AIFA|ASSIGNMENT 1 [AI61005]

IIT KHARAGPUR

Topic : Design Optimization in Mechanical Equipments

ABSTRACT

Study of the background and implementation of a teaching-learning based optimization algorithm (TLBO- It is a population-based method and uses a population of solutions to proceed to the global solution) for optimization task of a mechanical component (optimizing weight in a belt-pulley drive and hollow shaft). Simulation result on the optimization (mechanical components) problems reveals the ability of the proposed methodology to find better optimal solutions compared to other optimization algorithms.

INTRODUCTION

Majority of mechanical design includes an optimization task in which engineers always consider certain objectives such as weight, wear, strength, deflection, corrosion, and volume depending on the requirements. However, design optimization for a complete mechanical system leads to a cumbersome objective function with a large number of design variables and complex constraints. Hence, it is a general procedure to apply optimization techniques for individual components or intermediate assemblies rather than a complete assembly.

Numerical methods are traditionally used for finding the extremes of a function for engineering computations. These optimization procedures perform well in many practical cases. However , they may fail to perform in more complex design situations or in real time optimization (design) problems, where the number of design variables are very large, and their influence on the objective function to be optimized are very complicated.

Optimization is a method of obtaining the best result under the given circumstances. It plays a vital role in machine design because the mechanical components are to be designed in an optimal manner.

There are several methods available in the literature of optimization. Some of them are direct search methods and others are gradient methods. In direct search method, only the function value is necessary, whereas the gradient based methods require gradient information to determine the search direction. However, there are some difficulties with most of the traditional methods of optimization. The most commonly used evolutionary optimization technique is the genetic algorithm (GA).

Recently, a new optimization technique, known as teaching-learning based optimization (TLBO), has been developed. It is one of the recent evolutionary algorithms and is based on the natural phenomenon of teaching and learning process. It has already proved its superiority over other existing optimization techniques such as GA, ABC, PSO, harmony search (HS), DE, and hybrid-PSO. *TLBO is effective than other meta heuristics according to its characteristics viz. less computation effort, Parameter less, and high consistency. It outperforms some of the known met heuristics concerning continuous non-linear numerical optimization problems,constrained benchmark functions, constrained mechanical design.*

Mathematical Formulation

Case1:Optimum Design of Hollow Shaft

Objective function: The objective of this study is to minimize the weight of a hollow shaft which is given by the expression: W_s=cross sectional area*length *density

$$=(pi/4) * (d_0^2 - d_1^2) * L * \rho \qquad (1)$$

Substituting the values of L,ρ as 50 cm and 0.0083 kg/cm3, respectively, one finds the weight of the shaft and it is given by: $W_s = 0.326 d_0^2 * (1 - K^2)$ (2)

Constraint:

The twisting failure can be calculated from the torsion formula as given below:

$$\theta = \frac{TL}{GJ}. \qquad (3)$$

Now, θ applied should be greater than TL/GJ; that is, $\theta \geq TL/GJ$.

Substituting the values of θ, T, G, J as $2\pi/180$ per m length, 1.0×10^5 kg-cm, 0.84×10^6 kg/cm^2, and $[(\pi/32)d_0^4(1-k^4)]$, respectively, one gets the constraints as

$$d_0^4\left(1-k^4\right) - 1736.93 \geq 0. \tag{4}$$

The critical buckling load (T_{cr}) is given by the following expression:

$$T_{\mathrm{cr}} \leq \frac{\pi d_0^3 E(1-k)^{2.5}}{12\sqrt{2}(1-\gamma^2)^{0.75}}. \tag{5}$$

Substituting the values of T_{cr}, γ, and E to 1.0×10^5 kg-cm, 0.3, and 2.0×10^5 kg/cm^2, respectively, the constraint is expressed as

$$d_0^3 E(1-k)^{2.5} - 0.4793 \leq 0. \tag{6}$$

'he ranges of variables are mentioned as follows:

$$\begin{gathered} 7 \leq d_0 \leq 25, \\ 0.7 \leq k \leq 0.97. \end{gathered} \tag{7}$$

Case 2:Optimum Design of Belt-Pulley Drive

Objective function:

The weight of the pulley is considered as objective function which is to be minimized as: $W_p = \pi\rho b\left[d_1 t_1 + d_2 t_2 + d_1^1 t_1^1 + d_2^1 t_2^1\right]$.(8)

Assuming $t_1 = 0.1d_1$, $t_2 = 0.1d_2$, $t_1^1 = 0.1d_1^1$, and $t_2^1 = 0.1d_2^1$ and replacing d_1, d_2, d_1^1, and d_2^1 by N_1, N_2, N_1^1, and N_2^1, respectively, and also substituting the values of N_1, N_2, N_1^1, and N_2^1, ρ (to 1000, 250, 500, 500) 7.2×10^{-3} kg/cm^3, respectively, the objective function can be written as $W_p = 0.113047d_1^2 + 0.0028274d_2^2$. (9)

Constraint:

The transmitted power (P) can be represented as

$$P = \frac{(T_1 - T_2)}{75}V. \quad (10)$$

Substituting the expression for V in the above equation, one gets

$$P = (T_1 - T_2)\frac{\pi d_p N_p}{75 \times 60 \times 100}, \quad (11)$$

$$P = T_1\left(1 - \frac{T_2}{T_1}\right)\frac{\pi d_p N_p}{75 \times 60 \times 100}. \quad (12)$$

Assuming $T_2/T_1 = 1/2$, $P = 10$ hp and substituting the values of T_2/T_1 and P, one gets

$$10 = T_1\left(1 - \frac{1}{2}\right)\frac{\pi d_p N_p}{75 \times 60 \times 100}$$

or

$$T_1 = \frac{286478}{d_p N_p}. \qquad (13)$$

Assuming

$$d_2 N_2 < d_1 N_1,$$
$$T_1 < \sigma_b b t_b. \qquad (14)$$

And considering (26) to (28), one gets

$$\sigma_b b t_b \geq \frac{2864789}{d_2 N_2}. \qquad (15)$$

Substituting $\sigma_b = 30\ \mathrm{kg/cm^2}$ $t_b = 1\ \mathrm{cm}$, $N_2 = 250$ rpm in the above equation,

$$30b \times 1.0 \geq \frac{28864789}{d_2 250}$$

or

$$b \geq \frac{381.97}{d_2}$$

or

$$bd_2 - 81.97 \geq 0. \qquad (16)$$

Assuming that width of the pulley is either less than or equal to one-fourth of the dia of the first pulley, the constraint is expressed as

$$b \leq 0.25 d_1$$

or

$$\frac{d_1}{4b} - 1 \geq 0. \qquad (17)$$

The ranges of the variables are mentioned as follows:

$$15 \leq d_1 \leq 25,$$
$$70 \leq d_2 \leq 80,$$
$$4 \leq b \leq 10. \qquad (18)$$

So at this point we have completed formulating our problem into mathematical functions (objective function and other constraints) which we have to minimize / maximize under multiple constraints to get optimal solution.

