Expert Systems With Applications 206 (2022) 117866

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

A search algorithm for constrained engineering optimization and tuning the gains of controllers

Saeed Rafee Nekoo [1,*], José Ángel Acosta [2], Anibal Ollero [3]

GRVC Robotics Lab., Depto de Ingeniería de Sistemas y Automática, Escuela Técnica Superior de Ingeniería, Universidad de Sevilla, Seville, Spain

ARTICLE INFO

Keywords:
Search algorithm
Engineering optimization
Static
Dynamic
Tuning control gain

ABSTRACT

In this work, the application of an optimization algorithm is investigated to optimize static and dynamic engineering problems. The methodology of the approach is to generate random solutions and find a zone for the initial answer and keep reducing the zones. The generated solution in each loop is independent of the previous answer that creates a powerful method. Simplicity as its main advantage and the interlaced use of intensification and diversification mechanisms–to refine the solution and avoid local minima/maxima–enable the users to apply that for a variety of problems. The proposed approach has been validated by several previously solved examples in structural optimization and scored good results. The method is also employed for dynamic problems in vibration and control. A modification has also been done on the method for high-dimensional test functions (functions with very large search domains) to converge fast to the global minimum or maximum; simulated for several well-known benchmarks successfully. For validation, a number of 9 static and 4 dynamic constrained optimization benchmark applications and 32 benchmark test functions are solved and provided, 45 in total. All the codes of this work are available as supplementary material in the online version of the paper on the journal website.

1. Introduction

1.1. Literature review

The use of evolutionary algorithms has been highlighted in science due to their capability in various fields such as mathematics, electrical, civil, mechanical, and control engineering, physics, economics, finance, etc. The idea of the evolution in science changed the perspective of researchers to switch to simpler methods for solving more complex problems with the cost of more computations and numerous iterations. The classical mathematical optimization methods are fast and precise such as gradient descent method (Nesterov, 2013), linear programming (Kirk, 2012), second-order cone programming (Alizadeh & Goldfarb, 2003), nonlinear programming (Peressini & Sullivan, 1988), etc.; however, the application of them in different fields is difficult and they require modifications and tuning.

There are some techniques specifically for the context of dynamic optimization: calculus of variations (Kirk, 2012), dynamic programming, and optimal control which proposes lots of tools such as linear quadratic regulator (Bemporad, Morari, Dua, & Pistikopoulos, 2002), state-dependent Riccati equation (Nekoo, 2019), successive Galerkin approximation (Kim, Kim, & Lim, 2003), and interpolation (Beeler, Tran, & Banks, 2000). The characteristics of static and dynamic optimization are different. Static optimization problems have several objective functions and variables with constraints though the ultimate solution is constant (global optimum) to the cost function. Dynamic optimization is more complex. The cost function and variables are available and they are time-varying, moving from initial to final condition. The optimization task is usually the best transition for the variables between boundary values. The search method in this work has been used for both static and dynamic optimization problems. It should be noted that the search method is not an optimal control method to provide a state transition of a dynamical system between boundary conditions. It can be used for the parameter optimization of static and dynamic systems. The dynamical problems have some tuning parameters or physical parameters that define the behavior of the system or the

* Corresponding author.
E-mail address: saerafee@yahoo.com (S.R. Nekoo).
[1] ORCID: 0000-0003-1396-5082.
[2] ORCID: 0000-0003-0040-338X.
[3] ORCID: 0000-0003-2155-2472.

https://doi.org/10.1016/j.eswa.2022.117866
Received 16 May 2020; Received in revised form 14 May 2022; Accepted 11 June 2022
Available online 17 June 2022

```
initialize iteration numbers for zones N_1, N_2, N_3, ..., N_Z, and N_G; N_1 < N_2 < N_3, ..., < N_Z
initialize reduction-search-domain percentage P_1, P_2, ..., P_{Z-1};   P_1 > P_2 >, ..., > P_{Z-1}
set counter of generations i = 1 and counter of iterations j = 1
while any stop criteria are not satisfied or i < N_G
    if i > 2                                                                                          | zone 1
        if the current solution is not better than the previous one or the answer is not in the allowable domain
            repeat the task, set i = i - 1
        else
            reset the iteration counter j = 1
        end if
    end if
    generate a random solution in the entire search domain, Eq. (3)
    for r = 1,2, ..., Z - 1                                                                           | zones 1,2,...
        if the task has been repeated N_r times, N_r ≤ j < N_{r+1}
            reduce the search domain P_r percent around the last solution, Eqs. (4) and (5) with r = 1,2, ..., Z - 1
            generate a random solution in the r-th reduced domain, Eq. (8) with r = 1,2, ..., Z - 1
        end if
    end for
    if the task has been repeated N_Z times, j = N_Z, N_Z is the max iteration number                  | exit
        print (i - 1)-th solution as the best solution
        break while
    end if
    construct the objective function and constraints, Eqs. (1) and (2)
    update the counters i = i + 1 and j = j + 1
    if i ≤ 2
        if constraints are not satisfied, Eq. (2)
            repeat the task, set i = i - 1
        end if
    end if
end while
```

Fig. 1. Pseudocode of the search algorithm.

error of a task. The proposed method is applicable for finding those control or physical parameters.

Besides the mathematical search approach, heuristics and metaheuristics methods are powerful tools for optimization such as memetic algorithm (Burke, Newall, & Weare, 1995), differential evolution (Price, Storn, & Lampinen, 2006), dynamic relaxation (Collins & Cosgrove, 2019), hill-climbing with random restart (Yelmewad & Talawar, 2019), particle swarm optimization (Tharwat, Elhoseny, Hassanien, Gabel, & Kumar, 2019), and Tabu search (Brusco & Doreian, 2019). The heuristic optimization methods do not mathematically guarantee a solution to the defined problem though they are powerful, easy to implement, and can be used for a wide range of problems in multiple disciplines. Zuo et al. applied the Tabu search for optimization of the layout of a hospital emergency department as a multi-objective optimization (Zuo et al., 2019). Agrawal et al. presented a new hybrid adaptive Cuckoo search-squirrel algorithm for brain MR image analysis (Agrawal, Samantaray, Panda, & Dora, 2020). The vast application of heuristic optimization tools is the most important advantage of evolutionary search methods. Self-adaptive sine cosine algorithm (Gupta & Deep, 2019), and improved sine cosine algorithm (Long, Wu, Liang, & Xu, 2019), were also recorded precise results for high dimensional problems.

The main contribution of this work is to investigate the application of a search algorithm for both static and dynamic problems. The main advantages of the method are simplicity, applicability for so many problems, precision, few tuning parameters, and capability for static and dynamic optimization. The simplicity of the method enables the tuning of the control methods such as proportional derivative and the state-dependent Riccati equation.

The rest of the work is structured as follows. Sections 1 and 2 defines the characterization of the method. Section 2 describes the motivation of the method and details of the algorithm. Section 3 presents static and dynamic examples along with well-known test functions to check the performance of the method. A discussion is provided in Sections 4 and 5 summarizes the conclusions.

1.2. Search algorithm within the metaheuristic optimization framework

Heuristics are basic approximate algorithms that search the domain to find a good solution (also called search): constructive algorithms which generate an answer by putting pieces of solutions together, and local search methods that start with a pre-existing solution and try to improve it by modifying the components (Bianchi, Dorigo, Gambardella, & Gutjahr, 2009). Metaheuristics combine heuristics in a more general framework with a trade-off between *intensification* and *diversification*. So, metaheuristics are iterative master processes that guide subordinate heuristics intelligently and efficiently; the heuristics might be a low-level search or simple local searches (Osman & Laporte, 1996). The proposed method performs an inefficient global search in zone 1 and is guided to efficient local searches in other zones. The guidance of the local searches around the best solution in the previous zone imposes an intelligent way of a search on the overall search method. So, based on the above definitions, the search method is categorized as a metaheuristic approach. The following items are a summary of metaheuristics (Blum & Roli, 2003); also applicable to the proposed approach: guidance of a sublevel search process, efficiently exploring search space to find a near-optimal solution, approximate and non-deterministic, possess mechanisms to avoid getting trapped in local minima, and they are not problem-specific. The term “diversification” indicates the exploration of the search domain, whereas the term “intensification” represents the exploitation of the accumulated search experience (Blum & Roli, 2003). The balance between diversification and intensification (for the proposed search method) could be easily done by setting the number of iterations in each zone and reducing search domain parameters; as explained in Section 2. Consideration of a

large number for zone 1, the entire search space, enhances the diversification, and adding the number of zones increases the intensification by zooming in on the search space.

Population-based search methods work over a search domain and evaluation of set-points; however, single-point search methods work on a single solution, so-called trajectory methods. One could see the trajectory of evolution and the solution in the generations. The proposed approach is a trajectory method or single-point search method.

Does the method make use of search history? The search algorithm uses only the previous-loop information exclusively to determine the next action. So, we might refer to that as the usage of short-term memory. The definition of memory-based methods indicates the usage of accumulated information on search history (Blum & Roli, 2003). As a result, the search algorithm is a memoryless method.

1.3. Framework and practical implementation

The application of the search algorithm is multidisciplinary covering constraint, static, and dynamic optimization. Several examples were solved in the simulation section. This algorithm has been developed in the context of two recent cutting-edge projects: the ERC GRIFFIN (Permanent-URL(b)), and AERIAL-CORE (https://aerial-core.eu/) projects which are related to aerial robotics including several platforms such as multi-rotor drones and flapping-wing flying robots for perching and manipulation. Both projects are devoted to developing new complex prototypes for aerial manipulation and, among other applications, this metaheuristic algorithm will help to optimize them. The flapping-wing flying robots in GRIFFIN possess ultra-lightweight designs to increase the time of flight and better maneuverability. Inverse shape design in the fluid flow problem is an interesting topic in metaheuristics which is an optimum design approach for wings (Chegini, Bagheri, & Najafi, 2018). The optimization problem of wing design for the flying robot of the GRIFFIN project is also a good application for the proposed search algorithm. Although it is not a mature design yet, an example of the application to this subject has been also addressed in one of the simulations in Section 3.1.9. The problem is the optimization of wing design concerning weight and lift force. The currently proposed wing design optimizes the top view of the wing; however, the previous literature optimized the cross-sectional profile of the wing (Mirjalili et al., 2017).

2. The search algorithm

2.1. Algorithm

The idea of the search algorithm is very simple and is based on random solutions. The method, at least, has three phases for the generation of a solution. First, the algorithm provides solutions in the entire domain of variables. The best solution in stage 1 defines the bounds of zone 2 for search in the next phase. Consequently, the best solution in the second stage narrows the search area for the next look. At the end of zone 3, the best answer is found. The selection of the solutions is random and the method is extremely simple, as presented in Fig. 1. The number of zones can be extended for more complex problems depending on the dimension and vastness of the search area.

The simple idea of the method is as follows:

- Try random solutions in the entire search domain and check the objective function.
- Repeat the trial if the answer is worse than the previous one.
- If the trials have been repeated so many times, reduce the search domain around the last and best answer so far.
- Search again in the smaller search domain while checking the entire domain as well.
- If the trials have been repeated so many times, reduce the search domain again around the last and best solution up to now.
- Search again many times and declare the last answer as the best.

It should be noted that the mentioned steps are presenting the idea and the complete form of the algorithm is presented in Pseudocode, Fig. 1.

2.1.1. Pseudocode

$N_1, N_2, \cdots, N_Z$ are the number of iterations (Z is the total number of zones), e.g. $N_1 = 500, N_2 = 1000, N_3 = 2000, N_4 = N_Z = 5000$, here the ultimate number of iterations for each generation of a solution is 5000 and we have 4 zones. N_G is an independent limit for external loop counter *i*. *i* is the counter of the external loop and *j* is the counter of the internal loop, both of them are updated in line 24 of the algorithm, Fig. 1.

Zone 1 in Fig. 1 is presenting the operation for the search in the entire domain for the solution that optimizes the objective function:

minimize or maximize :

$$f(x) = \sum_{l=1}^{m} f_l(x), \tag{1}$$

where $f(\mathbf{x})$ includes m criteria to be minimized or maximized, $\mathbf{x} = [x_1, \cdots, x_n]$ collects the variables related to the criteria. The constraints are set as.

subject to :

$$g_{min,h}(x) \le g_h(x) \le g_{max,h}(x), \tag{2}$$

in which $h = 1, \cdots, k$ defines k constraints and the variables are bounded between $x_{\min,p} \le x_p \le x_{\max,p}$, for $p = 1, \cdots, n$ in which n represents the total number of variables. The solution to the first zone is random between the minimum and maximum allowable bound:

$$x_p(i) = \text{rand}[0,1]\left(x_{max,p} - x_{min,p}\right) + x_{min,p}, \tag{3}$$

where $\text{rand}[0,1]$ generates a random number between 0 and 1. The optimization is repeated N_1 times, e.g. 1000 times. During the N_1 iterations (related to j counter), there are M_1 generations of solutions (related to i counter), e.g. 50. So, not all the iterations lead to a newly generated solution though the last generated solution M_1 is the best up to now. After so many trials it is obvious that the best solution is around the last solution in zone 1. In the next stage, zone 2, the deviations are smaller and the search bounds are reduced P_1 percent, e.g. 50% or 30%. Now, the algorithm has more power to find a better answer in $(N_2 - N_1)$ iterations that provide M_2 generations. The last stage is again narrowing the search area drastically P_2 percent, e.g. 5% or 1% to find the ultimate solution by $(N_3 - N_2 - N_1)$ iterations. After N_3 iterations, the last solution is the best one by the approach. So, it is obvious that $N_1 < N_2 < N_3$ and $P_1 > P_2$ are held. The bounds of the solution in r-th zone are:

$$x_{max,p,N_r} = x_p(i-1) + \frac{\left(x_{max,p} - x_{min,p}\right)}{2P_r}, \tag{4}$$

$$x_{min,p,N_r} = x_p(i-1) - \frac{\left(x_{max,p} - x_{min,p}\right)}{2P_r}, \tag{5}$$

where $j > N_r$, $r = 1, 2,$ and the new bounds are limited to.

$$\text{if } x_{max,p,N_r} > x_{max,p}, \qquad x_{max,p,N_r} = x_{max,p}, \tag{6}$$

$$\text{if } x_{min,p,N_r} < x_{min,p}, \qquad x_{min,p,N_r} = x_{min,p}, \tag{7}$$

in which for the second zone $r = 1$ and $P_1 = 0.5$ shows a 50% reduction in the search zone. The solution to the problem in the r-th zone is:

$$x_p(i) = \text{rand}[0,1]\left(x_{max,p,N_r} - x_{min,p,N_r}\right) + x_{min,p,N_r}. \tag{8}$$

In line 6 of Fig. 1, it was stated that if the answer is "not better" than the previous one, repeat the task. The word "less" can be used instead of

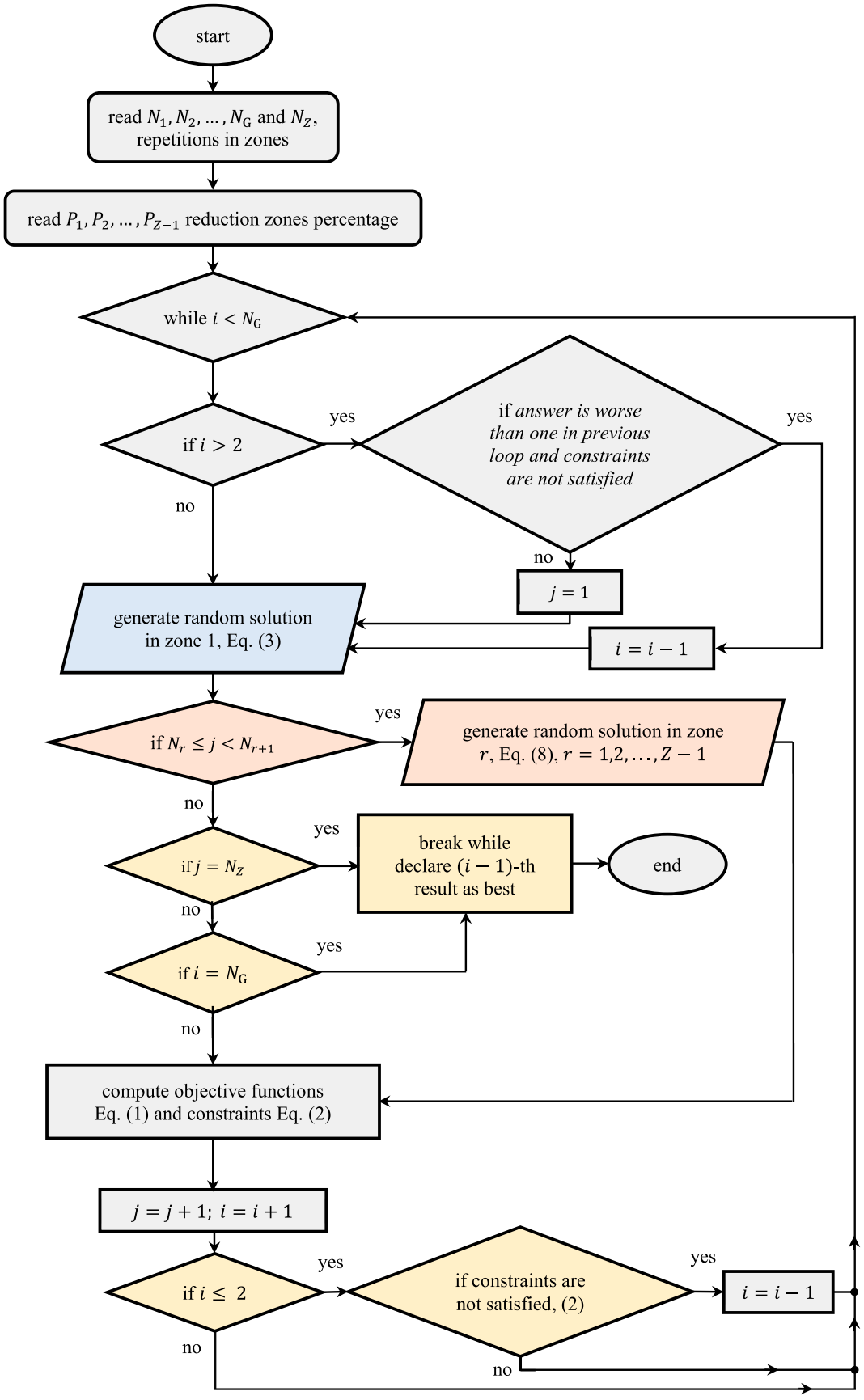


Fig. 2. The flowchart of the search algorithm; the zones can be extended.

"better" for minimization and "bigger" for maximization. Since the search algorithm is capable of both minimization and maximization, it was expressed "better" in the Pseudocode. It should be noted that the method could be enhanced by increasing the search zones, $r = 1,2,3,\cdots$, to find more precise answer though all the simulations in Sections 3.1 and 3.2 were done by three search zones $r = 1,2$.

2.1.2. Enhancement

The method works perfectly for constrained optimization problems presented in simulation Section 3.1 which covers so many static examples, and dynamic problems in Section 3.2. However, that might be weak for problems with very vast search domains such as high-dimensional test functions, Section 3.3. To present a very powerful option for the method, Eq. (8) is rewritten as.

$$x_p(i) = \big(-x_p(i-1) + \mathrm{rand}[0,1]\big)\big(x_{max,p,N_r} - x_{min,p,N_r}\big) + x_{min,p,N_r}. \tag{9}$$

This new edition speeds up the convergence and works for high-dimensional optimization functions. The proposed method with updating solution (9) is referred to as an improved search algorithm. All the simulations in Sections 3.1 and 3.2 are solved using the proposed search algorithm, without consideration of Eq. (9). In Section 3.3, both methods are compared to show the effectiveness of the simple version for fixed dimensional and constrained optimization and the enhanced version for high-dimensional problems.

The flowchart of the method is presented in Fig. 2. The program starts by reading the parameters, N_G in flowchart is the maximum number of generations for i counter. If the algorithm finds the answer in less than N_Z iterations (j counter), N_G finishes the while loop. For $i > 2$ the method checks the improvement with regards to the constraints. If it is not improving, the algorithm repeats the loop. For the two-first-generation loops, the constraints must be checked as well, presented at the bottom of the flowchart. If we do not check the first two generation loops for satisfying the constraints, we might find an infeasible solution and stay in the loop for entire N_Z iterations.

It should be noted that all solutions are random and in each generation, the previous answer does not play a role in the algorithm for finding the new guess. However, in the definition of the search area, the last solution plays a crucial role. According to Mitchell (Mitchell, 1998), the search methods that work for a large number of problems are weak methods and the ones especially defined to work for a specific problem are strong methods. Based on that definition and the fact that the method solves a large number of problems in static and dynamic optimizations, our proposed method is in the weak category though it is really simple. The simplicity does not imply that the solutions are not as good as the solutions by other search methods, and in some cases, the proposed algorithm scored better results. This point is also observable concerning large standard deviations by the search algorithm (see the results in tables) which implies the best solution is a mutation from multiple trials and sometimes results in astonishing values. Another highlight of the method is random solutions in the domain that eventually lead to the best answer, see Fig. 5.

The zones in the algorithm can be increased to give more power to the search method though most of the cases in this paper were solved by the proposed three-zone formalism. For test functions in Section 3.3, several examples were solved by simple and enhanced versions, with increased zones up to 9.

