer to the prey. So, the success rate increases. Hence, the diversity increases.

4. TROA for optimal control problems

In this section, to solve some real-time/optimal control problems were applied to TROA and was been compared with some optimization techniques. These problems include single link manipulator, single-link flexible manipulator, continuous stirred tank reactor(CSTR) problem, and Van der Pol oscillator.

According to the previous section, the TROA parameters setting is done in this section. This means the number of iterations is 500 and the population size is 50, the T-Rex run is 0.3, and the prey running away from the predator with the speed of *pr* is 0.25. The mathematical definition of these problems can be found in Refs. [15–22].

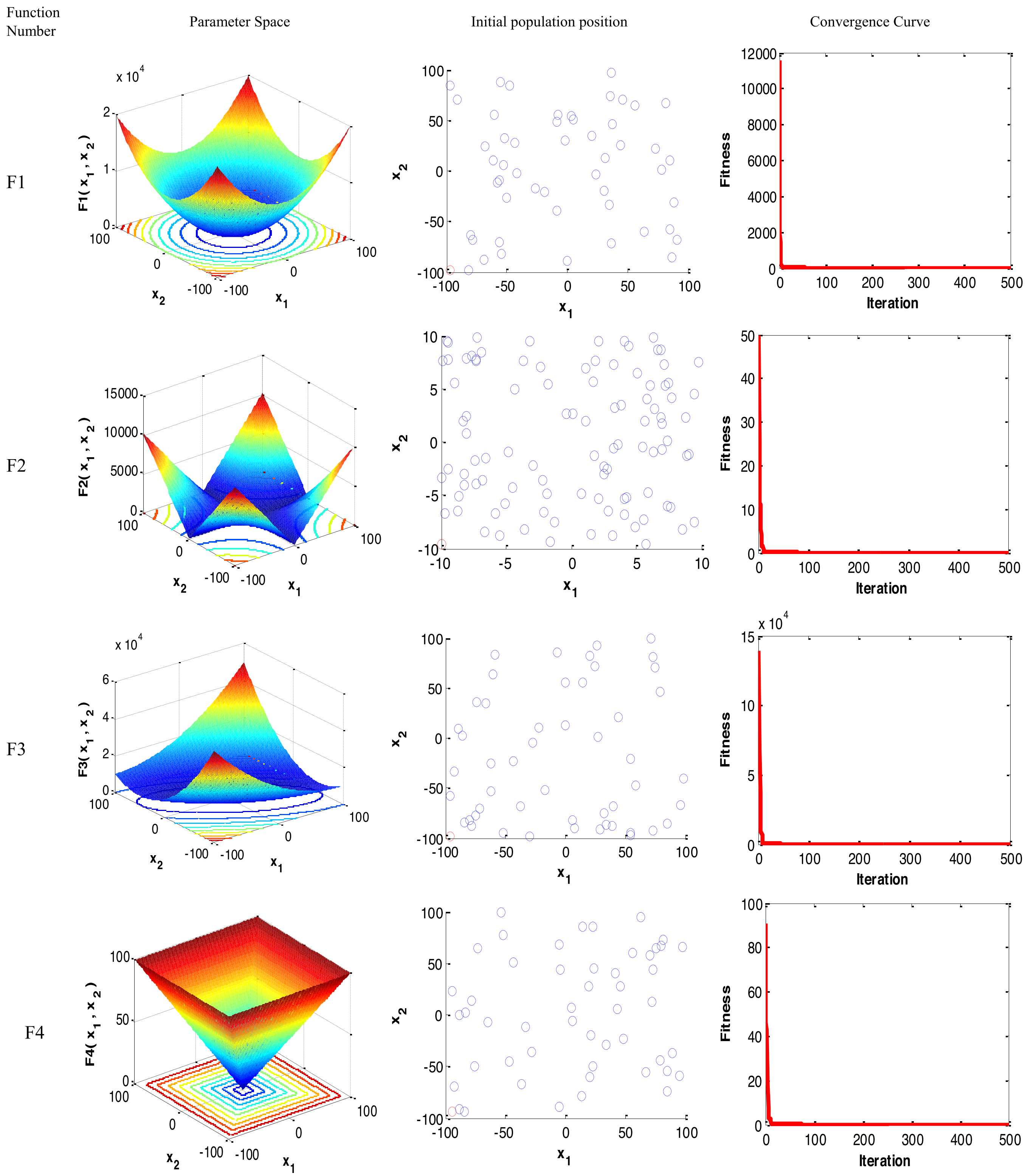

Fig. 4. Qualitative results of the selected functions.

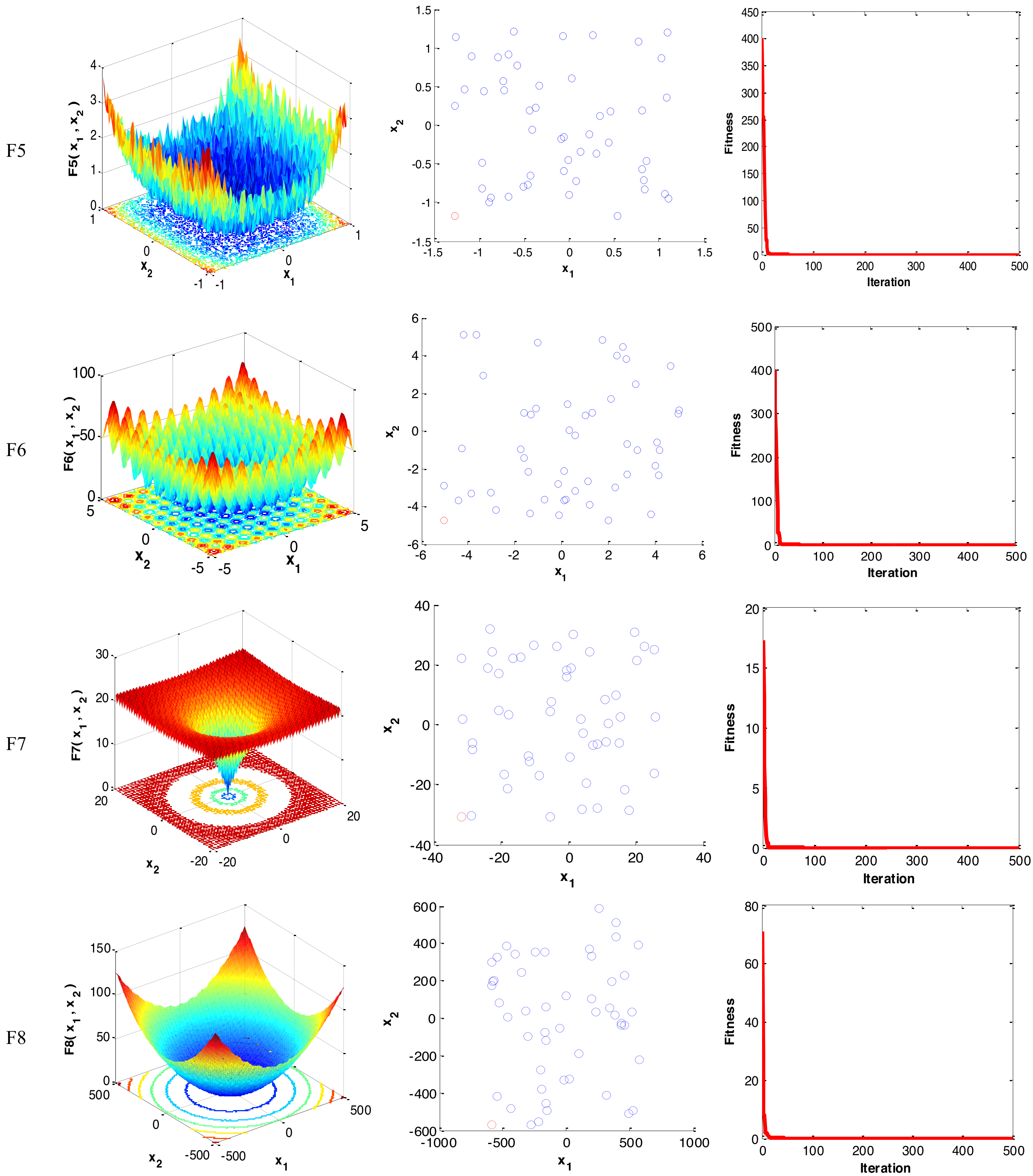

Fig. 4. (*continued*).