GENETIC ALGORITHM

Before going to TLBO (Teaching Learning Based Optimization) we will discuss one another nature inspired algorithm called as Genetic Algorithm.

Genetic algorithm (GA) is a metaheuristic inspired by the process of natural selection that belongs to the larger class of evolutionary algorithms (EA).

A genetic algorithm is used to solve complicated problems with a greater number of variables & possible outcomes/solutions. The combinations of different solutions are passed through the Darwinian based algorithm to find the best solutions.

Working of Genetic Algorithms in AI

The working of a genetic algorithm in AI is as follows:

- The components of the population, i.e., elements, are termed as genes in genetic algorithms in AI. These genes form an individual in the population (also termed as a chromosome).
- A search space is created in which all the individuals are accumulated. All the individuals are coded within a finite length in the search space.
- Each individual in the search space (population) is given a fitness score, which tells its ability to compete with other individuals.

- All the individuals with their respective fitness scores are sought & maintained by the genetic algorithm & the individuals with high fitness scores are given a chance to reproduce.
- The new offspring are having better ‘partial solutions’ as compared to their parents. Genetic algorithms also keep the space of the search space dynamic for accumulating the new solutions (offspring).
- This process is repeated until the offsprings do not have any new attributes/features than their parents (convergence). The population converges at the end, and only the fittest solutions remain along with their offspring (better solutions). The fitness score of new individuals in the population (offspring) are also calculated.

Key Terminologies in Genetic Algorithms

- **Selection Operator** – This operator in genetic algorithms in AI is responsible for selecting the individuals with better fitness scores for reproduction.
- **Crossover Operato**r – The crossover operator chooses a crossover site from where the merge will happen. The crossover sites in both the individuals available for mating are chosen randomly and form new individuals.
- **Mutation Operator** – This operator in the genetic algorithm is responsible for embedding random genes in the offspring to maintain diversity and avoid premature convergence.

- **Premature Convergence** – If a problem is optimized quickly, it means that the offspring were not produced at many levels. The solutions will also not be of optimal quality. To avoid premature convergence, new genes are added by the mutation operator.
- **Allele** – The value of a particular gene in a chromosome is termed as an allele. The specified set of alleles for each gene defines the possible chromosomes of that particular gene.

Benefits and Uses of Genetic Algorithms

- The solutions created through genetic algorithms are strong & reliable as compared to other solutions.
- They increase the size of solutions as solutions can be optimized over a large search scale. This algorithm also can manage a large population.
- The solutions produced by genetic algorithms do not deviate much on slightly changing the input. They can handle a little bit of noise.

- Genetic algorithms have a stochastic distribution that follows probabilistic transition rules, making them hard to predict but easy to analyze.
- Genetic algorithms can also perform in noisy environments. It can also work in case of complex & discrete problems.
- Due to their effectiveness, genetic algorithms have many applications like neural networks, fuzzy logic, code-breaking, filtering & signal processing.

Multi-objective optimization formulation:

A minimization multi-objective decision problem with K objectives is defined as follows: Given an n-dimensional decision variable vector $x = \{x_1,\ldots,x_n\}$ in the solution space X, find a vector x^* that minimizes a given set of K objective functions $z(x^*)= \{z_1(x^*),\ldots,z_K(x^*)\}$. The solution space X is generally restricted by a series of constraints, such as $g_j(x^*) = b_j$ for $j = 1,\ldots, m$, and bounds on the decision variables.

The procedure of a generic GA is given as follows:

Step 1: Set t=1. Randomly generate N solutions to form the first population, P_1. Evaluate the fitness of solutions in P_1.

Step 2: Crossover: Generate an offspring population Q_t as follows:

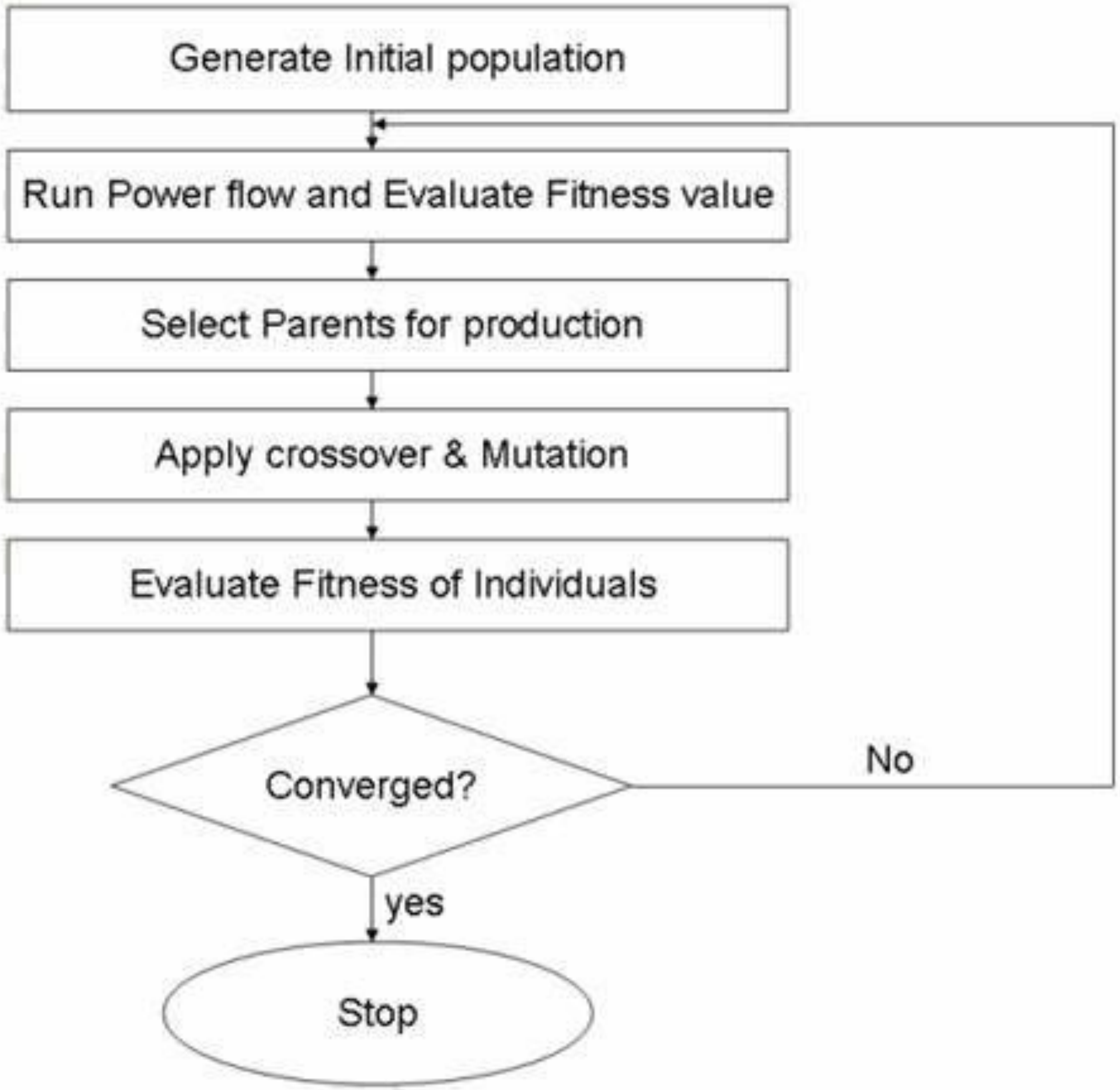
- **2.1.** Choose two solutions x and y from P_t based on the fitness values.
- **2.2.** Using a crossover operator, generate offspring and add them to Q_t.

Step 3: Mutation: Mutate each solution $x \in Q_t$ with a predefined mutation rate.

Step 4: Fitness assignment: Evaluate and assign a fitness value to each solution $x \in Q_t$ based on its objective function value and infeasibility.

Step 5: Selection: Select N solutions from Q_t based on their fitness and copy them to P_{t+1}.
Step 6: If the stopping criterion is satisfied, terminate the search and return to the current population, else,
set t = t+1 go to Step 2.

Flow chart of GA Algorithm



Optimization *(Using TLBO)*

What is TLBO (**T**eaching **L**earning **B**ased **O**ptimization)?

It is a teaching learning based algorithm inspired from nature (based on teaching-learning process in a class among the teacher and the students).
Like other nature inspired algorithms like GA (Genetic Algorithm) TLBO is also a population-based technique with a *predefined population size* that uses the population of solutions to arrive at the optimal solution . In this algorithm our ***populations are the students in a class*** and ***design variables are subjects taken by students***. The objective function value (In our case objective function is the weight of the pully and hollow shaft which need to be minimized in case 2 and case 1 respectively) symbolizes the knowledge of a particular student (Lower the value of the objective function in a minimization problem represents more knowledge). ***Each candidate solution comprises design variables responsible for the knowledge scale of a student***. The ***solution having best fitness in the population (among all students) is considered as the teacher.***

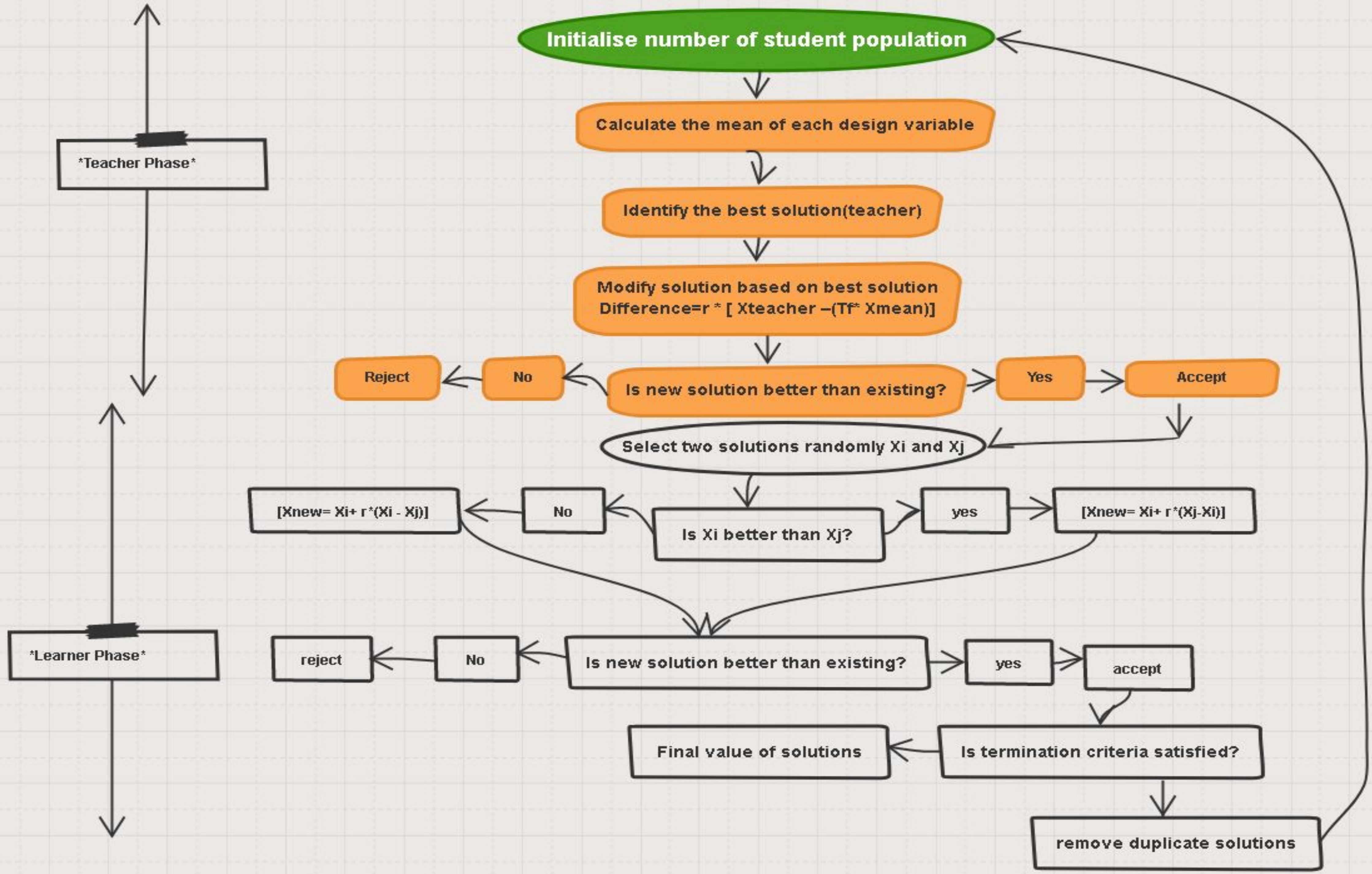
Let's consider an individual student (X_i) within the population represents a single possible solution to a particular optimization problem. X_i is a real-valued vector with D elements , where D is the dimension of the problem and is used to represent the number of subjects (Number of design variables) that an individual, either student or teacher, enrolls to learn/teach in the TLBO context. The algorithm then tries to improve certain individuals by changing these individuals during the ***Teacher and Learner Phases***, where an individual is only replaced if his/her new solution is better than his/her previous one. The algorithm will repeat until it reaches the maximum number of generations (specified).

