Proceedings of the Canadian Society for Mechanical Engineering International Congress
32nd Annual Conference of the Computational Fluid Dynamics Society of Canada
Canadian Society of Rheology Symposium
CSME-CFDSC-CSR 2025
May 25–28, 2025, Montréal, Québec, Canada

Multi-objective optimization of a robotic trajectory problem using NS-TLBO

Poonam Savsani[1], Vimal Savsani[2]
[1]Department of mechanical engineering, Cambrian college, Sudbury, Canada
[2]Department of mechanical engineering, Canadore college, North Bay, Canada

Abstract— This study introduces Non-dominated Teaching-Learning Based Optimization (NS-TLBO), a multi-objective extension of the Teaching-Learning Based Optimization (TLBO) algorithm. NS-TLBO enhances the fundamental TLBO algorithm by integrating concepts from NSGA-II, such as crowding distance and non-dominated sorting. The algorithm's performance is evaluated by comparing it to the standard NSGA-II in a multi-objective optimization problem involving robotic arm trajectory planning. The results highlight the better effectiveness of NS-TLBO over NSGA-II in this specific optimization scenario.

Keywords- Multi-objective optimization, Teaching learning based optimization (TLBO), Trajectory planning of a robotic arm. Non- dominated sorting algorithm

I. INTRODUCTION

Multi-objective optimization is used in many fields, such as science, engineering, economics, and logistics, where decisions must balance trade-offs between two or more conflicting goals. Many real-world problems involve multiple competing objectives, making multi-objective optimization (MOO) an important area of research. Over the past decade, many algorithms have been developed to solve these challenges.

In MOO problems, there is no single best solution. Instead, a set of solutions must be found by balancing different objectives. A multi-objective optimization problem (MOOP) can be defined as minimizing (or maximizing) a function $F(X)$, where $F(X) = \{f_1(X), \dots, f_n(X)\}$ represents different objectives, and X is a decision variable. The goal is to find a set of decision variables that provide the best trade-off among the objectives [1]. Since these objectives often compete, MOO results in multiple solutions rather than just one. The best solution set is called the non-dominated set, meaning no other solution is better across all objectives. This set forms the Pareto-optimal set, creating a surface in the objective space known as the Pareto front [2].

Research on MOO has classified optimization methods into different categories, including aggregative, lexicographic, sub-population, Pareto-based, and hybrid approaches. Among these, Pareto-based methods are the most widely used, as they select solutions based on the concept of Pareto dominance. Many algorithms have been proposed to solve multi-objective problems, with evolutionary algorithms being among the most popular. Some well-known evolutionary MOO algorithms include Archive-based Micro Genetic Algorithm (AMGA), NSGA-II, Multi-Objective Evolutionary Algorithm based on Decomposition (MOEAD), Multi-Objective Particle Swarm Optimization (MOPSO), Multi-Objective Bee Swarm Optimization, and the Multi-Objective Artificial Bee Colony Algorithm [3-8].

These algorithms perform well for many MOO problems. However, they have limitations when solving complex engineering optimization tasks, mainly due to the difficulty of fine-tuning their control parameters [10]. Therefore, ongoing research is needed to develop improved MOO algorithms. One such promising approach is the Teaching-Learning Based Optimization (TLBO) algorithm [11,12]. Unlike many other algorithms, TLBO does not require tuning control parameters (except for population size and the number of generations), making it simpler and more efficient.

Optimal path planning is crucial for efficiently controlling robotic manipulators. While optimization methods for robot control have been studied for decades, recent advances in computing power and numerical techniques have made it possible to apply optimal control to many real-world tasks. In robotics, optimization helps improve motion efficiency, especially in tasks requiring autonomous or adaptive path planning. Even small improvements in efficiency can significantly impact performance.

The first step in robot trajectory planning is offline path planning, where various methods are used to calculate an optimal trajectory while considering physical system constraints. For a robotic arm, the desired path is defined by the position and orientation of the tool being used [9]. The system's dynamics can be modeled using methods like the Lagrange-Euler method. The robot’s motion is then expressed in terms of joint coordinates and interpolated using standard techniques. Since motion planning is complex, the trajectory must be parameterized, scaled, and analyzed using derivatives. A cost function is then optimized while ensuring the solution meets kinematic constraints.

