Advances in Robotics Research, *Vol. 2, No. 1 (2018) 13-31*
DOI: https://doi.org/10.12989/arr.2018.2.1.013

Optimum static balancing of a robot manipulator using TLBO algorithm

R. Venkata Rao[*1] and Gajanan Waghmare[2a]

[1]*Department of Mechanical Engineering, S.V. National Institute of Technology, Ichchanath, Surat, Gujarat 395 007, India*
[2]*Department of Mechanical Engineering, Sandip Institute of Engineering and Management, Nashik, Maharashtra 422213, India*

(Received December 1, 2015, Revised February 27, 2017, Accepted December 7, 2017)

Abstract. This paper presents the performance of Teaching-Learning-Based Optimization (TLBO) algorithm for optimum static balancing of a robot manipulator. Static balancing of robot manipulator is an important aspect of the overall robot performance and the most demanding process in any robot system to match the need for the production requirements. The average force on the gripper in the working area is considered as an objective function. Length of the links, angle between them and stiffness of springs are considered as the design variables. Three robot manipulator configurations are optimized. The results show the better or competitive performance of the TLBO algorithm over the other optimization algorithms considered by the previous researchers.

Keywords: static balancing; robot manipulator; teaching-learning-based optimization algorithm

1. Introduction

Over the past few decades, the interest of researchers is growing in the field of design optimization of robot manipulator in order to improve robot system performance using advanced optimization techniques. The robot manipulator is a mechanism consisting of the major linkages, the minor linkages and the end effectors (gripper or tool).

Static balancing is very important aspect in the design of robot manipulator. Statically balanced systems are in equilibrium in every configuration in their workspace, even when no friction is present. As a consequence, these systems can usually be operated with much less effort as compared to the unbalanced situation. In Static balancing the weight of the links does not produce any force at actuators for any configuration of the manipulator. A draw bridge for instance is usually provided with a counterweight so that the heavy bridge structure can be lifted by hand. The counterweight provides continuous equilibrium, or alternatively, complements the potential energy to a constant value so that the system does not have a preferred configuration and becomes

∗Corresponding author, Professor, E-mail: ravipudirao@gmail.com
[a]Associate Professor, E-mail: waghmaregaju@yahoo.com

http://www.techno-press.org/?journal=arr&subpage=7 ISSN: 2287-4976 (Print), 2287-4984 (Online)

indifferent. The counterweight provides the energy for the bridge to move up and restores the energy when the bridge moves down. A statically balanced mechanism can be obtained by achieving a potential energy which remains constant irrespective of the position and orientation of the mechanism.

Static balancing involves ensuring that the motors do not contribute towards supporting the mechanism's weight, for any of the possible configurations. A mechanism can therefore remain stable in any position without the help of motors or brakes. This result can be obtained by using counterweights or springs. Balancing using springs has added advantages over mass balancing since it requires balancing of masses due to added counterweights. However, in spring balancing, a change in mass is insignificant since weight of the springs is very less compared to link weight.

Over the past few decades, the interest of researchers is growing in the field of balancing of robot manipulator using advanced optimization techniques. Efforts were made to attempt balancing of robot manipulator by optimization researchers. Filaretov and Vukobratović (1993) presented static balancing and dynamic decoupling of the motion of manipulation robots. The authors analyzed the mechanical unloading of a multi-link manipulator from the moments due to gravitational forces. The specificity of dynamic decoupling of the manipulator motion was considered. Examples of different types of balancing and unloading with manipulation mechanisms were presented.

Segla (1998) presented statical balancing of a robot manipulator using genetic algorithm. As objective function the average force on the gripper in the working area was used. The lengths of the links and angles between them as well as the stiffness of springs were considered as design variables. The author had considered as an industrial robot with 6-DOF. The robot had a spring balancing system that had to be optimized.

Simionescu and Ciupitu (2000a, b) presented some new constructional solutions for the balancing of the weight forces of the robot arms, using the elastic forces of the helical springs. The authors had defined a new notion, namely *efficaciousness coefficient*, for the performance study of the static balancing mechanisms. This coefficient was equal to the ratio of the mechanical work consumed for acting the unbalanced arm and the mechanical work consumed for moving the balanced arm.

Ouyang and Zhang (2003) described an integrated approach to design a real-time controllable (RTC) mechanism considering force balancing and trajectory tracking, simultaneously. A new approach called adjusting kinematic parameter (AKP) for the force balancing of RTC mechanisms was described. The authors had demonstrated that the force balanced mechanism by the AKP approach was more promising than those by other approaches in terms of the reduction of joint forces and torques in servomotors, and improvement of the trajectory tracking performance. Based on simulation, authors also showed the effects of two different control systems: PD and non-linear PD versus the AKP approach and the counterweight approach.

Ouyang and Zhang (2004) proposed a force balancing method called adjusting kinematic parameters (AKP) for robotic mechanisms or real-time controllable (RTC) mechanisms. A particular implementation of the AKP method for the RTC mechanisms where two pivots on a link were adjustable was presented. A comparison of the two methods, namely the AKP method and the counterweights (CW) method, was made for two RTC mechanisms with different mass distribution.

Russo *et al.* (2005) addressed the static balancing of spatial parallel manipulator and the conditions for balancing were derived. The authors had used two methods lead to static balancing, namely using counterweight and using springs. In both methods, the resulting mechanism was fully balanced for gravity.

Saravanan *et al.* (2008) presented optimum static balancing of an industrial robot mechanism. The authors had described the use of conventional and evolutionary optimization techniques such as Newton's method (NM), conjugate gradient method (CGM), Genetic algorithm (GA), Elitist Non-dominated Sorting Genetic Algorithm (NSGA-II) and differential evolution (DE) to solve static balancing of robot mechanism problems. The average force on the gripper in the working area was taken as objective function and length of the links, angle between them and stiffness of spring were considered as design variables.

Sangamesh and Ananthasuresh (2012) presented a technique to statically balance any planar revolute-jointed linkage having zero-free-length spring and constant load interactions between the bodies of the linkage. The technique involved only addition of zero-free-length springs but not any extra link, unlike spring-aided perfect static balancing techniques. Petković *et al.* (2012, 2013a, b, c, d, e) had presented many details of an adaptive compliant robot gripper.

Martini *et al.* (2015) analytically performed gravity compensation of a 3-DOF spatial parallel mechanism. The authors presented a feasible solution with two balancing springs and one auxiliary linkage. Static balancing reduces the robot estimated energy requirements for common tasks. The compensation effectiveness is marginally affected by potential design inaccuracies. Pierezan *et al.* (2017) proposed a modified self-adaptive differential evolution approach for static modeling of a humanoid robot and the optimization of its static force capability. Hassan and Abomoharam (2017) proposed a general robot modeling and optimal design process. The authors presented NSGA-II algorithm for a multi-objective design of the gripper. The authors also performed a local sensitivity analysis of an optimal solution to identify the most critical links of the gripper. Quaglia and Yin (2015) proposed a method to design balancing devices for articulated robots in industry, based on robotic dynamics. The authors presented two aspects: One is the optimization for the position of the balancing system; the other is the design of the spring parameters.

