Computers & Industrial Engineering 128 (2019) 192–218

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

Find-Fix-Finish-Exploit-Analyze (F3EA) meta-heuristic algorithm: An effective algorithm with new evolutionary operators for global optimization☆

Ali Husseinzadeh Kashan[a,*], Reza Tavakkoli-Moghaddam[b,c,*], Mitsuo Gen[d]

[a] Faculty of Industrial and Systems Engineering, Tarbiat Modares University, Tehran, Iran
[b] School of Industrial Engineering, College of Engineering, University of Tehran, Tehran, Iran
[c] Arts et Métiers ParisTech, LCFC, Metz, France
[d] Fuzzy Logic Systems Institute, Iizuka, Fukuoka, and Tokyo University of Science, Tokyo, Japan

ARTICLE INFO

Keywords:
Global optimization
Evolutionary computation
Selection, mutation and local search operators
F3EA targeting process

ABSTRACT

A novel population-based evolutionary meta-heuristic algorithm is introduced, which imitates the Find-Fix-Finish-Exploit-Analyze (F3EA) targeting process. It considers the surface of the objective function as the battlefield and executes Find-Fix-Finish-Exploit-Analyze steps in an iterative manner. Following the radar detection rationale, a new evolutionary selection operator is introduced during the Find step. It is justified how to model the Fix step as a one-dimensional optimization problem to attain a local search operator. To produce a new solution by the Finish step, the target solution selected in the Find step is actioned artificially. This is an adaptive mutation stage, in which the position of the new potential solution is identified via modeling of projectile motion. The Exploit step takes over opportunities provided by mating the generated solution and its parent solution. Finally, the Analyze step, updates the population. Extensive experiments are conducted based on engineering optimization problems and a large set of benchmark functions for performance assessment, sensitivity analysis of the control parameters, and effectiveness analysis of different steps of the algorithm. Results of statistical tests signify that equipping the algorithm with new selection, mutation and local search operators makes it effective and efficient enough to exceed or match the best of rivals.

1. Introduction

Algorithmic techniques developed within the field of optimization often fetch up being effective and efficient tools. For example, this is definitely true for meta-heuristic algorithms; a general algorithmic framework, which calls for low cost modifications to tackle variety of problems, and imitates natural phenomena rules combined with randomness. As a rich source of inspiration, nature has attracted many researchers' attention in many ways toward developing imitative algorithms for engineering optimization. Via mimicking a natural process (e.g., physical, ecological or social) and developing an optimization mechanism, nature-inspired meta-heuristic algorithms are talented to realize the global convergence.

There are numerous deterministic algorithms or search methods which are effective to solve engineering optimization problems, such as the generalized reduced gradient or recursive quadratic programming methods. However, these methods work under the assumptions of differentiability and continuity of the objective and constraints functions, which rarely addressed in many real world applications. Moreover, such methods fail to show effective in more complex problems where there are local optima. Beside such algorithms, there are meta-heuristic algorithms which are not limited to differentiability and continuity assumptions and use randomization to escape local optima (Mohamed, 2015).

Triggered by pioneer works of Rechenberg who introduced Evolution Strategy, Holland who introduced Genetic Algorithm, Glover who introduced Tabu Search and Kirkpatrick, Gelatt and Vecchi who introduced the so called Simulated Annealing algorithm, 1990s was a golden time for emerging a number of most celebrated meta-heuristic algorithms. The field of Swarm Intelligence was initiated by the work of

☆ The Matlab code of the F3EA algorithm is available at http://drkashan.ir/en/page/Archive/28/.

* Corresponding authors at: Faculty of Industrial and Systems Engineering, Tarbiat Modares University, Tehran, Iran (A. Husseinzadeh Kashan), School of Industrial Engineering, College of Engineering, University of Tehran, Tehran, Iran (R. Tavakkoli-Moghaddam).