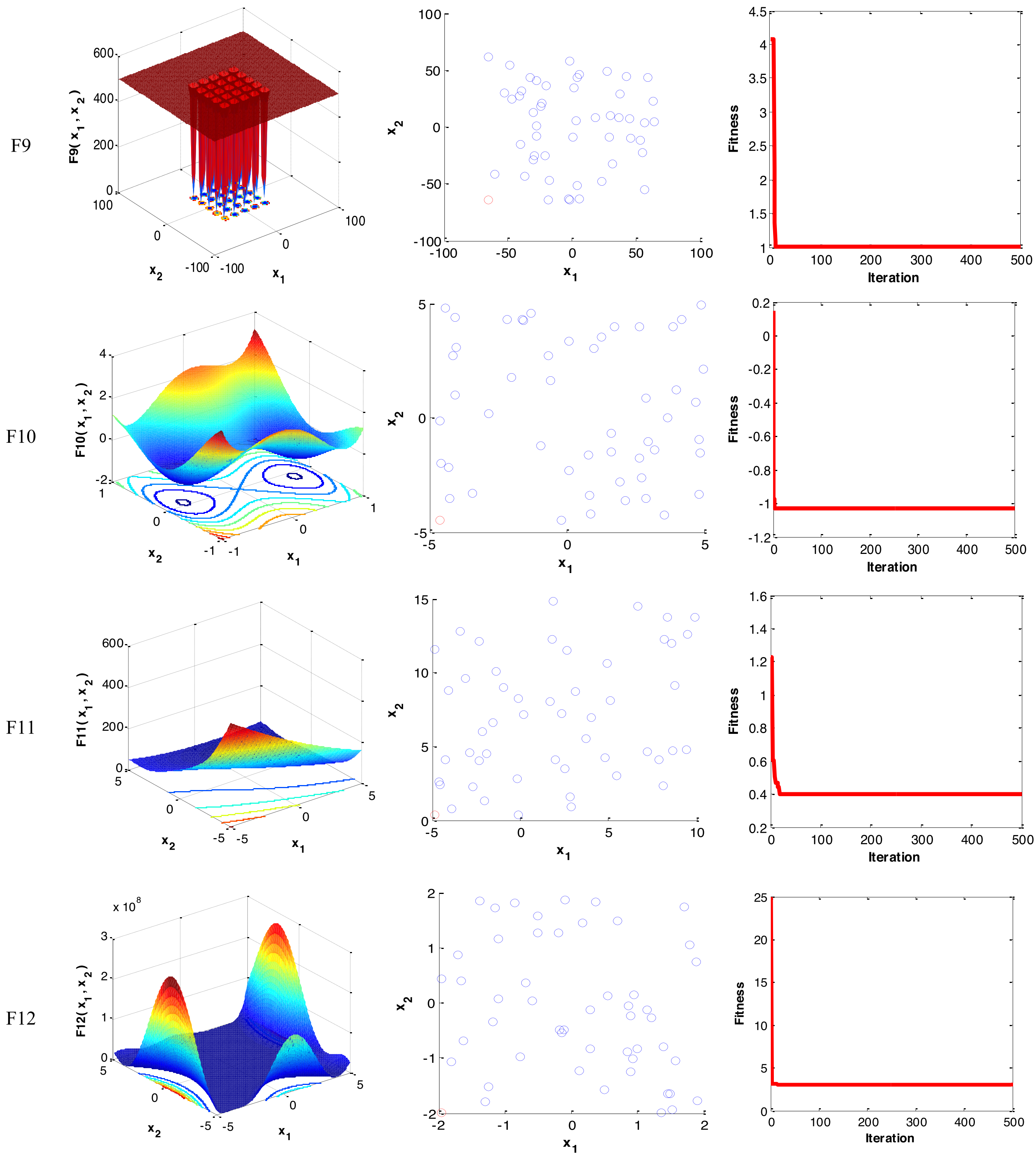

Fig. 4. (*continued*).

Table 3
Wilcoxon test results for benchmark function.

Function Name	TROA vs DE	TROA vs WSO	TROA vs CSA	TROA vs GWO	TROA vs JS	TROA vs GEO	TROA vs PSO
F1	7.0661e-18	7.0661e-18	7.0661e-18	7.0661e-18	7.0661e-18	7.0661e-18	7.0661e-18
F2	7.0661e-18	7.0661e-18	7.0661e-18	7.0661e-18	7.0661e-18	7.0661e-18	7.0661e-18
F3	1.8267e-04	1.8267e-04	1.8267e-04	1.8267e-04	1.8267e-04	1.8267e-04	1.8267e-04
F4	1.8267e-04	1.8267e-04	1.8267e-04	1.8267e-04	1.8267e-04	1.8267e-04	1.8267e-04
F5	1.8267e-04	1.8267e-04	1.8267e-04	1.8267e-04	1.8267e-04	1.8267e-04	1.8267e-04
F6	3.3111e-20	3.3111e-20	3.3111e-20	1.4155e-17	3.3111e-20	3.3111e-20	3.3111e-20
F7	6.3864e-05	6.3864e-05	6.3864e-05	5.5111e-05	6.3864e-05	6.3864e-05	6.3864e-05
F8	6.3864e-05	6.3864e-05	6.3864e-05	0.1681	0.1675	6.3864e-05	6.3864e-05
F9	6.2118e-20	3.9131e-18	2.4842e-18	2.2931e-06	3.3111e-20	3.5254e-17	2.0391e-12
F10	1.8267e-04	1.4939e-04	1.0997e-04	1.8267e-04	6.3864e-05	1.8267e-04	1.8267e-04
F11	3.3111e-20	7.6037e-19	3.3111e-20	5.6428e-16	3.3111e-20	7.0661e-18	3.3111e-20
F12	3.4937e-19	2.9797e-18	6.4157e-18	7.0661e-18	2.7547e-19	8.5606e-15	3.2433e-18

4.1. Continuous stirred reactor tank

In this subsection, the continuous stirred reactor tank is considered [21]. In this problem, the flow of a coolant through a coil installed in the reactor is regarded as the control variable that governs the reactor's first-order irreversible exothermic process. This CSTR is taken as an optimal control problem by considering the state equation as follows:

$$\begin{aligned}\dot{z}_1 &= -(2+u)(z_1+0.25)+(z_2+0.5)\exp\left(\frac{25z_1}{z_1+2}\right)\\ \dot{z}_2 &= 0.5-z_2-(z_2+0.5)\exp\left(\frac{25z_1}{z_1+2}\right)\end{aligned} \tag{5}$$

The variation from the dimensionless steady-state temperature is represented by z_1, while the divergence from the dimensionless steady-state concentration is represented by z_2. The control u stands for the adjustment of the cooling fluid's flow rate, which is introduced into the reactor by a coil. Finding the unconstrained u* that minimizes the performance index is the optimum control problem: In the initial condition of $Z(0) = [0 \quad 0]$, the performance of CSTR is measured by

$$J = \int_0^2 z_1^2 + z_2^2 + u^2 dt \tag{10}$$

The results of the TROA are compared with other techniques including DE, WSO, CSA, GWO, JS, and GEO. The Table 5 shows the maximum, mean, and standard deviation of all the algorithms for 500 iterations and 50 population sizes. It was clear that the best optimal value is 2.0630E-201 by TROA.

From Table 5, when comparing the TROA in terms of minimum performance is 99.99% more efficient than the other algorithms i.e. DE, WSO, CSA, GWO, JS, and GEO. When comparing the TROA in terms of maximum performance 10% more efficient than DE, 17% less efficient than WSO,13% more efficient than CSA,18.10% more efficient than GWO,5% less efficient than JS, and 13.07% efficient than GEO. When comparing the TROA in terms of mean performance 95.52% more efficient than DE, 92.14% more efficient than WSO, 99.63% more efficient than CSA, 5.17% more efficient than GWO,99.34% more efficient than JS, 99.63% more efficient than GEO.

Figs. 6 and 7, show that the TROA converges faster. Also, the mean and minimum value is less in TROA when compared with other algorithms.

From Fig. 8, it can be observed that the steady state temperature and the steady state concentration almost remain constant for TROA. Whereas, for the other algorithms there are variations of steady-state temperature and steady-state concentration. As a result for other algorithms, it is difficult to control. The TROA obtains nearly zero steady-state temperature, and zero steady-state concentration as shown in Fig. 9.

The Figs. 10 and 11 show the cooling fluid's flow rate of the CSTR. From Fig. 10 it can noticed that the lowest flow rate is obtained by the TROA.

4.2. Van der Pol oscillator

The van der Pol oscillator is an oscillator with nonlinear damping governed by the second-order differential equation described by

$$\ddot{y} + (y^2-1)\dot{y} + y = 0 \tag{7}$$

where, u is a scalar parameter representing the non-linearity and the intensity of the damping, and y is the location coordinate (a function of time t). Consider

$$z_1 = y \tag{8}$$

Using the Eq. (8), Eq. (7) can be rewritten as:

$$\begin{aligned}\dot{z}_1 &= z_2\\ \dot{z}_2 &= (1-z_1^2)z_2 - z_1\end{aligned} \tag{9}$$

With the initial condition of $Z(0) = [0 \quad 0]$, the performance of Van der Pol oscillator is measured by

$$J = \int_0^2 z_1^2 + z_2^2 + u^2 dt \tag{10}$$

The results of the TROA are compared with other techniques including DE, WSO, CSA, GWO, JS, GEO, and PSO. Table 6 shows the maximum, mean, and standard deviation of all the algorithms for 500 iterations and 50 population sizes. It was clear from this table that the best optimal value of zero is obtained by GWO. However, the TROA is the second best with the value of 5.9299E-08.

From Fig. 12 it can be observed that the TROA convergence is faster in comparison to DE, WSO, CSA, JS, GEO, and PSO. The TROA converges at 10th iteration as shown in Fig. 13.

4.3. Single-link manipulator model

In this section, a single link manipulator [17,22] is consider i.e. is robotic arm as shown in the Fig. 15. Length of the rod is l ,element mass is m,applied torque is u,angular position is θ. From this figure we can illustrate an equation as below

Table 4
Results of TROA and other algorithms with 500 iterations and 50 runs.