It has been observed that only few researchers had attempted the optimization of static balancing of a robot manipulator. Segla (1998) used GA, Saravanan *et al.* (2008) used conventional and evolutionary methods like NM, CGM, GA, NSGA-II and DE for optimum static balancing of robot manipulator. However, the parameters setting of the GA, NSGA-II and DE algorithms is a serious problem which influences their efficiency and affect the performance of the algorithms. For example, GA requires the crossover probability, mutation rate, and selection operator; NSGA-II requires crossover probability, real-parameter mutation probability, mutation parameter; DE requires the crossover probability and differential weight. Proper tuning of the algorithm-specific parameters is very crucial and affects the performance of the above mentioned algorithms. The improper tuning of algorithm-specific parameters either increases the computational effort or yields the local optimal solution. In addition to the tuning of algorithm-specific parameters, the common control parameters need to be tuned which further enhances the effort. Finding the optimized value of these algorithm-specific parameters is an optimization problem itself. Considering this fact, Rao *et al.* (2011, 2012a, b) introduced the teaching-learning-based optimization (TLBO) algorithm which does not require any algorithm-specific parameters thus making the implementation of TLBO algorithm simpler. This algorithm requires only the common control parameters and does not require any algorithm-specific control parameters. Since the tuning of any algorithm-specific parameters is not required in the TLBO algorithm, the achieved results can be more accurate.

In the literature, it is observed that the TLBO algorithm is not yet used in the field of static balancing of robot manipulator. Hence the same is now used for the parameter optimization of static balancing of robot manipulator under consideration. In this work, efforts are carried out to investigate the performance of the TLBO algorithm to obtain the optimum set of design parameters for static balancing of robot manipulator and comparisons are made with various other optimization algorithms.

The next section gives a brief description about the TLBO algorithm, which is used for the static balancing of the robot manipulator.

2. Teaching-learning-based optimization (TLBO)

The TLBO algorithm is a teaching-learning process inspired algorithm proposed by Rao *et al.* (2011, 2012a, b), Rao and Savsani (2012) and Rao and Patel (2012, 2013) based on the effect of influence of a teacher on the output of learners in a class. The algorithm describes two basic modes of the learning: (i) through teacher (known as teacher phase) and (ii) interacting with the other learners (known as learner phase). In this optimization algorithm a group of learners is considered as population and different subjects offered to the learners are considered as different

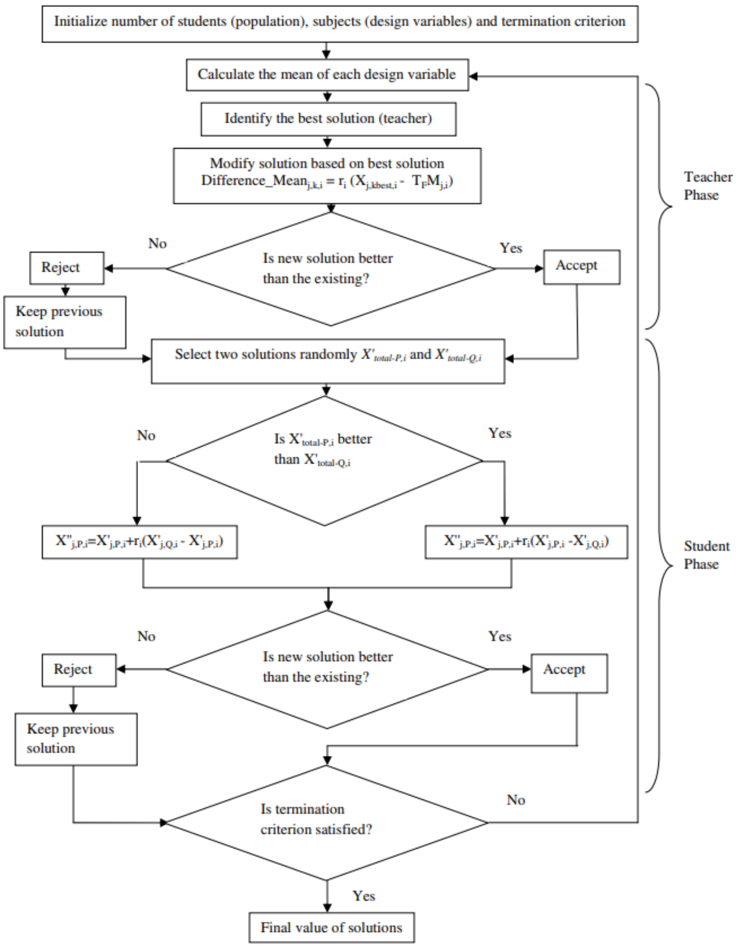

Fig. 1 Flowchart of TLBO algorithm (Rao *et al.* (2012)

design variables of the optimization problem and a learner's result is analogous to the 'fitness' value of the optimization problem. The best solution in the entire population is considered as the teacher. The design variables are actually the parameters involved in the objective function of the given optimization problem and the best solution is the best value of the objective function. The working of TLBO is divided into two parts, 'Teacher phase' and 'Learner phase'. The flowchart of teaching-learning-based optimization algorithm is shown in Fig. 1. For details of the algorithm and its code the readers may refer to https://sites.google.com/site/tlborao/. One complete iteration of the TLBO algorithm is demonstrated in the Appendix for minimization of a standard benchmark Sphere function.

2.1 Teacher phase

It is the first part of the algorithm where learners learn through the teacher. During this phase a teacher tries to increase the mean result of the class in the subject taught by him or her depending on his or her capability. At any iteration i, assume that there are 'm' number of subjects (i.e., design variables), 'n' number of learners (i.e., population size, k=1,2,…,n) and $M_{j,i}$ be the mean result of the learners in a particular subject 'j' (j=1,2,…,m) The best overall result $X_{total\text{-}kbest,i}$ considering all the subjects together obtained in the entire population of learners can be considered as the result of best learner *kbest*. However, as the teacher is usually considered as a highly learned person who trains learners so that they can have better results, the best learner identified is considered by the algorithm as the teacher. The difference between the existing mean result of each subject and the corresponding result of the teacher for each subject is given by

$$Difference\_Mean_{j,k,i} = r_i \, (X_{j,kbest,i} - T_F M_{j,i}) \qquad (1)$$

where $X_{j,kbest,i}$ is the result of the best learner (i.e. teacher) in subject j. T_F is the teaching factor which decides the value of mean to be changed, and r_i is the random number in the range [0, 1].