In this work, we propose a multi-objective variant of the TLBO algorithm and apply it to a robotic arm optimization problem. This new version of TLBO aims to improve performance in multi-objective optimization tasks.

II. NON-DOMINATED SORTING TLBO (NS-TLBO)

In this work, a multi-objective variant of the Teaching-Learning Based Optimization (TLBO) algorithm, called Non-dominated TLBO (NS-TLBO), is presented. The procedure for NS-TLBO is explained below:

1. A set of solutions is initialized for all design variables within their minimum and maximum limits.
2. These solutions are sorted into different fronts based on non-domination. The first front consists of entirely non-dominated solutions, while the second front contains solutions dominated only by those in the first front, and so on.
3. Each individual solution in every front is assigned a rank value. Solutions in the first front receive a rank of 1, those in the second front are assigned a rank of 2, and so forth.
4. Crowding distance is calculated for each solution. This measures how close a solution is to its neighbors. A larger average crowding distance helps maintain diversity in the solution set.
5. Solutions are updated using the teacher phase and learner phase of the TLBO algorithm.

Teacher Phase loop

$For\ i = 1{:}\ Solution\ size,$

$X_{new,i} = X_{old,i} + Difference_{Mean_i}$

End

Where, $Difference_{Mea\ \ i} = r_i(M_{new} - T_F M_i)$

T_F is a teaching factor that decides the value of the mean to be changed, and r_i is a random number in the range [0, 1]. The value of T_F can be either 1 or 2 which is a heuristic step and so it is decided randomly with equal probability as

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

Learner Phase loop

$For\ i = 1{:}\ Solution\ size$

$Randomly\ choose\ another\ learner\ Xj, such\ that\ i \neq j$

$If\ f(Xi) < f(Xj), \quad Xnew, i = Xold, i + ri(Xi - Xj)$

$Else, \quad Xnew, i = Xold, i + ri(Xj - Xi)$

$End.$

The current solution and updated solutions are merged and sorted again based on non-domination and only the best N solutions are selected, where N is the solution size. The selection is based on rank and the crowding distance on the last front.

III. PROBLEM FORMULATION OF A ROBOTIC ARM

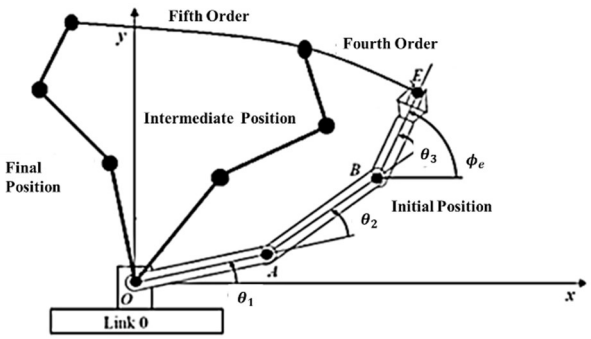


Figure 1. Intermediate position on planned trajectory

In this paper, a three-degree-of-freedom planar robotic arm is considered, as described in [13,14] and shown in Fig. 1. The end effector is required to move from an initial point to a final point within a workspace. This workspace may be free of obstacles or may contain obstacles, as shown in Fig. 1. For the problem under consideration, the complete trajectory is divided into two parts, resulting in one intermediate position between the initial and final positions of the robotic arm.

Objective functions, constrains and design variables are rewritten from [13]. Three objective functions are considered; total joint travelling distance (f_{joint}), total Cartesian trajectory length ($f_{clength}$) and the total time consumed (f_{time}) from the initial to the final position, simultaneously.

$$f_{joint} = \sum_{i=1}^{a} \sum_{j=2}^{b} |\theta_{ij} - \theta_{ij-1}|$$

$$f_{clength} = \sum_{j=2}^{b} d\big((x,y)_j, (x,y)_{j-1}\big)$$

$$f_{time} = T_{i,i+1} + T_{i+1,f}$$

All the joints are assigned a particular value of maximum torque (torque supplied by motor), which should not be exceeded. This leads to three constraints for three joints as given by Moreover, the constraints are imposed to ensure the robotic arm attains the final position. This leads to five constraints.