2.2. Stochastic mechanisms of the core algorithm

Although it is well-known that due to its complex nature, some of the metaheuristic algorithms do not have any mathematical proof of optimality or even convergence, in this approach we could provide at least a probabilistic formula of the evolution of the solutions as a function of zones and generations. Thus, in this section, we provide a combinatorial (rough) estimate of how the method generates a better result by the expansion of the zones (reduction of search space) along with generations in the random searching process. The process of generated answers is mainly defined by the "better" or "worse" condition in the flowchart, in Fig. 2. That condition asks "if the answer is worse than one in the previous loop and constraints are not satisfied",—no better estimate was found—, repeat the task by putting "YES" to the condition. If the answer is not worse than the previous one—a better estimate was found—, the algorithm goes to the next generation by putting "NO" to that condition, which is preferable. To simplify the analysis we define the event YES:= "a better estimate for the solution that satisfies the constraints is found". For the sake of clarity, in Fig. 3 we show a partial tree generated in the finding, where the event YES is denoted as y, and $\bar{y}$ denotes its complement.

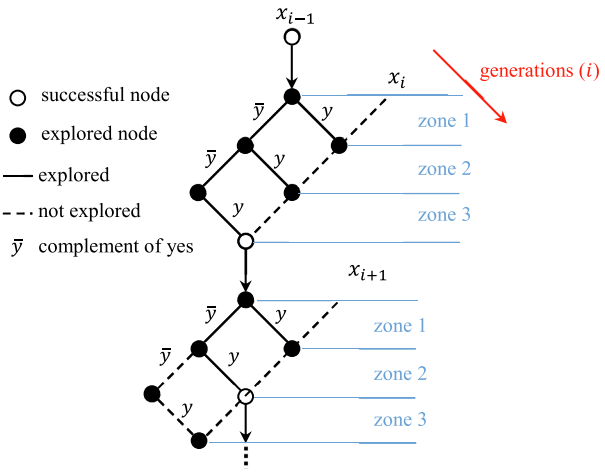


Fig. 3. Tree of the generations and solutions based on the probability of YES/NO.

Let us also denote Z as the number of zones with Z_i is i-th zone, and for simplicity but without any loss of generality, let N represent the search domain (set) in each zone, e.g. the number of points per zone, p is the probability of YES event in each zone $\langle p+q=1\rangle$ and finally, let N_G be the number of generations. Notice that we have imposed equal N for all the zones, which is not necessary, but it eases the analysis. This means, in turn, that p is likely to be equal for each zone because roughly speaking $p \propto 1/\dim(N)$. With all of these definitions, we can pose the problem of finding better estimates or solutions in a combinatorial framework. Thus, we can state the following result.

Theorem 1. Assume that for each generation a better estimate is found in one of the zones. The probability of finding a better estimate with Z zones and N_G generations $P_Z^{N_G}(y)$ yields.

$$P_Z^{N_G}(y) = \big(1 - q^Z\big)^{N_G}.$$

Proof. The discrete probability distribution of the success is a binomial distribution and hence the probability of a better estimate in the i-th generation reads.

$$P_i(y) = p + pq + pq^2 + \cdots = p\sum_{i=1}^{Z} q^{k-1}. \tag{10}$$

Moreover, it is straightforward to see that for the next generation this probability becomes.

$$P_{i+1}(y) = pP_i(y) + pqP_i(y) + pq^2P_i(y) + \cdots, \tag{11}$$

and therefore, summing up for N_G generations (last generation) yields.

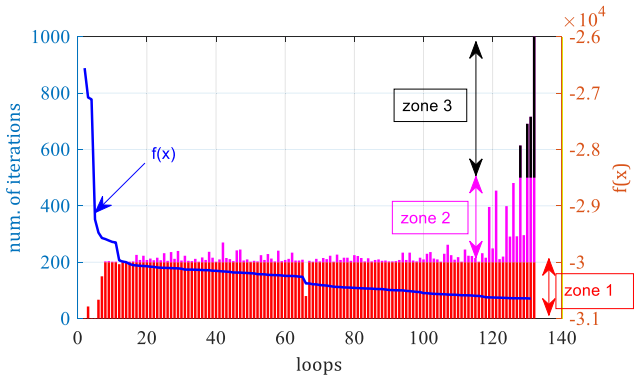


Fig. 4. Demonstration of 3 zones for an arbitrary solution.

$$P_Z^{N_G}(y) = \left(p\sum_{i=1}^{Z} q^{k-1}\right)^{N_G} = p^{N_G}\left(\frac{1-q^Z}{1-q}\right)^{N_G}, \tag{12}$$

where the last term comes from the sum of the convergent geometric series and $p+q=1$, concluding the proof. ■.

2.2.1. Discussion

On the one hand, Theorem 1 provides a simple and compact formula showing the evolutionary behavior of the algorithm as a function of the number of zones and generations. However, on the other hand, and more interestingly, it allows a qualitative analysis. Thus, let us say that we run the simplest random algorithm of searching for solutions in the entire space without any heuristic, and with a total number of independent trials $N_G = Z \times N$. Hence, since they are all independent, the probability of finding a better estimate in the entire space, namely $P_1^{N_G}$, becomes $P_1^{N_G} = \left(\frac{p}{Z}\right)^{N_G}$. When compared with the result of Theorem 1, it is straightforward to see that $P_Z^{N_G} > P_1^{N_G}$, for any $Z > 1$, and therefore increasing the number of zones will increase the chances of finding better estimates in non-convex optimization problems.

2.3. Computational cost

It should be reminded that two main loops, external (i) and internal (j), are playing the role in performing the algorithm. The first external loop is limited to maximum N_G generations. The number of generations varies between almost 10 for fixed-dimensional examples and 700–900 for high-dimensional test functions. It is always good to define N_G a large number. In that case, the algorithm decides to finish the generations when it has good enough results, defined by the internal loop. The internal loop is represented by j and is limited to the maximum number of N_Z. So, if the algorithm starts to work, several generations of answers will be found and after refining the answer and entering the other zones, N_Z sets the break command. There are two end switches for the algorithm though the correct way of finishing the optimization is by reaching N_Z limit. Every generation of solutions is gained by counting the internal loop less than N_Z. Consequently, the ultimate number of iterations is always [total num. iterations $<$ num. of generations $\times$ N_Z]. It could be counted for each case, though the general form is less than [num. of generations $\times$ N_Z]. The role of different zones is presented for an arbitrary example in Fig. 4. For this example, $N_1 = 200$, $N_2 = 500$ and $N_3 = 1000$; 132 generation were obtained which the last one caused the break command. The best answer is the 131st solution in this example. The total number of iterations is 30235, less than the estimated upper bound of 131000. It should be noted that the actual number of iterations is significantly lower than the upper bound.

3. Implementation and numerical experiments

3.1. Optimization for static problems

3.1.1. Himmelblau's non-linear optimization problem

The Himmelblau's optimization problem was proposed in 1972 (Himmelblau, 2018), and since then it has been widely used for testing search and optimization algorithms (Coello, 2000; Deb, 2000; Dimopoulos, 2007; Fesanghary, Mahdavi, Minary-Jolandan, & Alizadeh, 2008; Gandomi, Yang, & Alavi, 2013; Garg, 2016; Gen & Cheng, 1997; S. He, Prempain, & Wu, 2004; Homaifar, Qi, & Lai, 1994; Lee & Geem, 2005; Mehta & Dasgupta, 2012; Omran & Salman, 2009). It is a five-variable optimization problem (Lee & Geem, 2005):

Table 1
Results of Himmelblau's problem; NA represents not available and SD represents standard deviation; bold numbers are based on the second version of Himmelblau's problem.

Method	x	f(x)				SD
		Best	Median	Mean	Worst	
(Homaifar et al., 1994)	[80.39, 35.07, 32.05, 40.33, 33.34]	−30005.700	NA	NA	NA	NA
(Gen & Cheng, 1997)	NA	−30183.5760	NA	NA	NA	NA
(Coello, 2000)	**[78.0495, 33.007, 27.081, 45, 44.94]**	**−31020.859**	**−31017.21369**	**−30984.240703**	**−30792.407737**	**73.63**
(Deb, 2000)	NA	−30665.537	−30665.535	NA	−29846.654	NA
(Lee & Geem, 2005)	NA	−30665.500	NA	NA	NA	NA
(S. He et al., 2004)	[78, 33, 29.9952, 45, 36.7758]	−30665.539	NA	−30643.989	NA	70.04
(Dimopoulos, 2007)	[78, 33, 29.995256, 45, 36.775813]	−30665.54	NA	NA	NA	NA
(Fesanghary et al., 2008)	**[78, 33, 27.08515, 45, 44.92533]**	**−31024.3160**	NA	NA	NA	NA
(Omran & Salman, 2009)	NA	**−31025.5560**	NA	NA	NA	NA
(Gandomi et al., 2013)	[78, 33, 29.99616, 45, 36.77605]	−30665.2327	NA	NA	NA	11.62
(Mehta & Dasgupta, 2012)	[77.99, 32.99, 29.995, 44.99, 36.775]	−30665.538741	NA	NA	NA	NA
(Garg, 2014)	**[78, 33, 27.07097927, 45, 44.96902388]**	**−31025.57569195**	**−31025.5612911**	**−31025.55841263**	**−31025.49205458**	**0.0153528**
(Garg, 2016)	**[78, 33, 27.0709505, 45, 44.9691668]**	**−31025.574717**	**−31025.561141**	**−31025.557816**	**−31025.492054**	**0.01526**
(Garg, 2019)	**[77.961, 32.99948, 27.0728355, 45, 44.973943]**	−30668.004045 **−31027.6407622**	−30667.51076 **−31026.28126**	−30667.17208 **−31026.07246**	−30665.32463 **−31025.38705**	0.23158 **0.01803**
(Himmelblau, 2018)	[78, 33, 29.995, 45.0, 36.776]	−30665.5	NA	NA	NA	NA
this work	[78.03, 33.01, 30.01, 44.98, 36.77] **[78.06, 33.01, 27.08, 44.96, 44.92]**	−30682.06054 **−31049.88222**	−30644.23665 **−30966.08551**	−30624.11046 **−30957.36984**	−30396.37537 **−30866.50864**	74.75 **41.65**

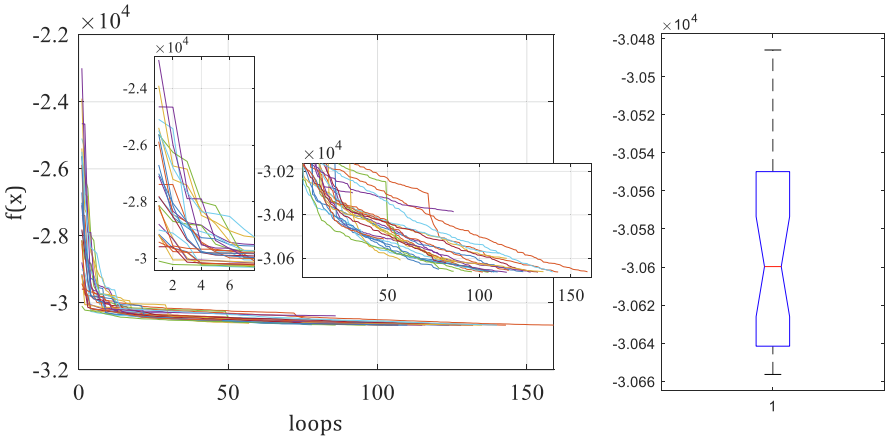


Fig. 5. The solutions to Himmelblau's problem; the left figure shows the flow of the solution and the right shows the statistics of the answers.

minimize :

$$f(\mathbf{x}) = 5.3578547x_3^2 + 0.8356891x_1x_5 + 37.293239x_1 - 40792.141, \quad (13)$$

in which $\mathbf{x} = [x_1, x_2, x_3, x_4, x_5]$ and it is limited by six constraints:

subject to :

$$g_1(\mathbf{x}) = 85.334407 + 0.0056858x_2x_5 + 0.0006262x_1x_4 - 0.0022053x_3x_5,$$

$$g_2(\mathbf{x}) = 80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 + 0.0021813x_3^2,$$

$$g_3(\mathbf{x}) = 9.300961 + 0.0047026x_3x_5 + 0.0012547x_1x_3 + 0.0019085x_3x_4, \quad (14)$$

where $0 \le g_1(\mathbf{x}) \le 90$, $90 \le g_2(\mathbf{x}) \le 110$ and $20 \le g_3(\mathbf{x}) \le 25$. The variables in equations (13) and (14) are defined in the domain of $78 \le x_1 \le 102$, $33 \le x_2 \le 45$ and $27 \le x_3, x_4, x_5 \le 45$. Here two versions of the Himmelblau's problem are studied, the first case is based on $g_1(\mathbf{x})$ in Eq. (14) and in the second version, the bold number 0.0006262 is replaced with 0.00026, according to some studies in Table 1. The parameters of the search algorithm were selected as $N_1 = 1000$, $N_2 = 2000$ and $N_3 = 10000$ with reduction search area terms $P_1 = 0.5$ and $P_2 = 0.01$. The results (repeated 30 times for calculating the best, median, mean and standard deviation) are reported in Table 1. The average time of runs is 4.7 s for the first case and 2.1 s for the second one, bolded in Table 1. The optimal answer to Himmelblau's problem was reported $\mathbf{x}^* = [78.0, 33.0, 29.995, 45.0, 36.776]$ with corresponding $f(\mathbf{x}) = -30665.5$ (Himmelblau, 2018); and better one by gravitational search algorithm and genetic algorithm as $f(\mathbf{x}) = -30665.56$ (Garg, 2019); though with the proposed search algorithm, this work, several good answers were found. For example, $\mathbf{x}^* = [78.03, 33.01, 29.93, 44.96, 36.82]$ with corresponding $f(\mathbf{x}) = -30667.99$ in 96 generations, or for another trial, $\mathbf{x}^* = [78.06, 33.07, 30.07, 44.92, 36.73]$ with corresponding $f(\mathbf{x}) = -30673.75$ in 114 generations; however, the minimum solution is presented in Table 1. The solutions to the optimization problem are illustrated in Fig. 5. Based on the policy of the algorithm, the number of internal loops for the generation of the answer to the optimization problem is defined by the search, so, the loops are different in Fig. 5, some of them needed less generation to gain the accepted solution. The T-test results in h-value of 1 and p-value of 6.94804447659251e-82; Wilcoxon signed-rank test also results in the same h-value and p-value of 1.73439762832058e-06.

3.1.2. Pressure vessel problem

The pressure vessel is a financial optimization problem based on the geometry of the vessel, allowable stress, welding, working pressure, and

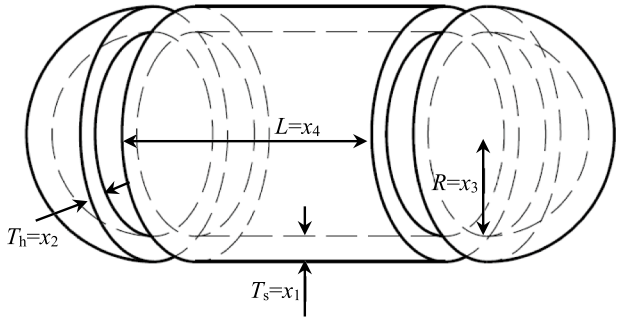


Fig. 6. Pressure vessel schematic and parameters definition.

design. The objective is the cost and the limitations are engineering factors. Four parameters are considered for optimization, the thickness of the cylinder $T_s = x_1$, the thickness of the hemisphere caps $T_h = x_2$, the inner radius of the cylinder $R = x_3$ and length of the cylinder $L = x_4$, Fig. 6. The working pressure is 3000psi and the minimum volume is 750ft^3 (Gandomi et al., 2013). The thickness of the elements could be integer multiples of 0.0625inches.

Based on the description, the optimization function is defined (Gandomi et al., 2013):

minimize :

$$f(\mathbf{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3, \quad (15)$$

which is limited by the constraints:

subject to :

$$g_1(\mathbf{x}) = -x_1 + 0.0193x_3 \le 0, g_2(\mathbf{x}) = -x_2 + 0.00954x_3 \le 0,$$

$$g_3(\mathbf{x}) = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 750 \times 1728 \le 0, g_4(\mathbf{x}) = x_4 - 240 \le 0, \quad (16)$$

where $1 \times 0.0625 \le x_1, x_2 \le 99 \times 0.0625$, $10 \le x_3 \le 200$ and case I: $10 \le x_4 \le 200$ and case II: $10 \le x_4 \le 240$.

One hundred trials were considered to test the approach for the pressure vessel optimization problem the best solution was found $f(\mathbf{x}) = 6059.7215907$ for case I with corresponding data $\mathbf{x} = [0.8125, 0.4375, 42.0984279262, 176.637033099]$. The average time for each trial was gained by almost 11.3 s. The parameters of the algorithm were selected as $N_1 = 50000$, $N_2 = 60000$ and $N_3 = 70000$ with reduced search area terms $P_1 = 0.1$ and $P_2 = 0.001$. Some of the previously published results in Table 2 violated the selection thickness, the thickness of the elements

Table 2
Results of the pressure vessel optimization problem; NA represents not available and SD represents standard deviation; bold numbers are based on the second version of the problem; 100 trials.

Method	x	$f(\mathbf{x})$				SD
		Best	Median	Mean	Worst	
(Sandgren, 1988)	[1.125, 0.625, 47.7, 17.701]	8129.1036	NA	NA	NA	NA
(Kannan & Kramer, 1994)	[1.125, 0.625, 58.291, 43.69]	7198.0428	NA	NA	NA	NA
(Coello, 2000)	[0.8125, 0.4375, 40.3239, 200]	6288.7445	NA	6293.8432	6308.1497	7.4133
(Coello & Montes, 2002)	[0.8125, 0.4375, 42.097398, 176.65405]	6059.9463	NA	6177.2533	6469.3220	130.9297
(He & Wang, 2007)	[0.8125, 0.4375, 42.091266, 176.7465]	6061.0777	NA	6147.1332	6363.8041	86.4545
(Mezura-Montes & Coello, 2008)	[0.8125, 0.4375, 42.098087, 176.640518]	6059.7456	NA	6850.0049	7332.8798	426
(Kaveh & Talatahari, 2010)	[0.8125, 0.4375, 42.098353, 176.637751]	6059.7258	NA	6081.7812	6150.1289	67.2418
(Kaveh & Talatahari, 2009)	[0.8125, 0.4375, 42.103566, 176.57322]	6059.0925	NA	6075.2567	6135.3336	41.6825
(Cagnina, Esquivel, & Coello, 2008)	[0.8125, 0.4375, 42.098445, 176.636595]	6059.714335	NA	6092.0498	NA	12.1725
(Gandomi et al., 2013)	[0.8125, 0.4375, 42.0984456, 176.6365958]	6059.7143348	NA	6447.7360	6495.3470	502.693
(dos Santos Coelho, 2010)	[0.8125, 0.4375, 42.0984, 176.6372]	6059.7208	6257.5943	6440.3786	7544.4925	448.4711
(He et al., 2004)	[0.8125, 0.4375, 42.098445, 176.636595]	6059.7143	NA	6289.92881	NA	305.78
(Mezura-Montes, Coello Coello, Velázquez-Reyes, & Muñoz-Dávila, 2007)	[0.8125, 0.4375, 42.098446, 176.636047]	6059.70166	NA	NA	NA	NA
(Akay & Karaboga, 2012)	[0.8125, 0.4375, 42.098446, 176.636596]	6059.714339	NA	6245.308144	NA	205
(Garg, 2014)	[0.7781977, 0.3846656, 40.32105455, 199.9802367]	5885.4032828	5886.149289	5887.557024	5895.126804	2.74529
(Garg, 2019)	[0.77819652, 0.3846644, 40.3210580446, 199.9799646]	5885.38533633	5884.58128	5884.24637	5884.462541	0.50281
(Dimopoulos, 2007)	**[0.75, 0.375, 38.86010, 221.36549]**	**5850.38306**	NA	NA	NA	NA
(Mahdavi, Fesanghary, & Damangir, 2007)	**[0.75, 0.375, 38.8601, 221.36553]**	**5849.76169**	NA	NA	NA	NA
(Gandomi, Yang, & Alavi, 2011)	**[0.75, 0.375, 38.8601, 221.36547]**	**5850.38306**	NA	**5937.3379**	**6258.96825**	**164.54747**
(Garg, 2014)	**[0.72759583, 0.359655288, 37.69913599, 239.999805]**	**5804.4486708**	**5805.073797**	**5805.47391**	**5811.977127**	**1.411462**
(Garg, 2019)	**[0.727610046, 0.359658023, 37.70 0 0 0238, 239.983428]**	**5804.4048008**	**5806.7764199**	**5806.596206**	**5808.16968**	**1.028072**
(Mirjalili, 2015)	[0.8125, 0.4375, 42.098445, 176.636596]	6059.7143	NA	NA	NA	NA
(Chegini et al., 2018)	[1.25, 0.0625, 64.7668, 11.9886]	3137.3	NA	NA	NA	NA
(Heidari, Abbaspour, & Jordehi, 2017)	**[0.727612, 0.359656, 37.771023, 238.510803]**	**5788.6492866**	**6100.186102**	**6103.774173**	**6971.110752**	**294.490733**
(Rizk-Allah, 2018)	[0.7781908, 0.383047, 40.32075, 199.98419]	5880.71150	NA	NA	NA	NA
(Mirjalili et al., 2014)	[0.8125, 0.4345, 42.089181, 176.758731]	6051.5639	NA	NA	NA	NA
This work	[0.8125, 0.4375, 42.0984279262, 176.637033099]	6059.7215907	6090.5976329	6149.9212110	6771.680943	144.90665
	[0.75, 0.375, 38.8600465508, 221.367130189]	**5850.4066129**	**6059.8373796**	**6036.8323191**	**6771.733292**	**176.71858**

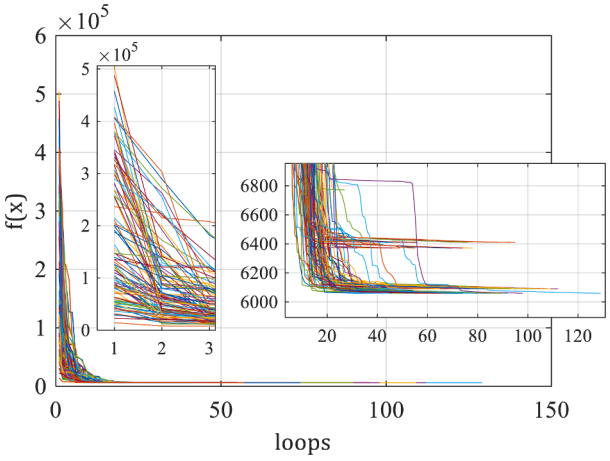


Fig. 7. Numerical solution based on the search algorithm for pressure vessel optimization problem, 100 trials, case I.

could be integer multiples of 0.0625 in., which resulted in lower values.