Value of T_F can be either 1 or 2. The value of T_F is decided randomly with equal probability as

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

T_F is not a parameter of the TLBO algorithm. The value of T_F is not given as an input to the algorithm and its value is randomly decided by the algorithm using Eq. (2). After conducting a number of experiments on many benchmark functions it is concluded that the algorithm performs better if the value of T_F is between 1 and 2. However, the algorithm is found to perform much better if the value of TF is either 1 or 2 and hence to simplify the algorithm, the teaching factor is suggested to take either 1 or 2 depending on the rounding up criteria given by Eq.(2).

Based on the $Difference\_Mean_{j,k,i}$, the existing solution is updated in the teacher phase according to the following expression

$$X'_{j,k,i} = X_{j,k,i} + Difference\_Mean_{j,k,i} \qquad (3)$$

where $X'_{j,k,i}$ is the updated value of $X_{j,k,i}$. Accept $X'_{j,k,i}$ if it gives better function value. All the accepted function values at the end of the teacher phase are maintained and these values become the input to the learner phase. The learner phase depends upon the teacher phase.

2.2. Learner phase

It is the second part of the algorithm where learners increase their knowledge by interaction among themselves. A learner interacts randomly with other learners for enhancing his or her knowledge. A learner learns new things if the other learner has more knowledge than him or her. Considering a population size of 'n', the learning phenomenon of this phase is expressed below.

Randomly select two learners P and Q such that $X'_{total\text{-}P,i} \neq X'_{total\text{-}Q,i}$ (where, $X'_{total\text{-}P,i}$ and $X'_{total\text{-}Q,i}$ are the updated values of $X_{total\text{-}P,i}$ and $X_{total\text{-}Q,i}$ respectively at the end of teacher phase). In the case of minimization problems

$$X''_{j,P,i} = X'_{j,P,i} + r_i\,(X'_{j,P,i} - X'_{j,Q,i}) \quad \text{if } X'_{total\text{-}P,i} < X'_{total\text{-}Q,i} \tag{4a}$$

$$X''_{j,P,i} = X'_{j,P,i} + r_i\,(X'_{j,Q,i} - X'_{j,P,i}) \text{ if } X'_{total\text{-}Q,I} < X'_{total\text{-}P,i} \tag{4b}$$

$X''_{j,P,i}$ is accepted if it gives a better function value.

The TLBO algorithm has been already tested on several constrained and unconstrained benchmark functions and proved better than the other advanced optimization techniques (Rao and Patel 2012, 2013, Rao 2016), Rao and Waghmare (2014) had evaluated the performance of the TLBO algorithm over a set of multi-objective unconstrained and constrained test functions and the results were compared against the other optimization algorithms. The TLBO algorithm was observed to outperform the other optimization algorithms for the multi-objective unconstrained and constrained benchmark problems.

It may be mentioned that various researchers like Niknam *et al.* (2012), Rao *et al.* (2014), Baykasoğlu *et al.* (2014), Satapathy and Naik (2014), Medina *et al.* (2014), Basu (2014), Zou *et al.* (2014), Camp and Farshchin (2014), Moghadam and Seifi (2014) and Sultana and Roy (2014) proved the better performance of the TLBO algorithm as compared to the other evolutionary algorithms.

Hence, literature showed that TLBO algorithm is proving better in various field of engineering applications. In the literature, it is observed that the TLBO algorithm is not yet used in the field of static balancing of robot manipulator. Hence the same is now used for the parameter optimization of static balancing of robot manipulator under consideration.

The next section presents the details of the problem formulation for static balancing of the robot manipulator.

3. Problem formulation

A formal representation of an optimization model can be stated as follows

To find $X = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ x_n \end{bmatrix}$ which maximizes $f(X)$

Subject to the constraints

$$g_i(X) \le 0, \;\; i = 1,2, \ldots., m$$
$$l_j(X) \le 0, \qquad j = 1,2, \ldots., p$$

where X is an n-dimensional vector called the design vector, $f(X)$ is called the objective function, and $g_i(X)$ and $l_j(X)$ are known as inequality and equality constraints, respectively.

In the present optimization model, the average force on the gripper in the working area is taken as an objective function. The design variables are the lengths of the links, angles between them

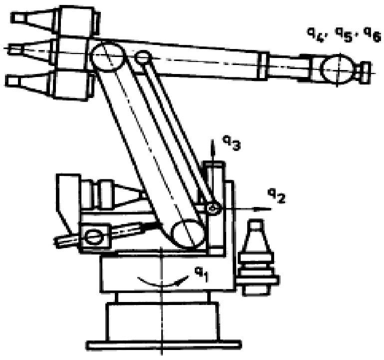

Fig. 2 Robot APR 20 (Segla 1998)

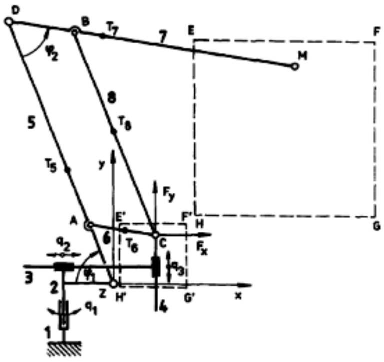

Fig. 3 Scheme of robot APR 20 (Segla 1998)

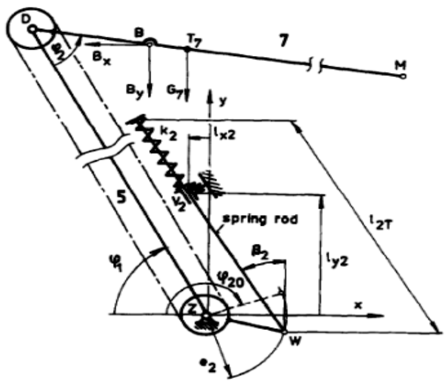

Fig. 4 Static equilibrium of the link 7 (Segla 1998)

and stiffness of springs. An industrial robot with 6-degree-of-freedom (6-DOF) (APR 20) is considered as a numerical example. The same robot was considered by Segla (1998) and Saravanan *et al.* (2008) and the details of the design variables, objective function and constraints are available in Segla (1998) and Saravanan *et al.* (2008). However, the important details are included in this paper for readers' convenience. The robot has a spring balancing system that has

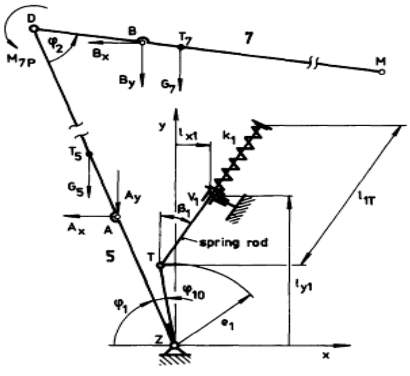

Fig. 5 Static equilibrium of the links 5 and 7 together (Segla 1998)

to be optimized. The method described here can be applied to any robot or mechanism that has to be designed to produce a certain kinematic, static or dynamic behaviour.