$g_i(X)$: $Tor_i \leq Tor_{imax}$ Where, $i = 1,2,3$ represents three joints,

$$g_4(X) : X_{calculated} = X_{final}$$

$$g_5(X) : Y_{calculated} = Y_{final}$$

The above optimization formulation requires 9 design variables given as under.

$$\{\theta_{in}\ , \theta_{int}\ , \theta_{int\,3}, \varphi, \dot{\theta}_{int\,1}, \dot{\theta}_{int\,2}, \dot{\theta}_{int\,3}, T_1, T_2\}$$

Where, $\theta_{int1}\ to\ \theta_{int3}$ represents intermediate joint angles, ϕ represents end effector angle for the final position, $\dot{\theta}_{int\,1\ to\ int\,3}$ represents intermediate velocity and $T_1, and\ T_2$ represents time

from initial to intermediate and from intermediate to final position respectively. Upper and lower limits for the design variable are considered such that it can cover whole work space generated by the robotic arm and it is given by:

$-\pi \leq \theta_{int\,i,(1,2,3,)} \leq +\pi, -\frac{\pi}{4} \leq \theta_{int\,i,(1,2,3)} \leq +\frac{\pi}{4}$,

$0.1 \leq T_{1,2} \leq 8, -\pi \leq \phi \leq +\pi$

The optimization problem considers both Task (Cartesian) space and Joint space. The Task space (Cartesian) is focused on the movement and trajectory of the end effector in a physical workspace, while the Joint space is involved in the optimization of joint angles and movements that enable the robotic arm to reach the desired positions in task space. Therefore, the problem spans both Cartesian (Task) space and Joint space.The procedure starts with the initialization of the robot and optimization algorithm parameters like population size, number of generations, number of design variables and bound on the design variables. The initial and final position is required to take as input, which is further checked for the feasibility as per the link lengths. If the position is not feasible as per the link lengths, it is required to change the starting or the final point of the robot motion. After defining the starting point, the joint angles are obtained using inverse kinematics. The redundancy in the solution is removed by considering the angle for the second link as positive for all the solutions. The optimization algorithm will start with the initial set of solutions (population), which produces intermediate positions equals to the population size. The coordinate of the end effector for the intermediate position is obtained using forward kinematics from the values of intermediate angles. Final configuration of the robotic arm is obtained from the value of Ø, which is checked for the required final configuration. Penalty is added to the objective function f, if final position is not attained. Tor_i is calculated using time T_1 and T_2, which is checked for the Tor_{imax} and penalty is added in objective function if torque constrains are violated. Objective function is calculated considering free workspace and workspace with obstacle. The procedure is repeated till the termination criterion is reached.

The robotic trajectory optimization problem is solved using NS-TLBO and NSGA-II. The investigation is done using a population size of 50 and number of function evaluations as 50000. Both the algorithms are run for 25 independent runs. As TLBO is an algorithm parameter free algorithm it does not require any algorithm-parameter for its working, which the parameter considered for NSGA-II are; crossover probability=0.8, mutation probability= 0.01, selection=tournament selection. Four different combinations are selected for the analysis as listed below:

a) Considering $f_{clength}(X)\ and\ f_{time}(X)$

b) Considering $f_{joint}(X)\ and\ f_{time}(X)$

c) Considering $f_{cleng}\ \ (X)\ and\ f_{joint}(X)$

d) Considering all the objective functions simultaneously.

