Korean J. Chem. Eng., 34(3), 628-641 (2017)
DOI: 10.1007/s11814-016-0317-x

pISSN: 0256-1115
eISSN: 1975-7220

A comparative study of teaching-learning-self-study algorithms on benchmark function optimization

Hyun-Jun Cho*, Faisal Ahmed, Tae Young Kim*, Beom Seok Kim*, and Yeong-Koo Yeo*,†**

*Department of Chemical Engineering, Hanyang University, Seoul 04763, Korea
**Department of Chemical Engineering, COMSATS Institute of Information Technology, Lahore, Pakistan
(*Received 8 June 2016 • accepted 10 November 2016*)

Abstract−In typical optimization problems, the number of design variables may be large and their influence on the specific objective function can be complicated; the objective function may have some local optima while most chemical engineers are interested only in the global optimum. For any new optimization algorithms, it is essential to validate their performance, compare with other existing algorithms and check whether they provide the global optimum solutions, which can be done effectively by solving benchmark problems. In this work, seven typical optimization algorithms including the newly proposed TLBO (Teaching-learning-based optimization) based algorithms such as the TLSO (Teaching-learning-self-study optimization) algorithm have been reviewed and tested by using a set of 20 benchmark functions for unconstrained optimization problems to validate the performance and to assess these optimization algorithms. It was found that the TLSO algorithm shows the fastest convergence speed to the optimum and outperforms other algorithms for most test functions.

Keywords: Optimization, Teaching-learning-self-study, Benchmark Function, Teaching-learning-based Optimization, Comparative Study

INTRODUCTION

The main objective of chemical process optimization is to minimize operating costs or to maximize profit while satisfying the prevailing constraints. In the operation of chemical plants, the use of optimization techniques in scheduling, control, production and transportation has resulted in substantial savings for an extensive range of the chemical industry. In the application of optimization method, effects of design variables on the system performance are analyzed and achievement of the optimality is assessed. But, in chemical process design problems, the number of design variables may be very large, and their influence on the specific objective function can be very complicated. Moreover, the objective function may have some local optima, whereas the engineer is interested only in the global optimum. These problems cannot be handled effectively by classical optimization methods because they search only local optima. To overcome these difficulties, many researchers have proposed efficient optimization techniques, and continuous research activity is being conducted in this area.

The genetic algorithm (GA), which has been widely used for various chemical engineering applications, is based on the principle of the Darwinian theory of evolution of living beings and of the survival of the fittest [1]. The differential evolution (DE) method, which is similar to GA with specialized crossover and selection method, was proposed by Storn and Price [2]. As another efficient evolutionary optimization algorithm, Runarsson and Yao developed the evolution strategy (ES) algorithm [3]. ES is based on the hypothesis that the laws of heredity are developed for fastest phylogenetic adaptation during biological evolution. In 1986, Farmer [4] proposed the artificial immune algorithm (AIA), which works on the immune system of human beings. Several algorithms based on swarm intelligence have been developed. Bacteria foraging optimization (BFO) [5], is based on the social foraging behavior of Escherichia coli. Particle swarm optimization (PSO) [6] is based on the principle of foraging behavior of the swarm of birds. The ant colony optimization (ACO) algorithm works on the principle of foraging behavior of ants for the food [7,8], and the artificial bee colony (ABC) scheme is based on the principle of foraging behavior of a honey bee [9,10]. In addition, there are many other algorithms based on the principles of various natural phenomena [11,12]. The optimization algorithms based on evolutionary and swarm intelligence are probabilistic, and knowledge about the parameters such as population size and number of generations is required. GA, one of the most popular evolutionary algorithm, requires mutation and crossover rates. For PSO, controlling of parameters, such as inertia weight, social and cognitive parameters, is required. These parameters affect significantly the performance of the related algorithms, and proper tuning of these parameters is crucial.

Recently the teaching-learning-based optimization (TLBO) algorithm was proposed to obtain global solutions for nonlinear functions with high consistency and less computational effort [13]. In this algorithm, any algorithm-specific parameters are not necessary and only basic parameters such as population size and number of generations are required. The TLBO algorithm is based on the effect of the influence of a teacher on the output of learners in a class. In this case, the output is considered in terms of results or grades. The

†To whom correspondence should be addressed.
E-mail: ykyeo@hanyang.ac.kr

teacher is generally considered as a highly learned person who shares knowledge with the learners [13]. Many algorithms have been proposed to improve the solution quality and to enhance the convergence property of the TLBO [14-16]. The modified TLBO (mTLBO) introduces changes to the learner phase of the TLBO algorithm to improve the solution quality [17]. In the orthogonal TLBO (oTLBO), the orthogonal design method used to solve multi-factor and multi-level problems is incorporated into the TLBO [18]. It is reported that the self-adapting control parameter can be applied to the DE method to yield the value of the most suitable variable [19]. The GA was combined with the TLBO to solve optimal control problems [20]. An ameliorated TLBO (ATLBO) was proposed to improve the solution quality and to quicken the convergence speed of the TLBO algorithm [21]. A novel teaching-learning-self-study-optimization (TLSO) was proposed by incorporating the natural phenomenon of a teacher's focus on bad learners into the TLBO, and by proposing once again the natural phenomenon of self-learning [22]. Integration of the first phenomenon mentioned above into the TLBO focuses the intensification, which causes acceleration of convergence compared to the TLBO. The TLSO keeps the inertia weight and acceleration coefficient in the teacher phase into the TLBO. The TLSO proposes the self-study concept in the learner phase to keep a trade-off between intensification and diversification.

For any new optimization algorithms, it is essential to validate their performance and compare with other existing algorithms. Benchmark test functions can be effectively used to assess the performance of optimization algorithms. Using these functions, various optimization algorithms may be validated and compared with each other to assess the performance of each optimization method. In this work, we review and test seven typical optimization algorithms, and compare the performance of the TLSO algorithm newly developed by one of the authors [22] red with other well-known optimization algorithms, including TLBO, ATLBO, ACO and PSO. A set of 20 benchmark functions has been selected and used for unconstrained optimization problems in the validation and comparison of the optimization algorithms.

REVIEW OF OPTIMIZATION ALGORITHMS

1. The Genetic Algorithm GA

GA is a direct and stochastic method to find a global optimum. It is based on the principle of the Darwinian theory of the evolution of living beings and of the survival of the fittest. GA is known to be adequate to solve complicated optimization problems, including adaptive control, recognition modeling, and optimal control problems. In GA, there are no definite rules, and probabilistic operators are commonly used. One of the significant characteristics of the genetic algorithm is the use of stochastic information in its implementation, and therefore this method may be considered as a global optimization method. A basic genetic algorithm can be summarized as follows [23]:

1) Set a set of solutions or chromosomes (population) P(0). Evaluate the initial solution for fitness. Set t=0.

2) Generate a set of children (crossover, mutation) using the genetic operators. Add a new set of randomly generated population and evaluate again the population fitness.

3) Determine which members will be part of next generation (competitive selection). Select population P(t+1).

4) If the convergence is not achieved, set t=t+1 and go to step 2).

Genetic algorithm starts with identification of the parameters of a given estimator as chromosomes. This is tracked by populating

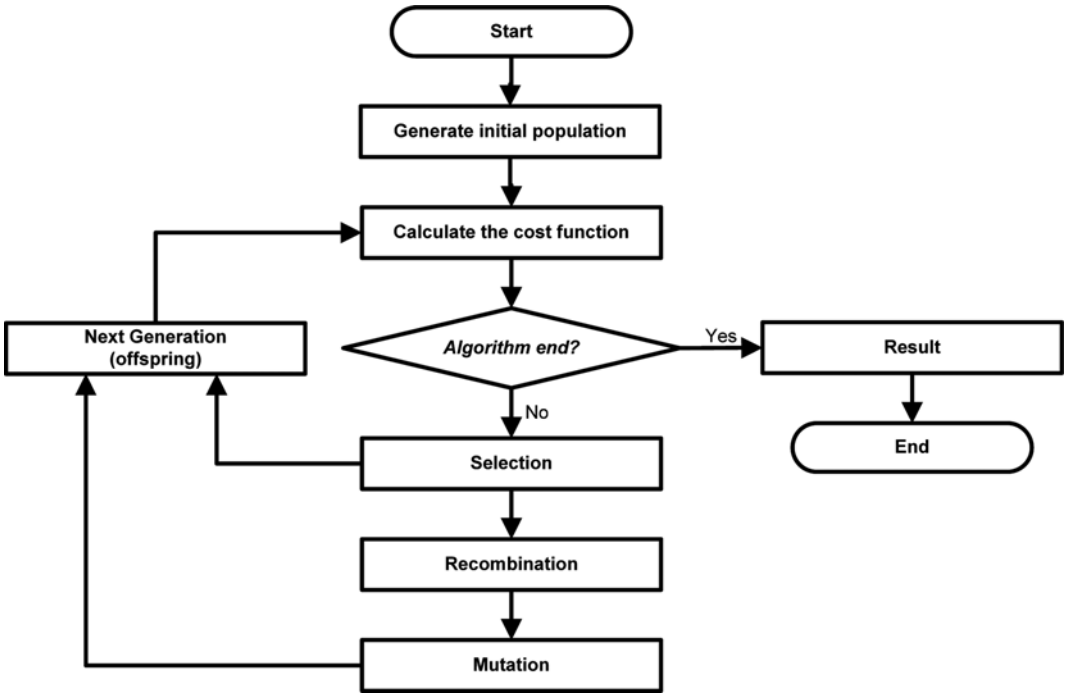

Fig. 1. Flowchart of genetic algorithm.