The solutions for the first case are illustrated in Fig. 7. For case II, $10 \leq x_4 \leq 240$, the best answer was obtained $f(\mathbf{x}) = 5850.4066129$ with corresponding geometry $\mathbf{x} = [0.75, 0.375, 38.8600465508, 221.367130189]$, and average time of each trial is 10.2 s. Most of the solutions to case I were around 6059 and for case II around 5850 which confirms the correctness and precision of the approach. The T-test results in h-value of 1 and p-value of 5.94904464016159e-151; Wilcoxon signed-rank test also results in the same h-value and p-value of 3.89655984509596e-18.

3.1.3. The welded beam optimization problem

A beam is welded to a support plate and is under a vertical load P with cross-sectional area bt and length $L + l$, see Fig. 8. The thickness of the welding is $h = x_1$, the length of the welding is $l = x_2$, width of the beam is $t = x_3$ and the thickness of the beam is $b = x_4$. The optimization task is to minimize the cost including material, setup, and welding labor (Garg, 2016).

The optimization problem is defined by (Garg, 2016):

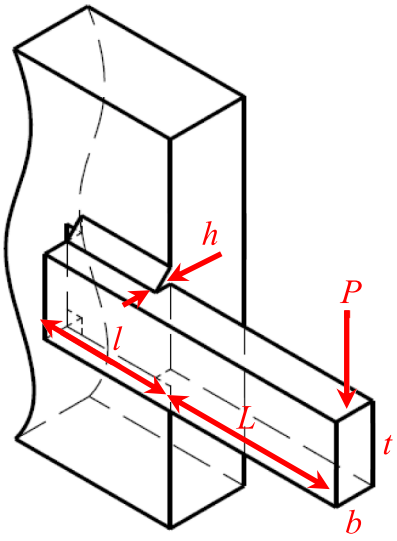


Fig. 8. Schematic design of welded beam optimization problem.

minimize :

$$f(\mathbf{x}) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14 + x_2), \tag{17}$$

where the limits are:

subject to :

$$g_1(\mathbf{x}) = \tau(\mathbf{x}) - \tau_{\max} \le 0, g_2(\mathbf{x}) = \sigma(\mathbf{x}) - \sigma_{\max} \le 0,$$

$$g_3(\mathbf{x}) = x_1 - x_4 \le 0, g_4(\mathbf{x}) = 0.125 - x_1 \le 0,$$

$$g_5(\mathbf{x}) = \delta(\mathbf{x}) - 0.25 \le 0, g_6(\mathbf{x}) = P - P_c(\mathbf{x}) \le 0, \tag{18}$$

where $0.1 \le x_1, x_4 \le 2$ and $0.1 \le x_2, x_3 \le 10$, and:

$$\tau(\mathbf{x}) = \sqrt{\tau_1^2(\mathbf{x}) + 2\tau_1(\mathbf{x})\tau_2(\mathbf{x})\frac{x_2}{2R(\mathbf{x})} + \tau_2^2(\mathbf{x})}, \tau_1(\mathbf{x}) = \frac{P}{\sqrt{2}x_1x_2}, \tau_2(\mathbf{x}) = \frac{M(\mathbf{x})R(\mathbf{x})}{J(\mathbf{x})},$$

$$M(\mathbf{x}) = P\left(L + \frac{x_2}{2}\right), R(\mathbf{x}) = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}, J(\mathbf{x}) = 2\left(\frac{x_1x_2}{\sqrt{2}}\left[\frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right),$$

$$\delta(\mathbf{x}) = \frac{4PL^3}{Ex_3^3x_4}, \sigma(\mathbf{x}) = \frac{6PL}{x_3^2x_4}, P_c(\mathbf{x}) = \frac{4.013\sqrt{\frac{EGx_3^2x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right),$$

in which $G = 12 \times 10^6$psi, $E = 30 \times 10^6$psi, $P = 6000$lb, $L = 14$in, $\tau_{\max} = 13600$psi and $\sigma_{\max} = 30000$psi. A second version (with an extra constraint and some minor changes) also exists in the literature presented in Ref. (Garg, 2019), the results of that are bolded in Table 3. The repetitions in the three zones are defined as $N_1 = 2500$, $N_2 = 5000$ and $N_3 = 10000$ with reduction search area terms $P_1 = 0.25$ and $P_2 = 0.01$. The best result of the proposed approach for 30 trials was gained $f(\mathbf{x}) = 2.38292535334716$ with corresponding design parameters $\mathbf{x} = [0.2442747104, 6.1965519331, 8.3186789293, 0.24427631231]$. The average time of each trial was found around 15 s. Concerning the best result in Table 3, it has a 0.002 deviation though it is close the most of the answers.

The second version of the problem (according to Ref. (Garg, 2019)) with bold numbers in Table 3, was found with the same parameters of the method; the best answer is $f(\mathbf{x}) = 1.6977033295$. The average time of each trial was defined 17 s. The results of the second version were also better than all the presented results in Table 3 except Refs (Chegini et al., 2018; Garg, 2014, 2016). Stochastic fractal search (SFS) scored the value 1.72485230 for the optimization function (Salimi, 2015), and this current work scored 1.6977033295, similar to Ref. (Garg, 2019). The SFS used 24,000 number-of-function-evaluations (NEF), and this work 17,500. The T-test results in an h-value of 1 and p-value of 4.66526787455877e-52; Wilcoxon signed-rank test also results in the same h-value and p-value of 1.73439762832058e-06, see ANOVA plot in Fig. 9.

3.1.4. Tension/compression spring design problem

Consider the spring optimization problem, a well-known example, with wire diameter $d = x_1$, coil diameter $D = x_2$ and the number of active coils $N = x_3$. All three variables are continuous. The objective is to find a spring with minimum possible weight satisfying the minimum deflection, shear, and surge frequency constraint, and limits on the coil diameter. The schematic view of the spring is presented in Fig. 10.

The mathematical optimization problem can be stated as (He et al., 2004):

minimize :

$$f(\mathbf{x}) = (x_3 + 2)x_1^2x_2, \tag{19}$$

where the limits are:

subject to :

$$g_1(\mathbf{x}) = 1 - \frac{x_2^3x_3}{71785x_1^4} \le 0,$$

$$g_2(\mathbf{x}) = \frac{4x_2^2 - x_1x_2}{12566(x_1^3x_2 - x_1^4)} + \frac{1}{5108x_1^2} - 1 \le 0,$$

$$g_3(\mathbf{x}) = 1 - \frac{140.45x_1}{x_2^2x_3} \le 0, g_4(\mathbf{x}) = \frac{x_1 + x_2}{1.5} - 1 \le 0, \tag{20}$$

in which $0.05 \le x_1 \le 2$, $0.25 \le x_2 \le 1.3$ and $2 \le x_3 \le 15$. Thirty trials were done to find the best, median, mean, worst, and standard deviation of the answers, reported in Table 4. The best solution was found $f(\mathbf{x}) = 0.012667928876$ with corresponding elements $\mathbf{x} = [0.051647047504, 0.35569064667, 11.351872381]$. There was a common range around 0.01266 which the approach resulted in less value than that. The average time for each trial was obtained 12.6 s. The repetitions in the three zones are defined as $N_1 = 2500$, $N_2 = 7500$ and $N_3 = 15000$ with reduction search area terms $P_1 = 0.2$ and $P_2 = 0.001$. Comparing with the SFS, the optimization function was found at 0.012665232788 for SFS, and 0.012667928876 for the proposed method. The NEF of the SFS was set at 100,000 and this work at 25,000. The T-test results in h-value of 1 and p-value of 1.10998075163599e-54; Wilcoxon signed-rank test also results in the same h-value and p-value of 1.73439762832058e-06.

3.1.5. Gear train optimization design

The gear train optimization problem was proposed as a discrete optimization case study (Sandgren, 1988). Here we use the example to test the capability of the approach. Consider a gearbox including four gears to reduce the angular velocity of the driven shaft, see Fig. 11. All the gears could have teeth between 12 and 60. The objective is the minimization of the cost. The variables are set as $T_d = x_1$, number of teeth of driving gear D, $T_b = x_2$, $T_a = x_3$ and the driven gear $T_f = x_4$ with transition ratio $\mathrm{TR} = T_dT_b/T_aT_f$.

The optimization function is (Sandgren, 1988):

minimize :

Table 3
Results of the welded beam problem; NA represents not available and SD represents standard deviation, 30 trials.

Method	x	$f(\mathbf{x})$				SD
		Best	Median	Mean	Worst	
(Ragsdell & Phillips, 1976)	[0.2455, 6.196, 8.273, 0.2455]	2.385937	NA	NA	NA	NA
(Rao, 2019)	[0.2455, 6.196, 8.273, 0.2455]	2.3860	NA	NA	NA	NA
(Ray & Liew, 2003)	[0.244438276, 6.2379672340, 8.2885761430, 0.2445661820]	2.3854347	3.0025883	3.2551371	6.3996785	0.9590780
(Deb, 2000)	NA	2.38119	NA	NA	NA	NA
(Hwang & He, 2006)	[0.2231, 1.5815, 12.8468, 0.2245]	2.25	NA	2.26	2.28	NA
(Mehta & Dasgupta, 2012)	**[0.24436895, 6.21860635, 8.29147256, 0.24436895]**	**2.3811341**	**2.3811641**	**2.3811786**	**2.3812614**	NA
(Garg, 2014)	[0.24436198, 6.21767407, 8.29163558, 0.24436883]	2.38099617	2.38107233	2.38108932	2.38146999	1.01227×10^{-4}
(Garg, 2019)	[0.24436822, 6.21754641, 8.29147808, 0.24436894]	2.3809597	2.3786824	2.3794755	2.380147	1.182265×10^{-4}
(Akhtar et al., 2002)	**[0.2407, 6.4851, 8.2399, 0.2497]**	**2.4426**	NA	2.5215	2.6315	NA
(Aragón, Esquivel, & Coello, 2010)	**[0.244369, 6.218613, 8.291474, 0.244369]**	**2.38113**	NA	2.439811	2.710406	0.093146
(Long et al., 2019)	**[0.24435, 6.2178, 8.2919, 0.24437]**	**2.3810**	NA	NA	NA	NA
(Chegini et al., 2018)	**[0.140946034, 2.8644127185, 9.036471193, 0.20573659]**	**1.57126326**	NA	NA	NA	NA
(Coello, 2000)	**[0.2088, 3.4205, 8.9975, 0.21]**	**1.748309**	NA	**1.771973**	**1.785835**	**0.011220**
(Coello & Montes, 2002)	**[0.205986, 3.471328, 9.020224, 0.20648]**	**1.728226**	NA	**1.792654**	**1.993408**	**0.07471**
(He & Wang, 2007)	**[0.202369, 3.544214, 9.048210, 0.205723]**	**1.728024**	NA	**1.748831**	**1.782143**	**0.012926**
(Dimopoulos, 2007)	**[0.2015, 3.5620, 9.041398, 0.205706]**	**1.731186**	NA	NA	NA	NA
(Mahdavi et al., 2007)	**[0.20573, 3.47049, 9.03662, 0.20573]**	**1.7248**	NA	NA	NA	NA
(Mezura-Montes et al., 2007)	**[0.205730, 3.470489, 9.036624, 0.205730]**	**1.724852**	NA	**1.725**	NA	$\mathbf{1 \times 10^{-15}}$
(Mezura-Montes & Coello, 2008)	**[0.199742, 3.61206, 9.0375, 0.206082]**	**1.73730**	NA	**1.813290**	**1.994651**	**0.0705**
(Cagnina et al., 2008)	**[0.205729, 3.470488, 9.036624, 0.205729]**	**1.724852**	NA	**2.0574**	NA	**0.2154**
(Kaveh & Talatahari, 2009)	**[0.205729, 3.469875, 9.036805, 0.205765]**	**1.724849**	NA	**1.727564**	**1.759522**	**0.008254**
(Kaveh & Talatahari, 2010)	**[0.2057, 3.471131, 9.036683, 0.205731,**	**1.724918**	NA	**1.729752**	**1.775961**	**0.0092**
(Gandomi et al., 2011)	**[0.2015, 3.562, 9.0414, 0.2057]**	**1.73121**	NA	**1.878656**	**2.3455793**	**0.2677989**
(Akay & Karaboga, 2012)	**[0.20573, 3.470489, 9.036624, 0.20573]**	**1.724852**	NA	**1.724852**	**1.741913**	**0.031**
(Garg, 2014)	**[0.2057245, 3.25325369, 9.03664438, 0.20572999]**	**1.69526388**	**1.69530879**	**1.69530842**	**1.69537060**	$\mathbf{2.83 \times 10^{-5}}$
(Garg, 2019)	**[0.20572943, 3.253123897, 9.03662392, 0.20572964]**	**1.695247383**	**1.6952473**	**1.6952473**	**1.6952473**	$\mathbf{1.978 \times 10^{-8}}$
(Mirjalili et al., 2014)	[0.205676, 3.478377, 9.03681, 0.205778]	1.72624	NA	NA	NA	NA
(Salimi, 2015)	[0.20572, 3.47048, 9.03662, 0.20572]	1.72485230	1.72485230	1.72485230	1.72485230	7.708×10^{-4}
This work	[0.24427471040,6.1965519331,8.3186789293,0.24427631231]	2.3829253533	2.4399704848	2.463045569	2.6474834886	0.081778
	[0.2046213849, 3.2726074634, 9.0412338219, 0.2058172266]	**1.6977033295**	**1.7172827764**	**1.731077882**	**1.836242963**	**0.037482**

$$f(\mathbf{x}) = \left(\frac{1}{6.931} - \frac{x_1 x_2}{x_3 x_4}\right)^2, \tag{21}$$

which is constrained by range $12 \leq x_1, x_2, x_3, x_4 \leq 60$ with $\{x_1, x_2, x_3, x_4\} \in \mathbb{Z}^+$. Due to the discrete selection of the teeth and having no other constraint, this example is solved fast with fewer iterations. The parameters are defined as $N_1 = 100$, $N_2 = 200$ and $N_3 = 500$ with reduction search area terms $P_1 = 0.7$ and $P_2 = 0.3$. The average time of each trial was found 0.49 s. The number of teeth was gained $\mathbf{x} = [16, 19, 43, 49]$, the same as other methods Table 5, $f(\mathbf{x}) = 2.7008571 \times 10^{-12}$. The nature of the example leads to solutions in Table 5 to a unique answer which verifies the capability of the method for solving different problems. The T-test results in an h-value of 1 and p-value of 0.000804033793725383; Wilcoxon signed-rank test also results in the same h-value and p-value of 1.64891051346443e-06.

3.1.6. Speed reducer problem

The speed reducer optimization problem is a popular case study for the comparison of algorithms in the literature. It is also another gearbox design with different constraints and objectives for Section 3.1.5. Here, a more sophisticated gearbox design is intended, the weight of the gearbox needs to be minimized with constraints related to the bending stress of the gear teeth, surface stress, transverse deflections, and stresses

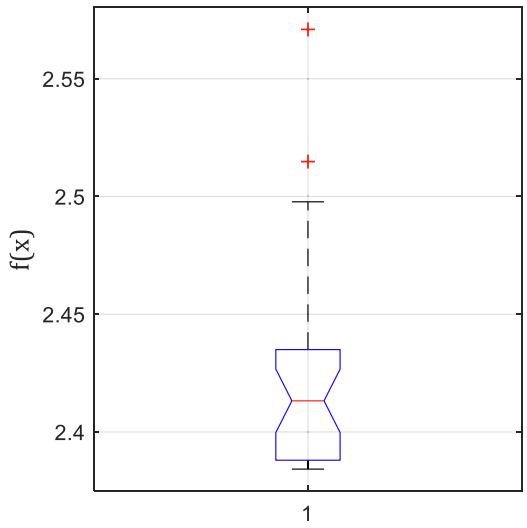


Fig. 9. ANOVA plot of the welded beam design.

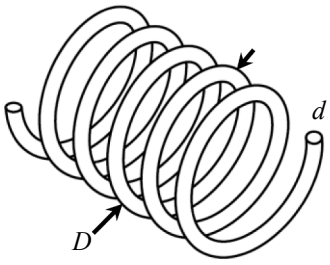


Fig. 10. Schematic view of the spring problem.

in shafts (Gandomi et al., 2013). The schematic view of the speed reducer is presented in Fig. 12. The width of the gears is $b = x_1$, gear module is $m = x_2$, number of the teeth for driving gear is $z = x_3$, length of the driving and driven shafts are $l_1 = x_4$ and $l_2 = x_5$ with respect, the diameter of the driving gear and driven one are $d_1 = x_6$ and $d_2 = x_7$ with respect. The number of teeth for driving gear is integer though the rest of the variables are continuous.

The mathematical optimization problem is stated as (Akhtar, Tai, & Ray, 2002):

minimize :

$$f(\mathbf{x}) = 0.7854x_1x_2^2\left(3.3333x_3^2 + 14.9334x_3 - 43.0934\right) - 1.508x_1\left(x_6^2 + x_7^2\right) + 7.4777\left(x_6^3 + x_7^3\right) + 0.7854\left(x_4x_6^2 + x_5x_7^2\right), \tag{22}$$

where the limits are:

subject to :

$$g_1(\mathbf{x}) = \frac{27}{x_1x_2^2x_3} - 1 \le 0, g_2(\mathbf{x}) = \frac{397.5}{x_1x_2^2x_3^2} - 1 \le 0,$$

$$g_3(\mathbf{x}) = \frac{1.93x_4^3}{x_2x_3x_6^4} - 1 \le 0, g_4(\mathbf{x}) = \frac{1.93x_5^3}{x_2x_3x_7^4} - 1 \le 0,$$

$$g_5(\mathbf{x}) = \frac{\sqrt{\left(\frac{745x_4}{x_2x_3}\right)^2 + 16.9 \times 10^6}}{110x_6^3} - 1 \le 0,$$

$$g_6(\mathbf{x}) = \frac{\sqrt{\left(\frac{745x_5}{x_2x_3}\right)^2 + 157.5 \times 10^6}}{85x_7^3} - 1 \le 0,$$

$$g_7(\mathbf{x}) = \frac{x_2x_3}{40} - 1 \le 0, g_8(\mathbf{x}) = \frac{5x_2}{x_1} - 1 \le 0,$$

$$g_9(\mathbf{x}) = \frac{x_1}{12x_2} - 1 \le 0, g_{10}(\mathbf{x}) = \frac{1.5x_6 + 1.9}{x_4} - 1 \le 0,$$

$$g_{11}(\mathbf{x}) = \frac{1.1x_7 + 1.9}{x_5} - 1 \le 0, \tag{23}$$

in which $2.6 \le x_1 \le 3.6$, $0.7 \le x_2 \le 0.8$, $17 \le x_3 \le 28$, $7.3 \le x_4 \le 8.3$, $7.8 \le x_5 \le 8.3$, $2.9 \le x_6 \le 3.9$ and $5 \le x_7 \le 5.5$. The second case, represented by bold numbers in Table 6, has a different value case I: 1.69 instead of case II: 16.9 in $g_5(\mathbf{x})$, Eq. (23). Similar to other sections, 30 trials have been done in this example to provide the results, each trial 63 s. The parameters are defined as $N_1 = 2500$, $N_2 = 5000$ and $N_3 = 10000$ with reduction search area terms $P_1 = 0.1$ and $P_2 = 0.05$. The best solution was found $f(\mathbf{x}) = 3002.1524434$ for the first case, and $f(\mathbf{x}) = 2899.1336453$ for the second one. The large number of constraints increased the time of search though the answer scored a good precision. As we are going forward in terms of computational tools and methods, we are losing the sense of engineering in problems. For example, this speed-reducer optimization is an engineering design, the module of gear is a fixed number, and finding a solution with many decimals is not helpful. Also, the thickness of gear could be built with almost 0.001mm in reality which is high precision for such a product. Therefore, we consider these examples just to test the capability of the search approach and validation with other previous sources. The T-test results in h-value of 1 and p-value of 7.40023725719789e-58; Wilcoxon signed-rank test also results in the same h-value and p-value of 1.73439762832058e-06.

3.1.7. Three-bar truss problem

The three-bar truss optimization problem is selected as a case study to show the performance of the method in structural optimization. The objective is to minimize the volume of a statically loaded truss, Fig. 13, by setting the design parameters A_1 and A_2 (Gandomi et al., 2013). The variables are selected as the area of the bars $A_1 = x_1$ and $A_2 = x_2$.