E-mail addresses: a.kashan@modares.ac.ir (A. Husseinzadeh Kashan), tavakoli@ut.ac.ir, reza.tavakkolimogahddam@ensam.eu (R. Tavakkoli-Moghaddam), gen@flsi.or.jp, gen@rs.tus.ac.jp (M. Gen).

https://doi.org/10.1016/j.cie.2018.12.033
Received 14 March 2017; Received in revised form 13 April 2018; Accepted 12 December 2018
Available online 14 December 2018
0360-8352/

Dorigo and his colleagues on Ant Systems, and the work of Kennedy and Eberhart on Particle Swarm Optimization. Besides, the Differential Evolution algorithm of Storn and Price had been one of the most successful evolutionary meta-heuristic algorithms. The emergent of new nature inspired algorithms have been accelerated after 2000. For example, the harmony search (HS), introduced in 2001, is conceptualized using the musical process of searching for a perfect state of harmony; the bacterial foraging optimization algorithm (BFOA), introduced in 2002, models foraging as an optimization process where an animal seeks to maximize energy per unit time spent for foraging; the bacterial chemotaxis algorithm (BC), introduced in 2002, has been proposed by analogy to the way bacteria react to chemoattractants in concentration gradients; the artificial bee colony algorithm (ABC), introduced in 2005, simulates the intelligent foraging behavior of a honeybee swarm; the central force optimization algorithm (CFO), introduced in 2007, makes an analogy between the process of searching a decision space for the maxima of an objective function and flying probes through physical space under the influence of gravity; the fire fly algorithm (FA), introduced in 2008, performs based on the idealization of the flashing characteristics of fireflies; the league championship algorithm (LCA), introduced in 2009, simulates the competitions followed in league games and models the match analysis process in which coaches modify their arrangement on the basis of their own game experiences and their opponent's style of play; the group search optimizer (GSO), introduced in 2009, simulates the animal searching behavior to find resources such as food, mates, oviposition, or nesting sites; Teaching–Learning Based Optimization (TLBO), introduced in 2011 mimics teaching–learning process in a class between the teacher and the students (learners); Krill herd algorithm (KH) introduced in 2012 works based on the simulation of the herding of krill swarms in response to specific biological and environmental processes; Gray Wolf Optimizer introduced in 2014 mimics the leadership hierarchy and hunting mechanism of grey wolves in nature; Optic Inspired Optimization, introduced in 2015, models the optical phenomena occurred in curved mirrors metaphorically into the searching process of optimization.

There are many ways to classify meta-heuristic algorithms. One way is based on the type of metaphor used to develop the algorithm. For example nature inspired algorithms (e.g., evolutionary algorithms, ant colony optimization and simulated annealing algorithm) social inspired algorithms (e.g. the well-known Tabu search, league championship algorithm (Husseinzadeh Kashan, 2011, 2014), and social engineering optimizer (Fathollahi Fard, Hajiaghaei-Keshteli, & Tavakkoli-Moghaddam, 2018) or political inspired algorithms (e.g., imperialist competitive algorithm (Atashpaz-Gargari & Lucas, 2007). Nature inspired algorithms solve the optimization problems by simulating a developing process in nature, which is considered as a new effective way for optimization. In these algorithms, the global convergence is realized by simulating the physical or ecological process in nature and the optimization mechanisms of algorithms themselves. One of the highest level sources which have been influential on the emergence of many recent meta-heuristic algorithms is Physics; the natural science that involves the study of matter and its motion through space and time. Typically, the research of Physics can be classified into areas such as classical mechanics, relativity, thermodynamics, electromagnetism, optics, and quantum mechanics. The simplicity of all these fundamental principles is not only the real beauty of Physics, but also the main momentum in developing innovative algorithms (Xing & Gao, 2013). There exist a relatively rich list of Physics inspired meta-heuristic algorithms. A first hand classification of the existing algorithms can be made as the one shown in Fig. 1.

This paper introduces a new meta-heuristic algorithm for numerical optimization based on a targeting process, namely Find-Fix-Finish-Exploit-Analyze (F3EA) approach. The physical rules, which make possible the act of military facilities, are also adopted and adapted to develop the required operators of the algorithm. The algorithm, is therefore called the F3EA meta-heuristic algorithm.

