DOI 10.52209/2706-977X_2024_2_79 **IRSTI 55.03.14** **UDC 621.396**

Weight Reduction of a Speed Reducer using Modified Particle Swarm Optimization and Shuffled Frog Leaping Algorithm

Dalavi A.*, Gomes A., Husain A.
Symbiosis Institute of Technology, Pune, India
*corresponding author

Abstract. Gear drives are extremely critical elements of transmission systems. Designing gears that satisfy the application's requirements like load carrying capacity and strength whilst also being compact, lightweight, and economical is quite a challenging task. In recent times, a lot of research has been dedicated to optimizing speed reducers by making use of metaheuristic nature-inspired algorithms. The present work optimizes the weight of a spur speed reducer by employing two modified nature-inspired algorithms: Particle Swarm Optimization (PSO) and Shuffled Frog Leaping Algorithm (SFLA). Parameter optimization was carried out to select the best combination of c_1, c_2 and ω for the selected case study. It was found that there was a 1.1619% and 13.41% decrease in the cost function evaluation, as compared to the Crude Monte-Carlo and Stray Process methods used in the original case study, and around an 11% decrease as compared to results in published literature.

Keywords: gearbox optimization, particle swarm optimization, shuffled frog leaping algorithm, parameter optimization, spur gear.

Introduction

Gears are extremely critical elements in mechanical systems and are typically used for short-distance power transmission between elements having different torque and speeds. Gears have been employed in a wide range of applications, including automobiles, aircraft, wind turbines, and industrial machines, among others. Designing gears with the least possible amount of volume is critical because it reduces energy consumption and the amount of material required to manufacture them [1]. Over the years, there have been great advances in research for achieving dimensional optimality while also preventing failure. Further, the ever-growing environmental concerns have led to the need for high efficiency of the designed product. As a result, design optimization is often regarded as the most crucial step in the design and development of a new gearbox [2].

Nature-inspired optimization techniques were introduced over half a century ago and have hence been used to optimize engineering problems using the myriad of algorithms developed. The most implemented algorithms for design problems include Genetic Algorithm (GA), Simulated Annealing (SA), Particle Swarm Optimization (PSO), and their variants [3]. In this paper, gear data from a previously investigated case of a spur reducer proposed by Dr Jan Golinski was considered [4]. The Golinski speed reducer problem is also a part of the NASA Langley Multidisciplinary Design Optimization (MDO) Test Suite [5]. Over the years, this case study has been extensively attempted using various optimization algorithms, both deterministic and metaheuristic. Modified versions of the PSO and SFLA algorithms were implemented on the reduced set of equations provided by [6]. The results obtained were then compared with those presented in previously published works.

2 Materials and Methods

2.1 Modified Particle Swarm Optimization Algorithm

The particle swarm optimization algorithm, a meta heuristic algorithm for optimising non-linear continuous functions [7]. It is based on the social behavior of swarm animals like birds and fish and depends on factors such as avoiding predators, seeking food, and environmental parameters like temperature, etc. The collective knowledge gathered and shared by the swarm members is known as social knowledge which is an integral part of the practice followed by these swarms to ensure their survival. The end goal is to find a spot that will maximise their survival advantage along with the food they are seeking. To achieve this, each member explores and judges using several criteria.

The PSO algorithm simulates this swarm behaviour, wherein the position and velocity vectors are given by X_i and V_i respectively, and n is the total number of variables in the problem. Soon after PSO was introduced, [8] proposed a modified version with a new parameter called inertia weight (ω), which greatly improved its performance by balancing out the local and global search of the algorithm. A larger ω tends to improve the global search, while a smaller ω tends to improve the local search. This modified PSO (with the inertia weight included) is considered as the canonical PSO. This was further modified to include limits for the minimum (Min V) and maximum (Max V) velocity [9].

Initially, the population is randomly generated within the specified ranges for each variable and V_i is set to 0. Then the cost function $f(X)$ is calculated, and the initial positions of the population is set as the personal best for each particle. The position with the least $f(X)$ value is set as the global best. Then the maximum and minimum allowed velocity limits are calculated using equations (1) and (2) and the velocity limits are applied as per equations (3) and (4):

$$Max\,V\ = 0.2(VarMax - VarMin) \tag{1}$$

$$Min\,V = -Max\,V \tag{2}$$

$$V_i = \max(V_i,\ Min\,V) \tag{3}$$

$$V_i = \min(V_i,\ Max\,V) \tag{4}$$

Then the algorithm proceeds by generating the new set of positions for the particles. For this, the velocity of the particle is calculated using equation (5) and is constrained as per equations (6) and (7):

$$V_{i+1} = \omega V_i + c_1 r_1 \left(p_{best,i} - X_i\right) + c_2 r_2 \left(g_{best,i} - X_i\right) \tag{5}$$

$$X_i = \max(X_i,\ VarMin) \tag{6}$$
$$X_i = \max(X_i,\ VarMax) \tag{7}$$

Where, $p_{best,i} =$ best personal particle position, $g_{best,i} =$ global best particle position, c_1 and c_2 = acceleration constants used to regulate the cognitive and social elements respectively, r_1 and r_2 = randomly generated numbers between 0 and 1 (to avoid early convergence).

The first term in equation (5) represents the particle's preceding velocity, the second term is the difference between the particle's own best position (cognitive term), and the third term oversees the process of sharing knowledge between particles (social term). This ensures that the best position is found regardless of which particle finds it. Then the new positions for the population are determined using equation (8) and the cost function is evaluated:

$$X_{i=1} = X_i + V_{i+1} \tag{8}$$

For any given particle, if the cost function evaluation for the new position is better than the previous one, the personal best is updated and set to the new position. Similarly, if the new cost function evaluation is better than the global best, this particle position is set as the new global best position. This procedure is repeated till the convergence criteria is met. The flowchart of the modified PSO algorithm used in this paper is shown in Figure 1.

2.2 Modified Shuffled Frog Leaping Algorithm

The Shuffled Frog Leaping Algorithm (SFLA) is a memetic metaheuristic method that seeks the global optimum of an optimization problem [10]. It is based on the concept of frogs leaping across stones in a swamp, wherein the frogs represent the population, which in turn represents feasible solutions to the optimization problem. The stones are at discrete locations in the swamp and have varying amounts of food on them. The frogs try to locate the stone that has the largest amount of food available as quickly as possible. They do this by improving their memes, which correspond to their coordinate position, and can only be altered by discrete values. The population is further subdivided into subsets called memeplexes, within which the frogs share information with each another and improve their memes. Based on this, the frog's leaping step size is adjusted, thus changing their positions. SFLA combines the local search methodology of PSO and the competitive nature and mixed information methodology of the Shuffled Complex Evolution (SCE) algorithm.

Initially the frog population X_i is randomly generated containing *n* number of frogs. Then the cost function is evaluated and the frogs in the population are arranged in descending order of the results obtained. Next, the population is divided into m number of memeplexes. The first frog will be sorted into the 1st memeplex, the second frog in the 2nd memeplex, …., the mth frog to the mth memeplex, the (m+1)th frog to the 1st memeplex, and so on. Then the worst (X_w) and best (X_b) frogs within each memeplex, and the global best frog (X_g) from the entire population are identified.