The industrial robot APR20 with 6-DOF (q_1, q_2, q_3, q_4, q_5 and q_6) is shown in Fig. 2. It is clear from Fig. 2 that the 6-DOF of the mechanism is to be balanced by only q_2 and q_3. When a mass of zero at the gripper is assumed, q_4, q_5 and q_6 have no influence. When the robot is in a correctly vertical position, the rotation about the vertical axis (q_1) is balanced. This simplifies the balancing considerably as the spatial robot can now be reduced to a planar mechanism with 2-DOF. It is proved that in the case of the links 5 and 7 of negligible mass the static gravity forces at the DOF q_2 and q_3 caused by gravitation forces of links 5 and 7 (Fig. 3) can be completely eliminated (Segla 1998).

The problem of the statically balancing of the robot taking into account masses of the links 6 and 8 can be formulated as an optimization problem. Spring mechanisms are used to balance the robot APR 20 (Figs. 4 and 5). Fig. 4 shows statically balancing of link 7. The balancing moment is transmitted to link 7 by a mechanical belt and pulley transmission. The first pulley wheel of the transmission placed on the rotating base of the robot can rotate independently of the rotation of link 5. The second one (of the same diameter) is attached to link 7. One end of the spring rod of this balancing mechanism is connected to the lower pulley by a revolute joint at W and a balancing spring 2 of constant stiffness k_2 is placed between the other end of the spring rod and a joint at V_2 (placed on the rotating base of the robot) which allows rotation and translation of the spring rod. Fig. 5 shows the balancing mechanism of link 5 that is similar to the previous one. The spring rod is connected to link 5 by a revolute joint at T. The parameters of both balancing mechanisms that minimize the forces F_x and $F_y$acting on point C (Fig. 3) are to be found. These forces have to ensure statical balance of the robot in any position of the robot gripper (M) in the working area of the robot EFGH. It should be noted that from a parallelogram ACBD, there is a simple relation between the position of the robot gripper (M) in the rectangle EFGH and the position of the joint C in the smaller rectangle E'F'G'H'.

The position of point M in the rectangle EFGH corresponds to the position of point C in the rectangle E'F'G'H' which is determined by intervals of coordinates x and y: 0.115 m $\leq x \leq$ 0.295 m and -0.025 m $\leq y \leq$ 0.155 m. The forces F_x and F_y have to be minimized for all occurring positions of the robot. For this purpose a rectangular grid is used. So there are four sets of F_x and F_y corresponding to the corner points of the rectangle E'F'G'H'. The force acting on point C is the average of forces acting on the corner points of rectangle E'F'G'H'. The objective function is the average force (fav) acting on point C that is needed to balance the robot. This average force is to be minimized (Saravanan *et al.* 2008):

The following are important parameters to be considered.

•Forces acting on point C in horizontal and vertical directions: F_x, F_y

•Coordinates of the robot mechanism: $ø_1$, $ø_2$

•Reaction forces: A_x, A_y, B_x, B_y

•Stiffness of springs 1 and 2: k_1, k_2

•Length of the unloaded springs 1 and 2: l_{01}, l_{02}

•Lengths of spring rods: l_1T, l_2T

•Distances determining position of point V_1: l_{x1}, l_{y1}

•Distances determining position of point V_2: l_{x2}, l_{y2}

•Distances between points ZT, ZW: e_1, e_2

•Angle determining position of point T: $ø_{10}$

•Angle determining position of point W (when $ø_1 = ø_2$): $ø_{20}$

•Masses of robot links: m_5=57.0 kg, m_6=1.0 kg, m_7 =97.43 kg (includes mass of robot gripper and nominal payload of 10 kg), m_8=5.16 kg.

•Length of links: l_5=1.098 m, l_6=0.18 m, l_7=1.098 m, l_8=0.918 m, Az=a=0.18m, DB=b= 0.18 m.

•Lengths determining centers of gravity: CT8=*c*=0.458 m, DT7=*d*, ZT5=*p*=0.5 m, AT6=*q*=0.105 m.

•Permissible spring deflections: t_{m1}= 0.15 m, t_{m2}=0.1 m.

The formal mathematical model can be represented as follows

$$Min.fav = \sum_{i=1}^{N} \left(\sqrt{F_{xi}^2 + F_{yi}^2}\right)/N \tag{5}$$

where N is the number of points in the rectangular E'F'G'H' at which the forces F_x and F_y are computed. For this purpose a rectangular grid will be used. The forces F_x and F_y have to be minimized for all occurring positions of the robot. These forces are functions of the coordinate's $ø_1$ and $ø_2$ and the design variables of the robot.

When the variables of the robot and its balancing mechanisms are known, the forces F_x and F_y can be computed from the following Eqs. (6) and (7) for all possible values of the coordinates $ø_1$ and $ø_2$. In an ideal situation the forces will be zero.

Where

$$F_x = cf_1cv\left(\frac{G_5p}{a} + G_6 + M5P/(cf_1a)\right) + \frac{l_5G_7}{a} - \frac{G_6q}{l_6} - \frac{G_8c}{l_8} - \frac{G_7d}{b} + M7P\ (cvb) + l_5G_8c/(al_8))/sf_2 \tag{6}$$

$$F_y = cf_1sv(-l_5cG_8/al_8) + \frac{G_8c}{l_8} - \frac{l_5G_7}{a} - \frac{G_5p}{a} - \frac{M5P}{(acf_1)} - G_6)sf_2 + sf_1cv\left(-\frac{M7P}{bcv}\right) + \frac{G_7d}{b} + G_6q/l_6)sf_2 + G_8 \tag{7}$$

The coordinates $ø_1$ and $ø_2$ can be computed from the coordinates x and y of point C using the following equations

$$ø_2 = 2arcsin\left(\sqrt{(x^2+y^2)/2a}\right) \tag{8}$$

$$ø_1 = {}^{\Pi}/_2 + {}^{ø}/_2 - arcsin\left(y/\sqrt{x^2+y^2}\right) \tag{9}$$

$$cv = cos(ø_1 - ø_2) \tag{10}$$

$$sf_1 = sinø_1 \tag{11}$$

$$sf_2 = sinø_2 \tag{12}$$

$$sv = sin(ø_1 - ø_2) \tag{13}$$

$$cf_1 = cosø_1 \tag{14}$$

The balancing moments M5P and M7P are determined by following equations

$$M5P = -K_1 \left\{ l_{01} - \left[l_1T - \sqrt{b_1^2 + b_2^2} \right] \right\} e_1 sin(\beta_1 + \Pi/2 - ø_1 - ø_{10}) \tag{15}$$