possible solutions. Effectiveness of each chromosome is evaluated by a fitness function. The better parent solutions are reproduced and the next generation of solutions (children) are created by applying the genetic operators (called crossover and mutation). The children solutions are evaluated and the whole cycle repeats until the best solution is reached [24]. Fig. 1 shows a step-by-step procedure of GA.

2. TLBO

The TLBO algorithm is a meta-heuristic optimization method that uses a population of solutions to proceed to the global solution. The TLBO algorithm does not require any algorithm-specific parameters and requires only common controlling parameters like population size and number of generations for its working [25]. The TLBO employs the influence of a teacher on the output of learners in a class. The class of learners in the TLBO may be considered as the population in the GA or PSO. The TLBO consists of two vital modes: teacher phase and leaner phase. In the teacher phase mode, the teacher teaches and influences all students in a classroom. In the learner phase mode, less knowledgeable students are influenced by more knowledgeable students and learn from them. In the teacher phase, learners learn from a teacher, and the teacher is considered as the most knowledgeable person in the society. The best learner in the class can be considered as a teacher:

$$T_i = X_{min f(x)} \tag{1}$$

where T_i is the mark of the teacher and $X_{min f(x)}$ denotes a student with the highest learning level. The teacher makes efforts to improve the learning level or ability of the group. The learners increase their knowledge and update their marks, and the newly obtained marks primarily depend upon the old mark and the difference between T_i and the mean. The learners learn and update their knowledge according to the following relation [21]:

$$X_{new,i} = X_{old,i} + r_i(X_i - T_F M_i) \tag{2}$$

where M_i is the mean of marks obtained by learners and r_i is a random number between 0 and 1, which decides the step size of the difference. The teaching factor, T_F, controls the movement of the mean value. The value of T_F is either 1 or 2, which is decided randomly as follows:

$$T_F = \text{round}[1 + \text{rand}(0, 1)(2-1)] \tag{3}$$

In the learner phase, the learners can update their marks and enhance their fitness values by interacting with other learners who have more knowledge than they in the class. A learner can communicate randomly with other learners to improve his/her knowledge with the help of group discussions, formal communications and presentations [21]. For a learner X_i, another learner $X_j (i \neq j)$ is randomly selected and X_i updates his/her marks according to the random learner [21,22]:

$$X_{new,i} = \begin{cases} X_{old,i} + r_i(X_i - X_j) & \text{if } f(X_i) < f(X_j) \\ X_{old,i} + r_i(X_j - X_i) & \text{if } f(X_i) \geq f(X_j) \end{cases} \tag{4}$$

$X_{new,i}$ is accepted if it gives better fitness value.

3. ATLBO

ATLBO algorithm was proposed to improve the search accuracy and to quicken the convergence speed of the TLBO algorithm. The ATLBO algorithm incorporates some additional parameters such as the inertia weight and acceleration coefficients to quicken the teaching and the learning processes. In the teacher phase, the new mark $X_{new,i}$ is modified as follows [21]:

$$X_{new,i} = \omega_i X_{old,i} + \Phi_i(X_i - T_F M_i) \tag{5}$$

where ω_i is the inertia weight and Φ_i is the acceleration coefficient which determines the maximum step size. These parameters are defined as follows:

$$\omega_i = \frac{1}{1 + \exp\left(-\frac{\text{fit(i)}}{\text{ap}}\right)^{iter}} \tag{6}$$

$$\Phi_i = \frac{1}{1 + \exp\left(-\frac{\text{fit(i)}}{\text{ap}}\right) \times \text{iter}} \tag{7}$$

where fit(i) is the fitness of the i^{th} learner, ap is the maximum fitness values among learners in the 1^{st} iteration, and iter denotes the current iteration. In the learner phase, a learner updates his/her mark by the following relation:

$$X_{new,i} = \begin{cases} X_{old,i} + \phi_i(X_j - X_i) & \text{if } f(X_i) \leq f(X_j) \\ X_{old,i} + \psi_i(X_{best} - X_i) & \text{if } f(X_i) > f(X_j) \end{cases} \tag{8}$$

$$\phi_i = 1 - \exp(\text{fit}(X_j) - \text{fit}(X_i)) \tag{9}$$

$$\psi_i = 1 - \exp(\text{fit}(X_{best}) - \text{fit}(X_i)) \tag{10}$$

where X_{best} represents the best learner in a class, and ϕ_i and ψ_i are acceleration coefficients that determine the step size depending on the difference between two learners [21].

4. ACO and PSO

The ACO algorithm draws its inspiration from the behavior of real ants as they move from their nest towards a food source [26]. The ACO is implemented as a collective group of intelligent agents that simulate the ants' behavior using mechanisms of cooperation and adaptation. The PSO algorithm, based on stochastic optimization scheme, is a heuristic search algorithm inspired by social behavior of bird flocking or fish schooling. The estimation is carried out by a population of potential solutions called particles [27]. The PSO is similar to GA in terms of population initialization with random solutions and searching for global optima, but the PSO does not undergo crossover and mutation. In the PSO, the particles move through the problem space following the current optimum particles. The particle associated with the best fitness value seems to be the leader and each particle keeps track of its coordinates in the problem space [28].

5. TLSO

To improve the average level of a class, a teacher must put more effort to a bad learner. But, in TLBO, a teacher's effort is randomly distributed to all learners and he/she offers a random amount of learning to both good and bad learners. In the learner phase, apart from learning from each other, the learners can increase their knowledge by self-learning. The TLSO algorithm incorporates the teacher's focus on bad learners and the phenomenon of self-learn-

ing into the TLBO. If the marks of a learner are too low, the teacher should have to put more effort on this learner than on the learner whose marks are higher. Hence, the learner with lower marks needs a big step, and the learner with higher marks needs a relatively small step towards the teacher. This concept can be integrated by introducing some parameters as follows:

$$X_{mod,i}=\Omega_i X_{old,i}+\Gamma_i(X_{teacher}-T_F M_i) \tag{11}$$

where Ω_i is the inertia weight that gives weight to the previous solution and Γ_i is the acceleration coefficient that accelerates the convergence by controlling the step size of the difference ($X_{teacher}-T_F M_i$). These parameters are defined as follows:

$$\Omega_i=\frac{1}{1+\exp\left(-\frac{\text{fit(i)}}{\text{fit(teacher)}}\right)^{iter}} \tag{12}$$

$$\Gamma_i=\frac{1}{1+\exp\left(-\frac{\text{fit(i)}}{\text{fit(teacher)}}\right)\times \text{iter}} \tag{13}$$

where fit(i) is the fitness value of the i^{th} learner and fit(teacher) the fitness value of the teacher. In TLBO, it is allowed in first iteration where learners get their marks randomly from the given min-max search space but start following the teacher and good learners in the subsequent iterations. This allowance in the first iteration is not enough, especially when teacher space is intensified, and needs to be implemented throughout the execution of all iterations to have significant contribution of diversification factor in the algorithm [22]. The diversification factor, D_m, which is the min-max search space, is defined as follows:

$$D_m=X_m^U-X_m^L \tag{14}$$

where X_m^U and X_m^L are the maximum and minimum marks of the m^{th} subject, respectively. To account for the self-study concept in the learner phase, the diversification factor can be incorporated as follows [22]:

$$\text{if rand} < 0.5 \quad X_{mod,i}=\begin{cases} X_{old,i}+r_i(X_i-X_j) & \text{if } f(X_i)<f(X_j) \\ X_{old,i}+r_i(X_j-X_i) & \text{if } f(X_i)\geq f(X_j) \end{cases} \tag{15}$$

$$\text{otherwise } X_{mod,i}=\begin{cases} X_{old,i}-r_i D_m & \text{if } r_i D_m<X_{old,i} \\ X_{old,i}+r_i D_m & \text{if } r_i D_m=X_{old,i} \\ r_i D_m-X_{old,i} & \text{if } r_i D_m>X_{old,i} \end{cases} \tag{16}$$

where rand is a uniformly distributed random number between 0 and 1. Fig. 2 shows a step-by-step procedure of TLSO.