To avoid early convergence at a local optimum, [11] suggested the use of a parameter called the ‘search acceleration factor’, denoted by c, which is used in the modified SFLA algorithm. At the beginning of the search, a larger c value would accelerate the global search, thus widening the search space. Once a probable optimal location is identified, c would focus on an in-depth local search. Hence, c balances the local and global search of the algorithm. [12] suggested the use of an inertia factor *ω*, similar to the inertia factor in PSO. The inertia factor in SFLA would help avoid premature convergence and widen the search space.

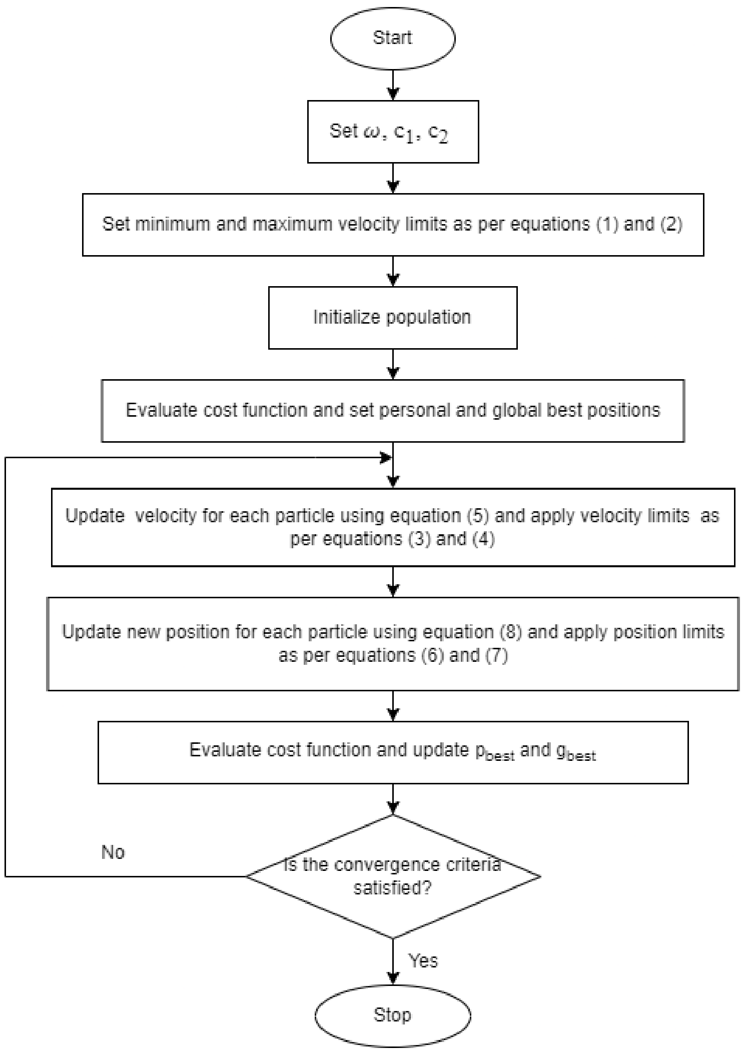
Start
Set ω, c_1, c_2
Set minimum and maximum velocity limits as per equations (1) and (2)
Initialize population
Evaluate cost function and set personal and global best positions
Update velocity for each particle using equation (5) and apply velocity limits as per equations (3) and (4)
Update new position for each particle using equation (8) and apply position limits as per equations (6) and (7)
Evaluate cost function and update p_{best} and g_{best}
Is the convergence criteria satisfied?
No
Yes
Stop

Fig. 1. - Flowchart for modified PSO algorithm

Now, the local search step is initiated using equation (9), by which the new position (X_{i+1}) of the frog is calculated. Every new frog generated is subject to the position range conformance criteria mentioned in equation (11) [13]. If this condition is satisfied, the cost function is evaluated, and if the evaluation is better, the new frog replaces the worst one. If not, the local search evaluation is skipped and the global search step is initiated using equation (10), by which the new position (X_{i+1}) is generated. Once again, the condition in equation (11) is checked. If satisfied, the cost function is evaluated, and if the evaluation is better, the new frog replaces the worst one. If equation (11) is not satisfied again, the global search step is skipped, and the frog is generated randomly.

$$X_{i+1} = \omega X_i + c_1 r(X_b - X_w) \tag{9}$$

$$X_{i+1} = \omega X_i + c_2 r\left(X_g - X_w\right) \tag{10}$$

$$all(x \geq VarMin) \quad \text{AND} \quad all(x \leq VarMax) \tag{11}$$

Where, X_{i+1} = new position, X_i = previous frog position, r = randomly generated number between 0 and 1.

This conditional procedure reduces the number of unnecessary cost function evaluations, thus speeding up the algorithm. The local / global search steps are repeated a set number of times, following which, the memeplexes are shuffled for meme (information) exchange at the global level. This sequence is repeated till convergence is achieved. An overview of the Modified SFLA algorithm is given in Figure 2.