$$b_1 = e_1 cos(ø_1 + ø_{10}) - l_{x1} \tag{16}$$

$$b_2 = -e_1 sin(ø_1 + ø_{10}) - l_{y1} \tag{17}$$

$$\beta_1 = arctg(b_1/b_2) \tag{18}$$

$$M7P = -K_2 \left\{ l_{02} - \left[l_2T - \sqrt{a_1^2 + a_2^2} \right] \right\} e_2 sin(3\Pi/2 - \beta_2 - ø_{20} - ø_1 + ø_2) \tag{19}$$

$$a_1 = e_2 cos(3\Pi/2 - ø_{20} - ø_1 + ø_2) + l_{y2} \tag{20}$$

$$a_2 = e_2 sin(3\Pi/2 - ø_{20} - ø_1 + ø_2) + l_{x2} \tag{21}$$

$$\beta_2 = arctg(a_2/a_1) \tag{22}$$

The appropriate lengths of the spring rods l_1T and l_2T are calculated from following equations

$$l_1T = \sqrt{(lx_1^2 + ly_1^2)} + tm_1 \tag{23}$$

$$l_2T = \sqrt{(e^2 + ly_2)^2} + lx_2^2 + tm_2 \tag{24}$$

For gravitational forces the relations are:
$G_i = m_i g$ (i= 5,…..,8) where g is the acceleration due to gravity.

Search intervals for design variables are given in Table 1. e_1 and e_2 are introduced as the two new variables in addition with design variables considered by Segla (1998). In Table 1, the search intervals for all independent design variables are given.

The optimization results are obtained for three different values of d (length determining C. G. of DT7) for three cases. In the first case the value of d is considered as 0.2225 m and as 0 for the second case. For the third case the value of d is considered as 0.122236 m (Saravanan *et al.* 2008). The optimum objective function (fav) for all the three cases using different optimization methods are given in Table 2.

Table 1 Search intervals

Variables (units)	Lower bound	Upper bound
k_1 (N/m)	0	4,000,000
l_{01} (m)	0.15	0.4
l_{x1} (m)	-0.08	0.08
l_{y1} (m)	0.035	0.234
$Ø_{10}$ (rad)	-0.3491	0.3491
k_2 (N/m)	0	600,000
l_{02} (m)	0.1	0.45
l_{x2} (m)	-0.04	0.04
l_{y2} (m)	0.024	0.18
$Ø_{20}$	2.7925	3.4906
e_1 (m)	0.075	0.125
e_2 (m)	0.075	0.125

Table 2 Optimum result obtained from various methods

Case no.	NM	CGM	GA	NSGA-II	DE	TLBO
1	26.38839785	26.37021773	16.8278733	16.7774657	16.52231619	16.36783591
2	21.74875683	21.84766492	17.42968778	16.6944685	16.48118857	16.32845249
3	26.05045689	22.35298544	12.17495621	12.01782274	11.9821382	11.61893467

The results of NM, CGM, GA, NSGA-II and DE are from Saravanan *et al.* (2008).

Table 3 Effects of new variable in the objective function for case 1

Variables	Without considering e_1 and e_2 as variables (Segla 1998)	With considering e_1 and e_2 as variables using DE (Saravanan 2008)	With considering e_1 and e_2 as variables using TLBO
k_1 (N/m)	64,957.72	76,215.15629	81,834.37846
l_{01} (m)	0.3595802	0.292022486	0.317847382
l_{x1} (m)	- 0.04250812	0.016286408	0.011389923
l_{y1} (m)	0.2047589	0.14144445	0.015795403
$Ø_{10}$ (rad)	0.2047383	-0.116328825	-0.1848926
k_2 (N/m)	23157.98	19,614.97714	18,493.25784
l_{02} (m)	0.299869	0.332661699	0.353795773
l_{x2} (m)	-0.01765858	0.010128801	0.009826437
l_{y2} (m)	0.08741526	0.115916625	0.127464335
$Ø_{20}$	3.331267	3.071067042	3.015784678
e_1 (m)	0.1	0.124999991	0.124999999
e_2 (m)	0.1	0.075000054	0.075000001
fav (N)	36.68841365	16.52231619	16.36783591

Table 4 Effects of new variable in the objective function for case 2

Variables	Without considering e_1 and e_2 as variables (Segla 1998)	With considering e_1 and e_2 as variables using DE (Saravanan 2008)	With considering e_1 and e_2 as variables using TLBO
k_1 (N/m)	77,470.14	74,815.38456	73,248.47826
l_{01} (m)	0.325919	0.294678383	0.302748823
l_{x1} (m)	0.01328951	0.000632288	- 0.000016753
l_{y1} (m)	0.17534	0.145051132	0.136895461
$\emptyset_{10}$ (rad)	-0.07568362	-0.006002453	-0.003216754
k_2 (N/m)	6885.527	62,203.58799	68,709.53797
l_{02} (m)	0.1778149	0.104730401	0.100045744
l_{x2} (m)	0.001941306	0.022781915	0.024994635
l_{y2} (m)	0.02669327	0.026304254	0.026218378
$\emptyset_{20}$	3.143471	3.460829819	3.480167922
e_1 (m)	0.1	0.125	0.124999998
e_2 (m)	0.1	0.075000025	0.075
fav (N)	35.56356433	16.48118857	16.32845249

Table 5 Effects of new variable in the objective function for case 3

Variables	Without considering e_1 and e_2 as variables (Segla 1998)	With considering e_1 and e_2 as variables using DE (Saravanan 2008)	With considering e_1 and e_2 as variables using TLBO
k_1 (N/m)	60,332.34	71,147.47273	78,753.28535
l_{01} (m)	0.3582281	0.289520497	0.283778434
l_{x1} (m)	0.07142384	0.015650483	0.012679357
l_{y1} (m)	0.1961656	0.138887075	0.115673479
$\emptyset_{10}$ (rad)	-0.3489188	-0.113532087	-0.173911432
k_2 (N/m)	9764.061	8548.302991	8264.624757
l_{02} (m)	0.331819	0.373253457	0.384256705
l_{x2} (m)	0.004234375	0.022364905	0.030167546
l_{y2} (m)	0.09313436	0.056932774	0.048356728
$\emptyset_{20}$	3.061444	2.792500368	2.717442461
e_1 (m)	0.1	0.125	0.124999978
e_2 (m)	0.1	0.125	0.124999997
fav (N)	42.40874757	11.9821382	11.61893467

4. Results and discussion

To check the effectiveness of the TLBO algorithm extensive computational experiments are conducted on static balancing of robot mechanism considered by Saravanan *et al.* (2008) and results are compared with other optimization algorithms. Saravanan *et al.* (2008) used optimization methods using 10000 function evaluations. Hence to make fair comparison of results, the same number of function evaluation is considered. Hence, Population size of 50 and maximum number of generations of 100 are considered (it may be mentioned here that the number of function evaluations in TLBO algorithm=2×population size×number of generations). Like other optimization algorithms (e.g., PSO, ABC, ACO, etc.), TLBO algorithm also has not any special mechanism to handle the constraints. So, for the constrained optimization problems it is necessary to incorporate any constraint handling techniques with the TLBO algorithm. In the present experiments, Deb's heuristic constrained handling method (Deb 2000) is used to handle the constraints with the TLBO algorithm. The TLBO code is written in MATLAB and implemented on a laptop having Intel core i3 2.53 GHz processor with 1.85 GB RAM.