6. Variant of TLSO

Variants of TLSO are similar to the TLSO with the exception of inclusion of some acceleration weights in the learner phase. The

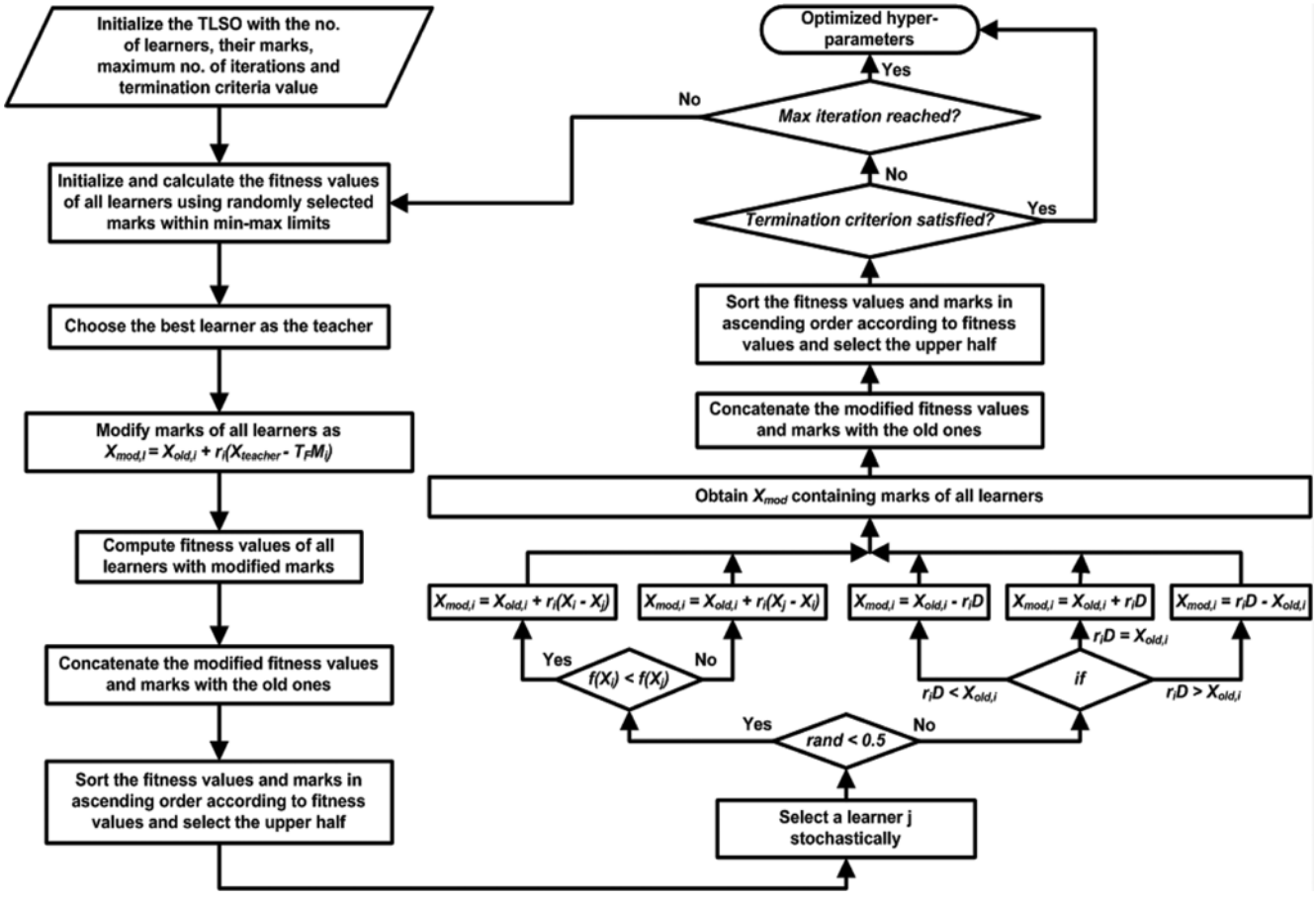

Fig. 2. Flowchart of TLSO algorithm.

acceleration weights control the step size according to the fitness values of learners. In the variant model, Φ_i and Ψ_i are used to control the step size according to the present efficiency of the learner and randomly selected learner j as follows [22]:

$$\Phi_i = 1 - \exp(\mathrm{fit}(X_j) - \mathrm{fit}(X_i)) \tag{17}$$

$$\Psi_i = 1 - \exp(\mathrm{fit}(X_{teacher}) - \mathrm{fit}(X_i)) \tag{18}$$

$$\text{if rand} < 0.5 \quad X_{mod,\,i} = \begin{cases} X_{old,\,i} + \Phi_i(X_i - X_j) & \text{if} \quad f(X_i) \le f(X_j) \\ X_{old,\,i} + \Psi_i(X_{teacher} - X_j) & \text{if} \quad f(X_i) > f(X_j) \end{cases} \tag{19}$$

$$\text{otherwise } X_{mod,\,i} = \begin{cases} X_{old,\,i} - r_i D_m & \text{if} \quad r_i D_m < X_{old,\,i} \\ X_{old,\,i} + r_i D_m & \text{if} \quad r_i D_m = X_{old,\,i} \\ r_i D_m - X_{old,\,i} & \text{if} \quad r_i D_m > X_{old,\,i} \end{cases} \tag{20}$$

EXPERIMENTAL STUDY ON UNCONSTRAINED BENCHMARK FUNCTIONS

1. Benchmark Functions

For a new optimization algorithm, it is essential to validate its performance and compare with other existing algorithms over a