Function Number	Optimization algorithms	TROA	DE	WSO	CSA	GWO	JS	GEO	PSO
1.	Minimum	4.6E-238	2.4E-08	24.4844	1.1584	7.02E-35	1.15E-19	52,466	6.879E-07
	Maximum	4.8E-204	3.58E-07	204.2714	5.09	1.06E-32	1.28E-17	78,977	0.0024
	Mean	1.5E-205	1.04E-07	66.8668	2.5424	1.61E-33	1.54E-18	65,405	8.574E-05
	Standard Deviation	0	6.21E-08	37.1801	0.8459	2.21E-33	2.07E-18	6752.4	0.0003428
	RANK	1	4	7	6	2	3	8	5
2.	Minimum	3.1E-123	2.74E-05	0.8665	1.2334	1.22E-20	2.71E-11	25,491,000	0.0002253
	Maximum	3.4E-104	0.000169	4.2054	6.2339	2.44E-19	6.42E-10	2.77E+13	0.0335
	Mean	7.8E-106	6.7E-05	2.0333	3.3043	7.19E-20	2.21E-10	2.74E+12	0.0037
	Standard Deviation	4.8E-105	2.39E-05	0.7259	1.0974	5.36E-20	1.58E-10	6.66E+12	0.0055
	RANK	1	4	6	7	2	3	8	5
3.	Minimum	1.7E-234	15,845	158.3682	131.7346	1.44E-11	0.0047	54,542	36.9159
	Maximum	7.3E-200	41,855	1001.7	617.0519	8.18E-07	0.5079	206,740	217.6414
	Mean	2.3E-201	26,600	437.9357	289.9584	2.72E-08	0.0875	121,680	119.3462
	Standard Deviation	0	5938.5	181.3907	90.0574	1.16E-07	0.1074	38,152	39.5034
	RANK	1	7	6	5	2	3	8	4
4.	Minimum	5.4E-124	0.9662	4.4329	3.6992	2.28E-09	7.17E-08	80.2792	0.62
	Maximum	6.4E-102	11.581	13.9795	10.78	1.11E-07	6.16E-07	91.2138	2.2043
	Mean	1.3E-103	2.1106	8.0455	6.7322	2.34E-08	2.37E-07	86.6208	1.3893
	Standard Deviation	9E-103	1.8117	1.8908	1.6511	2.2E-08	1.12E-07	2.5331	0.3266
	RANK	1	5	7	6	2	3	8	4
5.	Minimum	1.32E-07	0.0116	0.0132	0.0126	0.000145	0.000452	59.9082	0.004
	Maximum	0.000315	0.0377	0.1003	0.0745	0.0054	0.003	164.6145	0.023
	Mean	8.13E-05	0.0257	0.0382	0.0372	0.0011	0.0017	108.1176	0.0097
	Standard Deviation	6.92E-05	0.0061	0.0214	0.0136	0.000895	0.000621	23.1359	0.004
	RANK	1	5	7	6	2	3	8	4
6.	Minimum	0	116.786	16.5606	9.4273	0	0.0362	374.2596	16.9143
	Maximum	0	179.2605	150.9324	36.943	19.1707	26.9922	484.0314	83.5765
	Mean	0	156.5743	48.7502	19.5967	2.2802	9.5574	431.1227	36.1966
	Standard Deviation	0	11.3363	28.2921	7.7536	4.1005	5.118	29.711	13.532
	RANK	1	7	5	4	2	3	8	6
7.	Minimum	8.88E-16	5.03E-05	2.7596	2.3156	3.29E-14	6.89E-11	20.2621	0.0003111
	Maximum	8.88E-16	0.0002	5.9338	6.4416	6.48E-14	7.87E-10	20.8786	0.0047
	Mean	8.88E-16	8.84E-05	4.0808	4.1025	4.34E-14	3.25E-10	20.589	0.0013
	Standard Deviation	0	2.79E-05	0.6775	0.87	5.02E-15	1.8E-10	0.157	0.0011
	RANK	1	4	7	6	2	3	8	5
8.	Minimum	0	6.97E-08	1.1449	0.6875	0	0	456.8191	5.06E-06
	Maximum	0	0.0089	2.8214	1.0597	0.0197	9.99E-16	696.342	0.0454
	Mean	0	0.000246	1.5627	0.9728	0.0015	2.89E-17	585.1748	0.0102
	Standard Deviation	0	0.0013	0.3463	0.0642	0.0043	1.43E-16	56.9371	0.0122
	RANK	1	4	7	6	3	2	8	5
9.	Minimum	1.0061	0.998	0.998	0.998	0.998	0.998	4.269	0.998
	Maximum	8.2747	1.992	0.998	1.9998	12.6705	0.998	454.3508	7.874
	Mean	5.1333	1.0179	0.998	1.0832	3.4671	0.998	107.7592	2.0459
	Standard Deviation	2.6008	0.1406	5.52E-11	0.2741	3.7641	1.9394	11.0501	1.0777
	RANK	7	3	1	4	6	2	8	5
10.	Minimum	−1.0316	9.13E-42	−1.0316	−1.0316	−1.0316	−1.0316	−0.98	0
	Maximum	−1.0011	3.79E-34	−1.0316	−1.0316	−1.0316	−1.0316	4.2539	2.905E-09
	Mean	−1.0245	2.05E-35	−1.0316	−1.0316	−1.0316	−1.0316	0.0513	6.541E-11
	Standard Deviation	0.0074	6.66E-35	1.02E-06	2.94E-16	9.4E-09	2.37E-16	0.9635	4.116E-10
	RANK	5	8	4	2	3	1	6	7
11.	Minimum	0.3979	0.3979	0.3979	0.3979	0.3979	0.3979	0.4141	0.3979
	Maximum	0.4018	0.3979	0.398	0.3979	0.3995	0.3979	3.9832	0.3979
	Mean	0.3986	0.3979	0.3979	0.3979	0.3979	0.3979	1.2458	0.3979
	Standard Deviation	0.000767	3.36E-16	2.3E-05	3.36E-16	0.000225	3.36E-16	0.7853	3.365E-16
	RANK	4	1	2	1	3	1	5	1
12.	Minimum	3.0001	3.000	3.000	3.00000	3.00000	3.000	4.4177	3.00
	Maximum	53.1492	3.00000	3.000	3.00000	3.0001	3.000	162.5454	3.00
	Mean	5.745	3.00000	3.000	3.00000	3.0000	3.000	52.6833	3.00
	Standard Deviation	7.7821	3.39E-15	1.28E-15	1.74E-15	1.63E-05	3.54E-15	49.1436	1.692E-15
	RANK	7	4	1	3	6	5	8	2
		1	4	5.5	6	2	3	8	5

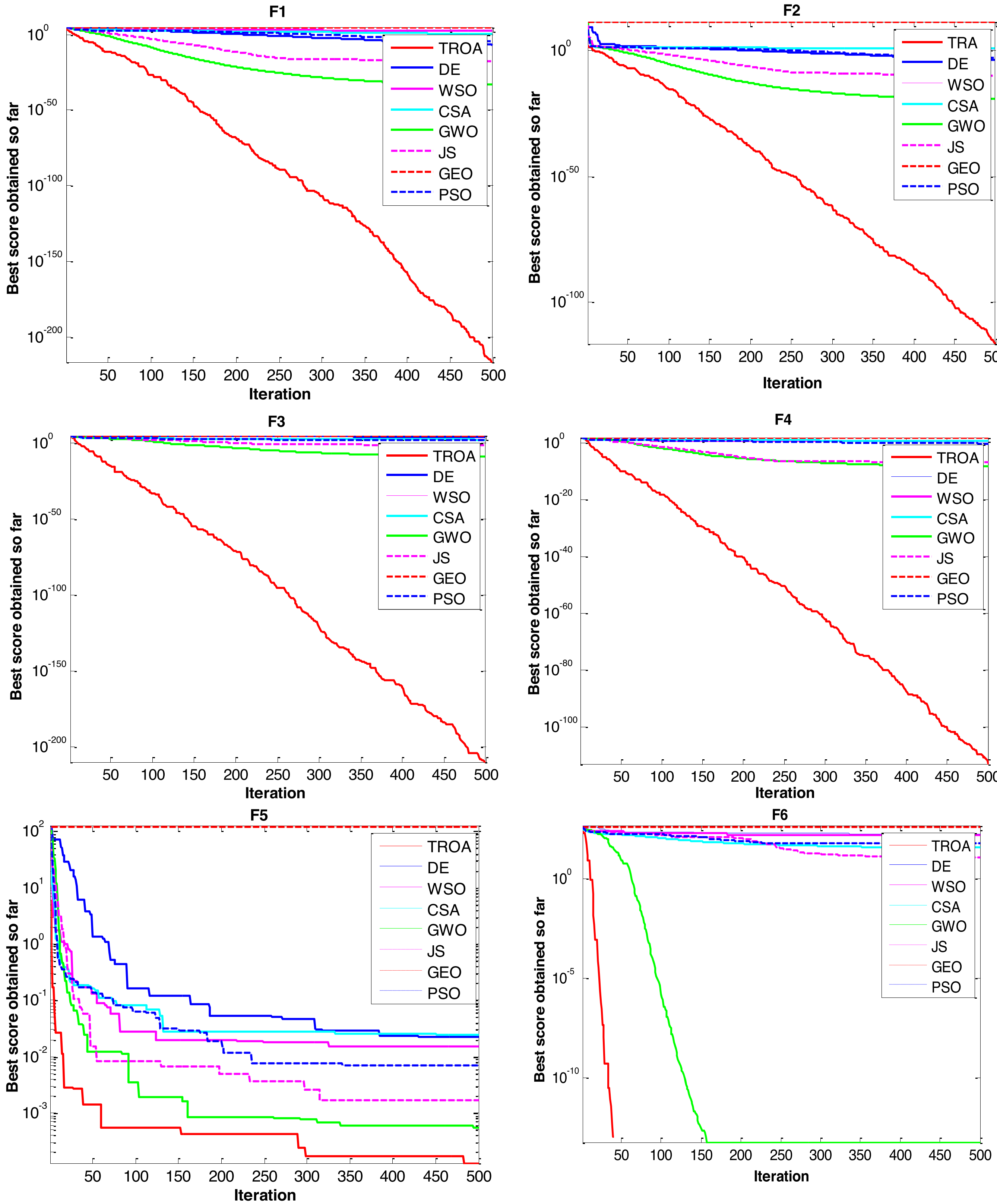

Fig. 5. Comparative analysis of convergence curve.