The optimum static balancing of robot mechanism is evaluated using TLBO algorithm for three cases. Three different values of d (length determining C. G. of DT7) are considered as three cases. In the first case the value of d is 0.2225 m (balancing of mechanism considering the payload 10 kg). For the second case the value of d is 0. It means the center of gravity T7 is at joint D. For the third case the value of m7 is 87.43 kg (without the payload of 10 kg) and therefore the value of d is 0.12236 m.

The Table 1 provides the search intervals for different variables. The optimum objective function (fav) for all the three cases using different optimization methods are given in Table 2. From Table 2, it can be seen that the optimum value of the objective function is 16.36783591 for case1 when d value is 0.2225 which is better than the value obtained by other optimization methods like Newton's method, Conjugate gradient method, GA, NSGA-II and DE. Also, it can be observed from the Table 2 for case 1 the function value is improved by 61.22%, 61.11%, 2.81%, 2.50% and 0.934% using TLBO algorithm as compared to the values obtained by Newton's method, Conjugate gradient method, GA, NSGA-II and DE respectively.

Similarly, for case 2, the TLBO algorithm performs better and provides best function value among the six optimization methods considered for this problem. The best function value gained is 16.32845249 for case 2 using TLBO algorithm when d value is 0. Also, it can be observed from the Table 2 for case 2 the function value is improved by 33.21%, 33.82%, 6.74%, 2.24% and 0.934% using TLBO algorithm as compared to the values obtained by the Newton's method, Conjugate gradient method, GA, NSGA-II and DE respectively.

For the case 3, the best optimum function value is 11.61893467 obtained using the TLBO algorithm which is much better than the value obtained by conventional optimization methods. Also, it can be observed from the Table 2 for case 2 the function value is improved by 124.37%, 92.50%, 4.85%, 3.44% and 3.18% using TLBO algorithm as compared to the values obtained by Newton's method, Conjugate gradient method, GA, NSGA-II and DE respectively.

The best result is gained for the third case when the value of d is 0.12236. It can be observed from Table 2 that TLBO gives better results than the other five optimization methods considered in all the three cases.

Table 3 presents the effect of two new variables e_1 and e_2 (distance between the points z and T, z and W, respectively) in the objective function for case 1. Table 4 presents the effect of two new variables e_1 and e_2 (distance between the points z and T, z and W, respectively) in the objective function for case 2. Table 5 presents the effect of two new variables e_1 and e_2 (distance between the points z and T, z and W, respectively) in the objective function for case 3. In this work, improved Segla model with 12 variables proposed by Saravanan *et al.* (2008) is considered. From Tables 3-5 it can concluded that the best results are obtained when two new variables e_1 and e_2 are included in the original Segla model.

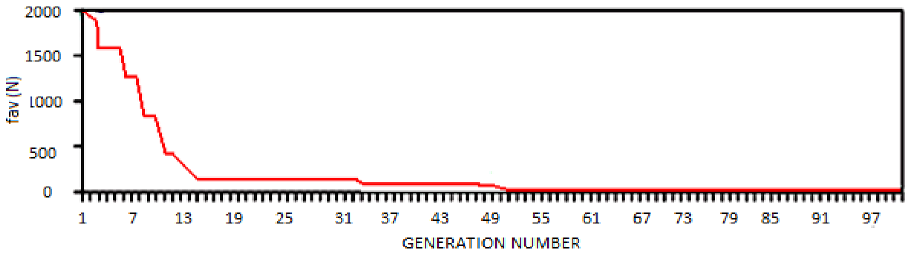

Fig. 6 Convergence plot of the TLBO algorithm for case 1

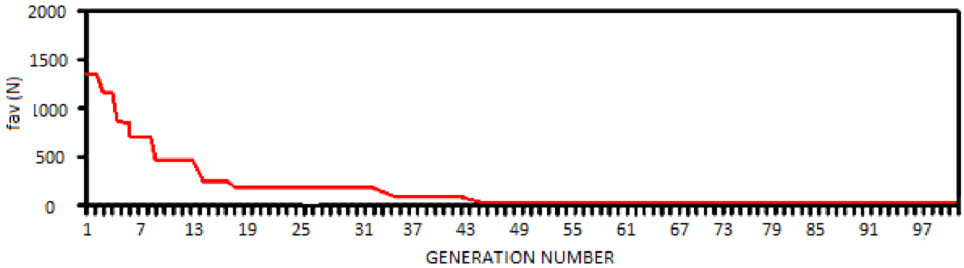

Fig. 7 Convergence plot of the TLBO algorithm for case 2

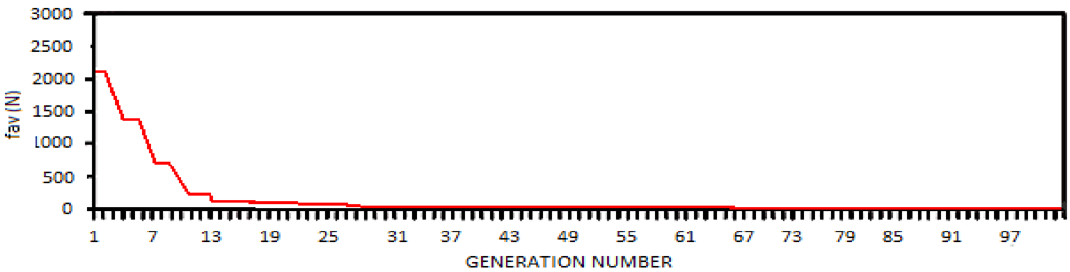

Fig. 8 Convergence plot of the TLBO algorithm for case 3

Figs. 6-8 present the convergence process using TLBO algorithm for case 1, case 2 and case3 respectively. From Figs. 6-8 it can be observed that the convergence pattern is smooth and are in lower region. Also, it can be seen that TLBO gives lowest value in comparison with other optimization methods.