IV. Performance Of Proposed Methods

The results obtained using NS-TLBO and NSGA-II for case (a) are shown in Fig. 2, which illustrates the final Pareto front after the specified number of function evaluations. It can be noted that for case (a), both algorithms, NS-TLBO and NSGA-II, exhibit nearly identical performance. However, it is also observed that a range of solutions are available for optimizing the length of the robotic arm, with a trade-off in time. The time range specified for this problem is up to 8 seconds from the starting to the intermediate position and 8 seconds from the intermediate to the final position, totaling 16 seconds. Interestingly, the results show that the maximum time required to achieve a range of optimized lengths is about 2.7 seconds, while the minimum time is around 2 seconds for the entire robotic arm travel from the initial to the final position. The result for optimizing joint angles and time simultaneously is given in Fig. 3, where NS-TLBO outperforms NSGA-II. The Pareto front produced by NS-TLBO contains more non-dominating solutions, indicating better performance. The variation of optimized time with respect to optimized joint travel distance is minimal, with time varying between 2 and 2.2 seconds, and joint travel variation being between 1.865 and 1.872 radians. The non-dominating solutions for optimized joint travel and optimized Cartesian length are shown in Fig. 4, where NS-TLBO provides more effective results than NSGA-II. Interestingly, the Pareto front produced by both algorithms is discrete. There is high sensitivity in joint travel when the Cartesian length varies from 3 to 3.3, which decreases as the length decreases from 2.8. Furthermore, the maximum possible Cartesian length for the given initial and final positions is around 3.3 units, and the maximum joint travel is about 4.9 radians, regardless of optimized time. It is essential to check the sensitivity of design variables across all optimal solutions in the Pareto front, as this information is crucial for design engineers to determine the practical limits of design variables for a specific robot manipulator task. Fig. 5 illustrates the variation of design variables for $f_{clength}\ and\ f_{time}$. It is observed that for the given initial and final positions, the useful range of θ1 (θ_1) is between 1.70 and 1.82 radians, θ_2 varies between 1.40 and 1.56 radians, and θ_3 ranges from -2.80 to -3.15 radians. The angle Ø is less sensitive and shows minimal variation. The variation of $velocity_1\ and\ velocity_2$ is near the bound points, with very little variation, while velocity-3 is at the lower bounds. $Time_1$ varies between 1 and 1.8 seconds, and $Time_2$ is less sensitive, remaining around 1. For the multi-objective optimization of $f_{joint}\ and\ f_{time}$, as shown in Fig. 6, $\theta_1, \theta_2, and\ \theta_3$ exhibit minimal sensitivity, varying around 1.84, 1.07, and 0.67 radians, respectively. The value of Ø remains nearly constant for all optimal solutions at around 3.14 radians. $Velocity_1\ and\ velocity_2$ are almost at the upper bounds, while $velocity_3$ is near 0. $Time_1\ and\ Time_2$ are also minimally sensitive and remain around 1. For the Pareto front of

f_{joint} and $f_{clength}$, as shown in Fig. 7, significant variations are observed in the limits of nearly all the design variables. The variation in $Time_1$ and $Time_2$ is considerably more when these are not treated as objective functions, as seen in earlier cases. It is crucial to consider all three objective functions simultaneously and examine the variations of all design variables. The Pareto front obtained using NS-TLBO in Matlab is shown in Fig. 8, where the variation of $f_{clength}$ is between 2 and 4, f_{joint} varies between 1 and 6, and f_{time} ranges from 1 to 6. Notably, these values are higher than those obtained when only two objective functions are considered. The variations of all design variables for the simultaneous optimization of all three functions are shown in Fig. 9. It is observed that $\theta_3, \Phi, velocity_1, and\ velocity_2$ utilize nearly the entire half-range of the specified range, while $velocity_3$ uses almost the full range. The range of $time_1$ and $time_2$ also increases when all three objectives are considered simultaneously. In conclusion, NS-TLBO proves to be a more effective algorithm than NSGA-II for the considered robotic trajectory optimization problems

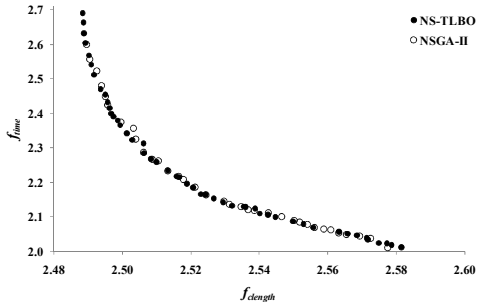


Figure. 2 Pareto front obtained by using NS-TLBO and NSGA-II for f_{time} and f_{cleng}

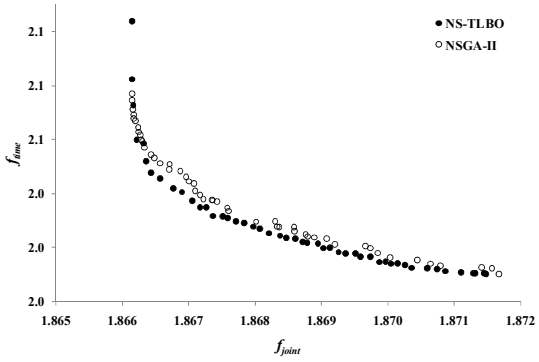


Figure. 3 Pareto front obtained by using NS-TLBO and NSGA-II for f_{time} and f_{joint}