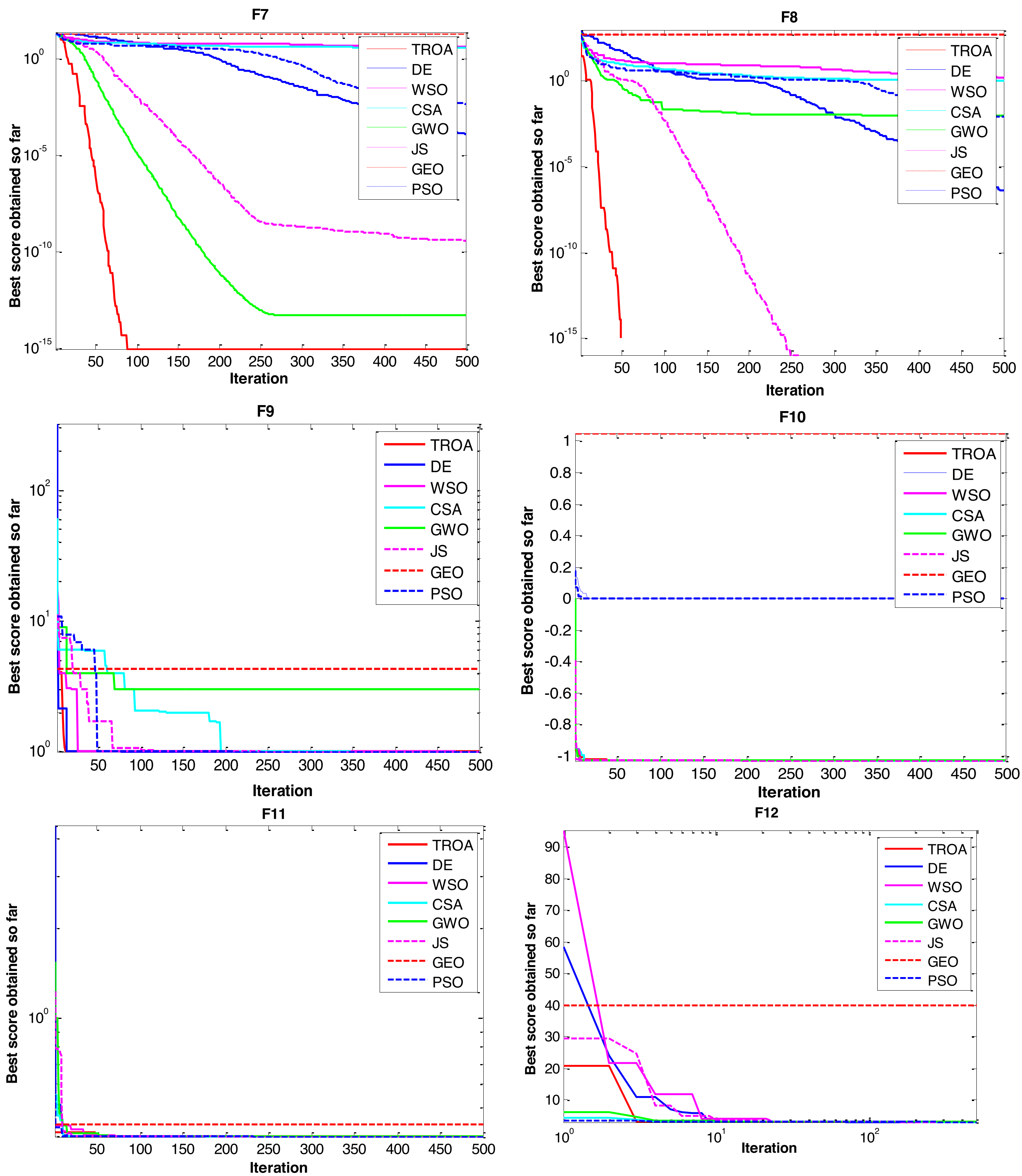

Fig. 5. (*continued*).

Table 5
Comparison analysis of techniques for CSTR.

Optimization algorithms	Minimum	Maximum	Mean	Standard deviation
TROA	2.0630E-201	1.3278	0.0055	0.0684
DE	0.0259	1.4811	0.1229	0.2057
WSO	0.0700	0.0700	0.0700	2.3616E-16
CSA	1.3898	1.5316	1.5023	0.0300
GWO	2.3986E-24	1.6214	0.0058	0.0774
JS	0.7932	1.2548	0.8435	0.0324
GEO	1.5276	1.5276	1.5276	1.0002E-14

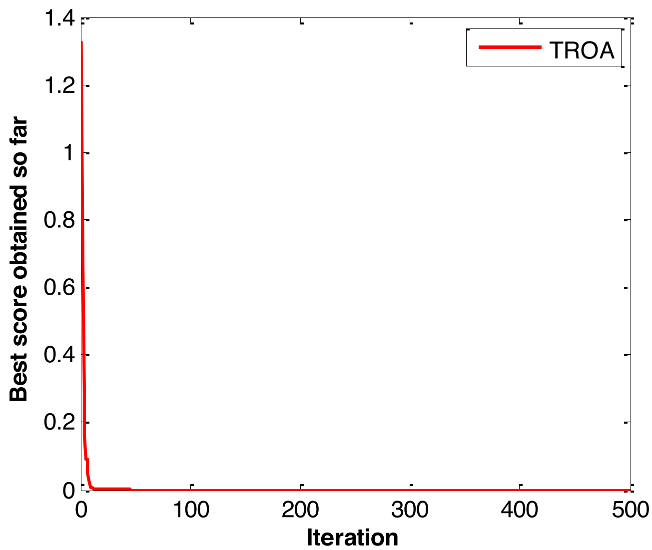

Fig. 6. Variation of performance w.r.t iteration for CSTR obtained by TROA.

$$\ddot{\theta}(t) = -\frac{g}{l}\sin(\theta(t)) - \frac{v}{ml^2}\dot{\theta}(t) + \frac{1}{ml^2}u(t) \tag{11}$$

The Eq. (11) can be rewritten by considering $m = 2kg, v = 6kgms$, and $l = 1$ m.Define $z_1(t) = \theta, z_2(t) = \theta$, So, the manipulator model can be written as:

$$\begin{aligned} \dot{z}_1(t) &= z_2(t) \\ \dot{z}_2(t) &= -9.8\sin(z_1(t)) - 3z_2(t) + 0.5u(t) \end{aligned} \tag{12}$$

So for our single-link manipulator, with the initial condition of $Z(0) = [0 \quad 0]$. The performance of Single link manipulator is measured by

$$J = \int_{t_0}^{t_f} z_1^2 + z_2^2 + u^2 dt \tag{13}$$

The results of the TROA are compared with other techniques including DE, WSO, CSA, GWO, JS, GEO, and PSO. Table 7 shows that the TROA obtains the best minimum and mean value in comparison to other algorithms. It was clear that the best optimal value obtained by TROA is 2.2106E-219 (Fig. 14).

From Table 7, when comparing the TROA in the terms of minimum performance is 99.99% more efficient than the other algorithms i.e. DE, WSO, CSA, GWO, JS, GEO, PSO. When comparing the TROA in terms of maximum performance 45.80% more efficient than DE, 98.71% less efficient than WSO, 40.10% more efficient than CSA, 63.15% more efficient than GWO, 15.55% more efficient than JS, 43.20% more efficient than GEO, 43.35% more efficient than PSO. When comparing the TROA in terms of mean performance 90.92% more efficient than DE,37.14% more efficient than WSO, 99.62% more efficient than CSA,37.14% more efficient than GWO,99.47% more efficient than JS, 99.67% more efficient than GEO, 99.67% more efficient than PSO.

From Fig. 16 it can be observed that the TROA convergence is faster than other algorithms. It can be noticed from Fig. 17 that the TROA achieves convergence around the 10th iteration.

Figs. 18 and 19 show the variations of the angular position and velocity for all the algorithms with respect to time. Among all the algorithms TROA has less velocity as depicted in Fig. 19.

Figs. 20 and 21 show the applied torque utilized by the SLM. It can be noticed that the TROA uses less amount of torque in the order of 10ˆ−109 as shown in Fig. 21.

4.4. Single link flexible manipulator

In this section, we have considered a single-link flexible manipulator model [23] as shown in Fig. 22. The basic equation is written as follows:

$$\begin{aligned} &(J_h + J_l)\ddot{\theta} + J_l\ddot{\alpha} - mgh\sin(\theta + \alpha) = \tau \\ &J_1\ddot{\theta} + J_l\ddot{\alpha} + K_s\alpha - mgh\sin(\theta + \alpha) = 0 \end{aligned} \tag{14}$$

The mathematical model of the dynamic of the relation with a flexible joint in its vertical position can be easily extracted using Lagrange's motion equations. The generalized coordinates are the motor's angular position θ and the flexible joint's angular displacement α.

The motor torque is defined here. The applied voltage v achieves the torque to the armature, and the control input of the system is shown. The relation between the voltage used and the torque is:

$$\begin{aligned} z_1 &= \theta \\ z_2 &= \alpha \\ z_3 &= \dot{\theta} \\ z_4 &= \dot{\alpha} \end{aligned} \tag{15}$$

and the above equation can be rewritten as follows

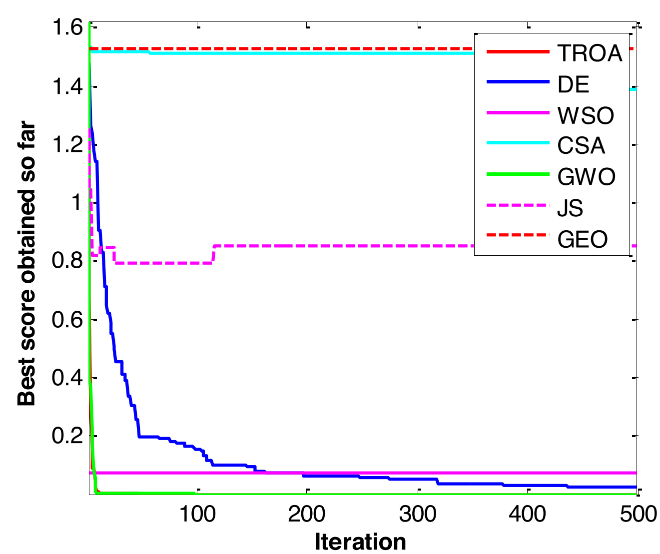

Fig. 7. Variation of performance w.r.t iteration for CSTR.

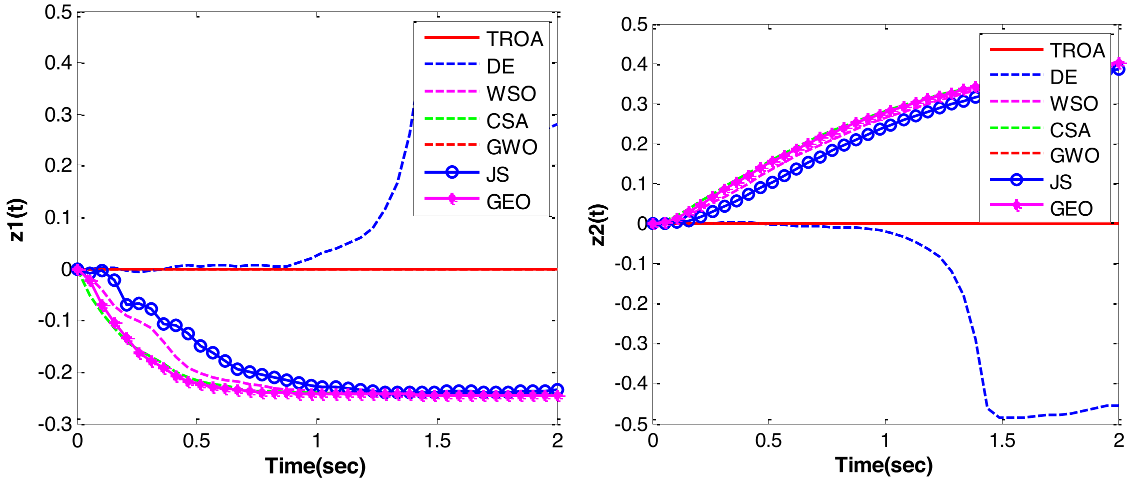

Fig. 8. Comparison state graph of the CSTR.