Teacher Phase

During the Teacher Phases, the teaching role is assigned to the best individual ($X_{teacher}$). The algorithm attempts to improve other individuals (X_i) by moving their position towards the position of the $X_{teacher}$ by referring to the current mean value of the individuals (X_{mean}). This is constructed using the mean values for each parameter within the problem space (dimension) and represents the qualities of all students from the current generation. Equation: $X_{new} = X_i + r * [X_{teacher} - (T_F * X_{mean})]$ simulates how student improvement may be influenced by the difference between the teacher's knowledge and the qualities of all students. For stochastic purposes, two randomly generated parameters are applied within the equation: r ranges between 0 and 1; T_F is a teaching factor which can be either 1 or 2, thus emphasizing the importance of student quality: $X_{new} = X_i + r * [X_{teacher} - (T_F * X_{mean})]$.

Learner Phase

During the Learner Phase, student (X_i) tries to improve his/her knowledge by peer learning from an arbitrary student (X_{ii}) , where i ≠ ii. In the case that X_{ii} is better than X_i , X_i moves towards X_{ii} , X_{new} is given by $[X_{new} = X_i + r*(X_{ii} - X_i)]$ .Otherwise, it is moved away from X_{ii} $[X_{new} = X_i + r*(X_i - X_{ii})]$. If student performs better by following by $[X_{new} = X_i + r*(X_{ii} - X_i)]$ or $[X_{new} = X_i + r*(X_i - X_{ii})]$, he/she will be accepted into the population. The algorithm will continue its iterations until reaching the maximum number of generations(specified).


Initialise number of student population
Calculate the mean of each design variable
Identify the best solution(teacher)
Modify solution based on best solution
Difference=r * [Xteacher –(Tf* Xmean)]
Reject
No
Is new solution better than existing?
Yes
Accept
Teacher Phase
Select two solutions randomly Xi and Xj
[Xnew= Xi+ r*(Xi - Xj)]
No
Is Xi better than Xj?
yes
[Xnew= Xi+ r*(Xj-Xi)]
Learner Phase
reject
No
Is new solution better than existing?
yes
accept
Final value of solutions
Is termination criteria satisfied?
remove duplicate solutions

For comparing two individuals, the TLBO algorithm, according to , utilizes Deb's constrained handling method.

RULES:-

(i)If both individuals are feasible, the fitter individual (with the better value of fitness function) is preferred.
(ii)If one individual is feasible and the other one infeasible, the feasible individual is preferred.
(iii)If both individuals are infeasible, the individual having the smaller number of violations (this value is obtained by summing all the normalized constraint violations) is preferred.

Refined TLBO Using Differential Operator (DTLBO-Data-driven Teaching–Learning-Based Optimization)

DTLBO is improved TLBO to solve expensive engineering optimization problems by making full use of the historical process data. The DTLBO algorithm is applied to solve a typical expensive engineering optimization problem–aerodynamic shape optimization design. The results demonstrate that the proposed DTLBO has the advantage of high efficiency, strong optimization ability, and non-parametric characteristic for expensive engineering problems.

To ensure that a student learns from good exemplars and to minimize the time wasted on poor directions, we allow the student to learn from the exemplars until the student ceases improving for a certain number of generations called the refreshing gap

Differences between the DTLBO algorithm and the original TLBO

1)Once the sensing distance is used to identify the neighboring members of each student, as exemplars to update the position, this mechanism utilizes the potentials of all students as exemplars to guide a student's new position.

(2)Instead of learning from the same exemplar students for all dimensions, each dimension of a student in general can learn from different students for different dimensions to update its position. In other words, each dimension of a student may learn from the corresponding dimension of different student based on the proposed equation .

(3)Finding the neighbor for different dimensions to update a student position is done randomly (with a vigil that repetitions are avoided). This improves the thorough exploration capability of the original TLBO with large possibility to avoid premature convergence in complex optimization problems.

The original TLBO is very efficient and powerful, but highly prone to premature convergence, So to evade this and further improve exploration ability of TLBO **a differential guidance is used to tap useful information in all the students to update the position of a particular student.**

Below Equation expresses the differential mechanism

$$Z_i - Z_j = (z_{i1}\ z_{i2}\ z_{i3}\ \cdots\ z_{in}) - (z_{\rho 1}\ z_{\rho 2}\ z_{\rho 3}\ \cdots\ z_{\rho n})$$

where

z_{i1} is the first element in the n dimension vector Z_i;
z_{in} is the nth element in the n dimension vector Z_i;
$z_{\rho 1}$ is the first element in the n dimension vector Z_p;
ρ is the random integer generated separately for each z, from 1 to n, but $\rho \neq i$.