The F3EA algorithm tries to mimic the targeting process; the process by which suitable objects or installations are selected for destruction. During the targeting process, those targets or resources that the enemy cannot afford to lose and seems to create desired outcome for achieving the objectives are analyzed and prioritized. Besides, appropriate fatal and nonfatal actions to those targets are matched. In 2014 General S.A. McChrystal wrote about the "F3EA" targeting cycle which had been practiced in Iraq and Afghanistan (Ferry, 2013). F3EA stands for:

1. *Find*: designates the process of identifying the targets and their elements (i.e., person or location).
2. *Fix*: describes the monitoring process for preparation of action against the targets. In this phase a precise location of the target is identified in the battle space and is maintained under track. This phase also includes the continual update of situational awareness of the local area and detection of indicators and warnings of changes to the situation.
3. *Finish*: describes the process of taking action on the targets to destroy them.
4. *Exploit*: describes the process by which the opportunities presented by the target effect are identified. This phase includes information gathering from the target area, sensitive target site exploitation, battle damage assessment etc.
5. *Analyze*: indicates an assessment step which is related to evaluation of the results of attacks on targets.

Although the underlying metaphor of F3EA algorithm inspires from the targeting process, it is established based on theories from projectile motion and Physics of radar sets to model Finish and Find steps. In this sense it can be classified in both of "Physics inspired algorithms" and "Other sources of inspiration (e.g., society, politics, sport, music and warfare)" categories in Fig. 1.

The mechanism of F3EA algorithm is composed of five steps. The Find step imitates the target selection process. In this step, which is essentially a selection step, we introduce a new selection mechanism to evolutionary computations with interesting properties. Here we imitate the detection process followed by military radar devices to analyze and prioritize targets. More specifically we adopt the radar range equation from Physics and develop our theory to decide whether an individual is detectable to bias the search toward it or not. The emphasis of the Find step is on identifying individual solutions that provide a great advantage if their surrounding solutions being searched. The Fix step, which constitutes a local search phase, artificially models the aiming process toward targets. This step is modeled as a one dimensional optimization problem to scan the function surface on a given path to find the precise location of a local optimum via line search techniques. The Finish step develops a new adaptive mutation operator. Here we use the projectile motion equations to develop a mathematical model for generation of new solutions by means of those solutions which were selected in the Find step to bias the search toward them. In the Exploit step, we seek for opportunities presented in mating the solutions resultant by Finish step with those individuals already exist in the population, so that the resultant solutions bear the information on both of shoot and target areas in the search space. In the last step i.e., the Analyze step, the new solutions generated by one of Fix or Exploit steps are entered into population if they provide better function values.

In the remaining part of the paper, the mechanism of F3EA algorithm is described in details. To do so, Section 2 reviews the required background. Section 3 puts forward the mathematical model of the algorithm. Section 4 investigates the computational experiments. We