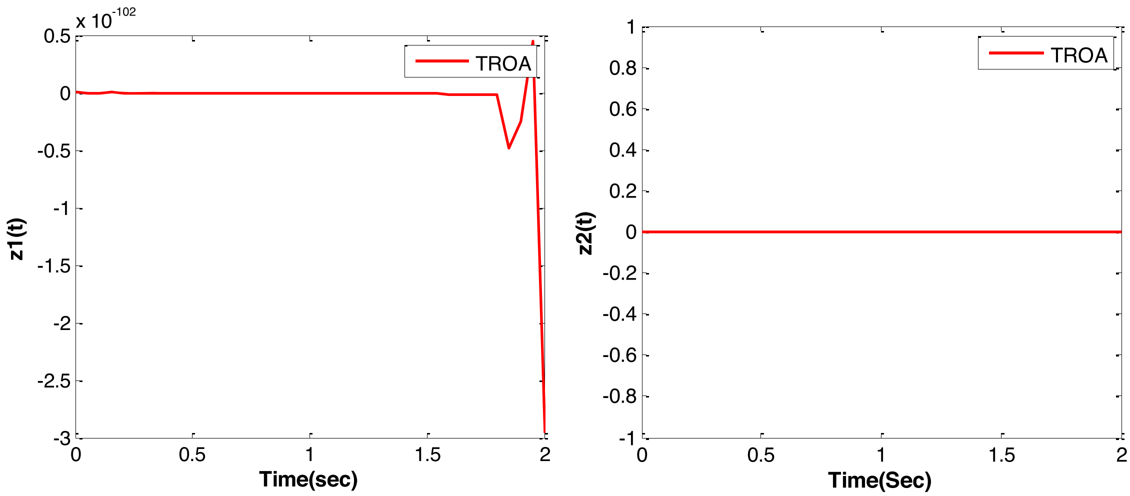

Fig. 9. State variables graph of CSTR for TROA.

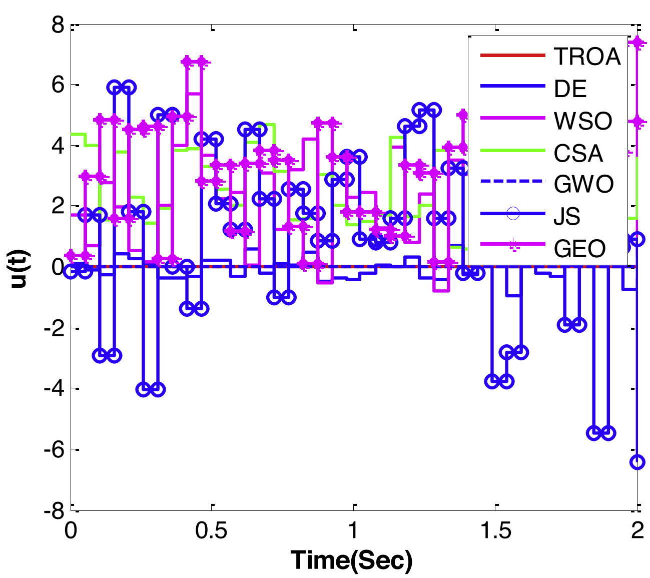

Fig. 10. Convergence curve of control parameter.

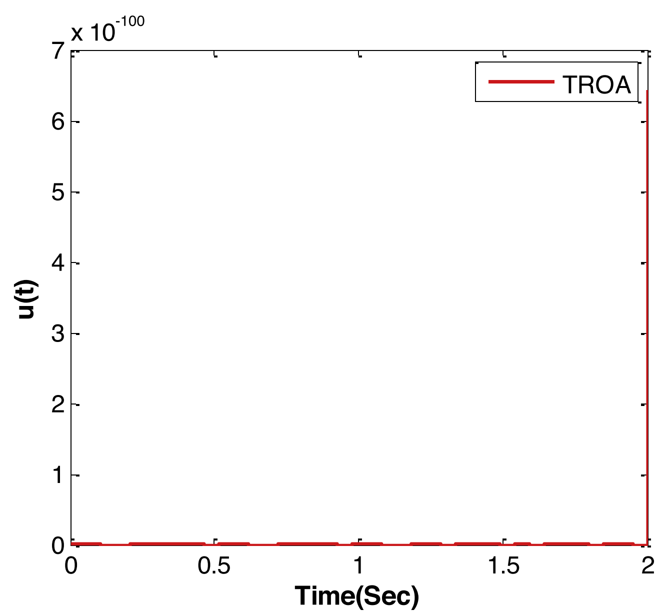

Fig. 11. Control parameter curve of the TROA.

Table 6
Comparison analysis of techniques for Van der Pol oscillator.

Optimization algorithms	Minimum	Maximum	Mean	Standard Deviation
TROA	5.9299E-08	9.2060	0.3165	0.6952
DE	3.6153E-05	7.9231	0.2326	0.8919
WSO	9.0430	9.0430	9.0430	3.5563E-15
CSA	5.1257	9.0466	6.1357	1.3076
GWO	0	11.1662	0.0338	0.5377
JS	0.9267	1.2351	0.9276	0.0149
GEO	8.8766	8.8766	8.8766	4.6232E-14
PSO	7.8373	8.8713	8.3684	0.4907

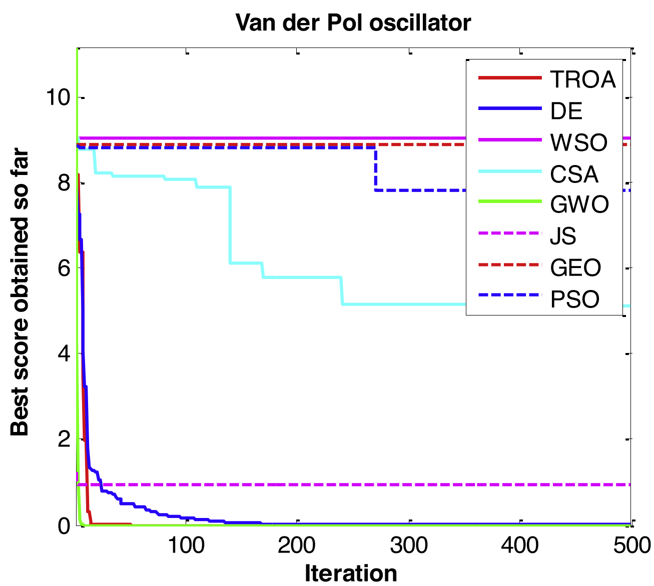

Fig. 12. Comparison of techniques for Van der Pol oscillator.

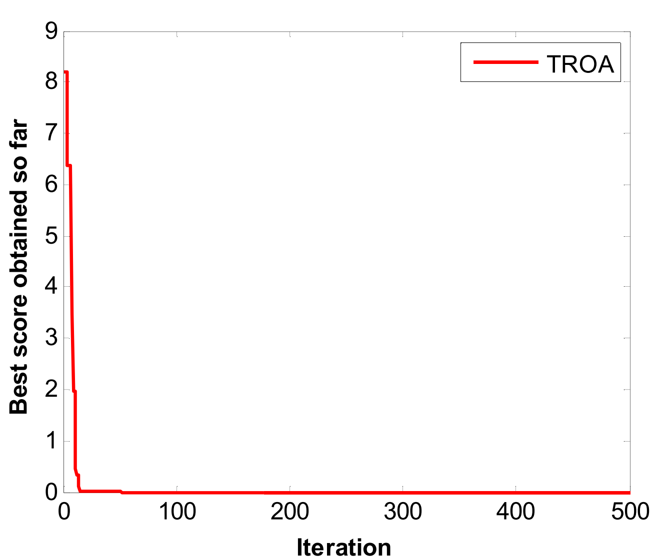

Fig. 13. Convergence curve of TROA for Van der Pol oscillator.

$$\begin{aligned}
\dot{z}_1 &= z_3 \\
\dot{z}_2 &= z_4 \\
\dot{z}_3 &= \frac{k_s}{J_h} z_2 - \frac{k_m^2 k_g^2}{R_m J_h} z_3 + \frac{k_m k_g}{R_m J_h} u \\
\dot{z}_4 &= -\frac{k_s}{J_h} z_2 + \frac{k_m^2 k_g^2}{R_m J_h} z_3 - \frac{k_m k_g}{R_m J_h} u + \frac{mgh}{J_1} \sin(z_1 + z_2) - \frac{k_s}{J_1} z_2
\end{aligned} \tag{16}$$

$m$is the effective mass of the arm end,g is gravitational,B_m is the loaded rotational friction,r is the radius,J_h is the moment of inertia of the motor,J_1 is the inertia of the arm.z_1 is the arm end position,z_2 is the arm

Table 7
Comparison analysis of techniques for single-link manipulator.

Optimization algorithms	Minimum	Maximum	Mean	Standard Deviation
TROA	2.2106E-219	0.7610	0.0044	0.0429
DE	2.1842E-09	1.4042	0.0485	0.1630
WSO	0.0700	0.0700	0.0700	2.3616E-16
CSA	1.1539	1.2719	1.1649	0.0247
GWO	3.4238E-21	2.0654	0.0070	0.1014
JS	0.5692	0.9010	0.8404	0.0469
GEO	1.3399	1.3399	1.3399	4.4453E-16
PSO	1.3434	1.3434	1.3434	1.2002E-14

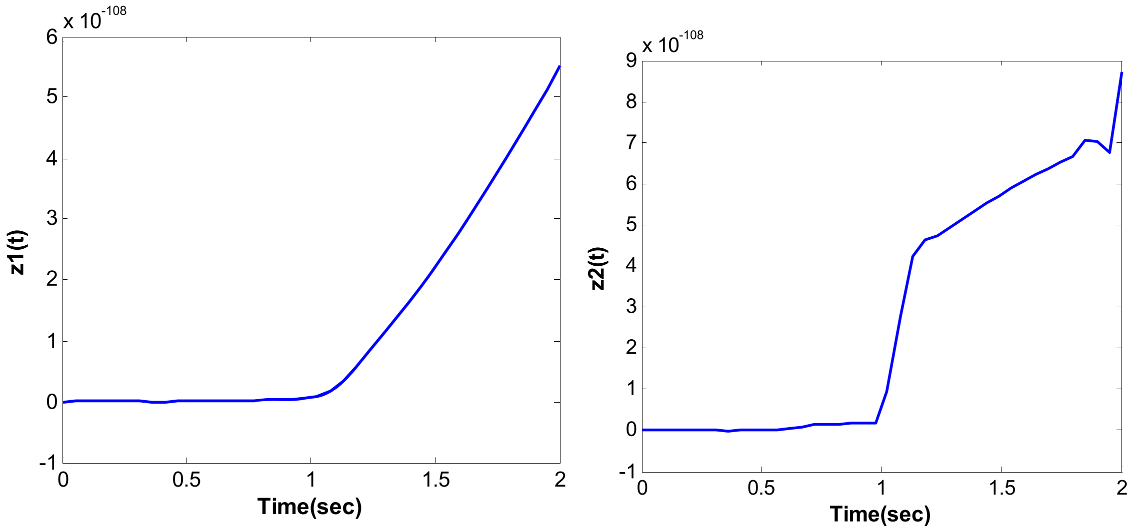

Fig. 14. State equation of Van der Pol oscillator.