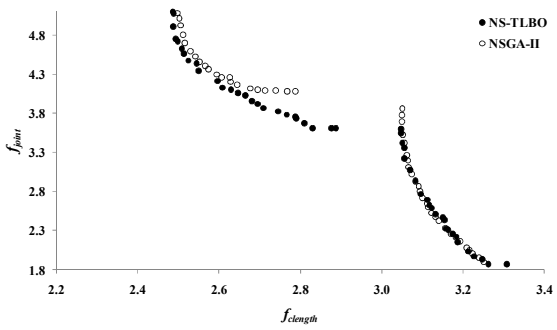


Figure. 4 Pareto front obtained by using NS-TLBO and NSGA-II for f_{joint} and $f_{clength}$

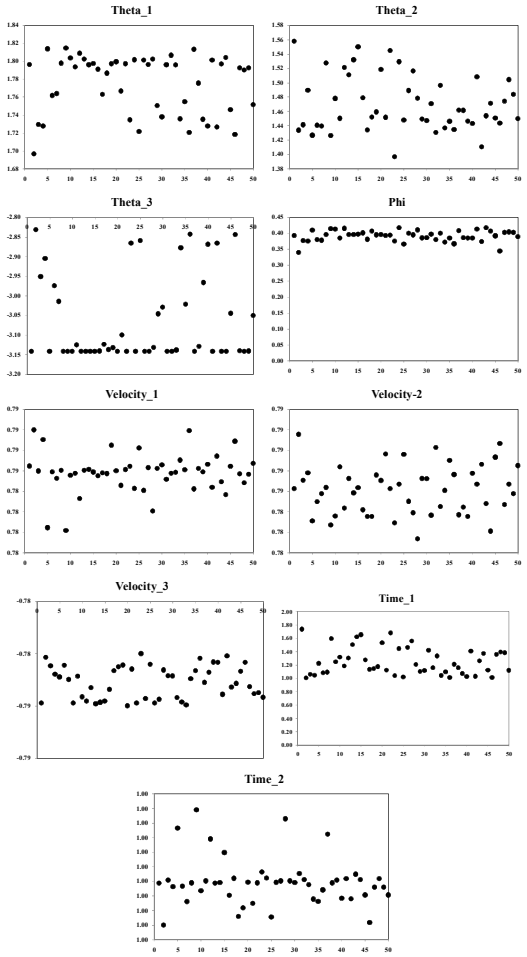


Figure. 5 Variations of design variables for the Pareto front solutions of f_{time} and f_{cleng}

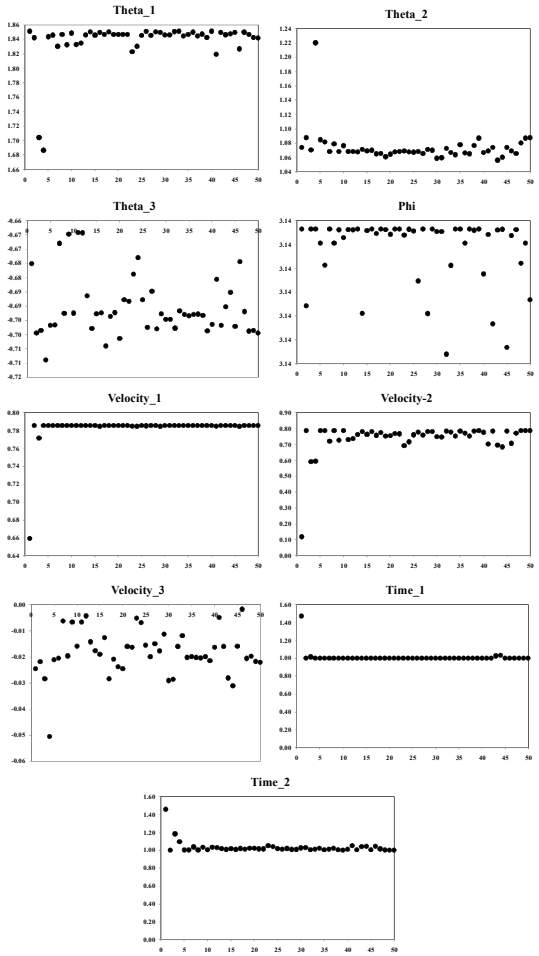
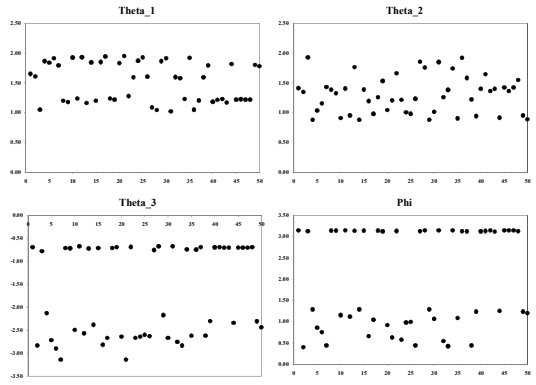