The mathematical optimization problem is formulated as (Gandomi et al., 2013):

minimize :

$$f(\mathbf{x}) = \left(2\sqrt{2}x_1 + x_2\right)L, \tag{24}$$

where the constraints are defined:

subject to :

$$g_1(\mathbf{x}) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P - \sigma \le 0, g_2(\mathbf{x}) = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P - \sigma \le 0,$$

$$g_3(\mathbf{x}) = \frac{1}{x_1 + \sqrt{2}x_2}P - \sigma \le 0, \tag{25}$$

where $0 \le x_1, x_2 \le 1$, $L = 100$cm, $P = 2$KN and $\sigma = 2$KN/cm^2. The repetitions in three zones of the method are defined as $N_1 = 5000$, $N_2 = 10000$ and $N_3 = 20000$ with reduction search area terms $P_1 = 0.5$ and $P_2 = 0.01$. The best result for 30 trials was gained $f(\mathbf{x}) = 263.89587188382$ with corresponding design parameters $\mathbf{x} = [0.78875226015122, 0.408030431521627]$. The average time of each trial was found around 9.1 s. The comparison is presented in Table 7. The results of the best solution were gained by 25 generation loops though the maximum generation loops for one of the answers was 35,

Table 4
Results of the spring optimization problem; NA represents not available and SD represents standard deviation, 30 trials.

Method	x	$f(\mathbf{x})$				SD
		Best	Median	Mean	Worst	
(Ray & Saini, 2001)	[0.050417, 0.321532, 13.979915]	0.01306	NA	0.015526	0.018992	NA
(Belegundu, 1982)	[0.05, 0.3159, 14.25]	0.0128334	NA	NA	NA	NA
(Coello & Montes, 2002)	[0.051989, 0.363965, 10.890522]	0.012681	NA	0.012742	0.012973	5.9×10^{-5}
(Coello, 2000)	[0.05148, 0.351661, 11.632201]	0.01270478	0.01275576	0.0127692	0.01282208	3.939×10^{-5}
(Ray & Liew, 2003)	[0.052160217, 0.368158695, 10.648442259]	0.01266924934	0.012922669	0.012922669	0.016717272	5.92×10^{-4}
(He et al., 2004)	[0.0516904, 0.35674999, 11.28712599]	0.0126652812	NA	0.01270233	NA	4.124×10^{-5}
(Raj & Sharma, 2005)	[0.053862, 0.41128365, 8.6843798]	0.0127484	NA	NA	NA	NA
(Tsai, 2005)	[0.05168906, 0.3567178, 11.28896]	0.01266523	NA	NA	NA	NA
(Mahdavi et al., 2007)	[0.05115438, 0.34987116, 12.0764321]	0.0126706	NA	NA	NA	NA
(Mezura-Montes et al., 2007)	[0.051688, 0.356692, 11.290483]	0.012665	NA	0.012666	NA	2×10^{-6}
(He & Wang, 2007)	[0.051728, 0.357644, 11.244543]	0.0126747	NA	0.01273	0.012924	5.19×10^{-5}
(Cagnina et al., 2008)	[0.051583, 0.35419, 11.438675]	0.012665	NA	0.0131	NA	4.1×10^{-4}
(Mezura-Montes & Coello, 2008)	[0.051643, 0.35536, 11.397926]	0.012698	NA	0.013461	0.16485	9.66×10^{-4}
(Zhang, Luo, & Wang, 2008)	[0.0516890614, 0.3567177469, 11.2889653382]	0.012665233	NA	0.012669366	0.012738262	1.25×10^{-5}
(Kaveh & Talatahari, 2010)	[0.051865, 0.3615, 11]	0.0126432	NA	0.01272	0.012884	3.48×10^{-5}
(dos Santos Coelho, 2010)	[0.051515, 0.352529, 11.538862]	0.012665	0.012957	0.013524	0.017759	0.001268
(Mirjalili et al., 2014)	[0.05169, 0.356737, 11.28885]	0.012666	NA	NA	NA	NA
(Akay & Karaboga, 2012)	[0.051749, 0.358179, 11.203763]	0.012665	NA	0.012709	NA	0.012813
(Garg, 2014)	[0.051689156131, 0.356720026419, 11.288831695483]	0.01266523278	0.012665314	0.0126689724	0.012710407	9.42×10^{-6}
(Garg, 2019)	[0.051689294, 0.356723362, 11.28863614]	0.01266523278	0.0126655	0.01266555	0.01266830	8.651×10^{-7}
(Long et al., 2019)	[0.0520217, 0.364768, 10.8323]	0.012667	NA	NA	NA	NA
(Chegini et al., 2018)	[0.0518836, 0.36141614, 11.018738]	0.012665923	NA	NA	NA	NA
(Salimi, 2015)	[0.05168, 0.35671, 11.28896]	0.012665232788	0.012665232788	0.012665232788	0.012665232788	1.585×10^{-16}
This work	[0.051647047504, 0.35569064667, 11.351872381]	0.012667928876	0.012697001508	0.012824071721	0.013801619673	0.000279

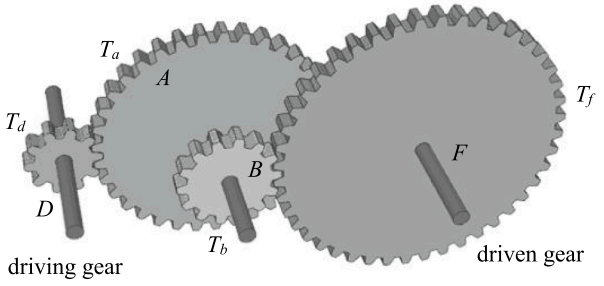


Fig. 11. Schematic of gear train optimization problem.

Fig. 14. The T-test results in h-value of 1 and p-value of 4.61600684122257e-173; Wilcoxon signed-rank test also results in the same h-value and p-value of 1.73439762832058e-06.

3.1.8. Tubular column design optimization

The schematic view of the tabular column design optimization is presented in Fig. 15. Optimization aims to minimize the cost (Hsu and Liu, 2007). The column is under a $P = 2500\text{kg.f}$ vertical load, with yield stress $\sigma_y = 500\frac{\text{kg.f}}{\text{cm}^2}$, module of elasticity $E = 0.85 \times 10^6\frac{\text{kg.f}}{\text{cm}^2}$ and length of $L = 250\text{cm}$. The constraints were defined based on the geometry of the design and allowable stress. The variables are selected as the nominal diameter of the column $d = x_1$, and half of the thickness of pipe wall $t = x_2$.

The optimization problem is presented as (Gandomi et al., 2013):

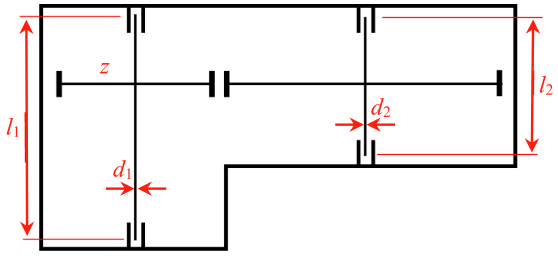


Fig. 12. Schematic view of the speed reducer example.

Table 5
Results of the gear train optimization case study; NA represents not available and SD represents standard deviation, 30 trials.

Method	x, Transmission Ratio	$f(\mathbf{x})$				SD
		Best	Median	Mean	Worst	
(Sandgren, 1988)	[18, 22, 45, 60], 0.146667	5.712×10^{-6}	NA	NA	NA	NA
(Kannan & Kramer, 1994)	[13, 15, 33, 41], 0.144124	2.146×10^{-8}	NA	NA	NA	NA
(Deb & Goyal, 1996)	[19, 16, 49, 43], 0.144281	2.701×10^{-12}	NA	NA	NA	NA
(Gandomi et al., 2013)	[19, 16, 43, 49], 0.144281	2.701×10^{-12}	NA	1.9841×10^{-9}	2.3576×10^{-9}	3.5546×10^{-9}
(Garg, 2019)	[19, 16, 43, 49], 0.14428096	$2.7008571 \times 10^{-12}$	$9.9215795 \times 10^{-10}$	1.2149276×10^{-9}	3.2999231×10^{-9}	8.77×10^{-10}
This work	[16, 19, 43, 49], 0.14428096	$2.7008571 \times 10^{-12}$	2.8287819×10^{-9}	1.7433495×10^{-8}	1.404475×10^{-7}	2.98×10^{-8}

Table 6
Speed reducer optimization data; NA represents not available and SD represents standard deviation, 30 trials.

Method	x	$f(x)$				SD
		Best	Median	Mean	Worst	
(Ku, Rao, & Chen, 1998)	[3.6, 0.7, 17, 7.3, 7.8, 3.4, 5]	2876.117623	NA	NA	NA	NA
(Ray & Saini, 2001)	[3.514185, 0.700005, 17, 7.497343, 7.8346, 2.9018, 5.0022]	2732.9006	NA	2741.5642	2757.8581	NA
(Akhtar et al., 2002)	[3.506122, 0.700006, 17, 7.549126, 7.85933, 3.365576, 5.289773]	3008.08	NA	3012.12	3028	NA
(Mezura-Montes, Coello, & Landa-Becerra, 2003)	[3.506163, 0.700831, 17, 7.460181, 7.962143, 3.3629, 5.3090]	3025.005	NA	3088.7778	3078.5918	NA
(Ray & Liew, 2003)	[3.50000681, 0.70000001, 17, 7.32760205, 7.71532175, 3.35026702, 5.28665450]	2994.744241	3001.758264	3001.7582264	3009.964736	4.0091423
(Raj & Sharma, 2005)	[3.5000711, 0.7, 17, 7.3, 7.8207280, 2.9001726, 5.0000046]	2724.055	NA	NA	NA	NA
(Mezura-Montes et al., 2007)	[3.500010, 0.7, 17, 7.300156, 7.800027, 3.350221, 5.286685]	2996.356689	NA	2996.367220	NA	8.2×10^{-3}
(Cagnina et al., 2008)	[3.5, 0.7, 17, 7.3, 7.8, 3.350214, 5.286683]	2996.348165	NA	2996.3482	NA	NA
(Zhang et al., 2008)	[3.5, 0.7, 17, 7.3, 7.7153199115, 3.3502146661, 5.286654465]	2994.471066	2994.471066	2994.471066	2994.471066	3.58×10^{-12}
(Gandomi et al., 2013)	**[3.5015, 0.7, 17, 7.6050, 7.8181, 3.3520, 5.2875]**	**3000.9810**	**NA**	**3007.1997**	**NA**	**4.9634**
(Garg, 2019)	**[3.499682, 0.699934, 16.999234, 7.300392, 7.799441, 2.89971, 5.286725]**	**2894.73832**	**2894.97128**	**2894.71248**	**2895.03219**	**4.96×10^{-4}**
This work	[3.50296695067,0.700184567453,17,7.386082258567,87534411985,3.350943775155,5.288576065641]	3002.1524434	3009.97423580	3016.20126195	3186.50080487	32.4107
	[3.500745324 6,0.700027445 98,17,7.440979027 2,7.821596143 1,2.902838593 39,5.287475083 6]	**2899.1336453**	**2907.9016035**	**2913.4026595**	**2951.5187977**	**13.3824**

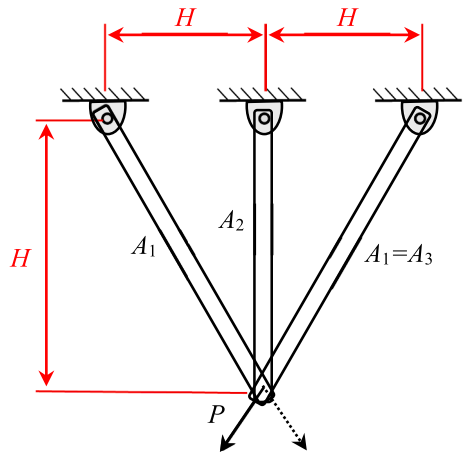


Fig. 13. Three-bar truss optimization problem.

minimize :

$$f(\boldsymbol{x}) = 9.82x_1x_2 + 2x_1, \tag{26}$$

where the constraints are defined:

subject to :

$$g_1(\mathbf{x}) = \frac{P}{\pi x_1 x_2 \sigma_y} - 1 \leq 0, g_2(\mathbf{x}) = \frac{8PL^2}{\pi^3 E x_1 x_2 (x_1^2 + x_2^2)} - 1 \leq 0,$$

$$g_3(\mathbf{x}) = \frac{2}{x_1} - 1 \leq 0, g_4(\mathbf{x}) = \frac{x_1}{14} - 1 \leq 0,$$

$$g_5(\mathbf{x}) = \frac{0.2}{x_2} - 1 \leq 0, g_6(\mathbf{x}) = \frac{x_2}{0.8} - 1 \leq 0, \tag{27}$$

where $2 \leq x_1 \leq 14$ and $0.2 \leq x_2 \leq 0.8$. The repetitions in the three zones are defined as $N_1 = 5000$, $N_2 = 10000$ and $N_3 = 20000$ with reduction search area terms $P_1 = 0.5$ and $P_2 = 0.01$. The best result for 30 trials was gained $f(\mathbf{x}) = 26.5315664916005$ with corresponding design parameters $\mathbf{x} = [5.45120833968508, 0.291965197101054]$. The average time of each trial was found around 7.9 s. The comparison is presented in Table 8. The results of the best solution were gained by 23 generation loops though the maximum generation loops for one of the answers was 28, Fig. 16. The T-test results in h-value of 1 and p-value of 1.32262244843709e-129; Wilcoxon signed-rank test also results in the same h-value and p-value of 1.73439762832058e-06.

3.1.9. Flapping wing design optimization

This design optimization problem aims to present an optimal wing in terms of weight and lift force. The area of the wing is proportional to the lift force and consequently must be maximized. The weight of the carbon-fiber rods, the skeleton of the wing, should be minimized as well. The structure of the flapping wing bird is presented in Fig. 17. There are 10 points for design optimization; the positions of them are set as variables to find the best shape, Fig. 18.

The position of nodes 1 and 2 are fixed, and the total length of the wing is also fixed $d_1 + d_2 + d_3 + d_4 = 230$mm, $Y_5 = Y_6 = 230$mm. The wing is supported by 4 carbon-fiber tubes, horizontal, between nodes 2,9 3,8 4,7 5,6 for l_1, l_2, l_3, l_4 with respect. Based on that, $Y_2 = Y_9$, $Y_3 = Y_8$, $Y_4 = Y_7$, and $Y_5 = Y_6$. The variables are the position of nodes, defined as $x_1 = X_2$, $x_2 = Y_2$, $x_3 = X_3$, $x_4 = Y_3$, $x_5 = X_4$, $x_6 = Y_4$, $x_7 = X_5$, $x_8 = X_6$, $x_9 = X_7$, $x_{10} = X_8$ and $x_{11} = X_9$. The nominal coordinates of the wing are $X_{\text{nom}} = [50, 47.5, 45, 40, 20, -40, -75, -90, -95, -100]$mm which present the X positions of the 10 points, concerning reference XY and similarly $Y_{\text{nom}} = [0, 40, 80, 145, 230, 230, 145, 80, 40, 0]$mm. The

Table 7
Results of the three-bar truss problem; NA represents not available and SD represents standard deviation, 30 trials.

Method	x	$f(\mathbf{x})$				SD
		Best	Median	Mean	Worst	
(Hernández, 1994)	[0.788, 0.408]	263.9	NA	NA	NA	NA
(Ray & Liew, 2003)	[0.7886210370, 0.4084013340]	263.8958466	263.8989	263.9033	263.96975	1.26×10^{-2}
(Ray & Saini, 2001)	[0.795, 0.395]	264.3	NA	NA	NA	NA
(Raj & Sharma, 2005)	[0.78976441, 0.40517605]	263.89671	NA	NA	NA	NA
(Tsai, 2005)	[0.788, 0.408]	263.68	NA	NA	NA	NA
(Zhang et al., 2008)	[0.7886751359, 0.40824 8286 8]	263.8958434	263.8958434	263.8958436	263.8958498	9.72×10^{-7}
(Gandomi et al., 2013)	[0.78867, 0.40902]	263.9716	NA	264.0669	NA	9×10^{-5}
(Garg, 2019)	[0.788676171219, 0.408245358456]	263.8958433	263.8958436	263.8958437	263.8958459	5.34×10^{-7}
(Ahmadianfar et al., 2022)	[0.788672734, 0.408255081]	263.8958434	NA	NA	NA	NA
This work	[0.78875226015122, 0.408030431521627]	263.89587188	263.896217121	263.896304847	263.897551326	3.98729×10^{-4}

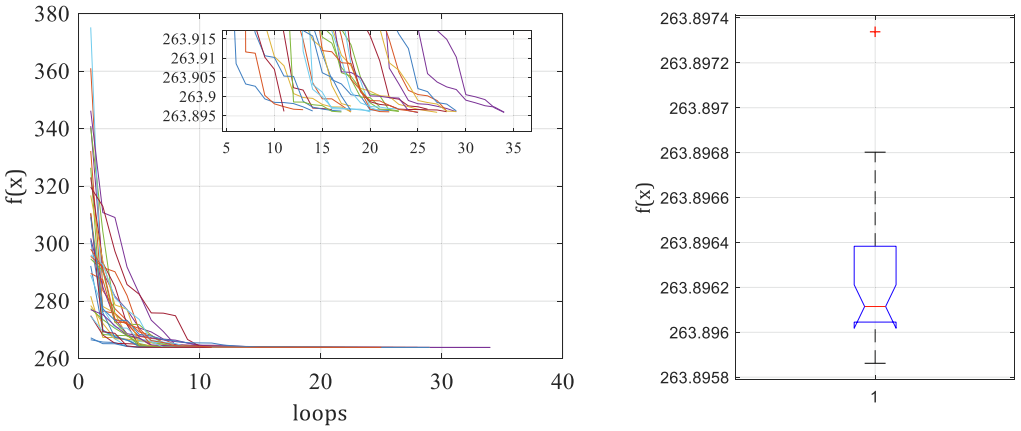


Fig. 14. The results of the three-bar truss; the left figure shows the flow of the solution and the right shows the statistics of the answers.

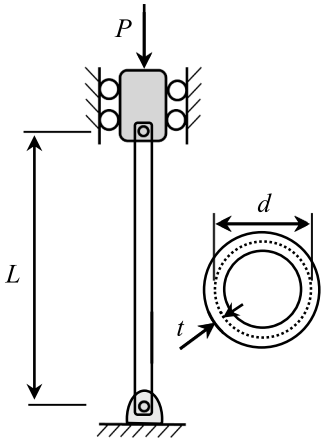


Fig. 15. Tabular column optimization problem with a cross-sectional view of the column.

density of the carbon-fiber is $\rho = 0.00175 \frac{kg}{mm^3}$ and the inner and outer diameters of the rods are 6 mm and 4 mm. The area of each zone is approximated by a trapezoid. The objective function is.

minimize :

$$f(\mathbf{x}) = \frac{1}{A_f(\mathbf{x})} + w\rho L(\mathbf{x})A_t, \tag{28}$$

where $A_f(\mathbf{x})$ is the area of the wing, $A_t(\mathbf{x})$ is the cross-sectional area of the carbon-fiber rods, $L(\mathbf{x})$ is the total length of carbon-fiber rods and $w = 0.0285$ is a weighting parameter to provide a trade-off between lift force and weight:

$$\begin{aligned} A_f(\mathbf{x}) &= A_1(\mathbf{x}) + A_2(\mathbf{x}) + A_3(\mathbf{x}) + A_4(\mathbf{x}) \\ &= \frac{150 + L_1(\mathbf{x})}{2} d_1(\mathbf{x}) + \frac{L_1(\mathbf{x}) + L_2(\mathbf{x})}{2} d_2(\mathbf{x}) + \frac{L_2(\mathbf{x}) + L_3(\mathbf{x})}{2} d_3(\mathbf{x}) \\ &\quad + \frac{L_3(\mathbf{x}) + L_4(\mathbf{x})}{2} d_4(\mathbf{x}), \end{aligned}$$

$$L(\mathbf{x}) = L_1(\mathbf{x}) + L_2(\mathbf{x}) + L_3(\mathbf{x}) + L_4(\mathbf{x}),$$

Table 8
Results of the tabular column optimization problem; NA represents not available and SD represents standard deviation, 30 trials.

Method	x	$f(\mathbf{x})$				SD
		Best	Median	Mean	Worst	
(Hsu & Liu, 2007)	[5.4507, 0.2920]	25.5316	NA	NA	NA	NA
(Rao, 2019)	[5.44, 0.293]	26.5323	NA	NA	NA	NA
(Gandomi et al., 2013)	[5.45139, 0.29196]	26.5321	NA	26.53504	26.53972	0.00193
(Garg, 2019)	[5.45115623, 0.29196548]	26.531328	26.531330	26.531332	26.55315	3.94×10^{-4}
This work	[5.45120833968, 0.291965197101]	26.5315664916	26.533128295	26.5336205207	26.5384331354	0.001604

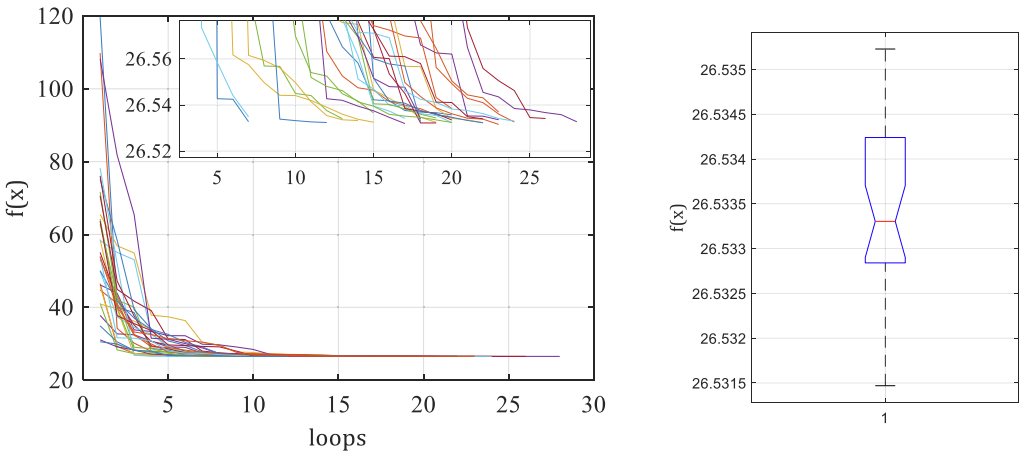


Fig. 16. The results of the tabular column optimization problem; the left figure shows the flow of the solution and the right shows the statistics of the answers.