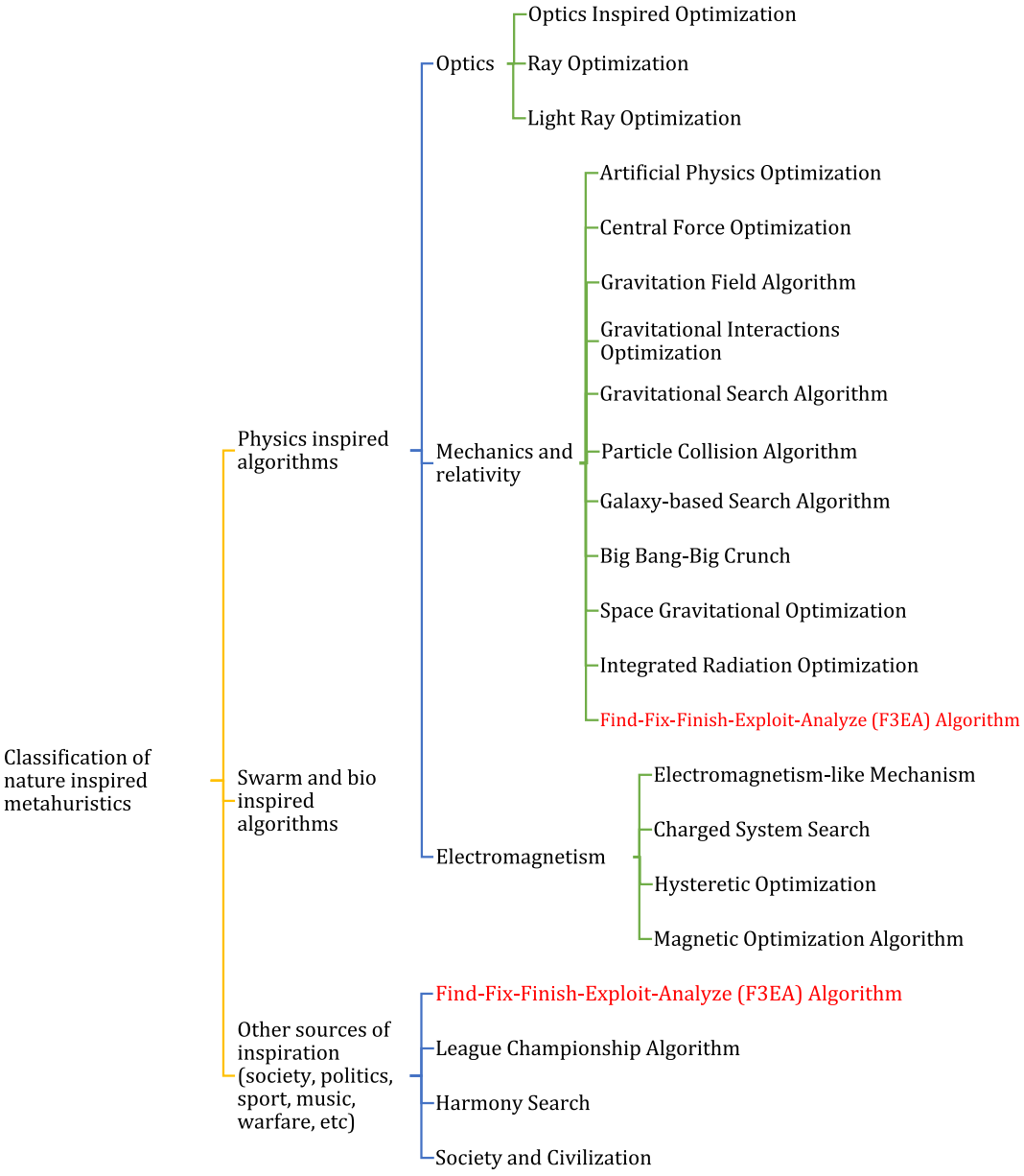


Fig. 1. The first hand classification of meta-heuristic methods.

compare the output of the proposed algorithm on several sets of engineering optimization problems and benchmark functions, with a large number of recent and advanced optimization algorithms. We do a sensitivity analysis on the control parameters of the F3EA algorithm and finally show that different steps of the algorithm add values during the optimization process. The last section concludes the paper.

2. Background

2.1. Radar range equation

Radar, which is stands for "RAdio Detecting And Ranging", is an electromagnetic device which is used for object detection. Radars are working based on transition of the radio-frequency electromagnetic energy pulses and receiving back its reflection from object. The running time of the transmitted signal and its transmission speed, which is actually the light speed, is used as a basis to determine the distance and location of the aim. The actual distance of an aim with respect to that of the radar antenna is the line of sight distance between them and is known as the slant range (R_{max}).

The radar range equation is the most well-known equation of the radar theory. The original radar equation was developed by the U.S. Naval Research Laboratory during the World War II (Norton & Omberg, 1947). Since then the radar equation was developed to analyze and design modern radars and to calculate the maximum range (R_{max}) at which the desired detection performance is achieved by the radar device. The maximum radar detection range is as follows:

$$R_{max} = K_{Radar} \sqrt[4]{\sigma} \quad (1)$$

where $K_{Radar} = \sqrt[4]{pGA/(4\pi)^2 S_{min}}$, in which p is the radar transmitted peak power. G is the radar antenna gain (the antenna ability to concentrate electromagnetic energy in form of a thin angular beam). A is

the antenna's aperture, which shows how effective it receives power from an incoming wave. $S_{\min}$ is a threshold on the amount of received signal. To be able to detect the signal, the amount of received power should be greater than $S_{\min}$.