Figure. 6 Variations of design variables for the Pareto front solutions of f_{time} and f_{joint}



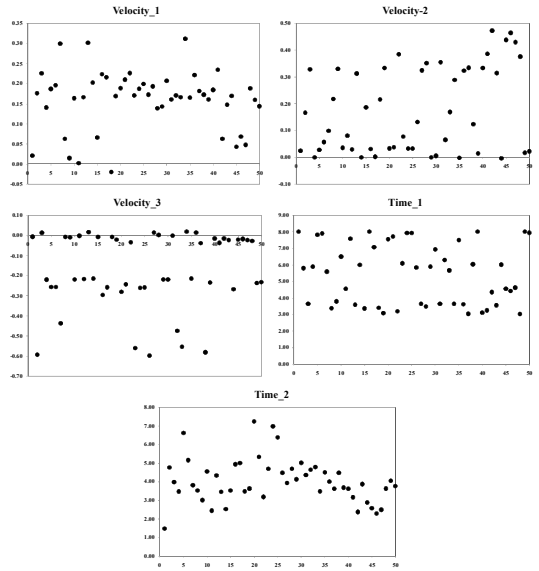


Figure. 7 Variations of design variables for the Pareto front solutions of f_{joint} and f_{cleng}

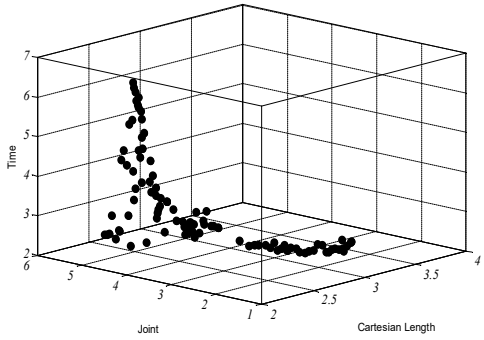
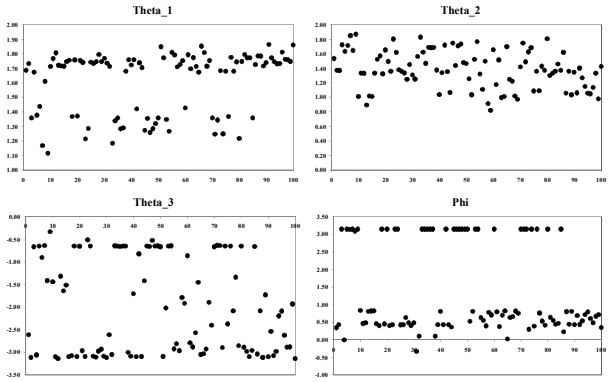


Figure. 8 Pareto front obtained by using NS-TLBO for ftime, f_{joint} and f_{cleng}