The Pseudocode of the Proposed Refined TLBO Algorithm

(1) Initialize the number of students (population), range of design variables, iteration count, and termination criterion.
(2)Randomly generate the students using the design variables.
(3)Evaluate the fitness function using the generated (new) students.*//Teacher Phase//*
(4)Calculate the mean of each design variable in the problem.
(5)Identify the best solution as teacher amongst the students based on their fitness value. Use differential operator scheme to fine-tune the teacher.
(6)Modify all other students with reference to the mean of the teacher identified in step 4. *//Learner Phase//*
(7)Evaluate the fitness function using the modified students in step 6.
(8)Randomly select any two students and compare their fitness. Modify the student whose fitness value is better than the other and use again the differential operator scheme. Reject the unfit student.
(9)Replace the student fitness and its corresponding design variable.
(10)Repeat (test equal to the number of students) step 8, until all the students participate in the test, ensuring that no two students (pair) repeat the test.
(11)Ensure that the final modified students strength equals the original strength, ensuring there is no duplication of the candidates.
(12)Check for termination criterion and repeat from step 4.

```
Step1: Randomly initialize Class of N students Xi ( i=1, 2, …, n)
Step2: Compute fitness value of all the students
Step3: For Iter in range(max_iter):  # loop max_iter times
          For i in range(N):  # for each student
               # Teaching phase-----------
                  Xteacher = student with least function or fitness value
                  Xmean = mean of all the students
                  TF ( teaching factor) = either 1 or 2 ( randomly chosen )
                  Xnew = class[i].position + r*(Xteacher - TF*Xmean)

                  # if Xnew < minx OR Xnew > maxx then clip it
                  Xnew = min(Xnew, minx)
                  Xnew = max(Xnew, maxx)

                  # compute fitness of new solution
                  fnew = fitness(Xnew)

                  # greedy selection strategy
                  if(fnew < class[i].fitness)
                     class[i].position = Xnew
                     class[i].fitness = fnew
```

```
# Learning phase------------
              Xpartner = randomly chosen student from class

              if(class[i].fitness < Xpartner.fitness):
                 Xnew = class[i].position + r*(class[i].position - Xpartner)
              else
                 Xnew = class[i].position - r*(class[i].position - Xpartner)

                # if Xnew < minx OR Xnew > maxx then clip it
                Xnew = min(Xnew, minx)
                Xnew = max(Xnew, maxx)

                # compute fitness of new solution
                fnew = fitness(Xnew)

                # greedy selection strategy
                if(fnew < class[i].fitness)
                   class[i].position = Xnew
                   class[i].fitness = fnew
         End-for
      End -for
Step 4: Return best student from class
```

Benchmark function: Unconstrained Himmelblau function

$$\min f(x_i) = \left(x_1^2 + x_2 - 11\right)^2 + \left(x_1 + x_2^2 - 7\right)^2$$

subject to

$-5 \le x_1, x_2 \le 5.$

known solution to this benchmark function is 0 for $x_1 = 3$ and $x_2 = 2$.

Assuming

random numbers $r_1 = 0.25$ for x_1 and

$r_2 = 0.43$ for x_2, and

$T_f = 1$,

the *difference_mean* values for x_1 and x_2 are calculated as,

Difference_Mean j,k,i = ri(Xj,kbest,i - TFMj,i)

(2 & 5)-new values of x1 and x2 for learner 2 are calculated as,

(x_1)new for learner 2 =0.191 + 0.47(2.49575-0.191) = 1.27423

(x_2)new for learner 2 = 2.289 + 0.33(0.31058-2.289) = 1.63612

DTLBO:

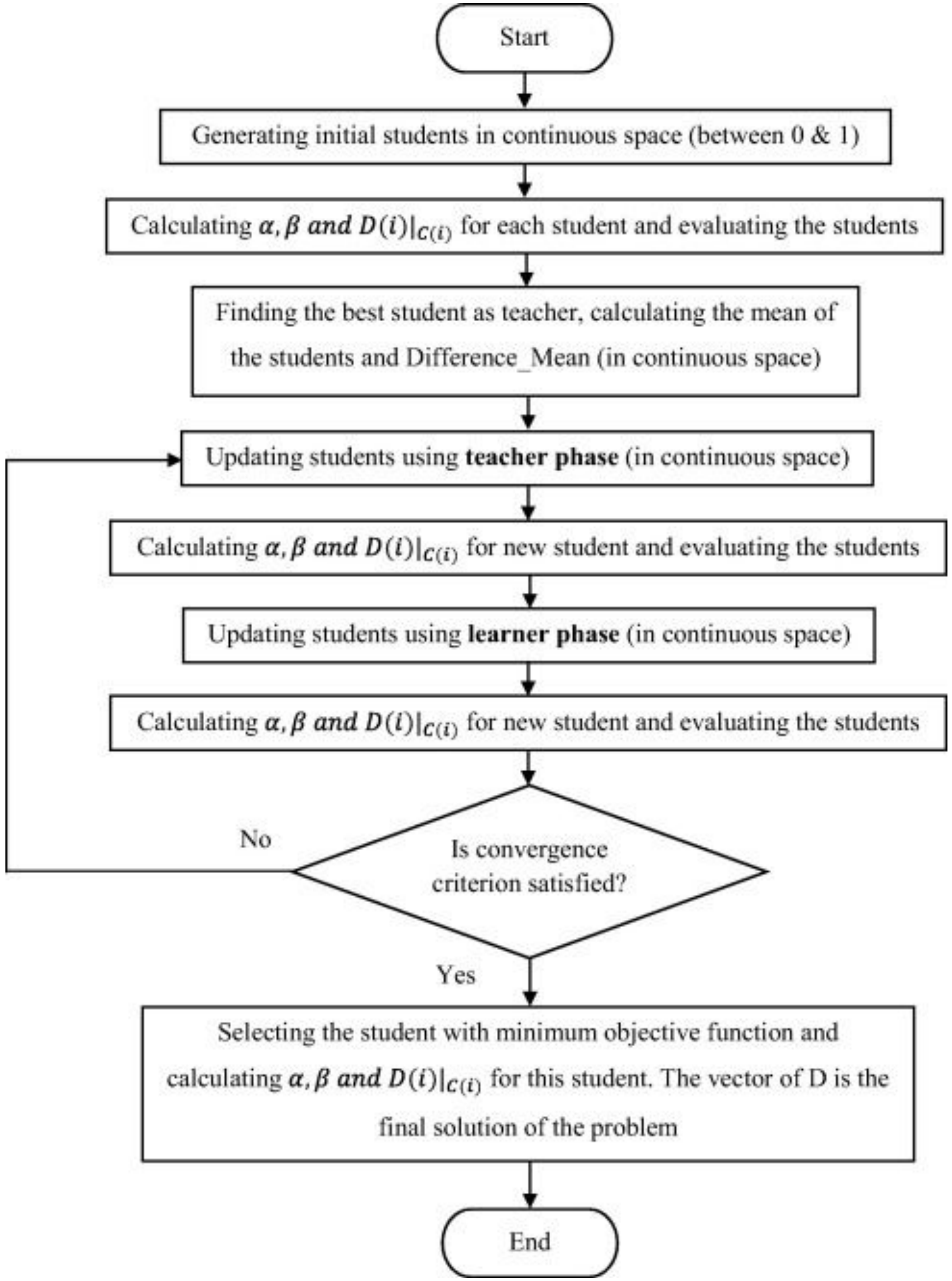
To make TLBO adapt to the DSR problem, a discrete Teaching–Learning-Based Optimization (DTLBO) algorithm is proposed. DTLBO is fast, converge to global optimal solution and is easily adapted for different distribution systems.