Table 1. 20 Benchmark functions used in experiment

No.	Function name	Function formulation	Dimension	Search range
1	Brad	$f(x)=\sum_{i=1}^{15}\left[y_i - x_1 - i \over (16-i)x_2 + \omega_i x_3\right]^2$	3	$-0.25 \le x_1 \le 0.25$ $0.01 \le x_2$ $x_3 \le 2.5$
2	Beale	$f(x)=(1.50 - x_1 + x_1x_2)^2 + (2.25 - x_1 + x_1x_2^2)^2 + (2.625 - x_1 + x_2x_2^3)^2$	2	$-4.5 \le x_i \le 4.5$
3	Brown	$f(x)=\sum_{i=1}^{n-1}(x_i^2)(x_{i+1}^2+1) + (x_{i+1}^2)(x_i^2+1)$	30	$-1 \le x_i \le 4$
4	Chen Bird	$f(x)=-\frac{0.001}{(0.001)^2 + (x_1 - 0.4x_2 - 0.1)^2} - \frac{0.001}{(0.001)^2 + (2x_1 + x_2 - 1.5)^2}$	2	$-500 \le x_i \le 500$
5	Chen V	$f(x)=-\frac{0.001}{(0.001)^2 + (x_1^2 + x_2^2 - 1)^2} - \frac{0.001}{(0.001)^2 + (x_1^2 + x_2^2 - 0.5)^2} - \frac{0.001}{(0.001)^2 + (x_1^2 - x_2^2)^2}$	2	$-500 \le x_i \le 500$
6	Chichinadze	$f(x)=x_1^2 - 12x_1 + 11 + 10\cos\left(\frac{1}{2}\pi x_1\right) + 8\sin\left(\frac{5}{2}\pi x_2\right) - \left(\frac{1}{5}\right)^{0.2}\exp(0.5 - (x_2 - 0.5)^2)$	2	$-30 \le x_i \le 30$
7	Csendes	$f(x)=\sum_{i=1}^{D} x_i^6\left(2 + \sin\frac{1}{x_i}\right)$	30	$-1 \le x_i \le 1$
8	Cube	$f(x)=100(x_2 - x_1^3)^2 + (1 - x_1)^2$	2	$-10 \le x_i \le 10$
9	Himmelblau	$f(x)=(x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2$	2	$-5 \le x_i \le 5$
10	Mishra 5	$f(x)=[\sin^2(\cos(x_1) + \cos(x_2))^2 + \cos^2(\sin(x_1) + \sin(x_2))^2 + x_1]^2 + 0.01x_1 + 0.1x_2$	2	$-10 \le x_i \le 10$
11	Powell Singular	$f(x)=\sum_{i=1}^{D/4}(x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + (x_{4i-2} - x_{4i-1})^4 + 10(x_{4i-3} - x_{4i})^4$	24	$-4 \le x_i \le 5$
12	Powell Singular 2	$f(x)=\sum_{i=2}^{D-2}(x_{4i-1} + 10x_i)^2 + 5(x_{i+1} - x_{i+2})^2 + (x_i - 2x_{i+1})^4 + 10(x_{i-1} - x_{i+2})^4$	30	$-4 \le x_i \le 5$
13	Price 4	$f(x)=(2x_1x_2 - x_2^3)^2 + (6x_1 - x_2^2 + x_2)^2$	2	$-500 \le x_i \le 500$
14	Schwefel 1.2	$f(x)=\sum_{i=1}^{D}\left(\sum_{j=1}^{i} x_j\right)^2$	30	$-100 \le x_i \le 100$
15	Schwefel 2.22	$f(x)=\sum_{i=1}^{D}\lvert x_i\rvert + \prod_{i=1}^{n}\lvert x_i\rvert$	30	$-100 \le x_i \le 100$
16	Shubert	$f(x)=\prod_{i=1}^{n}\left(\sum_{i=1}^{5}\cos((j+1)x_i + j)\right)$	2	$-10 \le x_i \le 10$
17	Step	$f(x)=\sum_{i=1}^{D}(\lvert x_i\rvert)$	30	$-100 \le x_i \le 100$
18	Carrom Table	$f(x)=-\frac{1}{30}\exp\left(2\left\lvert 1 - \frac{\sqrt{x_1^2 + x_2^2}}{\pi}\right\rvert\right)\cos^2(x_1)\cos^2(x_2)$	2	$-10 \le x_i \le 10$
19	Xin-She Yang	$f(x)=\sum_{i=1}^{D}\varepsilon_i\lvert x_i\rvert^i$	30	$-5 \le x_i \le 5$
20	Zakharov	$f(x)=\sum_{i=1}^{n}x_i^2 + \left(\frac{1}{2}\sum_{i=1}^{n}ix_i\right)^2 + \left(\frac{1}{2}\sum_{i=1}^{n}ix_i\right)^4$	30	$-5 \le x_i \le 10$

good set of test functions [29]. To test reliability, efficiency and validation of optimization algorithms, a set of common standard benchmark functions can be used. Note that the performance or effectiveness of an optimization algorithm against others may not be assessed by the problems that it solves if the set of problems is very specialized and without diverse properties such as unimodality, multimodality, regularity, and non-regularity. Test benchmark functions can be considered as artificial problems, and can be used to evaluate the performance of an optimization algorithm. These functions may include single global optimum, or multiple global optima in the presence of multiple local optima. Thus, these functions can be easily manipulated and modified to test the underlying optimization algorithms in various scenarios.

In this study, we focused on 20 test benchmark functions (see Table 1) selected from the collection of Jamil and Yang [29]. These functions include at least one optimum for seven optimization algorithms considered here. Among 20 benchmark functions, 8 functions (f2, f3, f8, f11, f12, f14, f15, f17) exhibit unimodality, and the remaining 12 functions exhibit multimodality. Functions with multimodality have many local optima and may be classified as a difficult class of problems for most optimization algorithms. Fig. 3 shows a plot of a unimodal function #2 (Beale), and as an example of a multimodal function, Fig. 4 depicts a plot of a multimodal function #16 (Shubert).

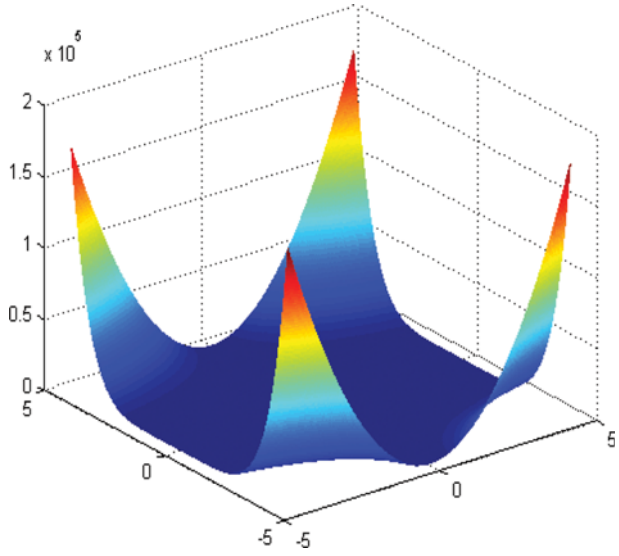

Fig. 3. Example of unimodal function graph: #2 Beale.

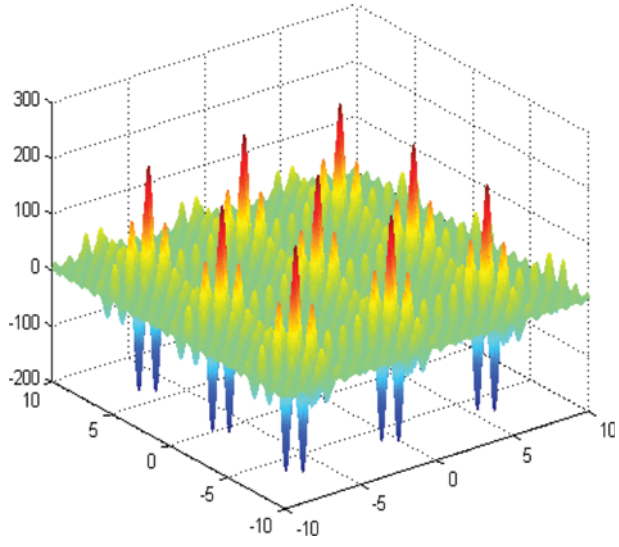

Fig. 4. Example of multimodal function graph: #16 Shubert.

2. Experiment Settings

The 20 benchmark functions were selected such that they have different characteristics such as unimodality or multimodality, separability or non-separability, and regularity or non-regularity. In this study, seven optimization algorithms (GA, TLBO, ATLBO, ACO, PSO, Variant, TLSO) were implemented on the selected 20 unconstrained benchmark functions. Table 1 shows the details of these functions. In experiments, it is required to set key parameters such as the population size, number of iterations, number of function evaluations, and dimension and search range of each function. A drawback of the algorithms having algorithm-specific parameters, such as ACO and PSO in this study, is that they need an extensive experimentation for the selection of the algorithm-specific parameters for each objective function prior to actual optimization. These parameters have substantial effects on the convergence accuracy and convergence speed. Therefore, the selection of suitable algorithm-specific parameters is essential for such algorithms. Extensive experimentation has been carried out to obtain the algorithm-specific parameters of ACO and PSO to get the best result for each benchmark function separately. In contrast, GA, TLBO, ATLBO, Variant and TLSO have the advantage of not having such algorithm-specific parameters, which saves much time consumed in algorithm-specific parameter selection [22].

Values of the population size or the class size affect the optimization calculations as well as optimization time through learning or communication. To identify the effect of population size for GA (the swarm size for ACO and PSO, the class size for TLBO-based algorithms), the algorithms are experimented with four different population sizes (25, 50, 75, 100). For all optimization algorithms, 30 optimization computations were performed to give average values and standard deviations. In each run, the maximum number of iterations was set as 500,000 for all functions regardless of dimensions for fair comparison purpose.

RESULTS AND DISCUSSION

1. Results by the TLSO

In the first experiment, the TLSO algorithm was implemented on 20 unconstrained benchmark functions. The results of each benchmark function are shown in Table 2 in the form of best solution, worst solution, average (mean) solution, and standard deviation.

In Table 2, all functions converged regardless of the population sizes. The population size does not affect the best value, while the standard deviation and computing time depend on it. It seems that large population size quickens the computation speed.