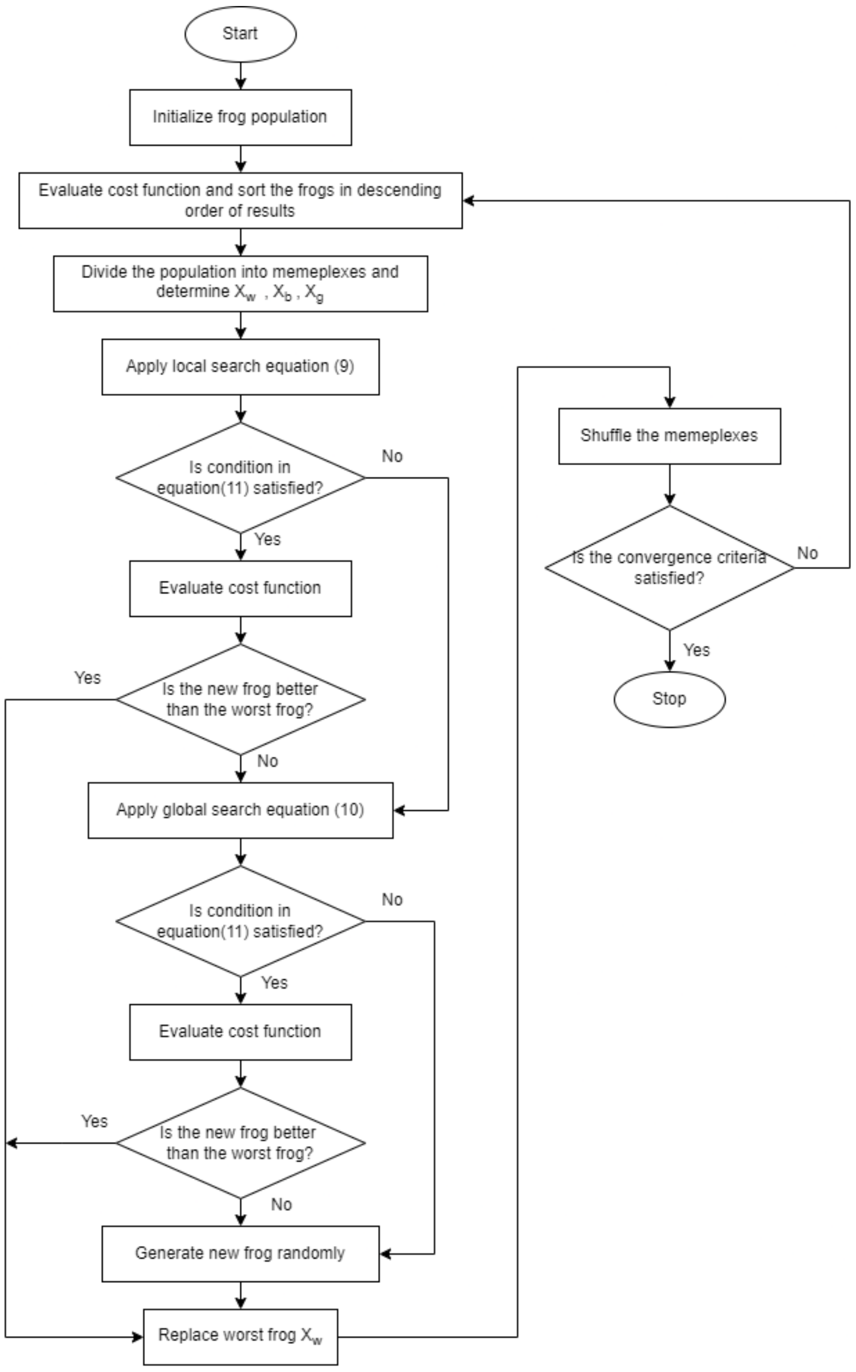
Start
Initialize frog population
Evaluate cost function and sort the frogs in descending order of results
Divide the population into memeplexes and determine X_w, X_b, X_g
Apply local search equation (9)
Is condition in equation(11) satisfied?
No
Yes
Evaluate cost function
Is the new frog better than the worst frog?
Yes
No
Apply global search equation (10)
Is condition in equation(11) satisfied?
No
Yes
Evaluate cost function
Is the new frog better than the worst frog?
Yes
No
Generate new frog randomly
Replace worst frog X_w
Shuffle the memeplexes
Is the convergence criteria satisfied?
No
Yes
Stop

Fig. 2 - Flowchart for modified SFLA algorithm

3. Theory and Calculation

3.1 Benchmarking and Validation

The code for PSO and SFLA was validated using 15 benchmark functions (refer to the Supplementary Materials for the 3D plots), which included both unimodal (functions 7, 1, and 2) and multimodal (functions 6, 9, and 12), functions with many local optima (functions 1, 7-9, 11, 13-14) and many global optima (functions 13 and 15). Each benchmark function was run 30 times, with 1000 iterations each. The total population size for both algorithms was 50 (5 memeplexes with 10 frogs per memeplex for SFLA).

The statistical results for the benchmark functions are given in Table 1. The main focus of this analysis was on the mean and standard deviation of the cost function evaluations since they are indicative of the repeatable optimality of the algorithms. This is especially important because the populations generated for each run of the algorithm differ.

From Table 1, we can see that in general, the mean results obtained using SFLA were better than those obtained through PSO. The exceptions to this observation were functions 11 (5d and 10d), 12, and 14, where the PSO results were better, and function 13, where the mean was the same. Consequently, the general trend observed for standard deviation was that it was lower for SFLA as compared to PSO. The average number of convergence iterations for SFLA were lesser compared to PSO, except for functions 1, 2 and 5. Conversely, the average run time (for 1000 iterations) for SFLA is higher.

Table 1 - Results of benchmark functions

N.	**Function**	**Range**	**d**	**PSO | $\omega = 0.65$, $c_1 = 1.65$, $c_2 = 1.75$**				**SFLA | $\omega = 1$, $c_1 = c_2 = 2$**			
				Avg RT	**Avg CI**	**Mean**	**Std. Dev.**	**Avg RT**	**Avg CI**	**Mean**	**Std. Dev.**
1	Sphere	[-100, 100]	30	1.1457	999	3.8051E-17	6.7902E-17	8.7681	1000	3.4576E-71	7.4977E-71
		[-10, 10]		1.1803	999	3.8051E-19	6.7902E-19	7.9281	1000	2.4754E-71	7.6592E-71
		[-5.12, 5.12]		1.2557	999	9.9749E-20	1.7800E-19	8.1669	1000	4.4202E-71	2.3782E-70
2	Sum Square	[-100, 100]	30	0.9550	990	3.3333E+02	1.8257E+03	5.9185	1000	1.1498E-65	6.2831E-65
		[-10, 10]		0.7007	990	3.3333E+00	1.8257E+01	6.1128	1000	1.0022E-70	3.7445E-70
		[-5.12, 5.12]		1.1036	990	8.7381E-01	4.7861E+00	6.0267	1000	1.2627E-69	6.8912E-69
3	Matyas	[-10, 10]	2	0.6696	996	1.4622E-97	7.3964E-97	5.6252	996	1.5040E-311	0.0000E+00
4	Booth	[-10, 10]	2	0.6571	261	0.0000E+00	0.0000E+00	6.3799	52	0.0000E+00	0.0000E+00
5	Zakharov	[-5, 10]	10	0.6560	998	2.4123E-32	5.3329E-32	5.8687	1000	3.7042E-98	1.9266E-97
			30	0.6969	996	1.5002E+01	3.1180E+01	6.1667	1000	3.3074E-06	1.5700E-05
6	Beale	[-4.5, 4.5]	2	0.6428	312	5.0805E-02	1.9334E-01	6.4111	86	9.2445E-34	5.0634E-33
7	Easom	[-100, 100]	2	0.6225	137	-1.0000E+00	0.0000E+00	6.6426	21	-1.0000E+00	0.0000E+00
8	Eggholder	[-512, 512]	2	0.6371	223	-8.3256E+02	1.0610E+02	6.2042	138	-8.8937E+02	7.5537E+01
9	Michale-wicz	[0, pi]	2	0.6618	159	-1.8013E+00	9.0336E-16	6.3598	38	-1.8013E+00	9.0336E-16
			5	0.7071	343	-4.5584E+00	1.9100E-01	6.2610	221	-4.4645E+00	1.9075E-01
			10	1.0580	546	-8.8041E+00	5.1623E-01	9.3777	367	-8.6069E+00	5.5608E-01
10	Dixon-Price	[-10, 10]	4	1.0869	396	3.6978E-32	0.0000E+00	8.4646	133	1.1710E-31	1.4504E-31
			10	1.0375	451	6.0000E-01	2.0342E-01	8.5383	265	4.2222E-01	3.2676E-01
			30	0.7108	983	6.6667E-01	3.7383E-05	6.5714	611	6.6667E-01	3.8246E-13
11	Ackley	[-32.768, 32.768]	2	0.5017	255	8.8818E-16	0.0000E+00	5.8666	36	8.8818E-16	0.0000E+00
			5	0.4981	507	1.9540E-15	1.6559E-15	5.8998	141	2.0724E-15	1.7034E-15
			10	0.5163	561	4.6777E-15	9.0135E-16	5.8978	159	4.8490E-01	6.7197E-01
12	Drop-Wave	[-5.12, 5.12]	2	0.4952	143	-1.0000E+00	0.0000E+00	5.9452	82	-9.8725E-01	2.5938E-02
13	Shubert	[-10, 10]	2	0.5047	250	-1.8673E+02	4.4157E-14	5.7148	228	-1.8673E+02	3.2534E-14
		[-5.12, 5.12]		0.5031	149	-1.8673E+02	2.4755E-14	5.7619	29	-1.8673E+02	2.4755E-14
14	Griewank	[-600, 600]	2	0.5010	327	2.4653E-04	1.3503E-03	9.2549	144	6.2457E-03	5.5594E-03
			5	0.5110	617	2.0609E-02	1.2647E-02	7.5310	269	1.2376E-01	8.2825E-02
			10	0.5272	700	7.0055E-02	3.9644E-02	5.8209	273	8.9725E-02	6.1086E-02
15	Cross-in-Tray	[-10, 10]	2	0.5025	127	-2.0626E+00	9.0336E-16	5.7631	20	-2.0626E+00	9.0336E-16