Flow chart of DTLBO Algorithm

$$\alpha = 1 + N \times C$$

$$\beta = \min(\lfloor \alpha \rfloor, N)$$

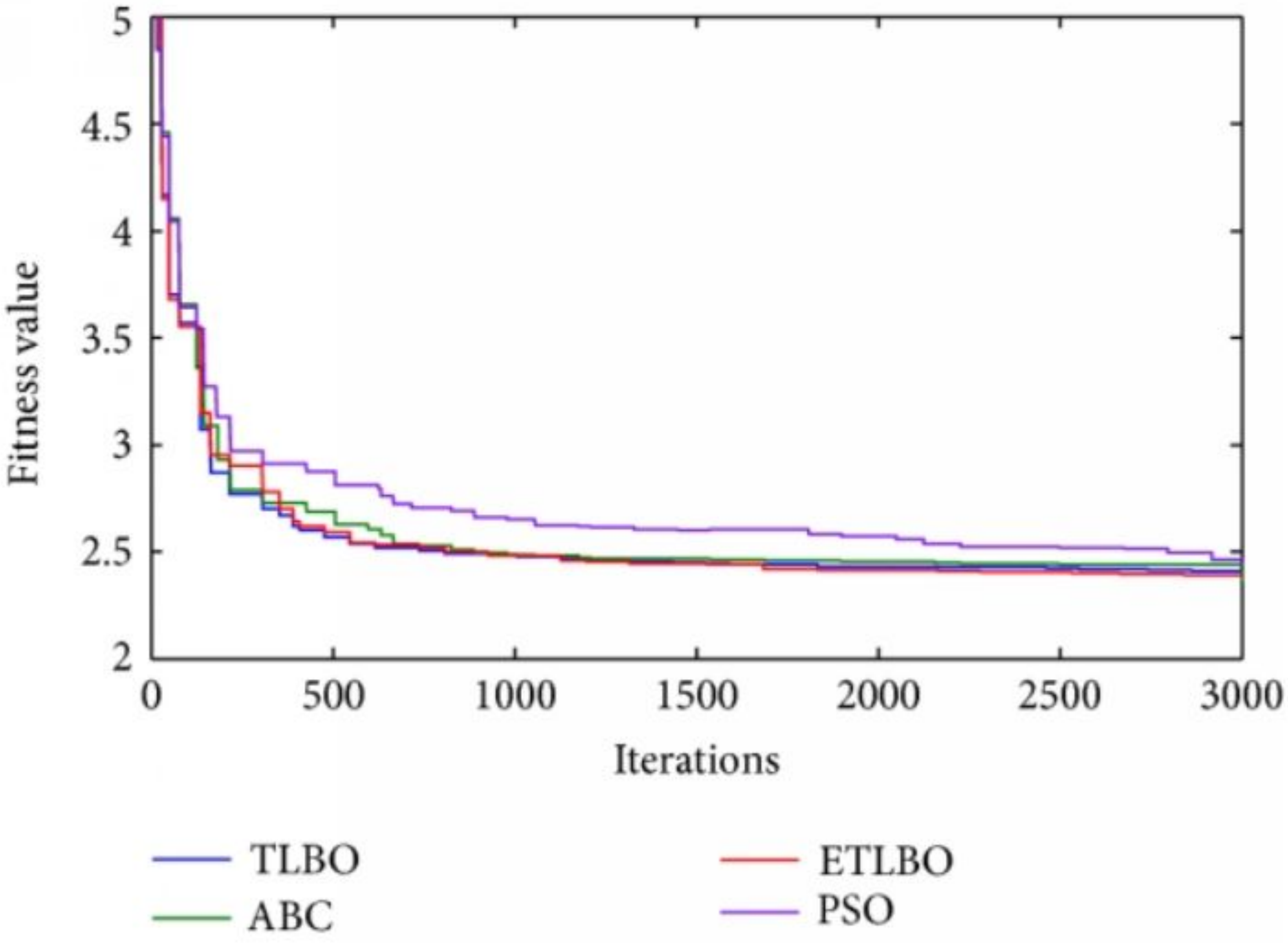
$$D(i)|_{C(i)} = K_{\beta}$$



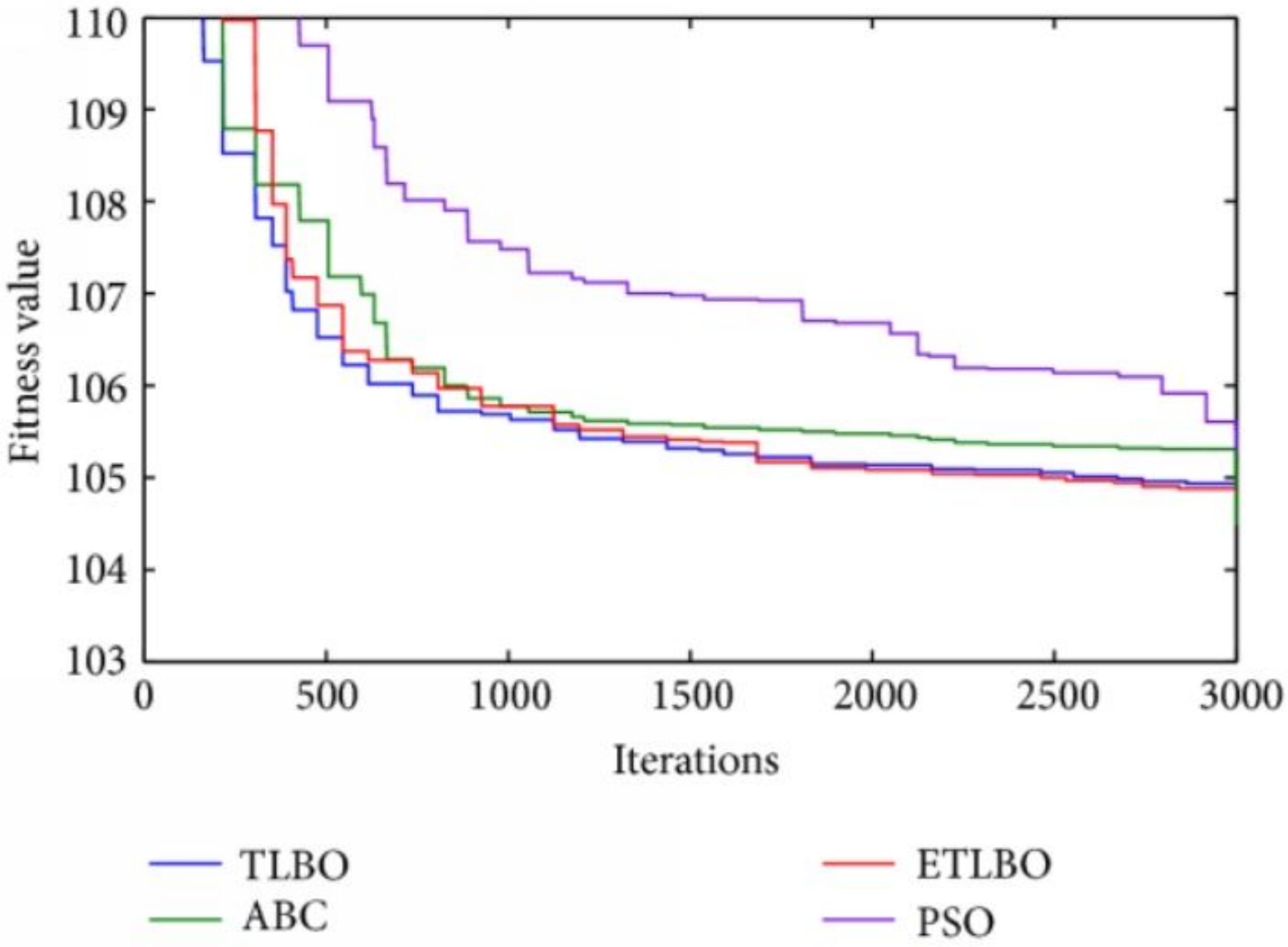
Comparison of the results obtained by GA with the published results (Case 2).

Optimal values	Results obtained by GA	Published result
Outer dia hallow shaft, cm	11.0928360	10.9000
Ratio of inner dia to outer dia	0.9699000	0.9685
Weight of hallow shaft, kg	2.3704290	2.4017