2. Comparative Results

The results obtained using the TLSO algorithm were compared with the results generated by other algorithms (GA, TLBO, ATLBO,

Table 2. TLSO results over 30 independent runs

No.	Function	Population	Optimum	Best	Worst	Mean	SD	Time
1	Brad	25	0.00821487	0.0082149	0.0099204	0.008406	0.0004207	12.521
		50		0.0082149	0.0082194	0.008215	8.299E-07	11.631
		75		0.0082149	0.0082149	0.0082149	9.712E-18	10.627
		100		0.0082149	0.0082149	0.0082149	5.931E-18	10.462
2	Beale	25	0	0	0	0	0	6.201
		50		0	0	0	0	4.365
		75		0	0	0	0	3.963
		100		0	0	0	0	3.791
3	Brown	25	0	0	0	0	0	9.239
		50		0	0	0	0	7.972
		75		0	0	0	0	7.373
		100		0	0	0	0	7.131
4	Chen Bird	25	-2000	−2000	−2000	−2000	0	6.036
		50		−2000	−1000	−1834.964	375.42752	4.158
		75		−2000	−1000	−1768.318	427.16722	3.776
		100		−2000	−1000	−1801.818	403.21491	3.632
5	Chen V	25	−2000	−2000	−1000	−1242.319	426.75619	6.229
		50		−2000	−1000	−1135.208	345.11905	4.208
		75		−2000	−1000	−1136.941	344.87055	3.793
		100		−2000	−1000	−1166.67	379.04925	3.656
6	Chichinadze	25	−42.94439	−42.94439	−42.49717	−42.92872	0.0815749	6.285
		50		−42.94439	−42.94439	−42.94439	3.592E-14	4.495
		75		−42.94439	−42.94439	−42.94439	3.613E-14	4.070
		100		−42.94439	−42.94439	−42.94439	3.613E-14	3.896
7	Csendes	25	0	0	0	0	0	11.070
		50		0	0	0	0	9.301
		75		0	0	0	0	8.158
		100		0	0	0	0	8.487
8	Cube	25	0	0	3.356E-26	1.119E-27	6.127E-27	6.398
		50		0	4.93E-30	1.643E-31	9.002E-31	4.577
		75		0	0	0	0	3.877
		100		0	0	0	0	3.738
9	Himmelblau	25	0	0	3.155E-30	2.63E-31	6.329E-31	6.439
		50		0	0	0	0	4.696
		75		0	7.889E-31	2.63E-32	1.44E-31	4.452
		100		0	7.889E-31	5.259E-32	2.001E-31	4.186
10	Mishra 5	25	−1.01983	−1.01983	−1.01003	−1.01885	0.0029906	6.264
		50		−1.01983	−1.01003	−1.01918	0.0024867	4.487
		75		−1.01983	−1.01983	−1.01983	6.78E-16	4.029
		100		−1.01983	−1.01983	−1.01983	6.78E-16	3.829
11	Powell Singular	25	0	0	0	0	0	8.086
		50		0	0	0	0	6.629
		75		0	0	0	0	6.212
		100		0	0	0	0	5.993

ACO, PSO, Variant) for the same number of function evaluations. Table 3 shows the comparative results of the optimization results of TLSO in the form of the best value. The numbers printed in bold font denote that the optimum was achieved.

From Table 3, the TLSO algorithm outperforms all the other algorithms (GA, TLBO, ATLBO, ACO, PSO, Variant): the TLSO

Table 2. Continued

No.	Function	Population	Optimum	Best	Worst	Mean	SD	Time
12	Powell Singular 2	25	0	0	0	0	0	10.865
		50		0	0	0	0	9.549
		75		0	0	0	0	9.093
		100		0	0	0	0	8.861
13	Price 4	25	0	0	0	0	0	6.767
		50		0	7.225E-87	2.408E-88	1.319E-87	4.973
		75		0	7.886E-82	2.629E-83	1.44E-82	4.647
		100		0	0	0	0	4.386
14	Schwefel 1.2	25	0	0	0	0	0	10.673
		50		0	0	0	0	9.321
		75		0	0	0	0	9.545
		100		0	0	0	0	8.528
15	Schwefel 2.22	25	0	0	0	0	0	7.405
		50		0	0	0	0	5.825
		75		0	0	0	0	5.547
		100		0	0	0	0	5.297
16	Shubert	25	−186.7309	−186.7309	−186.7308	−186.7309	2.473E-05	6.857
		50		−186.7309	−186.7309	−186.7309	3.58E-14	5.105
		75		−186.7309	−186.7309	−186.7309	4.75E-14	4.803
		100		−186.7309	−186.7309	−186.7309	4.352E-14	4.558
17	Step	25	0	0	0	0	0	7.405
		50		0	0	0	0	6.010
		75		0	0	0	0	5.627
		100		0	0	0	0	5.366
18	Table 3	25	−24.156816	−24.156816	−24.15609	−24.15679	0.0001315	6.713
		50		−24.156816	−24.156816	−24.156816	3.553E-15	4.959
		75		−24.156816	−24.156816	−24.156816	3.613E-15	4.672
		100		−24.156816	−24.156816	−24.156816	3.613E-15	4.462
19	Xin She Yang 1	25	0	0	0	0	0	9.088
		50		0	0	0	0	7.672
		75		0	0	0	0	7.267
		100		0	0	0	0	7.098
20	Zakharov	25	0	0	0	0	0	7.897
		50		0	0	0	0	6.799
		75		0	0	0	0	6.042
		100		0	0	0	0	5.727

converged for all 20 test functions and achieved an optimum in seven optimization tools. Only the PSO, which achieved optimum for the other optimization, matched the TLSO. But the PSO converged for 16 functions, followed by the Variant that converged for 15 functions.

Table 4 shows the comparative outcomes of the considered algorithms in the form of average solution (Mean) and standard deviation (SD). The numbers in bold font denote the best values. Small SD shows good performance because smaller SD value means closeness to the optimum.

From Table 4, the TLSO outperforms other algorithms for 17 functions followed by the TLBO, which exhibits the best results for 11 functions. While the TLSO and the PSO show similar performance in terms of the best value, the TLBO-based algorithms show better results in terms of average solution and standard deviation. Results of tests for all algorithms using 20 benchmark functions are summarized in Table 5. It shows the number of functions that a specific algorithm generates the best results in the form of the best value, average solution (Mean) and standard deviation (SD).

As can be seen from Table 5, the TLSO outperforms other algorithms in every aspect of comparison criteria. The TLSO outperformed other considered algorithms by giving better results for 17 functions in terms of mean solution and standard deviation, and 20 functions in terms of the best value out of 20 benchmark func-

Table 3. Comparative the best value of TLSO with other algorithms

No.	Opt. Val	GA	TLBO	ATLBO	ACO	PSO	Variant	TLSO
1	0.0082149	1.079437	**0.0082149**	0.0083759	0.0093381	**0.0082149**	0.0082337	**0.0082149**
2	0	3.12E-12	**0**	0.0004012	0.0001552	**0**	1.756E-06	**0**
3	0	1.33E-09	**0**	**0**	3.059E-08	0.0218808	**0**	**0**
4	−2000	**−2000**	−1000.0054	−1000.0062	−999.91318	**−2000**	−1999.9995	**−2000**
5	−2000	**−2000**	−1989.1766	**−2000**	−1000.0011	**−2000**	**-2000**	**−2000**
6	−42.944387	**−42.944387**	**−42.944387**	−42.943606	−42.944068	**−42.944387**	**−42.944387**	**−42.944387**
7	0	5.21E-23	**0**	**0**	5.211E-35	7.731E-25	0	**0**
8	0	2.29E-13	**0**	7.699E-05	1.791E-08	**0**	5.978E-05	**0**
9	0	5.02E-15	**0**	7.184E-06	0.0014653	**0**	4.579E-06	**0**
10	−1.01983	**−1.01983**	**−1.01983**	**−1.01983**	−1.01982	**−1.01983**	**−1.01983**	**−1.01983**
11	0	2.89E-05	1.603E-13	**0**	7.456E-08	0.2149297	**0**	**0**
12	0	3.49E-07	9.25E-241	**0**	3.978E-08	**0**	**0**	**0**
13	0	7.78E-16	8.993E-23	**0**	1.087E-06	**0**	**0**	**0**
14	0	0.090476	**0**	**0**	0.0824948	**0**	**0**	**0**
15	0	0.000124	**0**	**0**	0.0039814	**0**	**0**	**0**
16	−186.73091	**−186.73091**	**−186.73091**	−186.64225	−186.70698	**−186.73091**	**−186.73091**	**−186.73091**
17	0	0	**0**	**0**	**0**	**0**	**0**	**0**
18	−24.156816	−0.2463	**−24.156816**	−24.156815	**−24.156816**	**−24.156816**	**−24.156816**	**−24.156816**
19	0	1.02E-07	1.46E-282	**0**	1.548E-06	**0**	**0**	**0**
20	0	1.34E-07	6.21E-295	**0**	4.359E-07	0.9226756	**0**	**0**