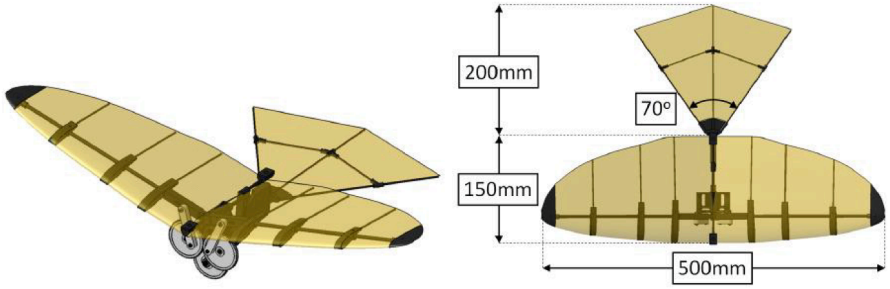


Fig. 17. The flapping wing flying robot (Martín-Alcántara, Grau, Fernandez-Feria, & Ollero, 2019).

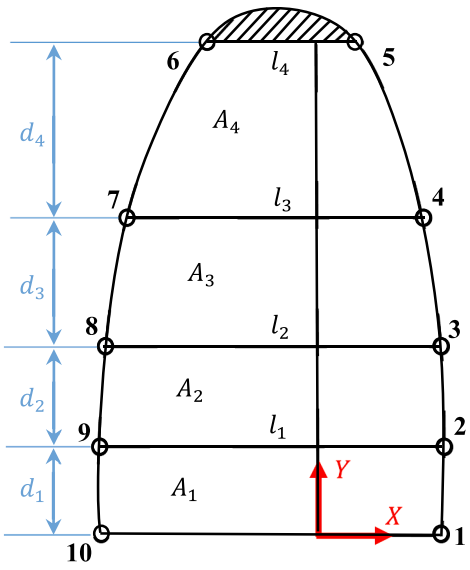


Fig. 18. Axis and points definition of one wing.

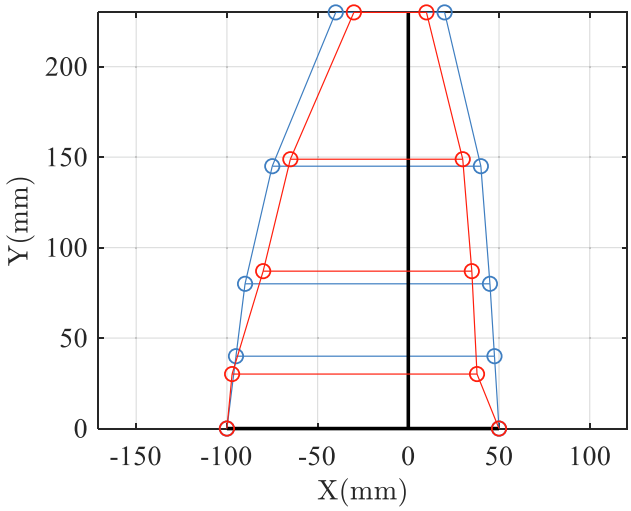


Fig. 19. Optimization of the wing design; blue lines are nominal points and red ones are the optimized form.

$$A_t = \frac{\pi}{4}(6^2 - 4^2), \tag{29}$$

in which $L_1(\mathbf{x}) = x_1 - x_{11}$, $L_2(\mathbf{x}) = x_3 - x_{10}$, $L_3(\mathbf{x}) = x_5 - x_9$, $L_4(\mathbf{x}) = x_7 - x_8$, $d_1(\mathbf{x}) = x_2$, $d_2(\mathbf{x}) = x_4 - x_2$, $d_3(\mathbf{x}) = x_6 - x_4$ and $d_4(\mathbf{x}) = 230 - x_6$. The constraints are imposed on $d_2(\mathbf{x})$ that must be bigger than 55 mm and $L_1(\mathbf{x})$ bigger than 135 mm:

subject to :

$$g_1(\mathbf{x}) = \frac{55}{d_2(\mathbf{x})} - 1 \le 0, g_2(\mathbf{x}) = \frac{135}{L_1(\mathbf{x})} - 1 \le 0, \tag{30}$$

where the bounds of the variables are $\mathbf{x}_{\min} = [37.5, 30, 35, 70, 30, 135, 10, -50, -85, -100, -105]$ and $\mathbf{x}_{\max} = [57.5, 50, 55, 90, 50, 155, 30, -30, -65, -80, -85]$. Setting the parameters as $N_1 = 1000$, $N_2 = 2500$, $N_3 =$

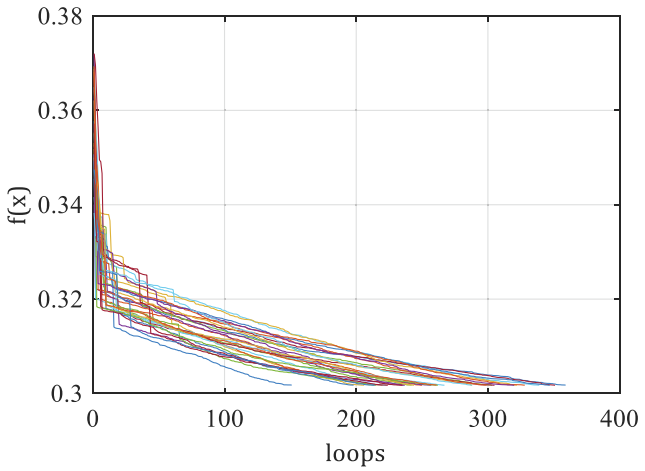


Fig. 20. The cost function of the wing optimization example.

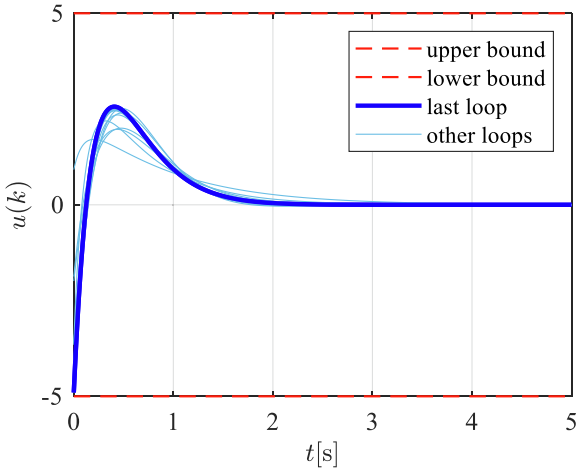


Fig. 21. The control signal for gain optimization of PD control for Van der Pol oscillator.

Table 9
The best position variables of the wing optimization example.

x(mm)	37.8764206883966
	30.122653515995
	35.0036196940535
	86.9952323879193
	30.0002460694343
	148.849196926496
	10.0058639345766
	−30.0074089732872
	−65.0213439814377
	−80.0139490298644
	−97.1419881529652

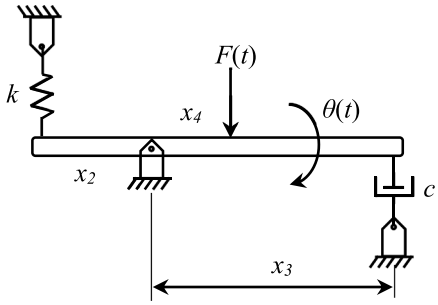


Fig. 22. Schematic of forced vibration of beam.

5000, and reduction terms $P_1 = 0.5$ and $P_2 = 0.01$, the simulation results are found. The simulation has been repeated 30 times, each trial recorded an average time of 15 s. The best result of the cost function was found $f(\mathbf{x}) = 0.301720621682752$; and median 0.301755377466941, mean 0.301754581714707, worst 0.301792260428475 and standard deviation $1.73201192480191 \times 10^{-5}$ were obtained. The new design of the wing based on the cost function is presented in Fig. 19. The flow of the solutions is also presented in Fig. 20. The best position variables of the example are also presented in Table 9. The T-test results in h-value of 1 and p-value of 1.33460623900505e-125; Wilcoxon signed-rank test also results in the same h-value and p-value of 1.73439762832058e-06.

3.2. Optimization for dynamic problems

3.2.1. Gain optimization of PD control design for a Van der Pol oscillator

The goal of this problem is to find the best gain for a proportional derivative (PD) controller to minimize the error of regulation for a Van der Pol oscillator. Consider the second-order differential equation in the state-space form:

$$\dot{\mathbf{x}} = \begin{bmatrix} x_2 \\ -x_1 + \mu(1 - x_1^2)x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u, \tag{31}$$

where $\mathbf{x} = [x_1, x_2]^T$ are the states and u is the input of the system. The initial condition is selected as $\mathbf{x}(0) = [1, -1]^T$, time of simulation 5 s and $\mu = 0.25$. Without a control input, the system will oscillate in a loop that μ changes the shape of oscillation. A PD input law is defined to regulate system (31) to zero to remove the oscillations:

$$u = -k_P x_1 - k_D x_2. \tag{32}$$

To have a stable control law, k_P and k_D, in (32) must be positive and bounded since the control law should be feasible. Here the objective is to minimize the error by defining the best $0 < k_P < 20$ and $0 < k_D < 20$ subjected to constraint $-5 \le u \le 5$. The repetitions parameters are selected as $N_1 = 1000$, $N_2 = 2000$ and $N_3 = 5000$ with reduction search area terms $P_1 = 0.1$ and $P_2 = 0.001$. The simulation repeated for 30 trials to produce the results. The best gains were found $k_P = 12.49515242$, $k_D = 7.495471616$ and resulted in best error (at the final time $t_f = 5$s) $E = \sqrt{x_1^2(5) + x_2^2(5)} = 8.78772391903852 \times 10^{-9}$ in 23 generations of gains. The median of errors is gained $1.09243776446006 \times 10^{-8}$, mean $1.32947634672588 \times 10^{-8}$, worst $3.57186809630302 \times 10^{-8}$ and standard deviations $6.7814411733861 \times 10^{-9}$. The control signal is presented in Fig. 21. The T-test results in h-value of 1 and p-value of 2.03817727850333e-06; Wilcoxon signed-rank test also results in the same h-value and p-value of 2.4051050298461e-12.

3.2.2. Forced vibration of a beam

Consider a beam subjected to a harmonic load $F(t) = 1000\sin 50t$(N) with a distance x_4(m) from the pivot point of the beam, see Fig. 22. The harmonic load generates rotary forced vibration for the beam θ(rad). The maximum allowable vibration of the beam is $\theta_{\max} = 1^\circ$. The optimization problem is to reduce the weight of the structure and reach the dynamic response of the beam near the maximum amplitude constraint $\theta(t) < \theta_{\max}$. To define the maximum amplitude of the vibrating system, the dynamic load factor is used $R_d(\mathbf{x}) = \theta_{\text{dyn, max}}/\theta_{\text{st, max}}$; the details of mathematical optimization problems could be followed in vibration reference textbooks (Kelly, 2012). The beam is attached to a frame by a spring $k = x_5$(N/m) with a distance x_2(m) and a damper $c = x_6$(N.s/m) with a distance x_3(m) at both ends. The mass of the beam is $m = x_1$(kg) and the length of that is $L(\mathbf{x}) = x_2 + x_3$(m).

The mathematical optimization problem can be stated as:

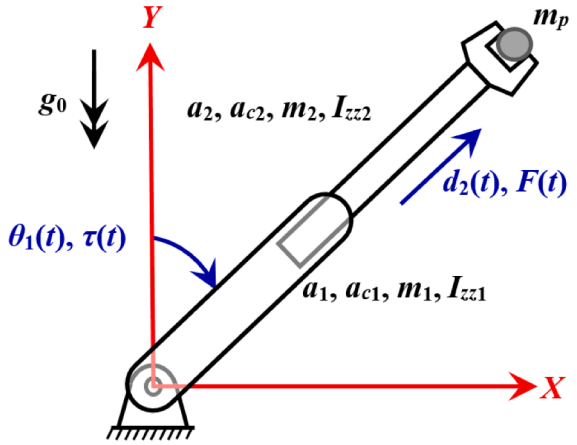


Fig. 23. Schematic of the revolute-prismatic manipulator (Nekoo & Irani Rahaghi, 2018).

$$f(\mathbf{x}) = \frac{1}{I_{\text{eq}}(\mathbf{x})} + c_{\text{eq}}(\mathbf{x}) + k_{\text{eq}}(\mathbf{x}) + \frac{k_{\text{eq}}(\mathbf{x})}{1000 x_4 R_{\text{d}}(\mathbf{x})}, \tag{33}$$

where the constraints are.

$$g_1(\mathbf{x}) = \frac{1000 x_4 R_{\text{d}}(\mathbf{x})}{k_{\text{eq}}(\mathbf{x})\left(\frac{1\times 2\pi}{360}\right)} - 1 \le 0, g_2(\mathbf{x}) = \frac{x_4}{x_3} - 1 \le 0, \tag{34}$$

in which $1 \le x_1 \le 5$, $0.1 \le x_2 \le 0.8$, $1 \le x_3 \le 1.5$, $0.1 \le x_4 \le 1.5$, $3 \times 10^4 \le x_5 \le 5 \times 10^4$, $50 \le x_6 \le 150$, and:

$$R_{\text{d}}(\mathbf{x}) = \frac{1}{\sqrt{(1 - r^2(\mathbf{x}))^2 + (2\zeta(\mathbf{x}) r(\mathbf{x}))^2}}, \zeta(\mathbf{x}) = \frac{c_{\text{eq}}(\mathbf{x})}{2\sqrt{k_{\text{eq}}(\mathbf{x}) I_{\text{eq}}(\mathbf{x})}},$$

$$r(\mathbf{x}) = \frac{50}{\omega_{\text{n}}(\mathbf{x})}, \omega_{\text{n}}(\mathbf{x}) = \sqrt{\frac{k_{\text{eq}}(\mathbf{x})}{I_{\text{eq}}(\mathbf{x})}}, c_{\text{eq}}(\mathbf{x}) = x_6 x_3^2,$$

$$k_{\text{eq}}(\mathbf{x}) = x_5 x_2^2, I_{\text{eq}}(\mathbf{x}) = \frac{1}{12} x_1 L^2(\mathbf{x}) + x_1 \left(\frac{L(\mathbf{x})}{2} - x_2\right)^2.$$

The parameters are set as $N_1 = 1000$, $N_2 = 5000$ and $N_3 = 10000$ with reduction search area terms $P_1 = 0.1$ and $P_2 = 0.01$. The number of trials is 30 and the best answer is found $f(\mathbf{x}) = 407.825142$ with corresponding design parameters $\mathbf{x} = [2.651252, 0.100024429, 1.00088121, 0.2596326, 30001.0209, 50.10418]$. The average time of each trial was found around 27 s. The maximum amplitude of θ was defined 0.9997734°. The median is gained 424.394493, mean 428.092205, worst 479.680436, and standard deviations 18.4445. The T-test results in h-value of 1 and p-value of 3.53780914123903e-41; Wilcoxon signed-rank test also results in the same h-value and p-value of 1.73439762832058e-06.

3.2.3. Control of two-DoF manipulator

The error minimization of a two-degree-of-freedom (DoF) revolute-prismatic manipulator is presented in this subsection as an optimization for dynamic problems. The dynamic of the manipulator and its specifications are based on Section 4-3-2 of Ref. (Nekoo & Irani Rahaghi, 2018). The states of the system are rotation of the first link, linear motion of the second link, the angular velocity of the first link, and linear velocity of the second links assembled in the state vector $\mathbf{x} = \left[\mathbf{q}^T, \dot{\mathbf{q}}^T\right]^T = \left[\theta, d, \dot{\theta}, \dot{d}\right]^T$ with respect, Fig. 23.

The state-space representation of the system is:

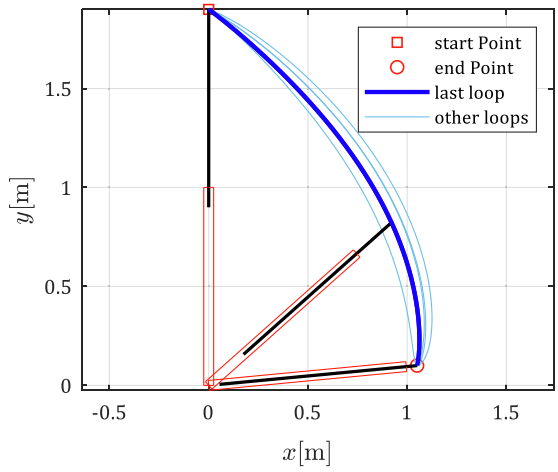


Fig. 24. Evolutions of the trajectories for one of the trials.

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{\mathbf{q}} \\ \mathbf{M}^{-1}(\mathbf{q})[\mathbf{u} - \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) - \mathbf{g}(\mathbf{q})] \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{0} & -\mathbf{M}^{-1}(\mathbf{q})[\mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q})] \end{bmatrix}}_{\mathbf{A}(\mathbf{x})} + \underbrace{\begin{bmatrix} \mathbf{0} \\ \mathbf{M}^{-1}(\mathbf{q}) \end{bmatrix}}_{\mathbf{B}(\mathbf{x})} \mathbf{u}, \tag{35}$$

where $\mathbf{M}(\mathbf{q})$ is inertia matrix, $\mathbf{c}(\mathbf{q}, \dot{\mathbf{q}})$ and $\mathbf{g}(\mathbf{q})$ are Coriolis-centrifugal and gravity vector with respect. The control law is the state-dependent Riccati equation:

$$\mathbf{u} = -\mathbf{R}(\mathbf{x})\mathbf{B}^T(\mathbf{x})\mathbf{K}(\mathbf{x})\mathbf{x}, \tag{36}$$

which is tuned by two weighting matrices $\mathbf{R}(\mathbf{x})$ and $\mathbf{Q}(\mathbf{x})$ in the cost function:

$$J = \int_0^{\infty} \{\mathbf{x}^T \mathbf{Q}(\mathbf{x})\mathbf{x} + \mathbf{u}^T \mathbf{R}(\mathbf{x})\mathbf{u}\} \mathrm{d}t. \tag{37}$$

The suboptimal gain $\mathbf{K}(\mathbf{x})$ is the solution to the SDRE (Nekoo, 2020):

$$\mathbf{A}^T(\mathbf{x})\mathbf{K}(\mathbf{x}) + \mathbf{K}(\mathbf{x})\mathbf{A}(\mathbf{x}) - \mathbf{K}(\mathbf{x})\mathbf{B}(\mathbf{x})\mathbf{R}^{-1}(\mathbf{x})\mathbf{B}^T(\mathbf{x})\mathbf{K}(\mathbf{x}) + \mathbf{Q}(\mathbf{x}) = 0. \tag{38}$$

The definition of weighting matrices are usually forcing us to select diagonal positive definite and semi-definite matrices for $\mathbf{R}(\mathbf{x})$ and $\mathbf{Q}(\mathbf{x})$, diagonal elements of $\mathbf{R}(\mathbf{x})$ less than 1, and for $\mathbf{Q}(\mathbf{x})$ greater than 1. So, the idea of this section is to assign the best weighting matrix $\mathbf{Q} = \text{diag}(Q_{11}, Q_{22}, Q_{33}, Q_{44})$ for the simulation to minimize the end-effector error. The

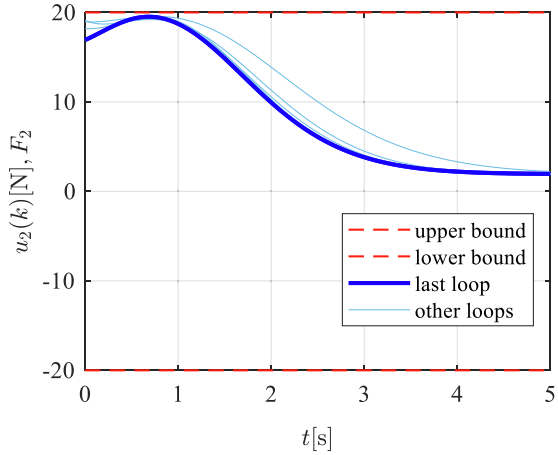


Fig. 25. The control signal of the second link.