Convergence plot of the various methods for Case 2.



Convergence plot of the various methods for Case 3.



Best, worst, and mean production cost produced by the various methods for Case 2.

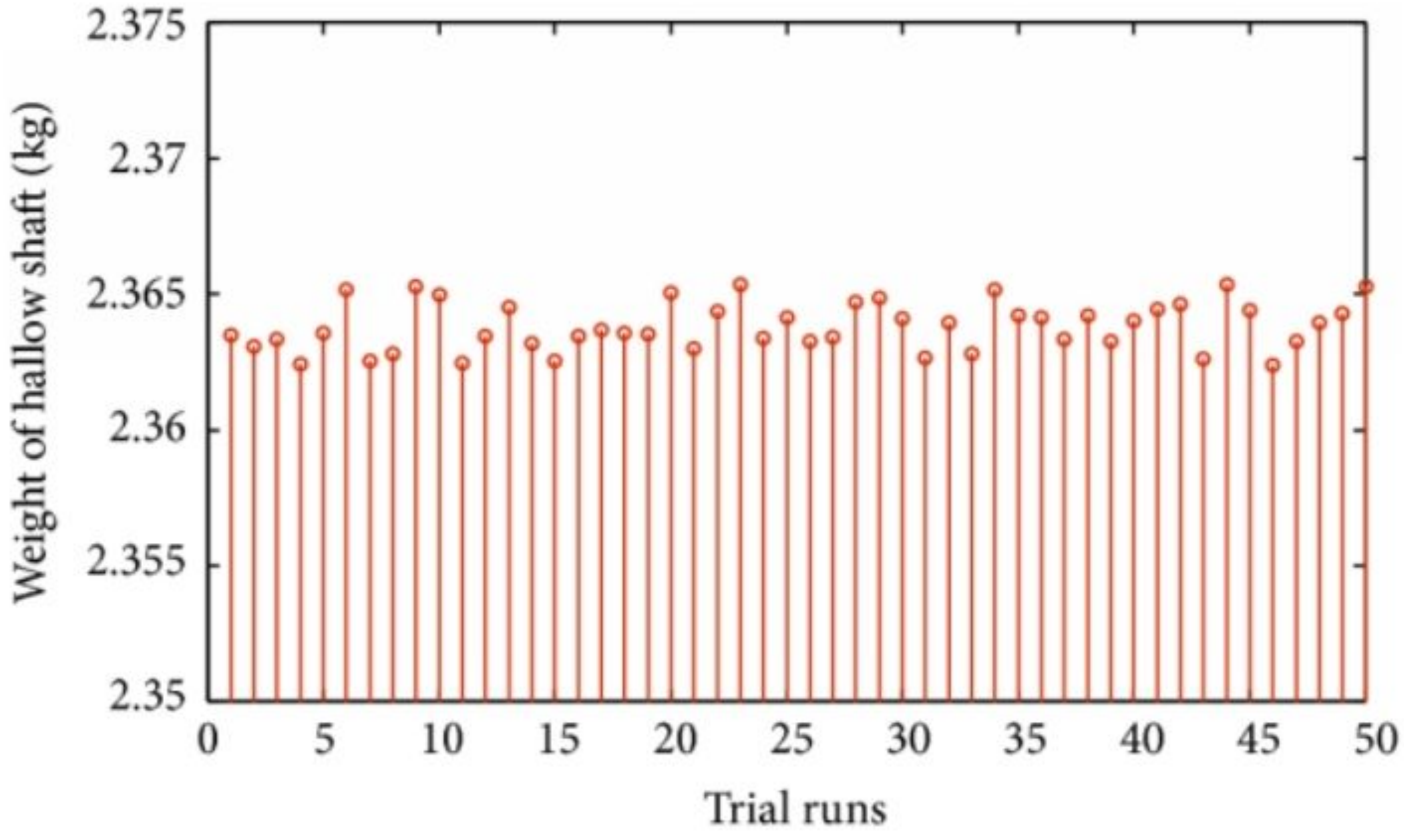
Method	Maximum	Minimum	Mean	Average time (min)	Minimum time (min)
Conventional	NA	46.5392	NA	NA	NA
GA	46.6932	46.6653	46.6821	3.2	3
PSO	46.6752	46.5212	46.6254	1.8	1.7
ABS	46.6241	46.5115	46.6033	2.5	2.3
TLBO	46.5214	46.3221	46.4998	2.2	2
DTLBO	46.4322	46.3012	46.3192	2.4	2.2