Table 4. Comparative mean and standard deviation of TLSO with other algorithms

No.		GA	TLBO	ATLBO	ACO	PSO	Variant	TLSO
1	Mean	1.079437	**0.0082149**	0.0151703	0.053274	0.0082152	0.0147913	0.0082149
	SD	2.7E-08	**3.659E-18**	0.0063262	0.083823	1.474E-06	0.0074474	5.931E-18
2	Mean	1.26E-11	**0**	0.1196782	0.163009	**0**	0.0001953	**0**
	SD	1.83E-11	**0**	0.2547912	0.208544	**0**	0.0002437	**0**
3	Mean	1.16E-08	**0**	**0**	1.16E-06	1.4856251	**0**	**0**
	SD	6.01E-09	**0**	**0**	1.33E-06	2.2239866	**0**	**0**
4	Mean	−1033.57	−1000	−584.1058	−275.733	−2000	−959.4457	**−2000**
	SD	182.5329	0.0009768	405.15092	355.775	3.895E-10	277.57921	**0**
5	Mean	−1666.68	−1044.847	−1135.682	−805.021	−1733.336	**−1863.627**	−1136.941
	SD	479.4529	183.4131	333.80557	335.3678	449.77824	**274.70658**	344.87055
6	Mean	−29.0057	−42.94439	−42.19939	−42.7106	−42.63134	−42.92157	**−42.94439**
	SD	2.6326	3.613E-14	1.1457741	0.179508	0.2084425	0.080008	**3.592E-14**
7	Mean	1.69E-22	**0**	**0**	1.37E-17	0.3861763	**0**	**0**
	SD	5.17E-23	**0**	**0**	1.68E-17	0.5554721	**0**	**0**
8	Mean	3.48E-06	**0**	1.2864531	0.000158	**0**	0.0043448	**0**
	SD	0.0000173	**0**	2.503885	0.000488	**0**	0.0071715	**0**
9	Mean	7.45E-13	**0**	0.3368475	0.443078	1.315E-31	0.0602188	**0**
	SD	9.36E-13	**0**	0.4245833	0.541638	2.99E-31	0.2647606	**0**
10	Mean	−1.01983	**−1.01983**	−1.01943	−1.01561	−1.01983	−1.01949	**−1.01983**
	SD	4.38E-10	**6.78E-16**	0.0017916	0.0113453	6.78E-16	0.0017884	**6.78E-16**
11	Mean	0.000146	1.571E-09	**0**	0.000244	14.968236	**0**	**0**
	SD	0.0000557	2.746E-09	**0**	0.000793	32.267206	**0**	**0**
12	Mean	0.000121	3.27E-123	**0**	0.000518	12.054143	**0**	**0**
	SD	0.000161	1.79E-122	**0**	0.001126	18.489478	**0**	**0**
13	Mean	6.7E-08	2.161E-16	3.859E-16	0.023113	7.969E-22	**0**	**0**
	SD	1.16E-07	3.784E-16	2.114E-15	0.04595	1.833E-21	**0**	**0**

Table 4. Continued

No.		GA	TLBO	ATLBO	ACO	PSO	Variant	TLSO
14	Mean	1.212715	**0**	**0**	2552048	11304.404	**0**	**0**
	SD	0.880296	**0**	**0**	13977798	36539.496	**0**	**0**
15	Mean	0.000964	**0**	**0**	0.154935	9.9933981	**0**	**0**
	SD	0.002403	**0**	**0**	0.208801	30.492723	**0**	**0**
16	Mean	−186.7309	**−186.7309**	−155.9899	−181.416	−186.7309	−181.5166	−186.7309
	SD	2.3E-10	**2.176E-14**	29.291176	9.915138	2.239E-14	12.190273	3.58E-14
17	Mean	**0**	**0**	**0**	**0**	3.3	**0**	**0**
	SD	**0**	**0**	**0**	**0**	18.074844	**0**	**0**
18	Mean	−0.2463	−24.15682	−23.88186	−24.1568	−24.15682	−24.15199	**−24.15682**
	SD	2.53E-08	4.067E-15	0.5232635	1.88E-06	6.663E-15	0.0091706	**3.613E-15**
19	Mean	0.0000386	1.412E-74	**0**	0.0000372	**0**	**0**	**0**
	SD	0.0000929	7.734E-74	**0**	0.000039	**0**	**0**	**0**
20	Mean	0.0000682	4.94E-281	**0**	0.011336	1.691238	**0**	**0**
	SD	0.0000993	**0**	**0**	0.022663	0.7216573	**0**	**0**

Table 5. The number of functions showing the best converged and the least mean and standard variation

	GA	TLBO	ATLBO	ACO	PSO	Variant	TLSO
Best value	6	13	12	2	16	15	**20**
Mean/SD	1	12	9	1	3	11	**17**

tions.

The performance of an algorithm can be assessed also by the convergence speed, because the maximum number of iterations was set equal for all algorithms. Fig. 5 and Fig. 6 show results of comparison for the two selected functions. Fig. 5 shows the results of comparison of seven algorithms (GA, TLBO, ATLBO, ACO, PSO, Variant, TLSO) for the benchmark function #5 (Chen V), and Fig. 6 shows those for the function #18 (Table 3).

For the benchmark function #5 (Chen V), the execution time of GA, ACO and TLBO was longer compared to other algorithms. Especially, TLBO and ACO failed to reach to the global optimum. The ATLBO converged fast, but only reached to local optimum. In contrast to other algorithms, PSO and TLSO outperformed other algorithms and did not generate significant noise. For the benchmark function #18, all the optimization algorithms generated acceptable optimization results. But, ATLBO, PSO and Variant algorithms generated intermittent local optimum. Overall, the TLSO scheme outperforms other algorithms in terms of average and standard deviation.

3. Ranks Test: Friedman, Friedman Aligned and Quade Ranks Test

The Friedman ranks test [30,31] is used to assess effects of experiments employing ranks derived from each column for given experimental data. This method can be regarded as an extended version of the Wilcoxon test in which ranks of corresponding pairs are used to evaluate differences. The Friedman aligned ranks test can be regarded as an upper level Friedman ranks test and is characterized by rank determination based on both column and raw. The Quade test [32] is used to perform multiple comparisons. In contrast to Friedman's test, the Quade test considers the fact that some problems are more difficult or that the differences registered on the run of various algorithms over them are larger (the Friedman test considers all problems to be equal in terms of importance). Therefore, the rankings computed on each problem could be scaled depending on the differences observed in the algorithms' performances. A weighted ranking analysis of the sample of results was performed. Let S_j be a statistic that represents the relative size of each observation within the problem. SW_j is adjusted to reflect the relative significance of the problem in which it appears. Let T_j denote the average ranking for the j^{th} algorithm [33].

To perform the ranks test, the error values of the Benchmark functions with nonzero optimum (f1, f4, f5, f6, f10, f16, f18) are set as test data (see Table 6). Results of rank tests are shown in Table 7 (Friedman ranks test), Table 8 (Friedman aligned ranks test), and Table 9 (Quade ranks test).

As can be seen from Tables 7, 8 and 9, the TLSO scheme shows the lowest average rank value both in Friedman and Friedman aligned ranks test. Moreover, the TLSO scheme also shows the lowest T_j value in Quade ranks test. From these results we can see that the TLSO scheme outperforms other algorithms in terms of ranks.

4. Example

As an example chemical engineering application, consider a gas pipeline transmission system where compressor stations are placed L miles apart. The annual cost is given by [34]

$$C(D, P1, L, \gamma)=7.84D^2P_1+450000+36900D + \frac{6.51\times10^6}{L}+\frac{772\times10^6}{L}(\gamma^{0.219}-1) \quad (21)$$

where D is the pipe inside diameter (in), P_1 is the compressor discharge pressure (psia), L is the length between compressor stations (miles), γ is the compression ratio (P_1/P_2) and P_2 is the compressor inlet pressure (psia).