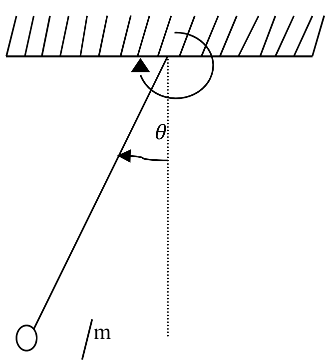

Fig. 15. Single link manipulator.

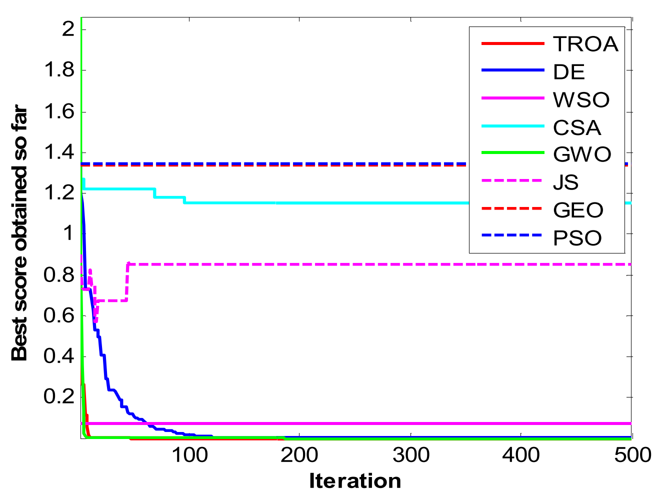

Fig. 16. Variation of performance w.r.t iteration for SLM.

axis velocity, z_3 is the arm position, z_4 is the arm end velocity. The values of the parameter are considered from the paper [20].

So for our single-link flexible manipulator, the performance measure is considered based on the above equation

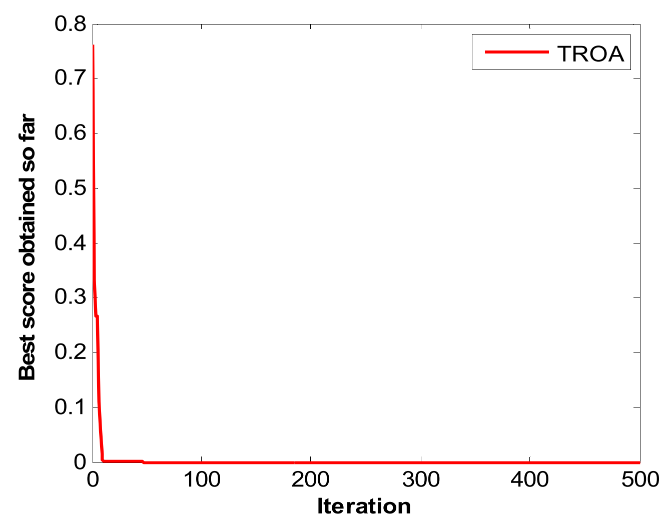

Fig. 17. Variation of performance w.r.t iteration for SLM obtained by TROA.

$$J = \int_{t_0}^{t_f} z_1^2 + z_2^2 + u^2 dt \tag{17}$$

Table 8 shows the maximum, mean, and standard deviation values for all the algorithms. It was clear from this table that the best optimal value is obtained by TROA having a value of 6.4543E-15.

From Fig. 23, it can be observed that the TRA convergence faster in comparison to other methods. Also, from the Fig. 24 it can be noticed that the TROA converges around 5th iteration. Hence, it can be concluded that the proposed approach shows superior convergence characteristics among other techniques.

Table 7 shows that the TROA performs 99.99% more efficiently than the other algorithms i.e. DE, WSO, CSA, GWO, JS, GEO,and PSO in terms of minimum performance. When comparing the TROA in the terms of mean performance 60.87% more efficient than DE,0.007% less efficient than WSO, 99.18% more efficient than CSA,60.87% more efficient than GWO,0.008% less efficient than JS, 93.55% more efficient than GEO, 99.50% more efficient than PSO.

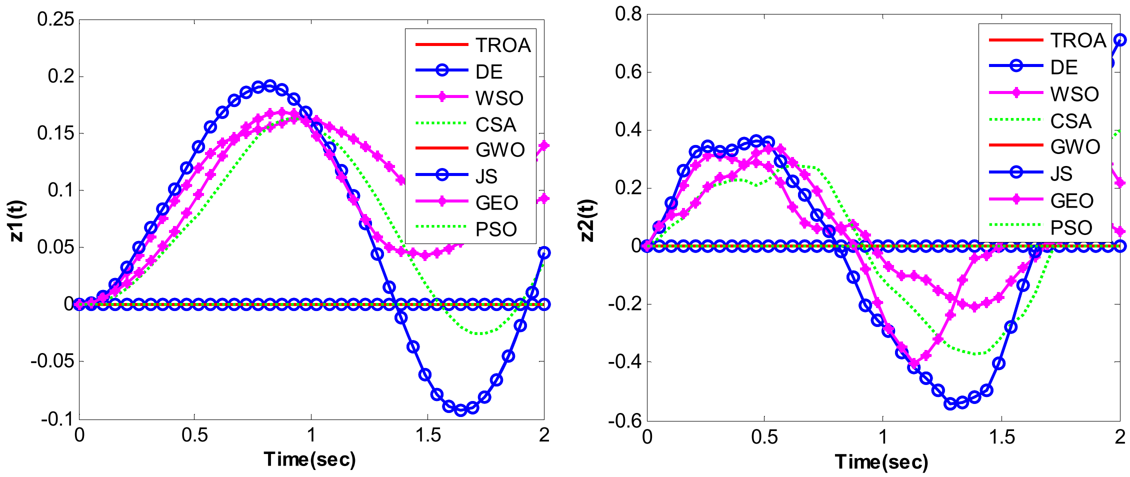

Fig. 18. Variation of state graph of the SLM.

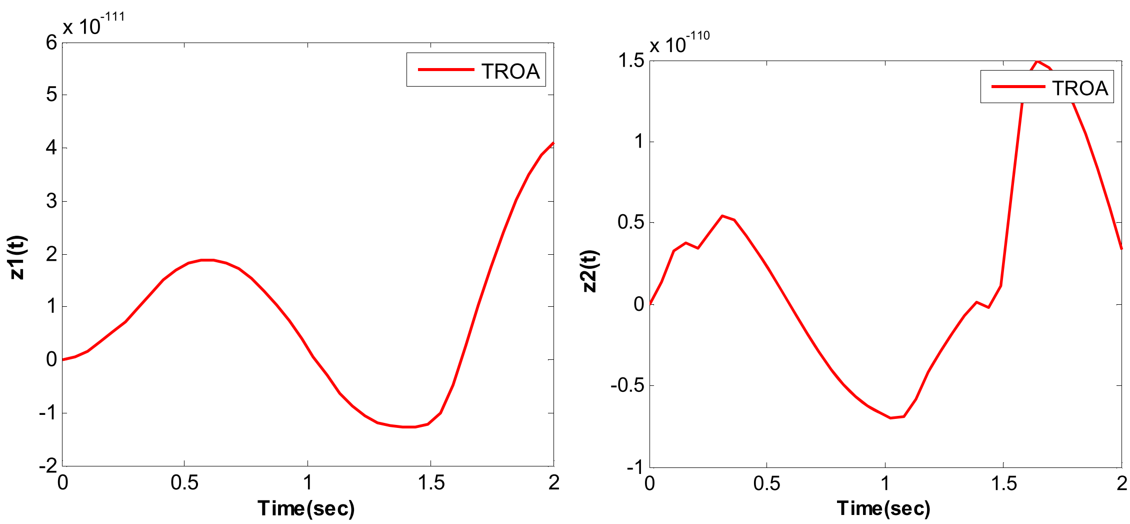

Fig. 19. Variation of state graph of the SLM obtained by TROA.

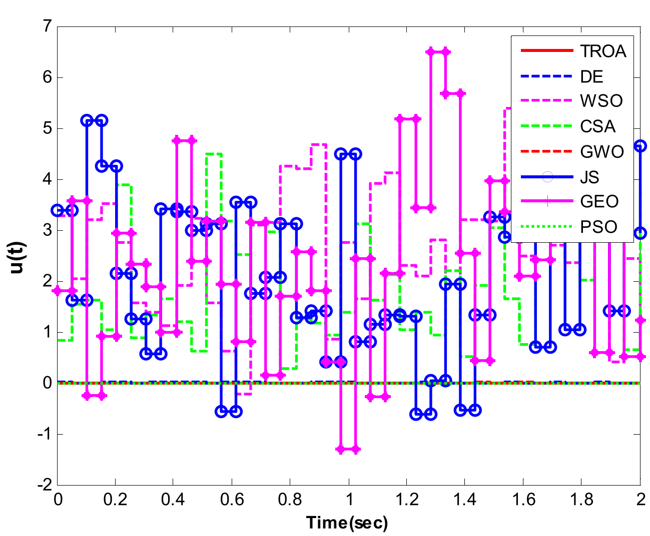

Fig. 20. Variation of control parameter for SLM.