Comparison of the results obtained by GA with the published results (Case 3).

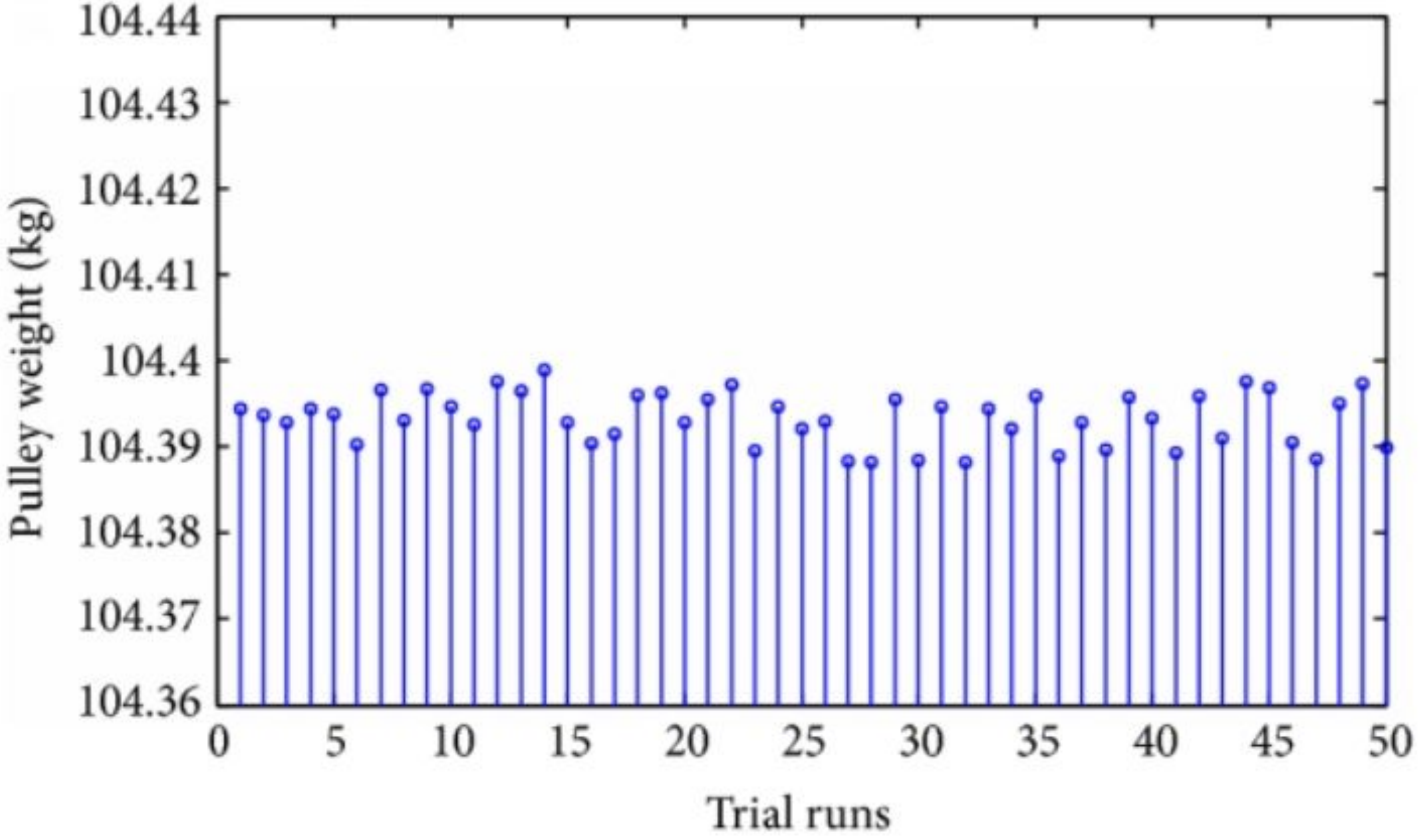
Optimal values	Results obtained by GA	Published result
Pulley dia (d_1), cm	20.957056	21.12
Pulley dia (d_2), cm	72.906562	73.25
Pulley dia (${d^1}_1$), cm	42.370429	42.25
Pulley dia (${d^1}_2$), cm	36.453281	36.60
Pulley width (b), cm	05.239177	05.21
Pulley weight, kg	104.533508	105120

Final cost of the optimization obtained for all test cases using DTLBO method

Case 2



CASE 3



104.44
104.43
104.42
104.41
104.4
104.39
104.38
104.37
104.36
Pulley weight (kg)
0
5
10
15
20
25
30
35
40
45
50
Trial runs