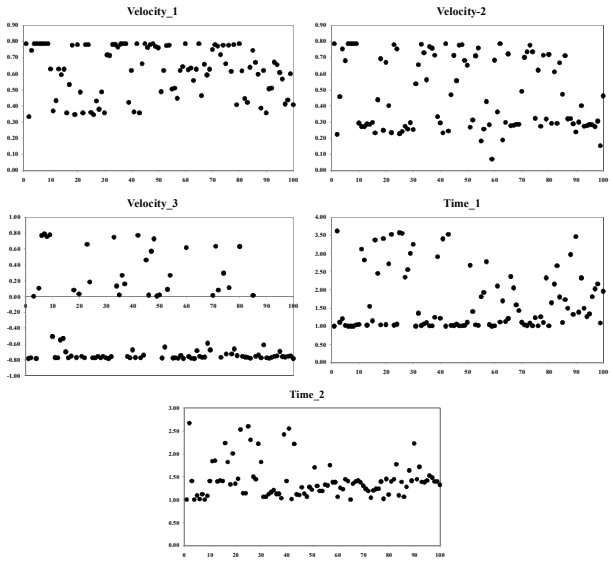


Figure. 9 Variations of design variables for ftime, f_{joint} and $f_{clength}$

V. CONCLUSIONS

A novel multi-objective optimization algorithm, Non-dominated Teaching Learning Based Optimization (NS-TLBO), has been developed by integrating the principles of non-dominated solutions with the Teaching Learning Based Optimization (TLBO) algorithm. The effectiveness of this algorithm is evaluated against the widely recognized Non-dominated Genetic Algorithm (NSGA-II) in the context of robot trajectory optimization, considering both two-objective and three-objective functions. The experimental results demonstrate that NS-TLBO outperforms NSGA-II, showing its ability to generate a more refined Pareto front for the robot trajectory optimization problem.

REFERENCES

[1] Adra, S.F., Dodd, T.J., Griffin, I.A. and Fleming, P.J., 2009. Convergence acceleration operator for multiobjective optimization. IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, 13(4), pp.825-847.

[2] Akbari, R., Hedayatzadeh, R., Ziarati, K. and Hassanizadeh, B., 2012. A multi-objective artificial bee colony algorithm. SWARM AND EVOLUTIONARY COMPUTATION, 2, pp.39-52.

[3] Deb, K., Pratap, A., Agarwal, S. and Meyarivan, T.A.M.T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, 6(2), pp.182-197.

[4] Zhang, Q., Liu, W. and Li, H., 2009, May. The performance of a new version of MOEA/D on CEC09 unconstrained MOP test instances. In 2009 IEEE CONGRESS ON EVOLUTIONARY COMPUTATION (pp. 203-208).

[5] Hedayatzadeh, R., Hasanizadeh, B., Akbari, R. and Ziarati, K., 2010, August. A multi-objective artificial bee colony for optimizing multi-objective problems. In 2010 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE) (Vol. 5, pp. V5-277).

[6] Coello, C.A.C., Pulido, G.T. and Lechuga, M.S., 2004. Handling multiple objectives with particle swarm optimization. IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, 8(3), pp.256-279.

[7] Got, A., Moussaoui, A. and Zouache, D., 2020. A guided population archive whale optimization algorithm for solving multiobjective optimization problems. EXPERT SYSTEMS WITH APPLICATIONS, 141, p.112972.

[8] Jin, R., Rocco, P. and Geng, Y., 2021. Cartesian trajectory planning of space robots using a multi-objective optimization. AEROSPACE SCIENCE AND TECHNOLOGY, 108, p.106360.

[9] Deb, K., Mohan, M. and Mishra, S., 2005. Evaluating the ϵ-domination based multi-objective evolutionary algorithm for a quick computation of Pareto-optimal solutions. EVOLUTIONARY COMPUTATION, 13(4), pp.501-525.

[10] Rao, R.V., Savsani, V.J. and Vakharia, D.P., 2011. Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems. COMPUTER-AIDED DESIGN, 43(3), pp.303-315.

[11] Rao, R.V., Savsani, V.J. and Balic, J., 2012. Teaching–learning-based optimization algorithm for unconstrained and constrained real-parameter optimization problems. ENGINEERING OPTIMIZATION, 44(12), pp.1447-1462.

[12] Savsani, P., Jhala, R.L. and Savsani, V.J., 2013, April. Optimized trajectory planning of a robotic arm using teaching learning based optimization (TLBO) and artificial bee colony (ABC) optimization techniques. In 2013 IEEE INTERNATIONAL SYSTEMS CONFERENCE (SysCon) (pp. 381-386).

[13] Savsani, P., Jhala, R.L. and Savsani, V.J., 2014. Comparative study of different metaheuristics for the trajectory planning of a robotic arm. IEEE SYSTEMS JOURNAL, 10(2), pp.697-708