Out of the 15 benchmark functions evaluated, it is evident that 1000 iterations were insufficient to achieve complete convergence for functions 1, 2, 3, and 5. This was the case even after the search space was reduced. Nonetheless, the solutions obtained for these functions were within desirable limits, but there was scope for further improvement if the number of iterations were increased. The only outlier to this observation is the results obtained for function 2 using PSO, even with reduced search space. Apart from this, the results obtained for function 14 (both algorithms) and 6 (using PSO) could have been better. On the other hand, both algorithms were unable to achieve optimality for functions 8 and 9 (10d). Based on the promising results obtained for the benchmark functions, the algorithms were used to attempt the case study.

3.2 Case Study

Figure 3 provides a diagrammatic representation of the speed reducer case study attempted. The original constraints, objective function, and assumptions by [4] can be found in the Supplementary Materials. These equations were simplified and re-written as listed below by [6] and were used in this case study.

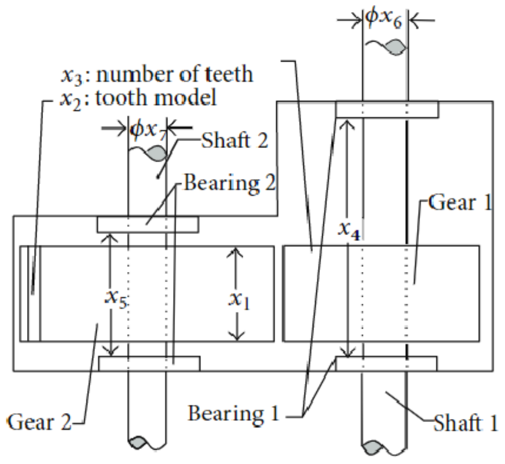

Fig. 3 - Schematic of the Golinski speed reducer [6]

Objective Function:

$$f(x_1,.., x_7) = (0.7854 \times 3.3333)x_1x_2^2x_3^2 + (0.7854 \times 43.0934)x_1x_2^2x_3 - 1.508x_1x_6^2 \\ -1.508x_1x_7^2 + 7.4777x_6^2 - 7.4777x_7^2 + 0.7854x_4x_6^2 + 0.7854x_5x_7^2$$

Constraints:

$$g_1 = 27x_1^{-1}x_2^{-2}x_3^{-1} - 1 \le 0,$$
$$g_2 = 397.5x_1^{-1}x_2^{-2}x_3^{-2} - 1 \le 0,$$
$$g_3 = 1.93x_4^3x_2^{-1}x_3^{-1}x_6^{-4} - 1 \le 0,$$
$$g_4 = 1.93x_5^3x_2^{-1}x_3^{-1}x_7^{-4} - 1 \le 0,$$
$$g_5 = 745^2x_4^2x_2^{-2}x_3^{-2} - 100^2x_6^6 + (16.9 \times 10^6) \le 0,$$
$$g_6 = 745^2x_5^2x_2^{-2}x_3^{-2} - 85^2x_7^6 + (157.5 \times 10^6) \le 0,$$
$$g_7 = x_2x_3 - 40 \le 0,$$
$$g_8 = 5x_2 - x_1 \le 0,$$
$$g_9 = x_1 - 12x_2 \le 0,$$
$$g_{10} = 1.5x_6 - x_4 + 1.9 \le 0,$$
$$g_{11} = 1.1x_7 - x_5 + 1.9 \le 0,$$

$$2.6 \le x_1 \le 3.6,$$
$$0.7 \le x_2 \le 0.8,$$
$$1.7 \le x_3 \le 28,$$
$$7.3 \le x_4 \le 8.3,$$
$$7.3 \le x_5 \le 8.3,$$
$$2.9 \le x_6 \le 3.9,$$
$$5 \le x_7 \le 5.5$$

Where x_1 = face width, x_2 = module, x_3 = number of pinion teeth, x_4 = length of the first shaft between bearings, x_5 = length of the second shaft between bearings, x_6 = diameter of the first shaft, x_7 = diameter of the second shaft.

The penalty approach employed by [14] was used to deal with the constraints of this optimization problem. The following are the steps involved:

- the cost function and constraints are evaluated for the set of variables obtained by the algorithm;
- if a constraint is violated, the penalty for that constraint is set to the value obtained on evaluating that constraint equation; else the penalty is set to zero. For example, if the evaluation of constraint g_8 is equal to 9, the constraint would be violated, and hence, the penalty for g_8 would be set to 9;
- after all the constraints are evaluated, the penalty for each constraint (if any) is added up and multiplied by a penalty factor R_m, which is chosen on a case-to-case basis. The basic purpose is to prevent the algorithm from

converging too early. In cases where the constraint violation is miniscule, larger penalty factors are picked, and vice versa;

- lastly, the product of total penalty and R_m are added to the cost function of that evaluation as shown in equation (12):

$$z = f(x) + (R_m \times \sum penalty) \tag{12}$$

The value of z was used as the cost function value for each evaluation of the objective function. In this way, every time the set of variables violated a constraint, the possibility of that evaluation being the global optimum was reduced, hence ensuring that the global optimum parameter combinations obtained had little to no constraint violation.

4. Results and discussion

4.1 Parameter Optimization

For benchmarking and validation of the code, certain standard values were used for the c_1, c_2, and ω parameters of the modified PSO and SFLA algorithms. Due to the affect c_1, c_2, and ω have on the progression of the algorithm, different combinations of these parameters work well for different search spaces [15, 16].