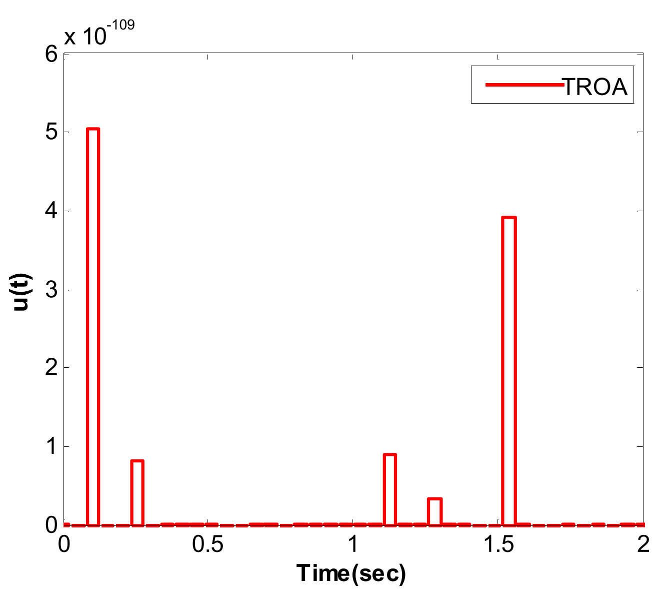

Fig. 21. Variation of control parameter for SLM obtained by TROA.

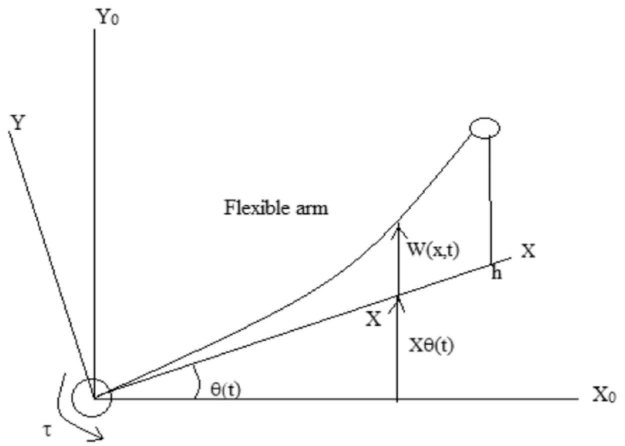

Fig. 22. Single-link flexible manipulator.

Table 8
Comparison analysis of techniques for single-link flexible manipulator.

Optimization algorithms	Minimum	Maximum	Mean	Standard Deviation
TROA	6.4543E-15	24.9328	0.0705	1.1388
DE	3.0251E-05	12.9226	0.1802	0.8103
WSO	0.0700	0.0700	0.0700	2.3616E-16
CSA	6.9377	12.5375	8.5369	1.0251
GWO	3.0251E-05	12.9226	0.1802	0.8103
JS	2.3801E-13	31.5527	0.0694	1.4164
GEO	1.0831	1.3982	1.0859	0.0258
PSO	13.8573	18.0601	14.0659	0.7855

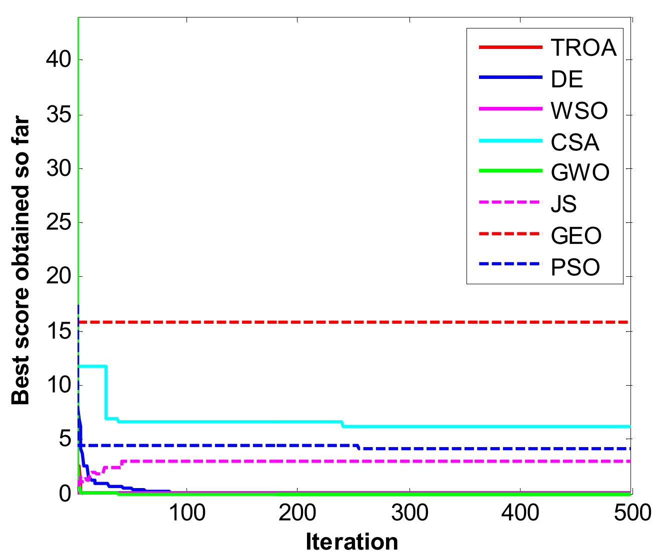

Fig. 23. Variation of performance w.r.t iteration for SLFM.

Fig. 25 shows the variation of the angular position and angular velocity of the SLFM with time for different algorithms. It can be observed that the TROA achieves nearly steady state variations of the angular position and angular velocity. Fig. 26 shows the variations of angular position and angular velocity for the SLFM are very low.

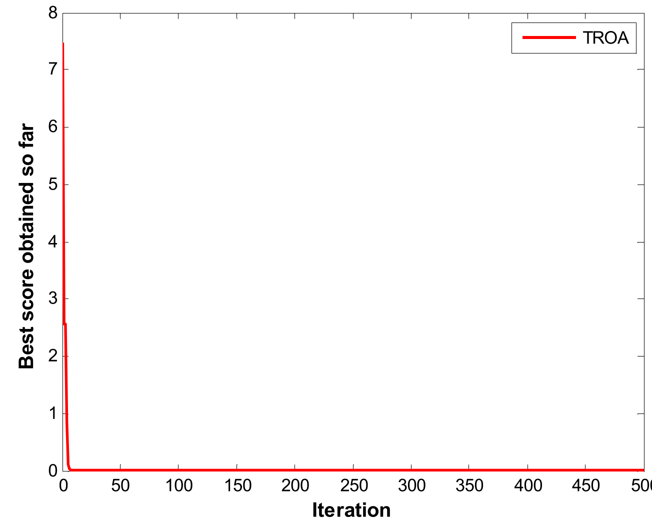

Fig. 24. Variation of performance w.r.t iteration for SLFM obtained by TROA.

Figs. 27 and 28 show that the TROA applied torque is very less when compared to the other algorithms. This is because the TORA requires less velocity to reach the desired position.

5. Conclusion

This paper introduces a new meta-heuristic algorithm inspired by the hunting behaviour of the dinosaur called tyrannosaurus (T-Rex) to solve optimization problems. The algorithm works on the prey and the T-Rex position. The T-Rex will chase after the prey with a definite success rate, while the prey will try to escape with a certain speed. The best parameters of this algorithm are set according to the Friedman mean ranking test. The tyrannosaurus algorithm has obtained better results when compared to the other techniques using 12 benchmark problems. In this paper, the TROA is found to converge faster in comparison to other meta-heuristic algorithms. In addition, the performance of the proposed algorithm is consistent statistically while solving different unimodal and multimodal benchmark test functions. Further, the application of a popular non-parametric test like; the Wilcoxon rank sum test indicates that the TORA performs better in comparison to other techniques for 11 benchmark test functions. Moreover, the TORA obtains better solutions when applied in the case of 4 optimal control problems. Future work is to extend the algorithm for multi-objective optimization problems.

Financial disclosure

The authors certify they have no affiliations with or involvement in any organization or entity with any financial or non-financial interest in the subject matter or materials discussed in this manuscript.

Ethical approval

This article does not contain any studies with human participants or animals performed by any of the authors.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence

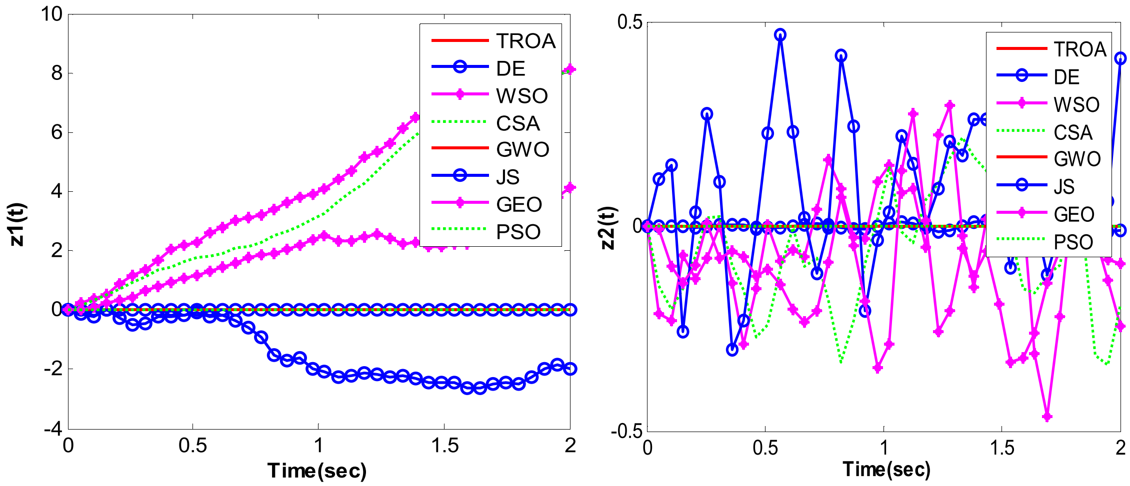

Fig. 25. Optimal states obtained by SLFM.

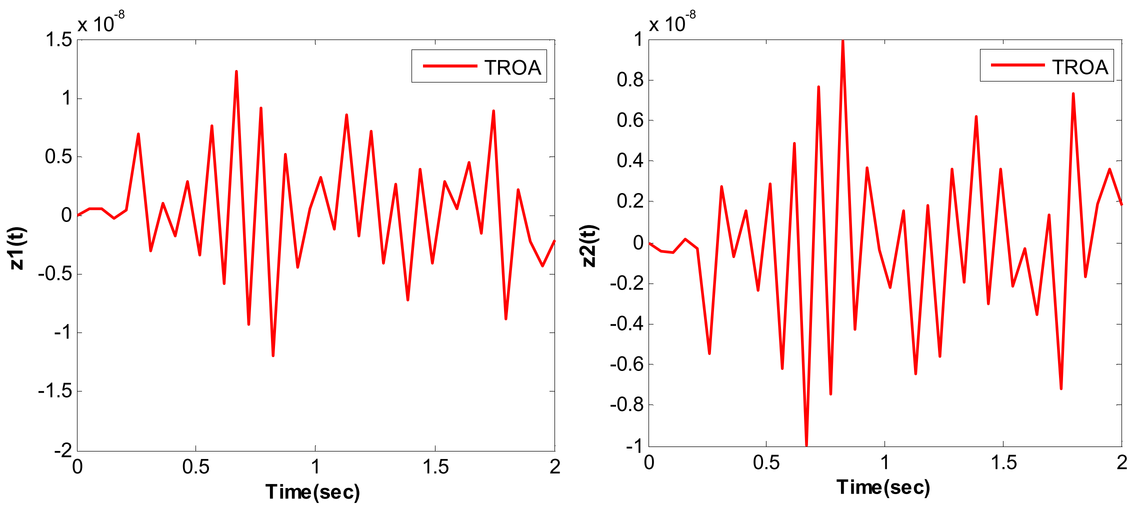

Fig. 26. Optimal states obtained by SLFM for TROA.