5. Conclusions

In this work, the performance of the proposed TLBO algorithm is checked for static balancing of a robot manipulator. Three cases are considered to verify the efficiency and accuracy of the proposed method. The results of TLBO algorithm are compared with conventional and evolutionary optimization methods such as NM, CGM, GA, NSGA-II and DE. For case 1, the optimum value of the average force on the gripper in the working area is obtained as 16.36783591 using the TLBO algorithm. The average force on the gripper is improved by 61.22%, 61.11%, 2.81%, 2.50% and 0.934% using the TLBO algorithm as compared to the values obtained by NM, CGM, GA, NSGA-II and DE algorithms respectively. For case 2, the optimum value of the average force on the gripper in the working area is 16.32845249 obtained using the TLBO algorithm. The average force on the gripper is improved by 33.21%, 33.82%, 6.74%, 2.24% and 0.934% using the TLBO algorithm as compared to the values obtained by NM, CGM, GA, NSGA-II and DE respectively. For case 3, the optimum value of the average force on the gripper in the working area is 11.61893467 obtained using the TLBO algorithm. The average force on the gripper is improved by 124.37%, 92.50%, 4.85%, 3.44% and 3.18% using the TLBO algorithm as compared to the values obtained by NM, CGM, GA, NSGA-II and DE respectively. The computational results showed that for all the three cases the TLBO algorithm has obtained comparatively more accurate solutions than those obtained by using other optimization methods. Therefore, it can be stated that the TLBO algorithm is effective and has a potential for solving static balancing of robot manipulator problem. The present work will be extended in the near future to solve other design problems related to the robot manipulator.

References

Basu, M. (2014), “Teaching-learning-based optimization algorithm for multi-area economic dispatch”, *Energy*, **68**, 21-28.

Baykasoğlu, A., Hamzadayi, A. and Köse, S.Y. (2014), “Testing the performance of teaching-learning based optimization (TLBO) algorithm on combinatorial problems: Flow shop and job shop scheduling cases”, *Inform. Sci.*, **276**, 204-218.

Camp, V. and Farshchin, M. (2014), “Design of space trusses using modified teaching-learning-based optimization”, *Eng. Struct.*, **62-63**, 87-97.

Deb, K. (2000), “An efficient constraint handling method for genetic algorithm”, *Comput. Meth. Appl. Mech. Eng.*, **186**(2-4), 311-338.

Filaretov, V.F. and Vukobratović, M.K. (1993), “Static balancing and dynamic decoupling of the motion of manipulation robots”, *Mechatronics*, **3**(6), 767-782.

Hassan, A. and Abomoharam, M. (2017), “Modeling and design optimization of a robot gripper mechanism”, *Robot. Comput. Integrat. Manufact.*, **46**, 94-103.

Martini, A., Troncossi, M., Carricato, M. and Rivola, A. (2015), “Static balancing of a parallel kinematics machine with linear delta architecture: Theory, design and numerical investigation”, *Mech. Machine Theor.*, **90**, 128-141.

Medina, M.A., Das, S., Coello, C.A.C. and Ramirez, J.M. (2014), “Decomposition-based modern metaheuristic algorithms for multi-objective optimal power flow-a comparative study”, *Eng. Appl. Artif. Intell.*, **32**, 10-20.

Moghadam, A. and Seifi, A.R. (2014), “Fuzzy-TLBO optimal reactive power control variables planning for energy loss minimization”, *Energy Convers. Manage.*, **77**, 208-215.

Niknam, T., Abarghooee, R.A. and Narimani, M.R. (2012), “A new multi objective optimization approach based on TLBO for location of automatic voltage regulators in distribution systems”, *Eng. Appl. Artif. Intell.*, **25**, 1577-1588.

Ouyang, P.R., Li, Q., and Zhang, W.J. (2003), “Integrated design of robotic mechanisms for force balancing and trajectory tracking”, *Mechatronics*, **13**(8-9), 887-905.

Ouyang, P.R. and Zhang, W.J. (2004), “Force balancing of robotic mechanisms based on adjustment of kinematic parameters”, *J. Mech. Des.*, **127**(3), 433-440.

Petković, D., Issa, M., Pavlović, N.D., Zentner, L. and Ćojbašić, Ž. (2012), “Adaptive neuro fuzzy controller for adaptive compliant robotic gripper”, *Expert Syst. Appl.*, **39**(18), 13295-13304.

Petković, D., Pavlović, N.D., Ćojbašić, Ž. and Pavlović, N.T. (2013a), “Adaptive neuro fuzzy estimation of underactuated robotic gripper contact forces", *Expert Syst. Appl.*, **40**(1), 281-286.

Petković, D., Issa, M., Pavlović, N.D. and Zentner, L. (2013b), "Intelligent rotational direction control of passive robotic joint with embedded sensors", *Expert Syst. Appl.*, **40**(4), 1265-1273.

Petković, D., Issa, M., Pavlović, N.D. and Zentner L. (2013c), "Design of compliant robotic joint with embedded-sensing elements of conductive silicone rubber", *Industr. Robot*, **40**(2), 143-157.

Petković, D., Issa, M., Pavlović, N.D. and Zentner, L. (2013d), "Application of the TRIZ creativity enhancement approach to design of passively compliant robotic joint", *J. Adv. Manufact. Technol.*, **67**(1), 865-875.

Petković, D., Issa, M., Pavlović, N.D. and Zentner, L. (2013e), "Application of the TRIZ creativity enhancement approach to the design of a passively adaptive compliant robotic gripper", *Assembly Autom.*, **33**(3), 231-239.

Pierezan, J., Freire, R.Z., Weihmann, L., Reynoso-Meza, G. and dos Santos Coelho, L. (2017), "Static force capability optimization of humanoids robots based on modified self-adaptive differential evolution", *Comput. Oper. Res.*, **84**, 205-215.

Quaglia, G. and Yin, Z. (2015), "Static balancing of planer articulated robots", *Front. Mech. Eng.*, **10**(4), 326-343.

Rao, R.V., Savsani, V.J. and Vakharia, D.P. (2011), "Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems", *Comput. Aid. Des.*, **43**(3), 303-315.

Rao, R.V., Savsani, V.J. and Balic, J. (2012a), "Teaching-learning-based optimization algorithm for unconstrained and constrained real parameter optimization problems", *Eng. Optim.*, **44**(12), 1447-1462.

Rao, R.V., Savsani, V.J. and Vakharia, D.P. (2012b), "Teaching-learning-based optimization: A novel optimization method for continuous non-linear large scale problems", *Inform. Sci.*, **183**(1), 1-15.

Rao, R.V. and Savsani, V.J. (2012), *Mechanical Design Optimization Using Advanced Optimization Techniques*, Springer-Verlag, London, U.K.

Rao, R.V. and Patel, V. (2012), "An elitist teaching-learning-based optimization algorithm for solving complex constrained optimization problems", *J. Ind. Eng. Comput.*, **3**(4), 535-560.

Rao, R.V. and Patel, V. (2013), "Comparative performance of an elitist teaching-learning-based optimization algorithm for solving unconstrained optimization problems", *J. Ind. Eng. Comput.*, **4**(1), 29-50.