In (1), R_{max} is influenced by K_{Radar} (which is dependent to the radar design) and σ (which is related to target). σ is the cross sectional area of the target which determines the portion of the radiated power reflected back by the target. The value of σ depends on its physical geometry, used material, type of the reflecting surface etc. The most important application of (1) is to determine the farthest possible distance at which the target will be detected with a high probability.

2.2. Projectile motion

Projectile motion is a form of motion, in which a thrown or launched projectile (an object projected into space by the action of an initial force) near the earth's surface moves along a curved path (which is called the ballistic trajectory of projectile) under the action of gravity, neglecting all other forces such as air friction (see Fig. 2). A projectile launched with specific initial conditions will have a predictable range. The mathematical equations of motion are used to analyze projectile - trajectory.

Fig. 3a shows a projectile fired up from A on an incline plane AB having inclination β with the horizontal. The projectile strikes the inclined plane at B. Let the time of flight taken by the projectile going from A to B be equal to T, then we have (Bansal, 1998):

$$T = \frac{2v_0 \sin(\alpha - \beta)}{g \cos\beta}. \tag{2}$$

where v_0, is the initial velocity at which the projectile is launched, α is the angle at which the projectile is launched and $g = 9.81\ \text{m s}^{-2}$. The range of the projectile on the inclined plane is given by the distance AB of Fig. 3a and is equal to (Bansal, 1998):

$$AB = \frac{2v_0^2 \cos\alpha \sin(\alpha - \beta)}{g\cos^2\beta} \tag{3}$$

If the projectile is launched from the top of the incline plane toward the bottom of the inclined plane as shown in Fig. 3b, then the expression for the time of flight and range on the incline are obtained by substituting $-\beta$ for β in (2) and (3).

Now, suppose that the projectile, with mass m, is subjected to an opposite air drag force and wind force, beside the force of gravity. The wind component can be either in the negative or in the positive direction to the projectile flight. Let assume that $\vec{w} = [w_x, w_y, w_z]$ is the wind (with both speed and direction) and b is the air resistance coefficient. It is always reasonable to do not consider any vertical wind component, that is $w_y = 0$. Under such situations and assuming that $C = m/b$, the motion equations are as follows[1]:

$$x(t) = w_x t + C(v_0 \cos\alpha - w_x)(1 - e^{-t/C}) \tag{4}$$

$$z(t) = w_z t - Cw_z(1 - e^{-t/C}) \tag{5}$$

$$y(t) = -Cgt + (C^2 g + Cv_0 \sin\alpha)(1 - e^{-t/C}) \tag{6}$$

Without loss of generality, we assume that $b = 1$ and replace C with m. For large values of m, the projectile trajectory tends to parabolic trajectory of Fig. 2. A projectile with a small mass is more affected by the air drag and wind forces. Depending on the wind speed and direction, and also depending on the value of m, the launched projectile experiences different trajectories. When the wind blows in an opposite direction with the projectile launching direction, the projectile may hit the ground even somewhere behind the launching station (see Fig. 4 for various trajectories).

[1] http://facstaff.cbu.edu/~jholmes/P380/Proj.doc.

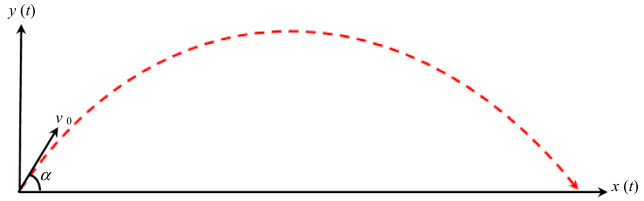


Fig. 2. Ballistic trajectory of projectile.

3. Developing the F3EA meta-heuristic algorithm

Encouraged by the above phenomena and their related physical theories, this section proposes the F3EA method for optimization over continuous spaces, which is composed of a new evolutionary selection operator inspired by the target recognition process via radar sets; a local search type operator inspired by the target monitoring process; and a new adaptive mutation strategy inspired by the target destruction (via projectiles) process.