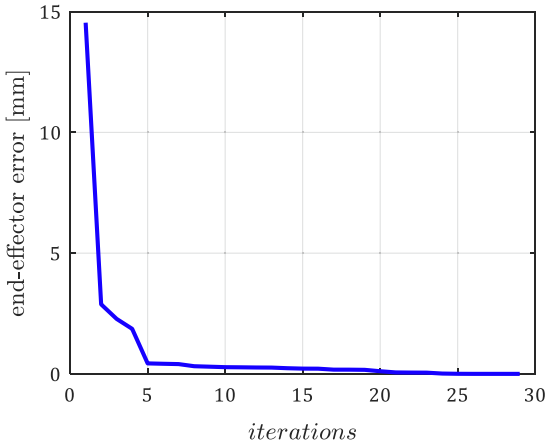


Fig. 26. Error reduction of one of the trials.

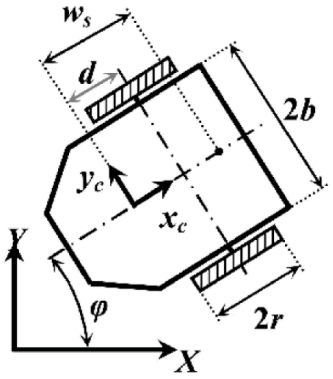


Fig. 27. Schematic view of differential drive wheeled mobile robot (Korayem et al., 2016).

bound of selection is $0 < Q_{ii} < 20$, for $i = 1,\cdots,4$. The parameters are set as $N_1 = 1000$, $N_2 = 2000$ and $N_3 = 5000$ with reduction search area terms $P_1 = 0.1$ and $P_2 = 0.01$. The constraints are set on the inputs $-u_{\min,i} < u_i < u_{\max,i}$, for $i = 1, 2$ and $\mathbf{u}_{\min} = [-24, -20]$ and $\mathbf{u}_{\max} = [24, 20]$. The initial and final conditions of the robot are $\mathbf{x}(0) = [0, 0.9, 0, 0]^T$ and $\mathbf{x}(5) = [1.4758, 0.05475, 0, 0]^T$ with respect. Thirty trials have been performed to find the best and least end-effector error $E =$ $2.80361172875433 \times 10^{-4}$mm with the best selection $\mathbf{Q} = \text{diag}(17.26150676,\ 18.18696608,\ 9.431774017,\ 10.36830457)$. The median of errors is gained $2.11306117961254 \times 10^{-3}$mm, mean $2.67879900515113 \times 10^{-3}$mm, worst $1.06725457555111 \times 10^{-2}$mm and standard deviations $2.23424734755049 \times 10^{-3}$. The evolutions of the trajectories for one of the trials are presented in Fig. 24. The control signal of the second link for one of the trials is illustrated in Fig. 25. The error reduction for the manipulator is presented in Fig. 26.

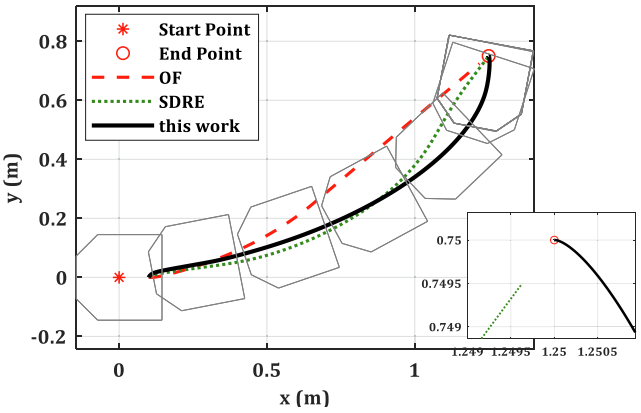


Fig. 29. Trajectory and configuration of the robot in comparison with output-feedback and SDRE method.

3.2.4. Mobile robot error minimization

Consider a wheeled mobile robot with nonlinear dynamics and non-holonomic constraints (Korayem, Nekoo, & Korayem, 2016), illustrated in Fig. 27. The task is to define a control gain for minimizing the error with respect to input constraints due to the limitation of DC motors. The initial and final regulation points are $(0, 0)$ m and $(1.25, 0.75)$m with respect. The proposed approach will find the control gains $\mathbf{K}_P = \begin{bmatrix} * & 0 \\ 0 & * \end{bmatrix}$ and $\mathbf{K}_D = \begin{bmatrix} * & 0 \\ 0 & * \end{bmatrix}$ in a way to minimize the regulation error. The simulation is compared with finite time SDRE and output-feedback linearization with constant gains $\mathbf{K}_P = \begin{bmatrix} 100 & 0 \\ 0 & 100 \end{bmatrix}$ and $\mathbf{K}_D = \begin{bmatrix} 80 & 0 \\ 0 & 80 \end{bmatrix}$. The time of simulation is 3 s and the rest of the parameters

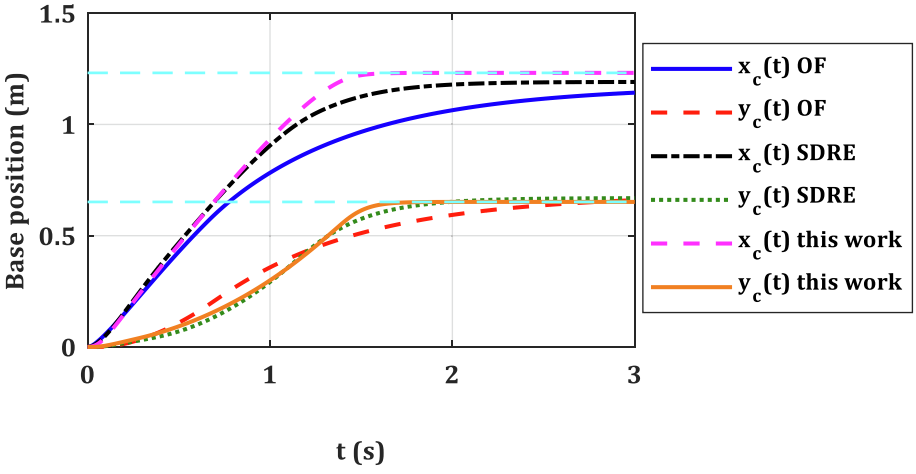


Fig. 28. Position variables of the mobile robot.

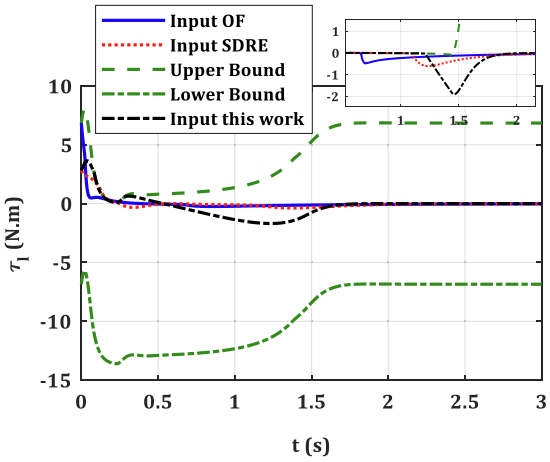


Fig. 30. Input torque of right wheel.

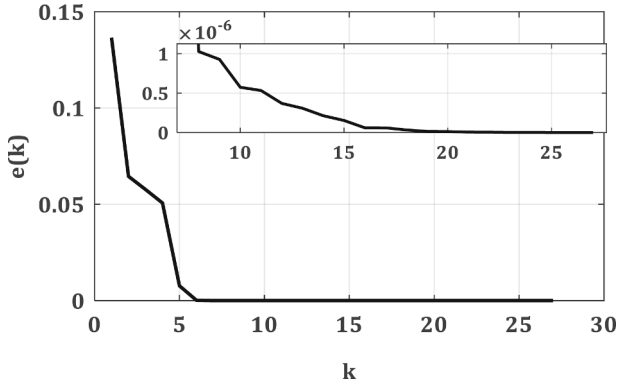


Fig. 31. Error reduction was generated in 27 loops.

are based on Ref. (Korayem et al., 2016).

Selecting parameters as $N_1 = 500, N_2 = 1000, N_3 = 2000, P_1 = 0.2$ and $P_2 = 0.005$, the error is reduced to $E = 5.6014 \times 10^{-7}$mm. The gains were obtained $\mathbf{K}_P = \begin{bmatrix} 100.6066 & 0 \\ 0 & 146.7633 \end{bmatrix}$ and $\mathbf{K}_D = \begin{bmatrix} 19.6680 & 0 \\ 0 & 23.9786 \end{bmatrix}$. The error of the output-feedback method was found at 40.23mm and the finite time SDRE 0.62mm (Korayem et al., 2016). The position variables of the robot are presented in Fig. 28. The trajectory and configuration of the robot in comparison with the output-feedback and SDRE methods are illustrated in Fig. 29. The input torque of the right wheel is shown in Fig. 30. The error reduction rate is also presented in Fig. 31 in 27 loops.

3.3. Test functions

3.3.1. Comparison with previous works

There are some specific test functions for the assessment of meta-heuristic search methods. Three of them were selected for checking the performance of the proposed approach, presented in Table 10. The results of the first function, a unimodal function, are presented in Fig. 32. The second function is multimodal with many local minima which makes it hard for the algorithm to find the global minimum. Lack of constraints and vast search bound, ±90000, are the challenges for the second function. The proposed method could reduce the initial guess from near the bounds to around 100. There are some methods capable of reducing the objective function near zero, the global minimum point for $f_{s,2}$ such as grey wolf optimizer 0.310521 (Mirjalili, Mirjalili, & Lewis, 2014), improved sine cosine algorithm 0 (Long et al., 2019), and sine cosine algorithm 12.7 (Long et al., 2019). This shows that the current approach is weak for high dimensional multimodal functions though the results of the fixed dimensional function $f_{s,3}$ and unimodal function $f_{s,1}$ were satisfactory. All three test functions were computed 30 times for finding standard deviation, best, mean, median and worst answer. The parameters were chosen for test functions as: $N = 10, 20, 30, 40, 50, 60, 70, 80, 100$ and $P = 0.75, 0.1, 0.01, 0.001, 0.0001, 0.00001, 0.000001, 0.0000001$ for the first case, $N = 10, 50, 60, 70, 80$ and $P = 0.5, 0.1, 0.01, 0.001$ for the second case and $N = 60, 90, 100, 110, 120, 130, 140,$

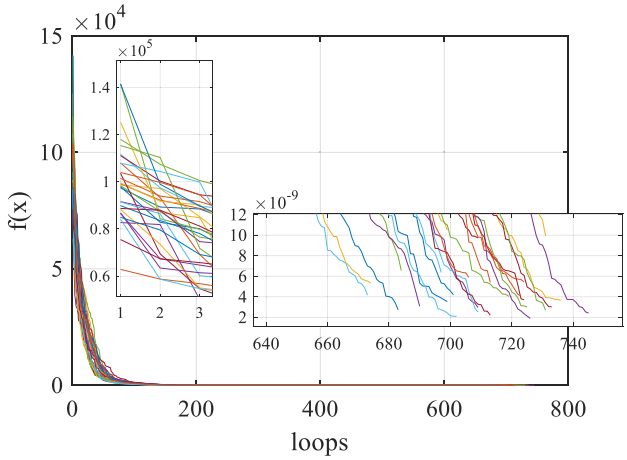


Fig. 32. The results of the test function $f_1(\mathbf{x})$.

Table 10
Test functions, 30 trials.

Test function	Dim,n	Range	Min	Literature	This work
$f_{s,1}(\mathbf{x}) = \sum_i^n x_i^2$	30	[−100,100]	0	Grey Wolf (Mirjalili et al., 2014):6.59×10^{-28} Sine Cosine (Long et al., 2019): 3.24×10^{-1} Improved Sine Cosine (Long et al., 2019): 0	Best:$1.8967425620358 \times 10^{-9}$ Median:$4.05773271258732 \times 10^{-9}$ Mean:$7.8982861739249 \times 10^{-9}$ Worst:$8.24300997218811 \times 10^{-9}$ SD:$1.46118358423316 \times 10^{-9}$
$f_{s,2}(\mathbf{x}) = \sum_i^n (x_i^2 - 10\cos 2\pi x_i + 10)$	30	[−5.12,5.12]	0	Grey Wolf (Mirjalili et al., 2014):0.310521 Sine Cosine (Long et al., 2019): 127 Improved Sine Cosine (Long et al., 2019): 0	Best:99.2426465014127 Median:144.807290814228 Mean:151.986610127804 Worst:216.024394808014 SD:33.0650604967027
$f_{s,3}(\mathbf{x}) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	[−5,5]	−1.0316	Grey Wolf (Mirjalili et al., 2014): −1.03163	Best: −1.03162845348987 Median: −1.03162845348983 Mean: −1.03162845348979 Worst: −1.03162845348913 SD:$1.40306419238579 \times 10^{-13}$

Table 11
Unimodal test functions.

Test function	Dim,n	Range	Min
$f_1(\mathbf{x}) = \left(\sum_{i=1}^{n} x_i^2\right)^2$	30	$[-100,100]$	0
$f_2(\mathbf{x}) = \sum_{i=1}^{n} x_i^2$	30	$[-100,100]$	0
$f_3(\mathbf{x}) = \sum_{i=1}^{n/4}(x_{4i-3} - 10x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + (x_{4i-2} - x_{4i-1})^4 + 10(x_{4i-3} - x_{4i})^4$	40	$[-4,5]$	0
$f_4(\mathbf{x}) = \sum_{i=2}^{n-2}(x_{i-1} - 10x_i)^2 + 5(x_{i+1} - x_{i+2})^2 + (x_i - 2x_{i+1})^4 + 10(x_{i-1} - x_{i+2})^4$	40	$[-4,5]$	0
$f_5(\mathbf{x}) = \sum_{i=1}^{n} \lvert x_i\rvert^{i+1}$	30	$[-1,1]$	0
$f_6(\mathbf{x}) = -\sum_{i=1}^{n} \lvert x_i\rvert$, concave function	30	$[-100,100]$	0
$f_7(\mathbf{x}) = \max\lvert x_i\rvert, i = 1,\cdots,n$	30	$[-100,100]$	0
$f_8(\mathbf{x}) = \sum_{i=1}^{n} \lvert x_i\rvert + \prod_{i=1}^{n} \lvert x_i\rvert$	30	$[-100,100]$	0
$f_9(\mathbf{x}) = \sum_{i=1}^{n} x_i^{10}$	30	$[-10,10]$	0
$f_{10}(\mathbf{x}) = \sum_{i=1}^{n} i x_i^2$	30	$[-10,10]$	0
$f_{11}(\mathbf{x}) = \sum_{i=1}^{n}\left(\sum_{j=1}^{i} x_j\right)^2$	30	$[-100,100]$	0
$f_{12}(\mathbf{x}) = \sum_{i=1}^{n-1}[\![100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]\!]$	30	$[-30,30]$	0
$f_{13}(\mathbf{x}) = \sum_{i=1}^{n}(i x_i^4 + \text{random}(0,1))$	30	$[-1.28,1.28]$	0

150, 160and $P = 0.75, 0.1, 0.01, 0.001, 0.0001, 0.00001, 0.000001$, 0.0000001 for the third case. Increasing the reduction search areas in the method is possible. For high dimensional optimization task such as the test functions in this section, the search areas increased from 3 zones to 9 zones.

3.3.2. Comparison with test functions

Enhanced version experiment and comparison: To cover the simple search algorithm weakness in high-dimensional test functions or optimization problems with vast search domains, the enhanced version is used. The modification is the proposed simple algorithm equipped with Eq. (8) instead of (3). All other specifications and structures are similar to the original version. The assessment and its comparisons are done for the test functions in Table 11 (unimodal), Table 12 (multimodal), and Table 13 (fixed-dimensional). The results are expressed in Table 14 and Table 15.

4. Discussion

The optimization problems are categorized as follows. **Engineering structural optimization**: this sort of problem has many constraints and limited search space which allow the method to perform the search with 3 zones with normal repetition loops. **Dynamic problems**: there exist fewer constraints though they are more time-consuming since, in each loop, the problem is simulated between two boundary conditions within a time span. The search area is limited and the behavior of the search parameters is expected. The application was shown for parameter optimization for limiting vibrations in the mechanical structures and control. The control examples were tuning the gains of three different controllers to indicate the wide range of applications. The first controller was PD, the second one was the state-dependent Riccati equation and the last one was output feedback linearization. The control platforms were also different, a Van der Pole oscillator, a manipulator, and a wheeled mobile robot. **Test functions**: three types of test functions were tested for assessment of the method. The first type is unimodal functions with one minimum. The dimension of the problem resulted in a very vast search space. To overcome the long reducing trajectory of the method from near boundaries to the optimum point, the reduction-search zones increased to 9 stages. The results were gained satisfactory. In the second case, multimodal functions with many local minima and extensive search areas created difficulties for the algorithm and it was not as good as in the unimodal case. With the fixed dimensional test function, the solutions were almost similar to the best available answers. The enhanced version of the proposed search algorithm, was tested and compared for 32 test functions with very vast domains to show the capability and flexibility of the proposed method. It has been concluded that for fixed-dimensional problems and constrained optimizations the simple version is better though, for high-dimensional problems, the enhanced version is more suitable.

The stochastic fractal search approach is a powerful method that considers convergence and accuracy simultaneously (Salimi, 2015). The current method in this work will be compared with SFS to clarify the difference between these two methods. The SFS uses Levy flight to model diffusion-limited aggregation and sweeps the domain of search in the first step of the algorithm. Then for each particle, the electrical potential energy will be computed and finally, a few best particles remain, also with the record of the rest of the particles. The exchange of information between the particles speeds up the convergence. On the contrary, the search algorithm in this current work is focused on the domain of search instead; narrows down the search domain without saving the particles' records. There is no exchange of information between the particles as well. Two examples were chosen to compare the results of the two methods. The welded beam example, Section 3-1-3, shows that SFS scored the value 1.72485230 for the optimization function and the search method in this work scored 1.6977033295, similar to Ref. (Garg, 2019). The SFS used 24, 000 number-of-function-evaluations (NEF), and this work 17, 500. For the second example, SFS scored a better result

Table 12
Multimodal test functions.

Test function	Dim,n	Range	Min
$f_{14}(\mathbf{x}) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i\right) + 20 + e$	30	$[-32,32]$	0
$f_{15}(\mathbf{x}) = \sum_{i=1}^{n} \lvert x_i \sin x_i + 0.1x_i\rvert$	30	$[-10,10]$	0
$f_{16}(\mathbf{x}) = \sum_{i=1}^{n-1}\left(\left(x_i^2\right)^{\left(x_{i+1}^2+1\right)} + \left(x_{i+1}^2\right)^{\left(x_i^2+1\right)}\right)$	30	$[-1,4]$	0
$f_{17}(\mathbf{x}) = -\exp\left(-0.5\sum_{i=1}^{n} x_i^2\right)$	30	$[-1,1]$	-1
$f_{18}(\mathbf{x}) = \frac{1}{4000}\sum_{i=1}^{n}(x_i^2) - \prod_{i=1}^{n}\left(\cos\frac{x_i}{\sqrt{i}}\right) + 1$	30	$[-600,600]$	0
$f_{19}(\mathbf{x}) = \sum_{i=1}^{n}\left(x_i^2 - 10\cos(2\pi x_i) + 10\right)$	30	$[-5.12,5.12]$	0
$f_{20}(\mathbf{x}) = \frac{\pi}{n}\left(10\sin\pi y_1 + \sum_{i=1}^{n-1}(y_i - 1)^2\left(1 + 10\sin^2\pi y_{i+1}\right) + (y_n - 1)^2\right) + \sum_{i=1}^{n} u(x_i, 10, 100, 4) y_i = 1 + \frac{x_i + 1}{4} u(x_i, a, k, m) =$ $\begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a \le x_i \le a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	30	$[-50,50]$	0
$f_{21}(\mathbf{x}) = 0.1\left(\sin^2 3\pi x_1 + \sum_{i=1}^{n}(x_i - 1)^2\left(1 + \sin^2(3\pi x_i + 1)\right) + (x_n - 1)^2\left(1 + \sin^2 2\pi x_n\right)\right) + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	30	$[-50,50]$	0
$f_{22}(\mathbf{x}) = \sum_{i=1}^{n} x_i^2 + \left(\sum_{i=1}^{n} 0.5 i x_i\right)^2 + \left(\sum_{i=1}^{n} 0.5 i x_i\right)^4$	30	$[-5,10]$	0

Table 13
Fixed-dimensional test functions.