Table 2. Effect of number of memeplexes and frogs per memeplex on convergence

No.	n	m	m × n	Min Cost	Max Cost	Mean Cost	Std Dev	Avg RT	Avg CI
1	10	2	20	2886.2279	2898.6941	2889.1941	3.9741E+00	3.1562	575
	10	3	30	2886.2279	2891.2402	2886.6852	1.1414E+00	4.4641	743
	10	4	40	2886.2279	2890.5270	2886.5325	9.8403E-01	5.7683	782
	10	5	50	2886.2279	2886.2628	2886.2291	6.3689E-03	7.0996	833
	10	6	60	2886.2279	2886.2280	2886.2279	2.2190E-05	11.9682	799
	10	7	70	2886.2279	2886.2279	2886.2279	7.0253E-10	12.5813	778
	10	8	80	2886.2279	2886.2279	2886.2279	3.6808E-12	14.9407	816
	10	9	90	2886.2279	2886.2279	2886.2279	4.4197E-11	12.6021	847
	10	10	100	2886.2279	2886.2279	2886.2279	4.0868E-11	13.8452	834
2	5	4	20	2886.2279	2886.3430	2886.2348	2.3653E-02	5.4799	309
	5	6	30	2886.2279	2886.2318	2886.2280	7.2008E-04	8.1512	314
	5	8	40	2886.2279	2886.2279	2886.2279	2.6503E-12	15.0238	321
	5	10	50	2886.2279	2886.2279	2886.2279	1.1852E-12	16.3917	327
	5	12	60	2886.2279	2886.2279	2886.2279	1.3978E-12	16.6759	344
	5	14	70	2886.2279	2886.2279	2886.2279	3.3778E-13	19.0712	345
	5	16	80	2886.2279	2886.2279	2886.2279	8.3596E-13	22.2465	350
	5	18	90	2886.2279	2886.2279	2886.2279	1.8882E-13	23.8131	364
	5	20	100	2886.2279	2886.2279	2886.2279	1.1942E-13	26.1906	367

Note: n = number of frogs per memeplex, m = number of memeplexes.

Hence, an experiment was carried out to understand the relation between number of memeplexes, frogs per memeplex, and convergence for the SFLA algorithm. These results are presented in Table 2 for a parameter combination of ω = 1, and $c_1 = c_2 = 2$. The results highlighted that the standard deviation of the cost function evaluations for each set of 30 runs in SFLA reduced with increasing total population size (Table 2). The average run time and number of iterations to achieve converge for 1000 iterations also increased with the increase in total population. Another important observation was that the standard deviation reduced when the number of memeplexes were increased (for the same total population size). The average run time increased as the total population increased, but the time taken for the experimental set with increased number of memeplexes was higher as compared to the ones with lesser number of memeplexes (for the same total population). One interesting observation was that, although the average number of iterations to achieve convergence was increased as the total population increased, the number of iterations required were far lesser than the number required with lesser memeplexes.

From Table 2, it can be noted that the optimal solution is around 2886 (including penalty). An experiment was carried out to select the best combination of parameters for both, PSO as well as SFLA, within the defined ranges for c_1, c_2, and ω. Since the value of c_1 and c_2 should lie between 0 and 2, and w, 0 to 1, this was done with increments of 0.5. The total number of combinations were 75. This experiment was also repeated for different total population sizes to check the effect of population size on the optimal solutions obtained. For SFLA, as per the results obtained in Table 2, the number of memeplexes was increased (with 10 frogs per memeplex). The complete result table for this experiment is given in the Supplementary Materials. Based on the results obtained, suitable combinations of parameters were selected for the modified PSO and SFLA algorithms.

It was observed that whenever any of the three parameters in the combination was set to zero for PSO, the minimum, maximum, and mean value of the cost function evaluations obtained were higher, as compared to combinations that were non-zero. This observation can be visualized from the sharp peaks in Figure 4 (a). A very similar trend can be seen for the standard deviation in Figure 4 (b). From Figure 4 (a) and (b), it is evident that the values of ω, c_1, and c_2 increase from 0, the number of iterations required for the solution to converge is higher. It was observed that as a general trend, as the values for the 3 parameters increases, the average CI required also increases. The elimination method was used to decide the best combination of parameters for modified PSO to solve this case study. This combination was: $\omega = 0.5$, $c_1 = 1.5$ and $c_2 = 1.5$, for a population size of 30.

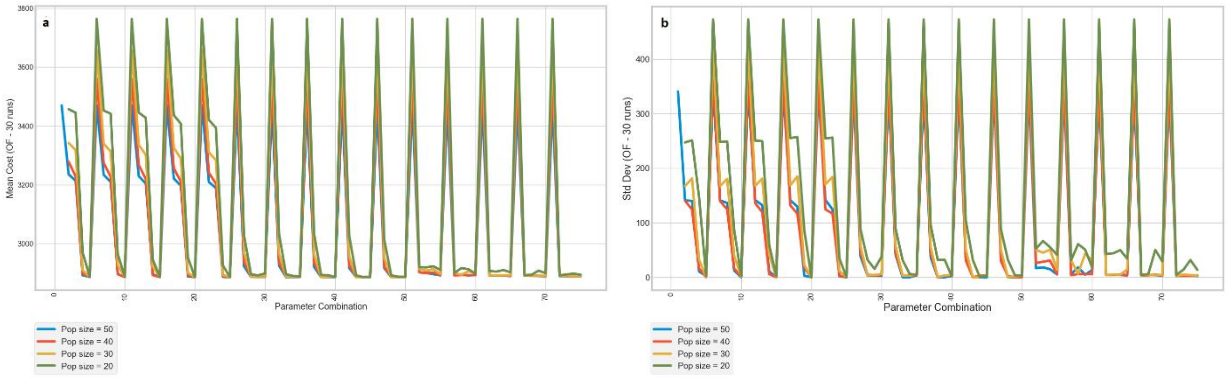

Fig. 4 - Effect of ω, c_1, c_2 on (a) Mean Cost, (b) Standard Deviation for Modified PSO

Note: The values on the x-axis represent the serial number for the parameter combination. Further information can be obtained in the Supplementary Materials.

Analysis of the results for SFLA indicate that for every combination where $\omega < 1$, there was negligible change in the cost function evaluations, standard deviation, average run time, and number of convergence equations required (supplementary materials). For combinations with $\omega = 1$, there is a trend similar to the one observed for PSO, except that the rise in the mean cost function value is more gradual, unlike the sharp peaks observed for PSO (Figure 5 (a)). This trend continues for 3 cycles and then converges once $c_1 \geq 1.5$. It can be noted that, the smaller the total population size, the greater is the cost function evaluation, for the same set of parameters (Figure 5, (a) and (b)). As was the case for modified PSO, the elimination method was used to identify the best combination of parameters for modified SFLA for this case study: $\omega = 1$, $c_1 = 2$ and $c_2 = 0.5$, for a population size of 150 (15 memeplexes with 10 frogs each).