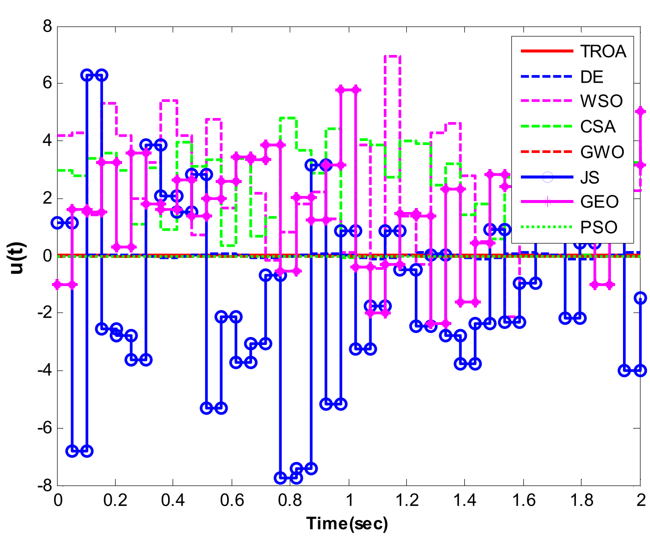

Fig. 27. Variation of applied torque for SLFM.

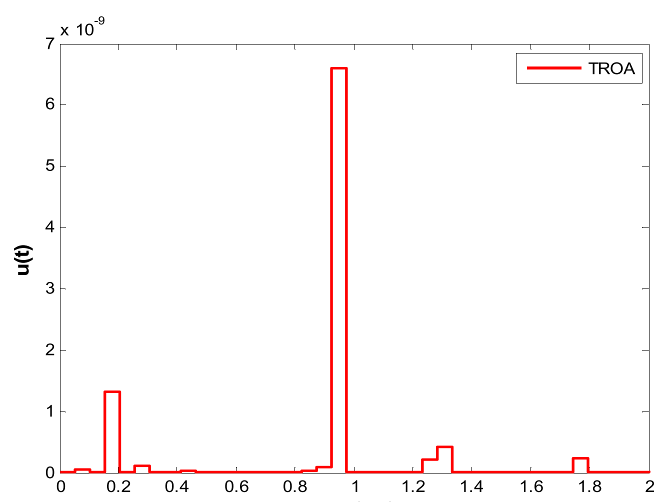

Fig. 28. Variation of applied torque for SLFM obtained by TROA.

the work reported in this paper.

Data availability

No data was used for the research described in the article.

References

[1] S.L. Brusatte, P. Larson, K. Carpenter, A Review of " Tyrannosaurus rex , the Tyrant King, 30, Indiana University Press, 2008, p. 456. Bloomington, Indiana US$49.95. ISBN 978-0-253-35087-9', *J. Vertebr. Paleontol.* pp. 304–305, Jan. 2010. 10.1080/02724630903416084.
[2] J.O. Farlow, Tyranivosaurus rex, The Tyrant King, Indiana University Press, 2008.
[3] K.P. Rainer Storn, Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces, J. Glob. Optim. (1997) 341–359.
[4] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of ICNN'95 - International Conference on Neural Networks 4, 1995, pp. 1942–1948, https://doi.org/10.1109/ICNN.1995.488968.
[5] A. Mohammadi-Balani, M. Dehghan Nayeri, A. Azar, M. Taghizadeh-Yazdi, Golden eagle optimizer: a nature-inspired metaheuristic algorithm, Comput. Ind. Eng. 152 (2021) 1–45, https://doi.org/10.1016/j.cie.2020.107050.
[6] S. Mirjalili, S.M. Mirjalili, A. Lewis, Grey wolf optimizer, Adv. Eng. Softw. 69 (2014) 46–61, https://doi.org/10.1016/j.advengsoft.2013.12.007.
[7] J.S. Chou, D.N. Truong, A novel metaheuristic optimizer inspired by behavior of jellyfish in ocean, Appl. Math. Comput. 389 (2021), 125535, https://doi.org/10.1016/j.amc.2020.125535.
[8] A.G. Hussien, et al., 'Crow search algorithm : theory , recent advances, and applications', IEEE Access 8 (2020) https://doi.org/10.1109/ACCESS.2020.3024108.
[9] M. Braik, A. Hammouri, J. Atwan, M.A. Al-Betar, M.A. Awadallah, White shark optimizer: a novel bio-inspired meta-heuristic algorithm for global optimization problems, Knowl. Based Syst. 243 (2022), 108457, https://doi.org/10.1016/j.knosys.2022.108457. May.
[10] J.R. Horner, Steak knives, beady eyes, and tiny little arms (a portrait of T. rex as a scavenger), Paleontol. Soc. Spec. Publ. 7 (1994) 157–164, https://doi.org/10.1017/S2475262200009497.
[11] B. Strauss, 'Was the Tyrannosaurus rex a hunter or scavenger?', 2019. https://www.thoughtco.com/tyrannosaurus-rex-hunter-or-scavenger-1092011 (accessed Aug. 28, 2019).
[12] B.T.V Wilson, Was Tyrannosaurus rex a predator or a scavenger?, 1970. https://science.howstuffworks.com/environmental/earth/geology/fossil-replica.htm.
[13] 'Tyrannosaurus' 2023. https://www.gettyimages.in/detail/illustration/rex-artwork-royalty-free-illustration/460716257.
[14] E. Gamillo, 'New study finds T. rex walked at a slow pace of three miles per hour', pp. 3–7, 2021 https://www.smithsonianmag.com/smart-news/new-study-finds-that-t-rex-walked-at-slow-pace-of-3-miles-per-hour-180977572/.
[15] L. Abualigah, M.A. Elaziz, P. Sumari, Z.W. Geem, A.H. Gandomi, Reptile Search Algorithm (RSA): a nature-inspired meta-heuristic optimizer, Expert Syst. Appl. 191 (2021), 116158, https://doi.org/10.1016/j.eswa.2021.116158. September2022.
[16] V.S.D.M. Sahu, P. Samal, C.K. Panigrahi, Modelling, and control techniques of robotic manipulators: a review, Mater. Today Proc. (2021), https://doi.org/10.1016/j.matpr.2021.10.009.
[17] V.S.D.M. Sahu, P. Samal, C.K. Panigrahi, Application of differential evolution algorithm to optimal control problem with state variable constraints, in: Proceedings of the 2020 IEEE 17th India Council of International Conference, INDICON 2020 i, 2020, https://doi.org/10.1109/INDICON49873.2020.9342273.
[18] H.H. Mehne, S. Mirjalili, A parallel numerical method for solving optimal control problems based on whale optimization algorithm, Knowl. Based Syst. 151 (2018) 114–123, https://doi.org/10.1016/j.knosys.2018.03.024.
[19] M. Alipour, M.A. Vali, A.H. Borzabadi, A hybrid parametrization approach for a class of nonlinear optimal control problems, Numer. Algebr. Control Optim. 9 (4) (2019) 493–506, https://doi.org/10.3934/naco.2019037.
[20] N. Razmjooy, M. Ramezani, A. Namadchian, A new LQR optimal control for a single-link flexible joint robot manipulator based on grey wolf optimizer, Majlesi J. Electr. Eng. 10 (3) (2016) 53–60.
[21] D.E. Kirk, Optimal control theory: an introduction, IEEE Trans. Autom. Control 17 (3) (1972) 423, https://doi.org/10.1109/tac.1972.1100008, 423.
[22] V.S.D.M. Sahu, P. Samal, C.K. Panigrahi, A novel technique based on crow search algorithm to solve optimal control problem of a single-link rigid robotic manipulator, in: Proceedings of the 2023 International Conference on Power Electronics and Energy (ICPEE), 2023, pp. 1–5, https://doi.org/10.1109/ICPEE54198.2023.10060674.
[23] K. Groves, A. Serrani, 'Modeling and nonlinear control of a single-link flexible joint manipulator', no. 3, p. 13, 2010 https://www2.ece.ohio-state.edu/~passino/lab5prelabnlc.pdf.
[24] D. Panigrahy, P. Samal, Modified lightning search algorithm for optimization, Eng. Appl. Artif. Intell. 105 (2021), 104419, https://doi.org/10.1016/j.engappai.2021.104419.

Venkata Satya Durga Manohar Sahu was born in 1990 and received the B.Tech. Degree from the Vikas college of engineering and technology, AP, India, and Master of Engineering degree in Control system in 2015 from AU college of Engineering, Andhra University (India). He has been pursuing a part-time Ph.D. in the School of Electrical Engineering, KIIT Deemed to be University in Bhubaneswar, Odisha, since 2019. His research interest is in the field of AI techniques, Control systems, and Optimization Techniques.

Padarbinda Samal was born in 1987 and received a Bachelor of Technology degree in electrical engineering from S.I.T, Bhubaneswar (Orissa) in 2009, as well as a Master of Technology degree from V.S.S University of Technology, Burla (Orissa). He also holds a Ph.D. in electrical engineering from the National Institute of Technology (NIT) in Rourkela (Orissa). From 2010 to 2011, he was an assistant professor in the electrical engineering department at SOA University in Bhubaneswar, Odisha, and he is currently continuing at KIIT Deemed to be University in Bhubaneswar, Odisha. His research interests include distribution system planning and soft computing applications to power systems.

Chinmoy Kumar Panigrahi was born in Orissa in 1967. In the years 1990 and 1997, he earned his Bachelor of Technology and Master of Technology Degrees from Sambalpur University, Orissa. In the year of 2007, he earned his Ph.D. from Jadavpur University, Kolkata. He has 23 years of teaching experience as well as 5 years of research experience. His resume has a long list of honors and recognitions. Power system control, renewable energy systems, and soft computing techniques are his areas of interest.