Test function	Dim,n	Range	Min
$f_{23}(\mathbf{x}) = \left(\frac{1}{500}+\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6}\right)^{-1}\ a=\begin{bmatrix}-32-1601632-32-1601632-32-1601632-32-1601632-32-1601632\\-32-32-32-32-32-16-16-16-16-16000001616161616323232323232\end{bmatrix}$	2	[−65,65]	1
$f_{24}(\mathbf{x}) = \sum_{i=1}^{11}\left(a_i-\frac{x_1\left(b_i^2+b_ix_2\right)}{b_i^2+b_ix_3+x_4}\right)^2\ a=[0.19570.19470.17350.160.08440.06270.04560.03420.03230.02350.0246]b=\frac{1}{[0.250.51246810121416]}$	4	[−5,5]	0.0030
$f_{25}(\mathbf{x}) = 4x_1^2-2.1x_1^4+\frac{1}{3}x_1^6+x_1x_2-4x_2^2+4x_2^4$	2	[−5,5]	−1.0316
$f_{26}(\mathbf{x}) = \left(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1-6\right)^2+10\left(1-\frac{1}{8\pi}\right)\cos x_1+10$	2	[−5,5]	0.398
$f_{27}(\mathbf{x}) = \left(1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)\right)\left(30+(2x_1-3x_2)^2(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)\right)$	2	[−2,2]	3
$f_{28}(\mathbf{x}) = -\sum_{i=1}^{4}c_i\exp\left(-\sum_{j=1}^{3}a_{ij}\left(x_j-p_{ij}\right)^2\right)\ c=[11.233.2],a=\begin{bmatrix}31030\\0.11035\\31030\\0.11035\end{bmatrix},p=\begin{bmatrix}0.36890.1170.2673\\0.46990.43870.747\\0.10910.87320.5547\\0.038150.57430.8828\end{bmatrix}$	3	[1,3]	−3.86
$f_{29}(\mathbf{x}) = -\sum_{i=1}^{4}c_i\exp\left(-\sum_{j=1}^{6}a_{ij}\left(x_j-p_{ij}\right)^2\right)\ c=[11.233.2],a=\begin{bmatrix}103173.51.78\\0.0510170.1814\\33.51.710178\\1780.05100.114\end{bmatrix}p=\begin{bmatrix}0.13120.16960.55690.01240.82830.5886\\0.23290.41350.83070.37360.10040.9991\\0.23480.14150.35220.28830.30470.6650\\0.40470.88280.87320.57430.10910.0381\end{bmatrix}$	6	[0,1]	−3.32
$f_{30}(\mathbf{x}) = x_1^2-12x_1+11+10\cos\frac{\pi x_1}{2}+8\sin\frac{5\pi x_1}{2}-0.2\sqrt{5}\exp\left(-0.5(x_2-0.5)^2\right)$	2	[−30,30]	−42.9443
$f_{31}(\mathbf{x}) = -\frac{1+\cos\left(12\sqrt{x_1^2+x_2^2}\right)}{0.5(x_1^2+x_2^2)+2}$	2	[−5.12,5.12]	−1
$f_{32}(\mathbf{x}) = 100(x_1^2-x_2)^2+(x_1-1)^2+(x_3-1)^2+90(x_3^2-x_4)^2+10.1\left((x_2-1)^2+(x_4-1)^2\right)+19.8(x_2-1)(x_4-1)$	4	[−10,10]	0

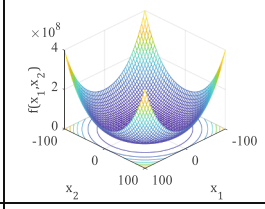
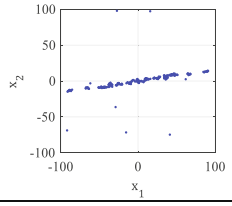
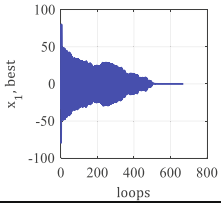
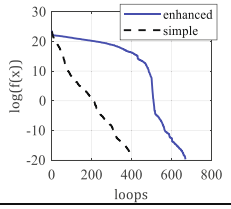
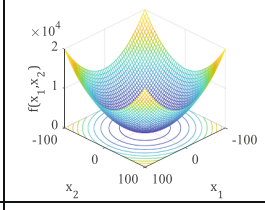
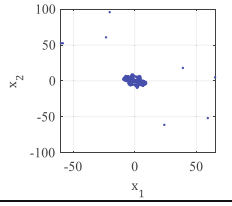
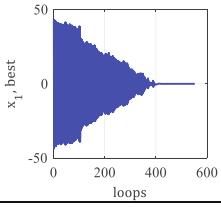
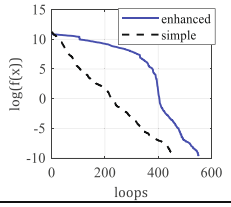
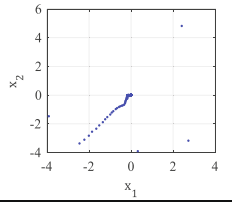
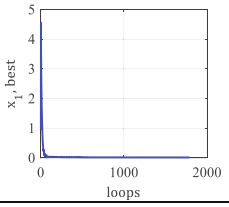
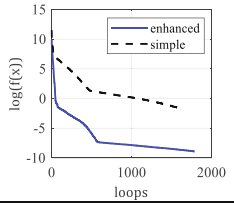
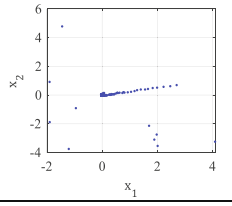
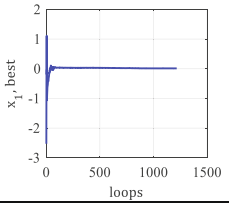
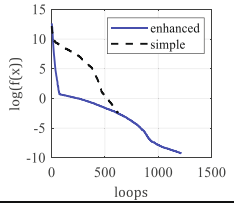
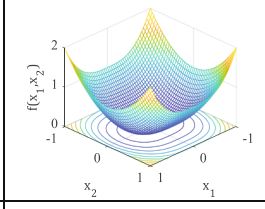
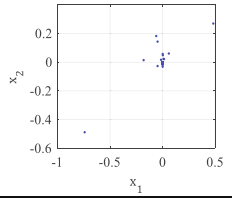
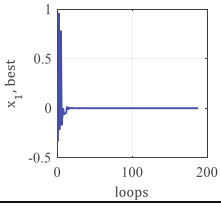
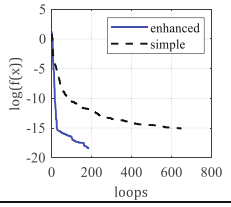
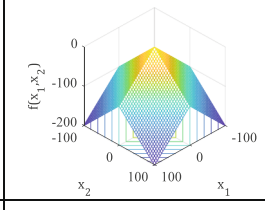
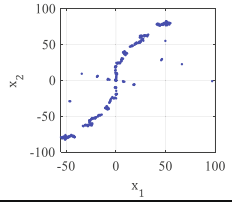
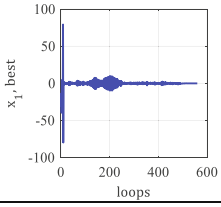
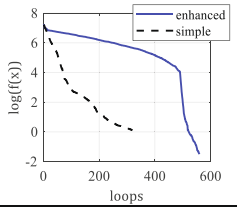
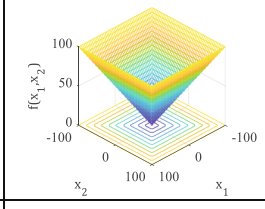
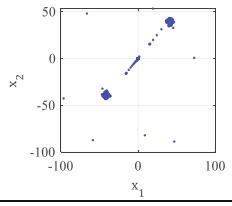
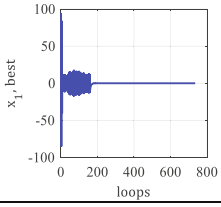
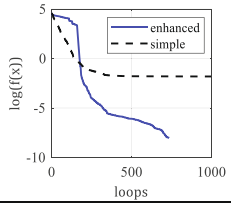
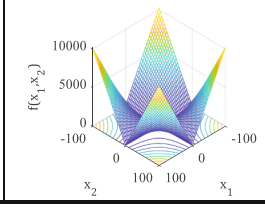
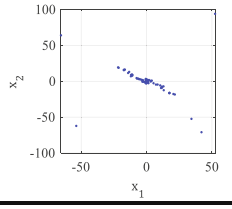
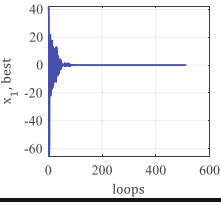
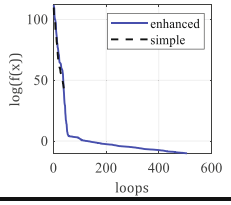
Table 14
The results of the test functions; SD shows standard deviations and PI presents possible improvement; 30 trials.

No.	Type	Best enhanced version	SD enhanced version	p-value T-test enhanced version	Best simple search algorithm	SD simple search algorithm	PI % of enhanced version
1	unimodal function	3.08675094599304E−9	5.44E−7	0.0147517845451004	6.95360195046262E−9	8.99E−7	55.6
2		4.40471211833798E−5	5.49E−4	5.79404620511476E−6	1.01674983925428E−4	6.95E−4	56.6
3		9.88129700602106E−5	1.21E−5	1.07243191421192E−8	4.63263465290848	2.1	99.9
4		3.04116840632009E−5	1E−4	5.02314701838995E−8	3.19784183105535	1.19	99.9
5		1.93901155110529E−9	1.98E−8	2.18161803721525E−7	1.83642379129164E−7	1.77E−7	98.9
6		−0.475324416827596	0.07	1.1869836819655E−25	−41.4024941956355	19.69	98.8
7		7.15459674642637E−5	8.24E−4	7.35392200988825E−6	0.267058877130073	0.4	99.9
8		2.33253899599805E−5	4.9E−3	0.325581988016065	991562.985192921	8.8E27	99.9
9		1.85451609892134E−22	4.78E−15	0.100241125822395	6.21648056136648E−21	8.64E−11	97
10		6.77705877779907E−5	2.79E−4	6.72336658073292E−6	2.88089105571225E−4	3E−3	76.4
11		4.32220719330256E−5	7.02E−5	2.85723356334505E−10	2.63339122311631E−2	0.47	99.9
12		0	1.82	8.5187666571308E−13	61	5.77	100
13		3.45681317080786E−3	9.47E−3	8.18871919539133E−11	4.10911845539854E−2	5.91E−2	91.5
14	multimodal	6.85408534373266E−7	1.05E−7	2.70707501875702E−32	7.67218730324259E−7	0.97	10.6
15	functions	8.62465167570393E−3	2.25E−3	1.97875967649453E−14	4.3815807893475	0.47	99.8
16		1.60696126565815E−14	1.3E−15	0.0542993726706154	1.69744729668881E−14	2.19E−15	5.3
17		−0.999999999999999	1.7E−16	0	−0.999999999999999	2.48E−16	0
18		3.22163407062703E−11	1.55E−11	8.57243944047098E−14	5.29469801335836E−11	5.69E−3	39.1
19		5.40012479177676E−12	3.95E−12	0.0834935549759329	117.404820444782	18.23	99.9
20		0.716974069579892	0.12	6.35745334908942E−30	3.90465374114241	5.84	81.6
21		3.20271431132239	19.72	7.53178654929205E−36	19.0781780322723	46.59	83.2
22		4.73698855301029E−13	1.18E−12	0.243502293843435	7.83523256647346E−13	3.68E−13	39.5
23	fixed-dimensional	0.998023179855449	9.52	0.000620367204066364	0.99800383779445	1.12E−16	1.9E−3
24	functions	3.07523873560042E−4	1.98E−2	0.0315035621578187	5.61917606175115E−4	5.16E−3	45.2
25		−1.0316269834269	3.71E−2	1.9461866476215E−38	−1.03162845348988	4.3E−16	−1.4E−4
26		0.398303289154891	0.85	1.02984186999034E−9	0.397887357729738	2.41E−14	−0.1
27		3.02334942534786	9.57	8.46779535951879E−8	2.99999999999995	6.05E−13	−0.7
28		−3.81616062024405	0.58	1.0466675198387E−20	fail	–	–
29		−3.17757870460904	0.21	5.07380505614456E−32	−3.32199462702961	5.99E−2	−4.3
30		−42.8431437431918	0.15	2.84697846083004E−69	−42.9443869406516	0.15	−0.23
31		−0.999992900529517	0.02	1.38871601492462E−48	−0.999991188979966	0.05	1.7E−4
32		2.53545204245043	8.02	1.14173644443228E−17	2.92047216903384E−12	4.9E−12	–

than this work, tension/compression spring, Section 3.1.4. The optimization function was found at 0.012665232788 for SFS, and 0.012667928876 for this work. The NEF of the SFS was set at 100,000 and this work at 25,000.

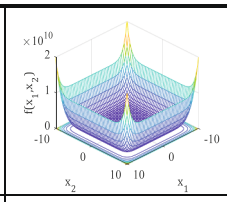
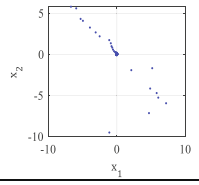
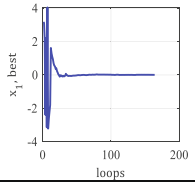
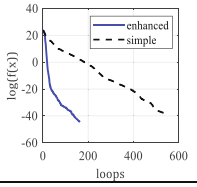
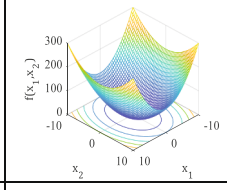
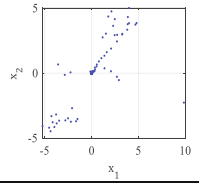
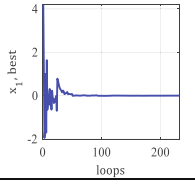
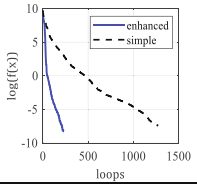
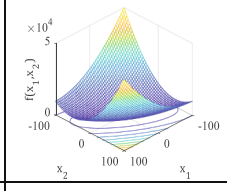
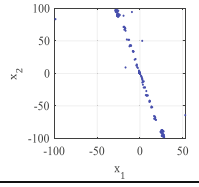
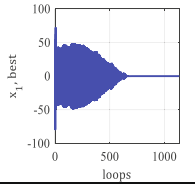
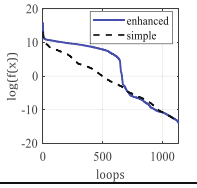
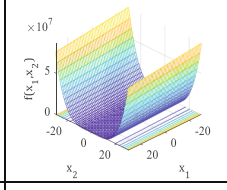
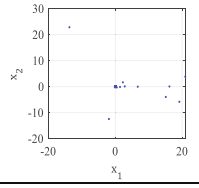
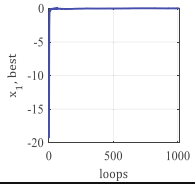
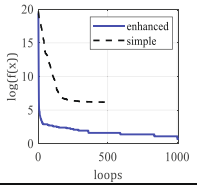
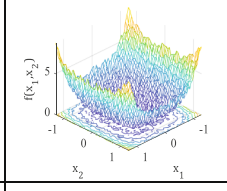
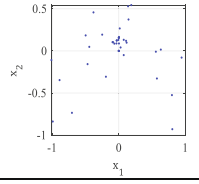
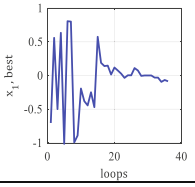
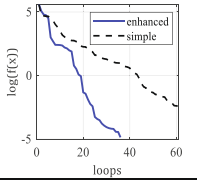
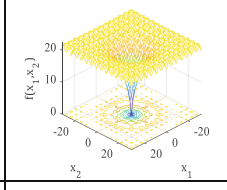
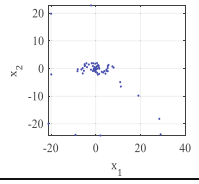
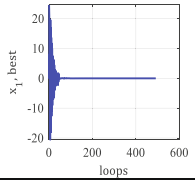
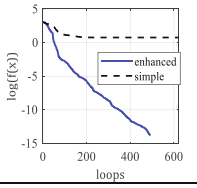
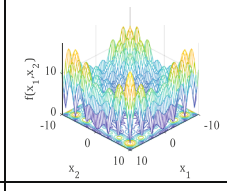
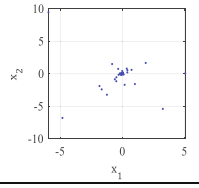
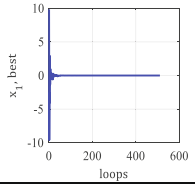
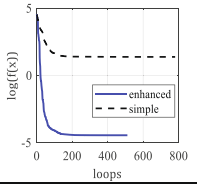
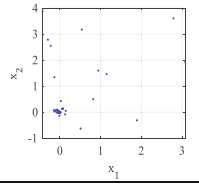
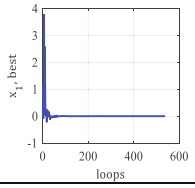
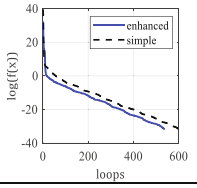
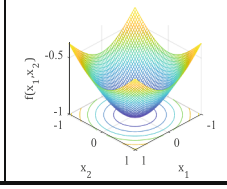
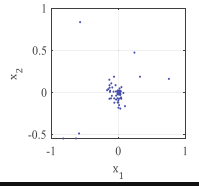
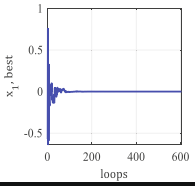
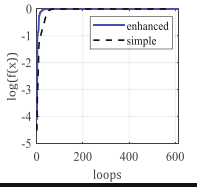
To justify the performance of the method and enhanced version, the results of the search algorithm are compared with the winner of the IEEE

Table 15
The plot results of the test functions.

No.	2D surface plot	2D trajectories	1st trajectory	log plot
1				
2				
3	NA			
4	NA			
5				
6				
7				
8				

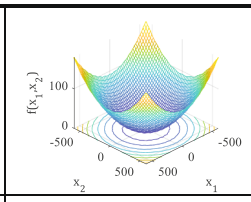
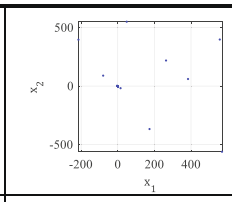
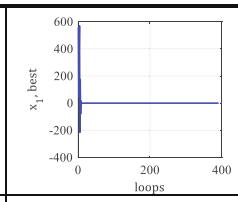
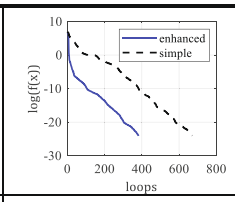
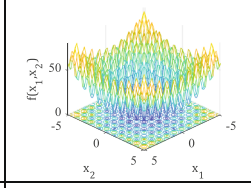
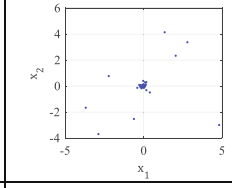
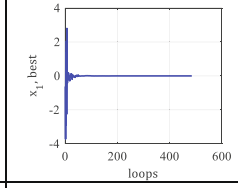
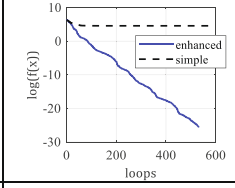
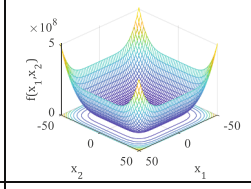
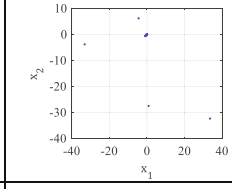
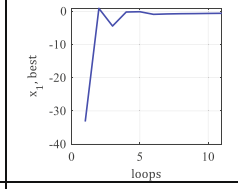
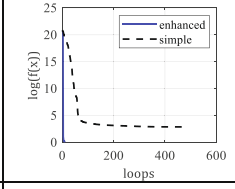
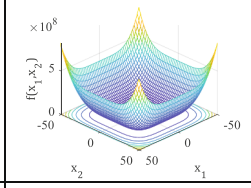
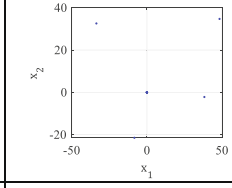
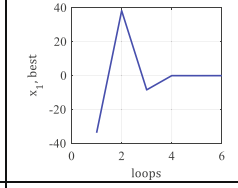
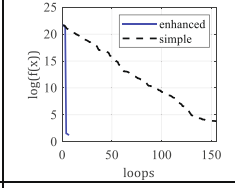
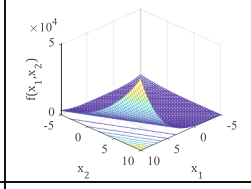
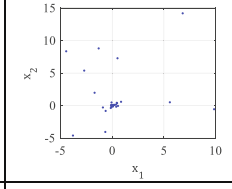
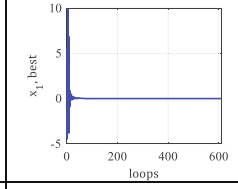
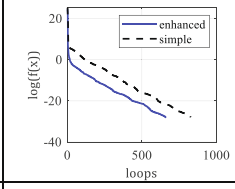
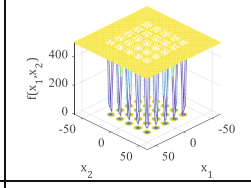
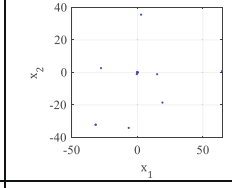
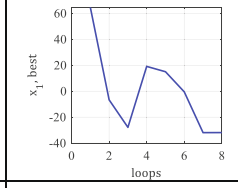
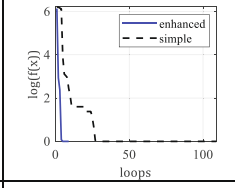
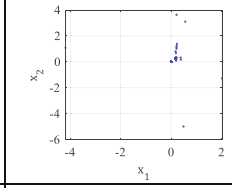
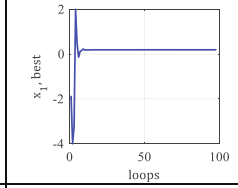
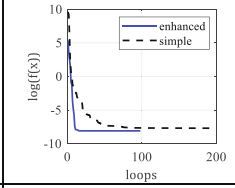
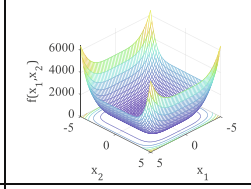
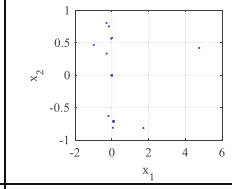
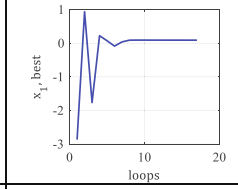
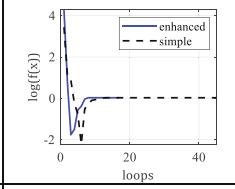
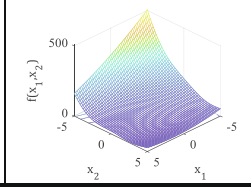
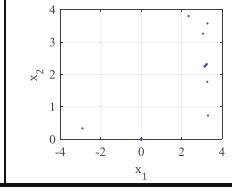
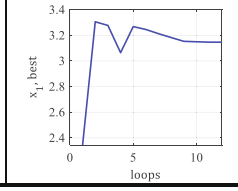
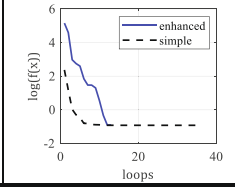
(*continued on next page*)