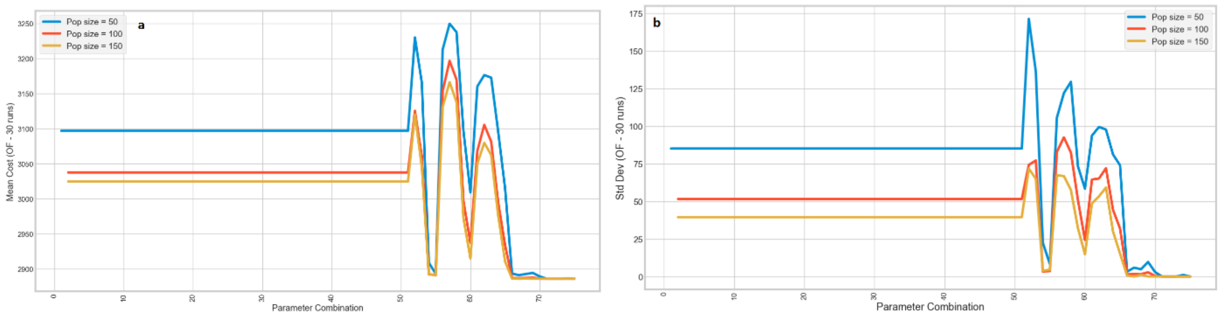

Fig. 5 - Effect of ω, c_1, c_2 on (a) Mean Cost, (b) Standard Deviation for Modified SFLA

Note: The values on the x-axis represent the serial number for the parameter combination. Further information can be obtained in the Supplementary Materials.

4.2 Comparison of Results with Previous Literature

The results of the case study using the modified PSO and SFLA algorithms were the same - 2886.2279 (with penalty) and 2640.9739 (without penalty). The total penalty was 1.2292 for PSO and 1.2263 for SFLA, for a penalty factor (r_m) of 200.

Table 3 gives us a broad comparison of the different reported optimal solutions for the Golinski speed reducer problem obtained by researchers over the years [6,17–26]. The OF and constraint equation values were re-calculated for each of the previous results (Table 4). As is clearly visible in Table 3, the results obtained using the modified PSO and SFLA algorithms (the OF value with the penalty added, i.e., 2886.2279) is one of the best solutions.

Table 3. Comparison of case study variables and OF (with penalty) with previous literature

No	**Authors**	**Algorithm**	**Variables**	**OF (with penalty, if any)**
1	K'uang J. Ku et al.	Taguchi method	$x_1 = 3.6$, $x_2 = 0.7$, $x_3 = 17$, $x_4 = 7.3$, $x_5 = 7.8$, $x_6 = 3.4$, $x_7 = 5.0$	2876.2200
2	Akhtar et al.	Socio-Behavioural Simulation Model	$x_1 = 3.506122$, $x_2 = 0.700006$, $x_3 = 17$, $x_4 = 7.549126$, $x_5 = 7.859330$, $x_6 = 3.365576$, $x_7 = 5.289773$	3008.1974
3	Rao and Xiong	Mixed Discrete Hybrid Genetic Algorithm (MDHGA)	$x_1 = 3.5$, $x_2 = 0.7$, $x_3 = 17$, $x_4 = 7.3$, $x_5 = 7.8$, $x_6 = 3.36$, $x_7 = 5.29$	3000.8300
4	Leticia C. Cagnina et al.	Simple Constrained Particle Swarm optimizer (SiC-PSO)	$x_1 = 3.5$, $x_2 = 0.7$, $x_3 = 17$, $x_4 = 7.3$, $x_5 = 7.8$, $x_6 = 3.350214$, $x_7 = 5.286683$	2996.3482
5a	Jaberipour and Khorram	Proposed Harmony Search Algorithm (PHS)	$x_1 = 3.5$, $x_2 = 0.7$, $x_3 = 17$, $x_4 = 7.3$, $x_5 = 7.71533233833903$, $x_6 = 3.35021510925684$, $x_7 = 5.28666403545462$	2994.4775
5b		Improving Proposed Algorithm Harmony Search (IPHS)	$x_1 = 3.5$, $x_2 = 0.7$, $x_3 = 17$, $x_4 = 7.3$, $x_5 = 7.71599501113801$, $x_6 = 3.35025375091328$, $x_7 = 5.28690759750734$	2994.9000
6	Li and Papalabros	Production System for Global Knowledge	$x_1 = 3.5$, $x_2 = 0.7$, $x_3 = 17$, $x_4 = 7.3$, $x_5 = 7.71$, $x_6 = 3.35$, $x_7 = 5.29$	2994.4000
7	Tosserams et al.	Augmented Lagrangian Decomposition Method	$x_1 = 3.5$, $x_2 = 0.7$, $x_3 = 17$, $x_4 = 7.3$, $x_5 = 7.72$, $x_6 = 3.35$, $x_7 = 5.29$	2996.6458
8	Lu & Kim	Regularized Inexact Penalty Decomposition Algorithm	$x_1 = 3.5$, $x_2 = 0.7$, $x_3 = 17$, $x_4 = 7.3$, $x_5 = 7.670396$, $x_6 = 3.542421$, $x_7 = 5.245814$	3019.5834
9	Huang C	Geometric Programming (GP)	$x_1 = 3.495652$, $x_2 = 0.7000002$, $x_3 = 17$, $x_4 = 7.30000007$, $x_5 = 7.7120386$, $x_6 = 3.343372$, $x_7 = 5.285352$	2990.1244
10	Lin M. et al.	Convexification strategies and piecewise linearization methods	$x_1 = 3.5$, $x_2 = 0.7$, $x_3 = 17$, $x_4 = 7.3$, $x_5 = 7.715319$, $x_6 = 3.350282$, $x_7 = 5.286654$	2994.4719
11a	**Golinski (Original case study)**	Crude Monte-Carlo	$x_1 = 4.4$, $x_2 = 0.6$, $x_3 = 17$, $x_4 = 7.3$, $x_5 = 8.1$, $x_6 = 3.4$, $x_7 = 5$	2236.3500
11b		Stray Process	$x_1 = 3.6$, $x_2 = 0.7$, $x_3 = 18$, $x_4 = 6.6$, $x_5 = 8.2$, $x_6 = 2.8$, $x_7 = 5.2$	2247.7900
A1	**Our solutions**	Modified PSO	$x_1 = 2.6$, $x_2 = 0.7$, $x_3 = 17$, $x_4 = 7.3$, $x_5 = 7.715319911$, $x_6 = 3.350214666$, $x_7 = 5.286654465$	2886.2279
A2		Modified SFLA	$x_1 = 2.6$, $x_2 = 0.7$, $x_3 = 17$, $x_4 = 7.3$, $x_5 = 7.715319911$, $x_6 = 3.350214666$, $x_7 = 5.286654465$	2886.2279

Note: OF represents the evaluation of the Objective Function – (with penalty values added for No. A1 and A2). x_1 to x_7 represent the values for each of the 7 variables.

The only two solutions with comparable results were those obtained using the Taguchi Method by [17] and the Crude Monte-Carlo and Stray Process algorithms by [4]. However, on closer observation, it is obvious from Table 4 that the constraint violations for these algorithms are far greater than the violations using the modified PSO and SFLA algorithms in the present work. Also, from Table 4, it is evident that the cost function value presented by [4] does not match the results presented in their paper. Based on these observations, the values obtained in the present work using modified PSO and SFLA as relatively more optimal.