The proposed F3EA algorithm works with a population of individuals. All members operate with a population Π^t that contains N_B individuals. The objective function surface in the F3EA algorithm is treated as the battlefield. The algorithm has five steps of Find, Fix, Finish, Exploit and Analyze. In what follows, we introduce the mathematical model of different steps of the algorithm. However, we introduce the Fix step at last, since it is an add-on local search module independently.

Let $f(\vec{X} = [x_1, x_2, ..., x_n])$: $\mathrm{R}^n \to \mathrm{R}$ be a scalar function for which we seek a global minimum. The decision space is defined by $x_{\min,d} \leqslant x_d \leqslant x_{\max,d},\ d = 1, ..,n$.

3.1. Find step

Here, a new evolutionary selection mechanism is proposed based on the object detection process followed by military radars. This step is activated whenever a new solution is being generated from $\vec{P}_i^t$ (the ith individual in the population Π^t at iteration t) which poses temporarily as an artificial radar. All other individuals in the population are treated as military facilities which may or may not be detected by the artificial radar (see Fig. 5). The aim is to choose among detectable facilities $\vec{P}_k^t,\ \forall k = 1, ...,N_B,\ k \neq i$ to generate a new solution by its aid.

Definition 1. Let $R_{ik}^t = |f(\vec{P}_i^t) - f(\vec{P}_k^t)|$. $\vec{P}_k^t$ is said to be detectable by $\vec{P}_i^t$ (and hence is a candidate for being the selected individual) if $R_{ik}^t \leqslant R_{\max_{ik}}^t$. Where, $R_{\max_{ik}}^t$ is the maximum range at which $\vec{P}_k^t$ will be detected by $\vec{P}_i^t$.

$R_{\max_{ik}}^t$ should be calculated based on (1). For this purpose, let $p(\vec{P}_i^t)$, $G(\vec{P}_i^t)$, $A(\vec{P}_i^t)$, $S_{\min}(\vec{P}_i^t)$ be the parameters of the artificial radar associated to $\vec{P}_i^t$ (see Section 2.1). Let us also consider an ideal (the most powerful) artificial radar in the search space whose parameters are introduced by $p(ideal)$ that is maximal, $G(ideal)$ that is maximal, $A(ideal)$ that is maximal and $S_{\min}(ideal)$ that is minimal. Such an ideal radar is able to detect any candidate solution anywhere in the search space. In this way $R_{\max}^t(ideal)$ is maximal. Let also set $f_{\max}^t = \max_{k\in\Pi^t}\{\vec{P}_k^t\}$ and $f_{\min}^t = \min_{k\in\Pi^t}\{\vec{P}_k^t\}$. We can accept that:

$$\frac{p(\vec{P}_i^t)}{p(ideal)} \equiv \frac{f(\vec{P}_i^t) - f_{\min}^t}{f_{\max}^t - f_{\min}^t} \Rightarrow \frac{p(\vec{P}_i^t)}{p(ideal)} = c\frac{f(\vec{P}_i^t) - f_{\min}^t}{f_{\max}^t - f_{\min}^t} \tag{7}$$

$$\frac{G(\vec{P}_i^t)}{G(ideal)} \equiv \frac{f(\vec{P}_i^t) - f_{\min}^t}{f_{\max}^t - f_{\min}^t} \Rightarrow \frac{G(\vec{P}_i^t)}{G(ideal)} = c\frac{f(\vec{P}_i^t) - f_{\min}^t}{f_{\max}^t - f_{\min}^t} \tag{8}$$