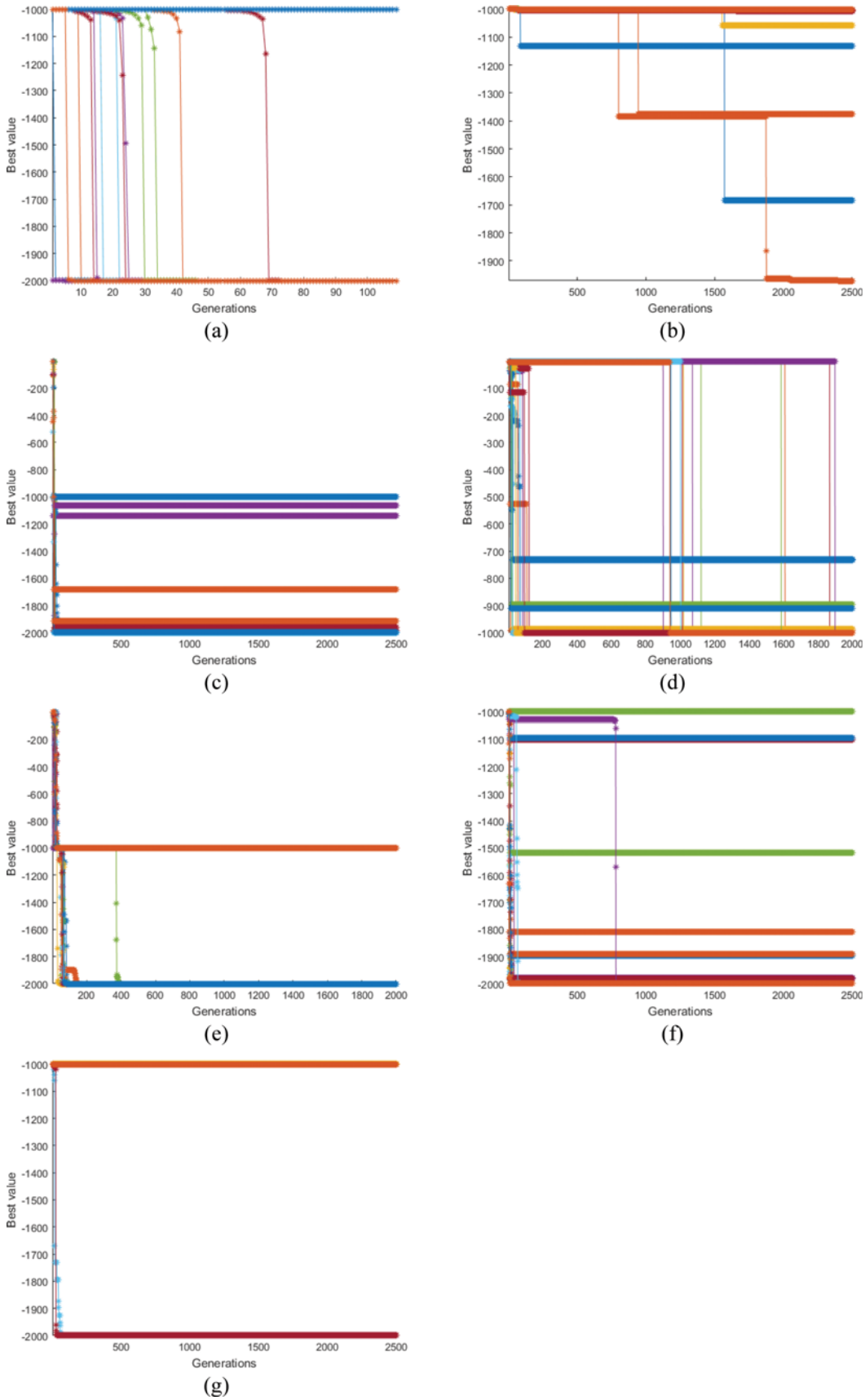

Fig. 5. Best value of objective function f5 Chen V against the number of generation (a) GA (b) TLBO (c) ATLBO (d) ACO (e) PSO (f) Variant (g) TLSO.

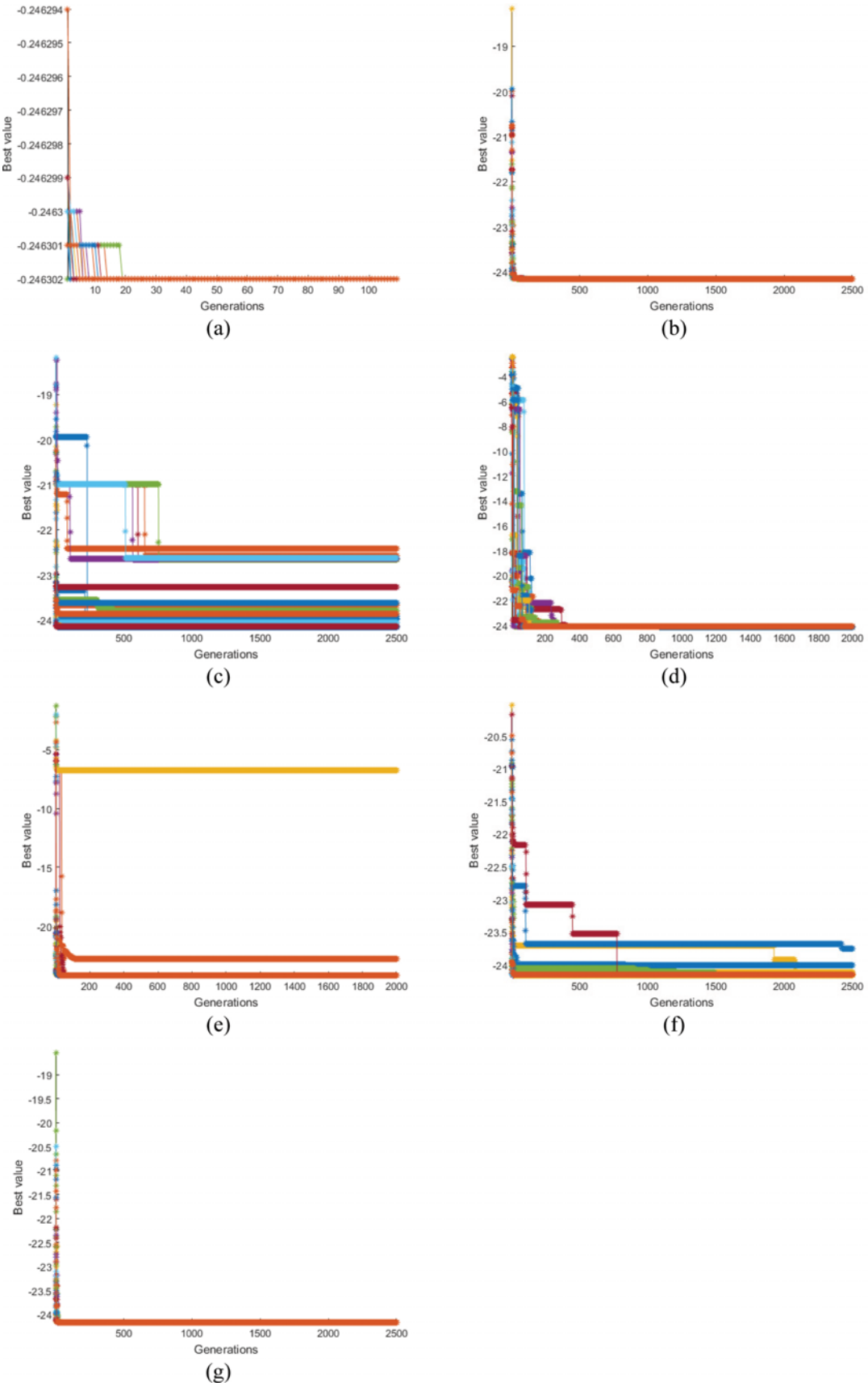

Fig. 6. Best value of objective function f18 Table 3 against the number of generation (a) GA (b) TLBO (c) ATLBO (d) ACO (e) PSO (f) Variant (g) TLSO.