Table 4. Comparison of results of re-calculated OF and constraint equations (values obtained from previous literature)

No	**f(x)**	**Constraints**
1	2876.219475	g(1) = -0.09964, g(2) = -0.220276, g(3) = -0.527868, g(4) = -0.876856, g(5) = -1.5833E+06, **g(6) = 4.4848E+07**, g(7) = -28.1, g(8) = -0.1, g(9) = -4.8, g(10) = -0.3, g(11) = -4.0000E-01
2	3008.19744	g(1) = -0.075548, g(2) = -0.199413, g(3) = -0.456175, g(4) = -0.899442, g(5) = -4.6162E+05, g(6) = -5.5031E+05, g(7) = -28.099898, g(8) = -0.006092, g(9) = -4.89395, g(10) = -0.600762, g(11) = -1.4058E-01
3	3000.959715	g(1) = -0.073915, g(2) = -0.197999, g(3) = -0.504981, g(4) = -0.901719, g(5) = -3.0203E+05, g(6) = -5.9471+05, g(7) = -28.1, g(8) = 0, g(9) = -4.9, g(10) = -0.36, g(11) = -8.1000E-02
4	2996.347849	g(1) = -0.073915, g(2) = -0.197999, g(3) = -0.499172, g(4) = -0.901472, **g(5) = 2.0410E+01**, **g(6) = 4.1132E+01**, g(7) = -28.1, g(8) = 0,

		g(9) = -4.9, g(10) = -0.374679, g(11) = -8.4649E-02
5a	2994.477531	g(1) = -0.073915, g(2) = -0.197999, g(3) = -0.499173, g(4) = -0.904644, g(5) = -1.3579E+01, g(6) = -1.7125E+03, g(7) = -28.1, g(8) = 0, g(9) = -4.9, g(10) = -0.374677, g(11) = -1.8993E-06
5b	2994.656655	g(1) = -0.073915, g(2) = -0.197999, g(3) = -0.499196, g(4) = -0.904637, g(5) = -1.1976E+03, g(6) = -4.5280E+04, g(7) = -28.1, g(8) = 0, g(9) = -4.9, g(10) = -0.374619, g(11) = -3.9665E-04
6	2996.425995	g(1) = -0.073915, g(2) = -0.197999, g(3) = -0.499044, g(4) = -0.905082, **g(5) = 6.5765E+03**, g(6) = -6.0018E+05, g(7) = -28.1, g(8) = 0, g(9) = -4.9, g(10) = -0.375, **g(11) = 9.0000E-03**
7	2996.645783	g(1) = -0.073915, g(2) = -0.197999, g(3) = -0.499044, g(4) = -0.904712, **g(5) = 6.5765E+03**, g(6) = -5.9959E+05, g(7) = -28.1, g(8) = 0, g(9) = -4.9, g(10) = -0.375, g(11) = -1.0000E-03
8	3019.583344	g(1) = -0.073915, g(2) = -0.197999, g(3) = -0.599338, g(4) = -0.903348, g(5) = -6.8015E+06, **g(6) = 7.1687E+06**, g(7) = -28.1, g(8) = 0, g(9) = -4.9, g(10) = -0.086369, g(11) = -6.0000E-07
9	2990.124384	g(1) = -0.072764, g(2) = -0.197001, g(3) = -0.49506, g(4) = -0.904672, **g(5) = 2.0860E+05**, **g(6) = 2.3282E+05**, g(7) = -28.099997, **g(8) = 0.004349**, g(9) = -4.90435, g(10) = -0.384842, **g(11) = 1.8486E-03**
10	2994.48791	g(1) = -0.073915, g(2) = -0.197999, g(3) = -0.499213, g(4) = -0.904644, g(5) = -2.0633E+03, **g(6) = 8.3184E+01**, g(7) = -28.1, g(8) = 0, g(9) = -4.9, g(10) = -0.374577, **g(11) = 4.0000E-07**
11a	2672.019394	**g(1) = 0.002674,** g(2) = -0.131671, g(3) = -0.44918, g(4) = -0.839109, g(5) = -1.5078E+06, **g(6) = 4.4959E+07**, g(7) = -29.8, g(8) = -1.4, g(9) = -2.8, g(10) = -0.3, g(11) = -7.0000E-01
11b	3049.97544	g(1) = -0.14966, g(2) = -0.304506, g(3) = -0.283549, g(4) = -0.884491, **g(5) = 1.1221E+07**, **g(6) = 1.4892E+07**, g(7) = -27.4, g(8) = -0.1, g(9) = -4.8, g(10) = -0.5, g(11) = -5.8000E-01
A1	2640.97393	**g(1) = 0.246653,** **g(2) = 0.0796174,** g(3) = -0.499172, g(4) = -0.904644, g(5) = -7.0781E-08, **g(6) = 2.9802E-07**, g(7) = -28.1, **g(8) = 0.9**, g(9) = -5.8, g(10) = -0.374678, g(11) = -7.5495E-15
A2	2640.97393	**g(1) = 0.246653,** **g(2) = 0.0796174,** g(3) = -0.499172, g(4) = -0.904644, g(5) = -7.0781E-08, **g(6) = 2.9802E-07**, g(7) = -28.1, **g(8) = 0.9**, g(9) = -5.8, g(10) = -0.374678, g(11) = -7.5495E-15

Note: f(x) is the evaluation of the OF (Objective Function) – without penalty values added for No. A1 and A2. g(1) to g(11) represents the evaluation of the 11 constraints. The cells highlighted in bold text represent the constraints that were violated.

The percentage difference in the results obtained using modified PSO and SFLA (without penalty) in comparison with previous results is shown in Table 5.

Table 5. Percentage improvement in results

No.	**Author(s)**	**OF (without penalty)**	**% Improvement**
1	K'uang J. Ku et al.	2876.2200	8.1790%
2	Akhtar et al.	3008.1974	12.2074%
3	Rao and Xiong	3000.8300	11.9919%
4	Leticia C. Cagnina et al.	2996.3482	11.8602%
5a	Jaberipour and Khorram	2994.4775	11.8052%
5b		2994.9000	11.8176%
6	Li and Papalabros	2994.4000	11.8029%
7	Tosserams et al.	2996.6458	11.8690%
8	Lu and Kim	3019.5834	12.5385%
9	Huang	2990.1244	11.6768%
10	Lin et al. (2012)	2994.3410	11.8012%
11	Lin et al. (2013)	2994.4719	11.8050%
12a	Dr Jan Golinski (Original case study)	2236.3500	-18.0931%
12b		2247.7900	-17.4920%
12a*		2672.0194	1.1619%
12b*		3049.9754	13.4100%
A1	Modified PSO	2640.9739	
A2	Modified SFLA	2640.9739	

*Note: * indicates the actual OF values obtained after re-calculating results*

The results of this project indicate that there is a negative difference in the cost function value (18.0931%) as compared to the result obtained by [4] using the Crude Monte-Carlo algorithm, and a 17.4920% difference in comparison with the Stray Process result. But as mentioned earlier, the values of the OF evaluations presented by Golinski do not match the re-calculated values. Based on these re-calculated values, the percentage difference between Golinski's results and the results obtained in this project is 1.1619% and 13.41% for both algorithms respectively. Apart from the results presented by Golinski, the smallest percentage improvement is 8.1790%, in comparison with the Taguchi method suggested by [17]. A total of 4 constraints (constraints 1, 2, 6, and 8) were violated by the combination of parameters obtained using the modified PSO and SFLA algorithms. Since the violation in the constraints are negligible, the design of the speed reducer can be considered satisfactory:

- constraint 1 (Bending condition) violation: 0.246653;
- constraint 2 (Compressive stress limitation) violation: 0.079617;
- constraint 6 (Stress condition for shaft 2) violation: 0.00000029802;
- constraint 8 (Relative face width condition) violation: 0.9.