Table 15 (*continued*)

9				
10				
11				
12				
13				
14				
15				
16	NA			
17				

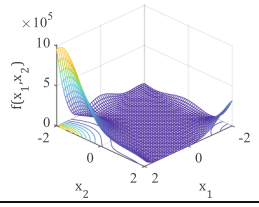
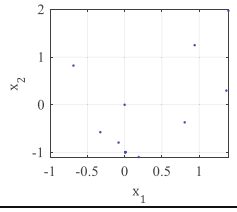
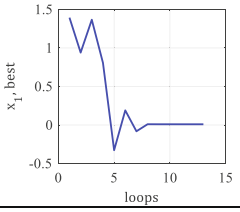
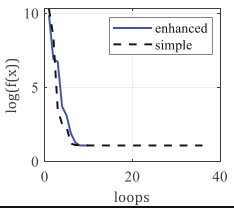
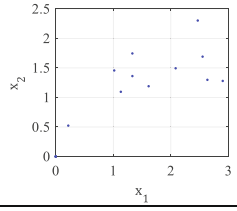
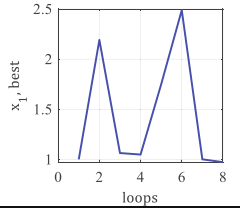
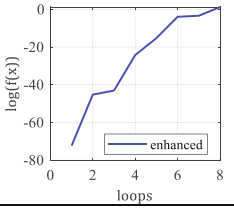
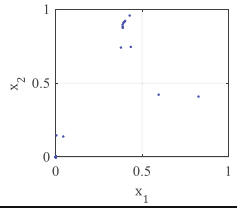
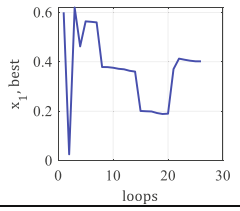
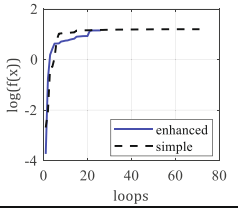
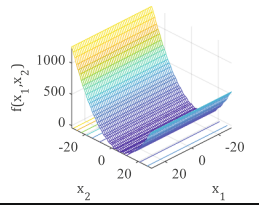
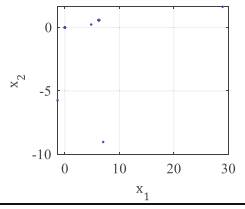
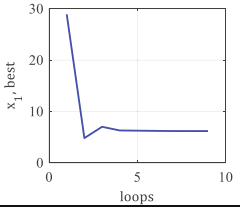
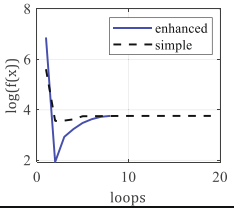
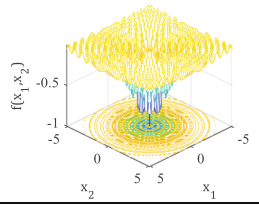
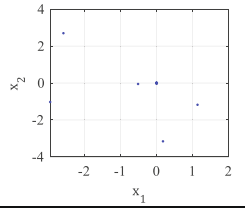
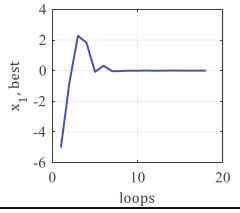
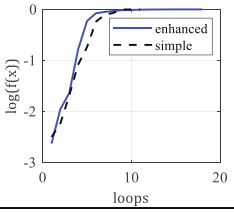
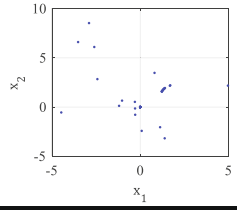
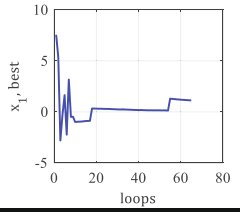
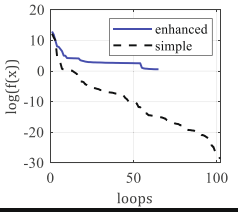
(*continued on next page*)

Table 15 (*continued*)

18				
19				
20				
21				
22				
23				
24	NA			
25				
26				

(*continued on next page*)

Table 15 (*continued*)

27				
28	NA			
29	NA			
30				
31				
32	NA			

Congress on Evolutionary Computation (CEC) 2014. Ackley's function, presented in Table 12, number 14, was modified to be compatible with Ref. (Tanabe & Fukunaga, 2014). The dimension of the equation is 30, and the range of search is $[-100, 100]^{30}$, and a number of 51 iterations were done to compute the statistics. The results of the L-SHADE method scored the best solution 2.0E + 01, the worst answer 2.0E + 01, the median value 2.0E + 01, the mean value 2.0E + 01, and the standard deviation 3.7E −02, which are very good results with respect to the domain of the search (Tanabe & Fukunaga, 2014). This work also scored with the same condition, the best solution 20.9517965443784, the worst solution 21.1939668780108, the median and mean values 21.0872585611148 and 21.0811858734007 respectively, and the standard deviation 0.0575.

INFO: an efficient optimization algorithm based on the weighted mean of vectors, is a recent search algorithm for multi-objective optimization (Ahmadianfar, Heidari, Noshadian, Chen, & Gandomi, 2022). This method is also compared with the proposed search algorithm to provide a performance assessment. Function 9 of Table 3, in Ref. (Ahmadianfar et al., 2022), has been compared with this current work. The INFO scored 00E + 00, similar to this work by increasing the zones. The three-bar truss optimization problem also scored the optimization function value of 263.8958434 for INFO, and the same problem for this work scored 263.89587188, Section 3.1.7, which confirms the correctness of the proposed approach in comparison with other trendy techniques.

5. Conclusions

The search algorithm generates a random solution that makes it a powerful approach for seeking the best answer. The first level of the method searches zone 1 to find the best answer, then focuses around it to find the best answer. It does not rely on the previously generated

solution, in the previous loop. The implementation is easy, simple, and fast. The convergence of the method might be a little time-consuming due to the nature of the method though the wide range of applications justifies the drawback. Most of the evolutionary methods are slow with respect to mathematical methods without trials. The method was applied for mathematical functions, structural optimization in static form, and also for dynamic problems such as vibration and control of a manipulator. The method has been validated by several examples and other previously established methods, scoring a good result. An enhancement was also introduced for optimization problems with excessive large search domains.

All the codes of this work are available as supplementary material in the online version of the paper on the journal website.

Author contributions

SRN: writing initial draft, coding, programming, simulations, review and editing, conceptualization, methodology, validation, investigation. JAA: review and editing, supervision, conceptualization, investigation. AO: supervision, project administration, funding acquisition.

CRediT authorship contribution statement

Saeed Rafee Nekoo: Writing – review & editing, Coding, Programming, Simulations, Conceptualization, Validation, Investigation. **José Ángel Acosta:** Writing – review & editing, Supervision, Conceptualization, Investigation. **Anibal Ollero:** Supervision, Project administration, Funding acquisition.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is supported by the European Research Council as part of GRIFFIN ERC Advanced Grant 2017, Action 788247, and by the European Commission H2020 Programme under AERIAL-CORE project contract number 871479; and received partial support by PAIDI 2020 through the Project HOMPOT under Grant PY20_00597.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.eswa.2022.117866.

References

Agrawal, S., Samantaray, L., Panda, R., & Dora, L. (2020). A new hybrid adaptive Cuckoo search-squirrel search algorithm for brain MR image analysis *Hybrid Machine Intelligence for Medical Image Analysis* (pp. 85-117): Springer.

Ahmadianfar, I., Heidari, A. A., Noshadian, S., Chen, H., & Gandomi, A. H. (2022). INFO: An efficient optimization algorithm based on weighted mean of vectors. *Expert Systems with Applications, 195*, Article 116516.

Akay, B., & Karaboga, D. (2012). Artificial bee colony algorithm for large-scale problems and engineering design optimization. *Journal of Intelligent Manufacturing, 23*(4), 1001–1014.

Akhtar, S., Tai, K., & Ray, T. (2002). A socio-behavioural simulation model for engineering design optimization. *Engineering Optimization, 34*(4), 341–354.

Alizadeh, F., & Goldfarb, D. (2003). Second-order cone programming. *Mathematical Programming, 95*(1), 3–51.

Aragón, V. S., Esquivel, S. C., & Coello, C. A. C. (2010). A modified version of a T-Cell algorithm for constrained optimization problems. *International Journal for Numerical Methods in Engineering, 84*(3), 351–378.

Beeler, S. C., Tran, H. T., & Banks, H. T. (2000). Feedback control methodologies for nonlinear systems. *Journal of Optimization Theory and Applications, 107*(1), 1–33.

Belegundu, A. D. (1982). A study of mathematical programming methods for structural optimization. *PhD thesis Department of Civil and Environmental Engineering*. University of Iowa.

Bemporad, A., Morari, M., Dua, V., & Pistikopoulos, E. N. (2002). The explicit linear quadratic regulator for constrained systems. *Automatica, 38*(1), 3–20.

Bianchi, L., Dorigo, M., Gambardella, L. M., & Gutjahr, W. J. (2009). A survey on metaheuristics for stochastic combinatorial optimization. *Natural Computing, 8*(2), 239–287.

Blum, C., & Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys, 35*(3), 268–308.

Brusco, M. J., & Doreian, P. (2019). Partitioning signed networks using relocation heuristics, tabu search, and variable neighborhood search. *Social Networks, 56*, 70–80.

Burke, E. K., Newall, J. P., & Weare, R. F. (1995). A memetic algorithm for university exam timetabling. *Paper presented at the international conference on the practice and theory of automated timetabling*.

Cagnina, L. C., Esquivel, S. C., & Coello, C. A. C. (2008). Solving engineering optimization problems with the simple constrained particle swarm optimizer. *Informatica, 32*(3).

Chegini, S. N., Bagheri, A., & Najafi, F. (2018). PSOSCALF: A new hybrid PSO based on Sine cosine algorithm and levy flight for solving optimization problems. *Applied Soft Computing, 73*, 697–726.

Coello, C. A. C. (2000). Use of a self-adaptive penalty approach for engineering optimization problems. *Computers in Industry, 41*(2), 113–127.

Coello, C. A. C., & Montes, E. M. (2002). Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. *Advanced Engineering Informatics, 16*(3), 193–203.

Collins, M., & Cosgrove, T. (2019). Dynamic relaxation modelling of braced bending active gridshells with rectangular sections. *Engineering Structures, 187*, 16–24.

Deb, K. (2000). An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering, 186*(2–4), 311–338.

Deb, K., & Goyal, M. (1996). A combined genetic adaptive search (GeneAS) for engineering design. *Computer Science and informatics, 26*, 30–45.

Dimopoulos, G. G. (2007). Mixed-variable engineering optimization based on evolutionary and social metaphors. *Computer methods in applied mechanics and engineering, 196*(4–6), 803–817.

dos Santos Coelho, L. (2010). Gaussian quantum-behaved particle swarm optimization approaches for constrained engineering design problems. *Expert Systems with Applications, 37*(2), 1676–1683.

Fesanghary, M., Mahdavi, M., Minary-Jolandan, M., & Alizadeh, Y. (2008). Hybridizing harmony search algorithm with sequential quadratic programming for engineering optimization problems. *Computer methods in Applied Mechanics and Engineering, 197*(33–40), 3080–3091.

Gandomi, A. H., Yang, X.-S., & Alavi, A. H. (2011). Mixed variable structural optimization using firefly algorithm. *Computers & Structures, 89*(23–24), 2325–2336.

Gandomi, A. H., Yang, X.-S., & Alavi, A. H. (2013). Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems. *Engineering with Computers, 29*(1), 17–35.

Garg, H. (2014). Solving structural engineering design optimization problems using an artificial bee colony algorithm. *Journal of Industrial and Management Optimization, 10*(3), 777–794.

Garg, H. (2016). A hybrid PSO-GA algorithm for constrained optimization problems. *Applied Mathematics and Computation, 274*, 292–305.

Garg, H. (2019). A hybrid GSA-GA algorithm for constrained optimization problems. *Information Sciences, 478*, 499–523.

Gen, M., & Cheng, R. (1997). *Genetic algorithms and engineering design*. New York: John Wily and Sons.

Gupta, S., & Deep, K. (2019). A hybrid self-adaptive sine cosine algorithm with opposition based learning. *Expert Systems with Applications, 119*, 210–230.

He, S., Prempain, E., & Wu, Q. H. (2004). An improved particle swarm optimizer for mechanical design optimization problems. *Engineering Optimization, 36*(5), 585–605.

He, Q., & Wang, L. (2007). An effective co-evolutionary particle swarm optimization for constrained engineering design problems. *Engineering Applications of Artificial Intelligence, 20*(1), 89–99.

Heidari, A. A., Abbaspour, R. A., & Jordehi, A. R. (2017). An efficient chaotic water cycle algorithm for optimization tasks. *Neural Computing and Applications, 28*(1), 57–85.

Hernández, S. (1994). *Multi-objective structural optimisation* (pp. 341–363). Elsevier Applied Science: Geometry and Optimisation Techniques for Structural Design.

Himmelblau, D. M. (2018). *Applied nonlinear programming*. McGraw-Hill.

Homaifar, A., Qi, C. X., & Lai, S. H. (1994). Constrained optimization via genetic algorithms. *Simulation, 62*(4), 242–253.

Hsu, Y.-L., & Liu, T.-C. (2007). Developing a fuzzy proportional–derivative controller optimization engine for engineering design optimization problems. *Engineering Optimization, 39*(6), 679–700.

https://aerial-core.eu/.

Hwang, S.-F., & He, R.-S. (2006). A hybrid real-parameter genetic algorithm for function optimization. *Advanced Engineering Informatics, 20*(1), 7–21.

Kannan, B. K., & Kramer, S. N. (1994). An augmented Lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design.

Kaveh, A., & Talatahari, S. (2009). Engineering optimization with hybrid particle swarm and ant colony optimization. *Asian Journal of Civil Engineering, 10*(6), 611–628.

Kaveh, A., & Talatahari, S. (2010). An improved ant colony optimization for constrained engineering design problems. *Engineering Computations, 27*(1), 155–182.

Kelly, S. G. (2012). *Mechanical vibrations: Theory and applications*. USA: Cengage learning Stamford.

Kim, Y. J., Kim, B. S., & Lim, M. T. (2003). Composite control for singularly perturbed nonlinear systems via successive Galerkin approximation. *Dynamics of Continuous, Discrete and Impulsive Systems Series B: Applications and Algorithms, 10*(1–3), 247–258.

Kirk, D. E. (2012). *Optimal control theory: An introduction*. Courier Corporation.
Korayem, M. H., Nekoo, S. R., & Korayem, A. H. (2016). Finite time SDRE control design for mobile robots with differential wheels. *Journal of Mechanical Science and Technology, 30*(9), 4353–4361.
Ku, K. J., Rao, S. S., & Chen, L. (1998). Taguchi-aided search method for design optimization of engineering systems. *Engineering Optimization, 30*(1), 1–23.
Lee, K. S., & Geem, Z. W. (2005). A new meta-heuristic algorithm for continuous engineering optimization: Harmony search theory and practice. *Computer Methods in Applied Mechanics and Engineering, 194*(36–38), 3902–3933.
Long, W., Wu, T., Liang, X., & Xu, S. (2019). Solving high-dimensional global optimization problems using an improved sine cosine algorithm. *Expert Systems with Applications, 123*, 108–126.
Mahdavi, M., Fesanghary, M., & Damangir, E. (2007). An improved harmony search algorithm for solving optimization problems. *Applied Mathematics and Computation, 188*(2), 1567–1579.
Martín-Alcántara, A., Grau, P., Fernandez-Feria, R., & Ollero, A. (2019). A simple model for gliding and low-amplitude flapping flight of a bio-inspired UAV. *Paper presented at the 2019 International Conference on Unmanned Aircraft Systems (ICUAS)*.
Mehta, V. K., & Dasgupta, B. (2012). A constrained optimization algorithm based on the simplex search method. *Engineering Optimization, 44*(5), 537–550.
Mezura-Montes, E., & Coello, C. A. C. (2008). An empirical study about the usefulness of evolution strategies to solve constrained optimization problems. *International Journal of General Systems, 37*(4), 443–473.
Mezura-Montes, E., Coello Coello, C. A., Velázquez-Reyes, J., & Muñoz-Dávila, L. (2007). Multiple trial vectors in differential evolution for engineering design. *Engineering Optimization, 39*(5), 567–589.
Mezura-Montes, E., Coello, C. A. C., & Landa-Becerra, R. (2003). Engineering optimization using simple evolutionary algorithm. *Paper presented at the Proceedings. 15th IEEE International Conference on Tools with Artificial Intelligence*.
Mirjalili, S. (2015). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-Based Systems, 89*, 228–249.
Mirjalili, S., Gandomi, A. H., Mirjalili, S. Z., Saremi, S., Faris, H., & Mirjalili, S. M. (2017). Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Advances in Engineering Software, 114*, 163–191.
Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in engineering software, 69*, 46–61.
Mitchell, M. (1998). *An introduction to genetic algorithms*. MIT press.
Nekoo, S. R. (2019). Tutorial and review on the state-dependent Riccati equation. *Journal of Applied Nonlinear Dynamics, 8*(2), 109–166.
Nekoo, S. R. (2020). A PDE breach to the SDRE. *Asian Journal of Control, 22*(2), 667–676.
Nekoo, S. R., & Irani Rahaghi, M. (2018). Recursive approximate solution to time-varying matrix differential Riccati equation: Linear and nonlinear systems. *International Journal of Systems Science, 49*(13), 2797–2807.
Nesterov, Y. (2013). Gradient methods for minimizing composite functions. *Mathematical Programming, 140*(1), 125–161.
Omran, M. G. H., & Salman, A. (2009). Constrained optimization using CODEQ. *Chaos, Solitons & Fractals, 42*(2), 662–668.
Osman, I. H., & Laporte, G. (1996). Metaheuristics: A bibliography (Vol. 63, pp. 513-623): Springer, Annals of Operations Research.
A.L. Peressini F.E. Sullivan J.J. Uhl The mathematics of nonlinear programming: Springer-Verlag 1988 New York.
Permanent-URL(b). https://griffin-erc-advanced-grant.eu/.
Price, K., Storn, R. M., & Lampinen, J. A. (2006). *Differential evolution: A practical approach to global optimization*. Springer Science & Business Media.
Ragsdell, K. M., & Phillips, D. T. (1976). Optimal design of a class of welded structures using geometric programming.
Raj, K. H., & Sharma, R. S. (2005). An evolutionary computational technique for constrained optimisation in engineering design.
Rao, S. S. (2019). *Engineering optimization: Theory and practice*. John Wiley & Sons.
Ray, T., & Liew, K.-M. (2003). Society and civilization: An optimization algorithm based on the simulation of social behavior. *IEEE Transactions on Evolutionary Computation, 7*(4), 386–396.
Ray, T., & Saini, P. (2001). Engineering design optimization using a swarm with an intelligent information sharing among individuals. *Engineering Optimization, 33*(6), 735–748.
Rizk-Allah, R. M. (2018). Hybridizing sine cosine algorithm with multi-orthogonal search strategy for engineering design problems. *Journal of Computational Design and Engineering, 5*(2), 249–273.
Salimi, H. (2015). Stochastic fractal search: A powerful metaheuristic algorithm. *Knowledge-Based Systems, 75*, 1–18.
Sandgren, E. (1988). Nonlinear integer and discrete programming in mechanical design. *Paper presented at the Proceeding of the ASME design technology conference*.
Tanabe, R., & Fukunaga, A. S. (2014, July). *Improving the search performance of SHADE using linear population size reduction.* Paper presented at the 2014 IEEE congress on evolutionary computation (CEC), Beijing, China.
Tharwat, A., Elhoseny, M., Hassanien, A. E., Gabel, T., & Kumar, A. (2019). Intelligent Bézier curve-based path planning model using chaotic particle swarm optimization algorithm. *Cluster Computing, 22*(2), 4745–4766.
Tsai, J.-F. (2005). Global optimization of nonlinear fractional programming problems in engineering design. *Engineering Optimization, 37*(4), 399–409.
Yelmewad, P., & Talawar, B. (2019). Parallel iterative hill climbing algorithm to solve TSP on GPU. *Concurrency and Computation: Practice and Experience, 31*(7), e4974.
Zhang, M., Luo, W., & Wang, X. (2008). Differential evolution with dynamic stochastic selection for constrained optimization. *Information Sciences, 178*(15), 3043–3074.
Zuo, X., Li, B., Huang, X., Zhou, M.-C., Cheng, C., Zhao, X., & Liu, Z. (2019). Optimizing hospital emergency department layout via multiobjective tabu search. *IEEE Transactions on Automation Science and Engineering, 16*(3), 1137–1147.