Table 6. Error rates of Benchmark functions with nonzero optimum

No.	GA	TLBO	ATLBO	ACO	PSO	Variant	TLSO
1	1.071222	7.31E-09	0.000161	0.001123	7.31E-09	1.89E-05	7.31E-09
4	0	999.9946	999.9938	1000.087	0	0.000465	0
5	0.004	10.82744	6.09E-10	1000.003	0	0	0
6	0.3715	0.371513	0.372294	0.371832	0.371513	0.371513	0.371513
10	0	0.91092	0.91092	0.91092	0.91092	0.91092	0.91092
16	0	9.12E-05	0.088749	0.024018	9.12E-05	9.15E-05	9.12E-05
18	23.91052	0	1.10E-07	1.21E-09	0	6.46E-12	0

Table 7. Results of ranks test: Friedman ranks test

No.	GA	TLBO	ATLBO	ACO	PSO	Variant	TLSO
1	7	2.5	5	6	2.5	4	1
4	2	6	5	7	2	4	2
5	5	6	4	7	2	2	2
6	1	3	7	6	3	5	3
10	1	4	4	7	4	4	4
16	1	3.5	7	6	3.5	5	2
18	7	2	6	5	2	4	2
Average	3.428571	3.857143	5.428571	6.285714	2.714286	4	2.285714

Table 8. Results of ranks test: Friedman aligned ranks test

No.	GA	TLBO	ATLBO	ACO	PSO	Variant	TLSO
1	44	19	22	23	19	21	19
4	2	47	46	48	2	4	2
5	9	10	8	49	6	6	0
6	29	31	35	34	31	33	31
10	17	40	40	43	40	40	40
16	24	26.5	37	36	26.5	28	25
18	45	12	16	15	12	14	12
Average	24.28571	26.5	29.14286	35.42857	19.5	20.85714	19.28571

Table 9. Results of ranks test: Quade ranks test

No.	GA	TLBO	ATLBO	ACO	PSO	Variant	TLSO
1	18	−3	3	4	−6	0	−12
4	−4	14	7	21	−4	0	−4
5	4	12	0	18	−4	−2	−4
6	−15	−4	15	8	−6	6	−6
10	−6	0	0	15	0	0	0
16	−6	−1.5	12	6	−2.5	4	−10
18	21	−2	4	1	−4	0	−4
S_j	12	15.5	41	73	−26.5	8	−40
T_j	4.428571	4.553571	5.464286	6.607143	3.053571	4.285714	2.571429

The flow rate (Q) is related to the design variables as

$$Q = 3.39\sqrt{\frac{(P_1^2 - P_2^2)^2 D^5}{fL}} \quad (22)$$

where f is the friction factor (=$0.008D^{-1/3}$). The purpose is to determine the lowest annual cost when the flow rate is 100×10^6 scf/day [32]. In this problem, there are four continuous independent variables (D, P_1, L and γ) to yield a four-dimensional optimization problem. Upper limits of D, P_1, L and γ are 1000 inch, 500 psia, 1000 miles and 5, respectively. Table 10 shows results of the solution of this optimization problem using optimization methods described before. The numbers printed in bold font denote the best perfor-

Table 10. Results of optimizations: minimum annual cost

	GA	TLBO	ATLBO	ACO	PSO	Variant	TLSO
Optimal value	**13307605**	**13307605**	45939299	**13307605**	**13307605**	13314978	**13307605**
Mean	13409254	**13307605**	1.12E+09	13318030	13753851	13943347	**13307605**
SD	364674.2	7.65E-09	1.38E+09	55312.97	55312.97	477178.1	**7.48E-09**

mance. The penalty function method was used to consider the violation of each constraint [35].

As can be seen from Table 10, GA, TLBO, ACO, PSO, and TLSO generated the same optimal value. Among these methods, TLBO and TLSO gave the lowest mean value, and only TLSO showed the lowest standard deviation. The values of D, P_1, L and γ at the optimum point achieved by TLSO method are 107.2036, 65.8561, 12.3221 and 1.1901, respectively.

CONCLUSION

Benchmark test functions are used to validate and to compare the performance of specific optimization methods. The performance of different optimization algorithms is validated and compared while solving benchmark problems. The Teaching-learning-self-study-optimization (TLSO) algorithm is one of the newly developed optimization schemes based on the Teaching-learning-based optimization (TLBO) algorithm. In TLBO, the teacher's effort is distributed at random to all learners and it offers a random amount of learning to both good and bad learners. In the learner phase, apart from learning from each other, the learners can increase their knowledge by self-learning. The TLSO algorithm incorporates the teacher's focus on bad learners and a phenomenon of self-learning into the TLBO. The performance of the TLSO algorithm is analyzed and compared with other well-known optimization algorithms including GA, TLBO, ATLBO, ACO and PSO. A set of 20 benchmark functions has been reviewed and tested for unconstrained optimization problems to validate and to compare the optimization algorithms. In a chemical engineering example, the TLSO algorithm exhibited the best results compared to other algorithms. The TLSO algorithm showed the fastest convergence speed to the optimum and outperformed other algorithms for most test functions.

REFERENCES

1. D. E. Goldberg, *Genetic algorithms in search optimization and machine learning* (Vol. 412), Reading Menlo Park Addison-Wesley (1989).
2. R. Storn and K. Price, *J. Global Optimization*, **11**(4), 341 (1997).
3. T. P. Runarsson and X. Yao, *IEEE Transactions on Evolutionary Computation*, **4**(3), 284 (2000).
4. J. D. Farmer, N. H. Packard and A. S. Perelson, *Physica D: Nonlinear Phenomena*, **22**(1), 187 (1986).
5. K. M. Passino, *IEEE Control Systems*, **22**(3), 52 (2002).
6. M. Clerc, *Particle swarm optimization* (Vol. 93), Wiley (2010).
7. M. Dorigo, M. Birattari and T. Stützle, *IEEE Computational Intelligence*, **1**(4), 28 (2006).
8. C. Blum, *Physics of Life Reviews*, **2**(4), 353 (2005).
9. D. Karaboga, *An idea based on honey bee swarm for numerical optimization*, Technical report-tr06, Erciyes University, Computer Engineering Department (2005).
10. D. Karaboga and B. Basturk, *Applied Soft Computing*, **8**(1), 687 (2008).
11. S. Mirjalili, S. M. Mirjalili and A. Hatamlou, *Neural Computing and Applications*, 1 (2015).
12. M. Yazdani and F. Jolai, *J. of Computational Design and Eng.*, **3**(1), 24 (2016).
13. R. V. Rao, V. J. Savsani and D. P. Vakharia, *Computer-Aided Design*, **43**(3), 303 (2011).
14. A. Verma, S. Agrawal, J. Agrawal and S. Sharma, in *Proceedings of 3rd International Conference on Advanced Computing*, Networking and Informatics, Springer, India (2016).
15. D. Chen, F. Zou, Z. Li, J. Wang and S. Li, *Information Sciences*, **297**, 171 (2015).
16. D. Chen, R. Lu, F. Zou and S. Li, *Neurocomputing*, **173**, 1096 (2016).
17. S. C. Satapathy and A. Naik, *Recent Patents on Computer Science*, **6**(1), 60 (2013).
18. S. C. Satapathy, A. Naik and K. Parvathi, *SpringerPlus*, **2**(1), 130 (2013).
19. J. Brest, S. Greiner, B. Bošković, M. Mernik and V. Zumer, *IEEE Transactions on Evolutionary Computation*, **10**(6), 646 (2006).
20. N. Patel and N. Padhiyar, *J. Process Control*, **26**, 35 (2015).
21. G. Li, P. Niu, W. Zhang and Y. Liu, *Chemometrics and Intelligent Laboratory Systems*, **126**, 11 (2013).
22. A. Faisal, Ph. D. Dissertation, Hanyang University (2016).
23. D. Whitley, *Statistics and Computing*, **4**(2), 65 (1994).
24. M. Afshar, A. Gholami and M. Asoodeh, *Korean J. Chem. Eng.*, **31**(3), 496 (2014).
25. R. Rao and V. Patel, *International J. Industrial Eng. Computations*, **4**(1), 29 (2013).
26. L. G. Zheng, H. Zhou, K. F. Cen and C. L. Wang, *Expert Systems with Applications*, **36**(2), 2780 (2009).
27. C. E. de Araújo Padilha, N. K. de Araújo, D. F. de Santana Souza, J. A. de Oliveira, G. R. de Macedo and E. S. dos Santos, *Korean J. Chem. Eng.*, **33**(9), 2650 (2016).
28. S. Sumathi and P. Surekha., *Computational Intelligence Paradigms*, CRC Press (2010).
29. M. Jamil and X. S. Yang, *International J. Mathematical Modelling and Numerical Optimization*, **4**(2), 150 (2013).
30. M. Friedman, *J. the American Statistical Association*, **32**, 674 (1937).
31. M. Friedman, *Annals of Mathematical Statistics*, **11**, 86 (1940).
32. D. Quade, *J. the American Statistical Association*, **74**, 680 (1979).
33. J. Derrac, S. García, D. Molina and F. Herrera, *Swarm and Evolutionary Computation*, **1**(1), 3 (2011).
34. H. Adidharma and V. Temyanko, *Mathcad for chemical engineers, 2nd Ed.*, Trafford Publishing (2009).
35. R. Rao Jaya, *Int. J. Ind. Eng. Computations.*, **7**(1), 19 (2016).