Conclusions

Modified PSO and SFLA are both powerful nature-inspired algorithms that can obtain optimal results for a variety of benchmark functions (unimodal, multimodal, multiple local and global optima). However, it is important to note that different parameter combinations work better for different optimization problems with different search spaces.

The results obtained using modified PSO and SFLA described in the present work had an 8.1790% reduction in cost function evaluation as compared to the Taguchi method by [17], 1.1619% reduction compared to the Crude Monte-Carlo method and 13.41% reduction compared to the Stray Process used by [4], and around an 11% reduction compared to results obtained by other researchers. Modified SFLA generally takes lesser number of convergence iterations (CI) on average as compared to modified PSO, but the average run time (RT) for the same number of iterations (1000) is greater for modified SFLA. So, it's essentially a trade-off between average CI and average RT.

Further, the results for variables x_1 to x_7 obtained using modified PSO and SFLA can be verified by developing CAD models and conducting finite element analysis (FEA). In this way, the minor constraint violations can be evaluated to check the degree of damage caused by them.

As per the No Free Lunch theorem, no single algorithm is best suited for all optimization problems. Along the same lines, certain nature-inspired algorithms work better than others for speed reducer problems. The algorithms with promising results can be analysed to understand their components/mechanisms that support their optimal results. For example, the inertia weight factor in PSO is attributed to its success in a variety of applications. Similarly, in memetic algorithms like SFLA, since information can be directly transmitted among frogs within the same iteration instead of waiting for the next one, the propagation of optimal memes is faster, which contributes to the faster convergence observed in this algorithm. For future work, these algorithm-specific mechanisms could be identified and combined in the form of a hybrid algorithm that could be used to solve families of similar optimization problems.

References

[1] Zolfaghari A., Goharimanesh M., Akbari A.A. Optimum design of straight bevel gears pair using evolutionary algorithms //Journal of the Brazilian Society of Mechanical Sciences and Engineering, 39, 2017, 2121–2129

[2] Qin Z., Wu Y.T., Lyu S.K. A Review of recent advances in design optimization of gearbox //International journal of precision engineering and manufacturing, 19, 2018, 1753–1762

[3] Dalavi A.M., Gomes A., Javed Husain A. Bibliometric analysis of nature inspired optimization techniques //Comput Ind Eng, 169, 2022,108161

[4] Golinski J. Optimal synthesis problems solved by means of nonlinear programming and random methods, Journal of Mechanisms 5,1970, P. 287–309. https://doi.org/10.1016/0022-2569(70)90064-9.

[5] Ray T. Golinski's speed reducer problem revisited //AIAA Journal, 41, 2003, P. 556–558

[6] Lin M.H., Tsai J.F., Hu N.Z., Chang S.C. Design optimization of a speed reducer using deterministic techniques //Math Probl Eng, 2013, 2, P.1-7

[7] Kennedy J., Eberhart R. Particle Swarm Optimization, 1995.

[8] Shi Y., Eberhart R. Modified particle swarm optimizer, Proceedings of the IEEE Conference on Evolutionary Computation, ICEC, 1998, P. 69–73

[9] Heris M.K. Particle Swarm Optimization (PSO) in MATLAB - Video Tutorial, Yarpiz, 2016.

[10] Eusuff M.M., Lansey K.E. Optimization of water distribution network design using the shuffled frog leaping Algorithm // J Water Resour Plan Manag, 129, 2003. P. 210–220

[11] Elbeltagi E., Hegazy T., Grierson D. A modified shuffled frog-leaping optimization algorithm: Applications to project management, Structure and Infrastructure Engineering 3, 2007, P. 53–60. https://doi.org/10.1080/15732470500254535.

[12] Dalavi A.M., Pawar P.J., Singh T.P. Tool path planning of hole-making operations in ejector plate of injection mould using modified shuffled frog leaping algorithm //J Comput Des Eng, 3, 2016, P. 266–273

[13] Heris M.K. Shuffled Frog Leaping Algorithm in MATLAB, Yarpiz, 2015

[14] Parsopoulos K., Vrahatis M.N. Particle Swarm Optimization Method for Constrained Optimization Problems, in: P. Sincak, J. Vascak, V. Kvasnicka, J. Pospichal (Eds.), Intelligent Technologies: Theory and Applications: New Trends in Intelligent Technologies, IOS Press (Frontiers in Artificial Intelligence and Applications series), Fairfax, VA, U.S.A., 2002: P. 1–7.
[15] Rao R.V., Savsani V.J. Mechanical Design Optimization Using Advanced Optimization Techniques, Springer-Verlag London, 2012
[16] Van Den Bergh F., Engelbrecht A.P. A study of particle swarm optimization particle trajectories //Inf Sci (N Y), 176, 2006, P. 937–971
[17] Ku K.J., Rao S.S., Chen L. Taguchi-aided search method for design optimization of engineering systems //Engineering Optimization 30, 1998, P. 1–23
[18] Akhtar S., Tai K., Ray T. A socio-behavioural simulation model for engineering design optimization //Engineering Optimization 34(4), 2002, P.341–354
[19] Rao S.S., Xiong Y. A hybrid genetic algorithm for mixed-discrete design optimization, //Journal of Mechanical Design, Transactions of the ASME 127, 2005, P. 1100–1112
[20] Tomassetti G., Cagnina L. Particle swarm algorithms to solve engineering problems: A comparison of performance, Journal of Engineering, 2023, P. 1–13
[21] Jaberipour M., Khorram E. Two improved harmony search algorithms for solving engineering optimization problems, Undefined 15, 2010, P.3316–3331
[22] Li H.L., Papalambros W. A Production System for Use of Global Optimization Knowledge //Journal of Mechanisms, Transmissions, and Automation in Design, 107, 1985, P.277–284
[23] Tosserams S., Etman L.F.P., Rooda J.E. An augmented Lagrangian decomposition method for quasi-separable problems in MDO //Structural and Multidisciplinary Optimization, 34, 2007, P. 211–227
[24] Lu S., Kim H.M. A Regularized inexact penalty decomposition algorithm for multidisciplinary design optimization problems with complementarity constraints // Journal of Mechanical Design, Transactions of the ASME, 2010, P. 0410051–04100512
[25] Huang C.H. Engineering design by geometric programming //Math Probl Eng, 2013,
[26] Lin M.-H., Tsai J.-F., Wang P.-C. Solving Engineering optimization problems by a deterministic global optimization approach //Applied Mathematics & Information Sciences, 2012, P. 1101–1107.

Information of the authors

Amol Dalavi, PhD, assistant professor, Symbiosis Institute of Technology, Pune, India
e-mail: amol.dalavi@sitpune.edu.in

Alyssa Gomes, BTech student, Symbiosis Institute of Technology, Pune, India

Aaliya Husain, BTech student, Symbiosis Institute of Technology, Pune, India