Rao, R.V. (2016), *Teaching Learning Based Optimization Algorithm*, in *Innovative Computational Intelligence: A Rough Guide to 134 Clever Algorithms*, Springer International Publishing, Cham, Switzerland.

Rao, R.V. and Waghmare, G.G. (2014), "A comparative study of a teaching-learning-based optimization algorithm on multiobjective unconstrained and constrained functions", *J. King Saud Univ. Comput. Inform. Sci.*, **26**(3), 332-346.

Rao, R.V., Kalyankar, V.D. and Waghmare, G. (2014), "Parameters optimization of selected casting processes using teaching-learning based optimization algorithm", *Appl. Math. Modell.*, **38**(23), 5592-5608.

Russo, A., Sinatra, R., and Xi, F. (2005), "Static balancing of parallel robots", *Mech. Machine Theor.*, **40**(2), 191-202.

Saravanan, R., Ramabalan, S. and Babu, P.D. (2008), "Optimum static balancing of an industrial robot mechanism", *Eng. Appl. Artif. Intell.*, **21**(6), 824-834.

Sangamesh, R.D. and Ananthasuresh, G.K. (2012), "Perfect static balance of linkages by addition of springs but not auxiliary bodies", *J. Mech. Robot.*, **4**(2), 1-12.

Satapathy, S.C. and Naik, A. (2014), "Modified teaching-learning-based optimization algorithm for global numerical optimization- A comparative study", *Swarm Evol. Comput.*, **16**, 28-37.

Segla, S. (1998), "Statical balancing of a robot mechanism with the aid of a genetic algorithm", *Mech. Machine Theor.*, **33**(1-2), 163-174.

Simionescu, I. and Ciupitu, L. (2000a), "The static balancing of the industrial robot arms: Part II: Continuous balancing", *Mech. Machine Theor.*, **35**(9), 1299-1311.

Simionescu, I. and Ciupitu, L. (2000b), "The static balancing of the industrial robot arms: Part I: Discrete balancing", *Mech. Machine Theor.*, **35**(9), 1287-1298.

Sultana, S. and Roy, P.K. (2014), “Optimal capacitor placement in radial distribution systems using teaching learning based optimization”, *J. Elec. Power Energy Syst.*, **54**, 387-398.
Zou, F., Wang, L., Hei, X., Chen, D. and Yang, D. (2014), “Teaching-learning-based optimization with dynamic group strategy for global optimization”, *Inform. Sci.*, **273**, 112-131.

CC

Appendix: Demonstration of the working of TLBO algorithm

A standard benchmark function of Sphere is considered for demonstration of working of the TLBO algorithm. The objective function is to find out the values of x_i that minimize the value of Sphere function.

Function: Minimize

$$f(x_i) = \sum_{i=1}^{D} x_i^2$$

Variables range: $-100 \leq x_i \leq 100$

The known solution to this benchmark function is 0 for all x_i=0. Now to demonstrate the TLBO algorithm, let us assume a population size of 5 (i.e., number of learners), two design variables x_1 and x_2 (i.e., number of subjects) and one iteration as the termination criterion. The initial population is randomly generated within the ranges of the variables and the corresponding values of the objective function are shown in Table A1.

Table A1 Initial population

	x_1	x_2	Value of objective function	
	-55	36	4321	
	0	41	1681	
	96	-86	16612	
	-64	31	5057	
	-18	-27	1053	Teacher
Mean	-8.2	-1		

Table A2 Teacher phase

x_1	x_2	Value of objective function
-60.684	23.26	4224
-5.684	28.26	830.9
90.32	-98.74	17907
-69.68	18.26	5189
-23.68	-39.74	2140

The mean values of $x_1$and $x_2$are also shown in Table A1. As it is a minimization function, the lowest value of $f(x)$ is considered as the best learner (and is considered as equivalent to teacher). Now the teacher tries to improve the mean result of the class. Assuming random numbers r_1= 0.58 for $x_1$and r_2= 0.49 for x_2, and T_f= 1, the difference mean values for x_1 and x_2 are calculated as,

difference_mean(x_1) = 0.58*(-18-(-8.2))=-5.684

difference_mean(x_2) = 0.49*(-27-(-1))=-12.74

The value of difference_mean(x_1) is added to all the values under the x_1 column and the value of difference_mean(x_2) is added to all the values under the $x_2$column of Table A1. Table A2 shows the new values of x_1 and x_2 and the corresponding values of the objective function. Now, the values of $f(x)$ of Tables A1 and A2 are compared and the best values of $f(x)$ are considered and placed in Table A3. This completes the teacher phase of the TLBO algorithm.

Table A3 Updated values of the variables and the objective function (teacher phase)

x_1	x_2	fitness
-60.684	23.26	4224
-5.684	28.26	830.9
96	-86	16612
-64	31	5057
-18	-27	1053

Table A4 New values of the variables and the objective function (learner phase)

x_1	x_2	Value of objective function	Interaction
-16.134	27.86	1036	1&2
41.55	25.74	2389	2&4
3.66	-31.72	1020	3&5
-61.31	23.88	4330	4&1
-110.3	27.28	12919	5&3

Now, the learner phase starts and any student can interact with any other student for knowledge transfer. This interaction can be done in a random manner. In this example, interactions between learners 1 and 2, 2 and 4, 3 and 5, 4 and 1, and 5 and 3 are considered. It is to be noted that every learner has to interact with any other learner. That is why, in this example, 5 interactions are considered (i.e., one interaction for each learner). Table A4 shows the new values of x_1 and x_2 for the learners after the interactions and considering random numbers r_1=0.81 for x_1 and r_2=0.92 for x_2. For example, the new values of x_1 and x_2 for learner 1 are calculated as explained below. As it is a minimization function, the value of $f(x)$ is better for learner 1 as compared to that of learner 2 and hence the knowledge transfer is from learner 1 to learner 2. Hence the new values of $x_1$and $x_2$for learner 1 are calculated as

(x_1)new for learner 1=-60.684+0.81(-5.684-(-60.684)) =-16.134

(x_2)new for learner 1=23.26+0.92(28.26-23.26) =27.86

Now, the values of $f(x)$ of Tables A3 and Table A4 are compared and the best values of $f(x)$ are considered and placed in Table A5.

Table A5 Updated values of the variables and the objective function (learner phase)

x_1	x_2	Value of objective function
-16.13	27.86	1036
-5.684	28.26	830.9
3.66	-31.72	1020
-61.31	23.88	4330
-18	-27	1053

This completes the learner phase and one iteration of the TLBO algorithm. The value of $f(x)$ is reduced from 1053 at the beginning of iteration to 830.9 at the end of first iteration. Increasing the number of iterations will soon find the value of $f(x)$ reaching the known solution of 0 for x_1=0 and x_2=0.