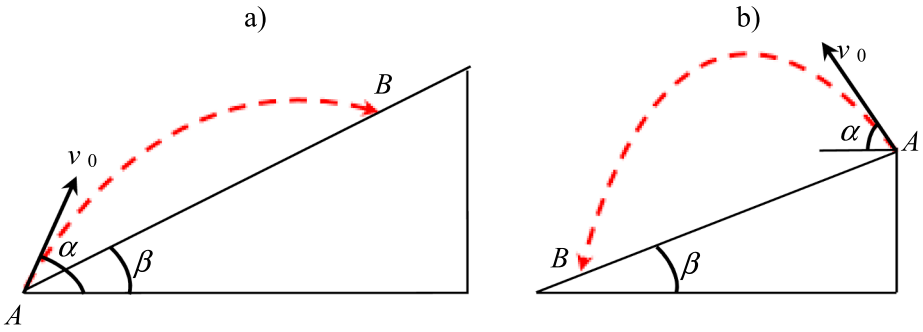


Fig. 3. Projectile is fired (a) up an incline plane, (b) down an incline plane.

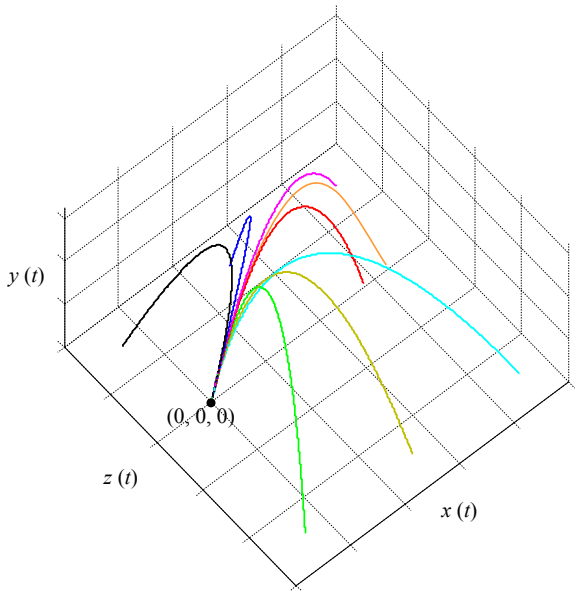


Fig. 4. Trajectory of projectile influenced by the air resistance and wind speed and direction.

$$\frac{A(\vec{P}_i^t)}{A(ideal)} \equiv \frac{f(\vec{P}_i^t) - f^t_{\min}}{f^t_{\max} - f^t_{\min}} \Rightarrow \frac{A(\vec{P}_i^t)}{A(ideal)} = c\frac{f(\vec{P}_i^t) - f^t_{\min}}{f^t_{\max} - f^t_{\min}} \tag{9}$$

$$\frac{S_{\min}(\vec{P}_i^t)}{S_{\min}(ideal)} \equiv \frac{1/(f(\vec{P}_i^t) - f^t_{\min})}{1/(f^t_{\max} - f^t_{\min})} \Rightarrow \frac{S_{\min}(\vec{P}_i^t)}{S_{\min}(ideal)} = c\frac{1/(f(\vec{P}_i^t) - f^t_{\min})}{1/(f^t_{\max} - f^t_{\min})}. \tag{10}$$

Without loss of generality we set $c = 1$. From (1) and (7)–(10), we have:

$$\frac{K_{\text{Radar}}(\vec{P}_i^t)}{K_{\text{Radar}}(ideal)} = \sqrt[4]{\frac{p(\vec{P}_i^t)G(\vec{P}_i^t)A(\vec{P}_i^t)}{(4\pi)^2 S_{\min}(\vec{P}_i^t)}} \Big/ \sqrt[4]{\frac{p(ideal)G(ideal)A(ideal)}{(4\pi)^2 S_{\min}(ideal)}} = \frac{f(\vec{P}_i^t) - f^t_{\min}}{f^t_{\max} - f^t_{\min}} \tag{11}$$

Based on relation (11), a larger value for $f(\vec{P}_i^t)$ results in a larger value for $K_{\text{Radar}}(\vec{P}_i^t)$. This implies that as $\vec{P}_i^t$ becomes worse (better), it should be more relaxed (restricted) to detect more individuals and hence to have more (less) options for biasing the search towards them. Intuitively we can define $R^t_{\max}(ideal) = f^t_{\max} - f^t_{\min}$.

Let $\sigma(\vec{P}_k^t)$ be the artificial radar cross section associated to $\vec{P}_k^t$. From (1) and (11) we have:

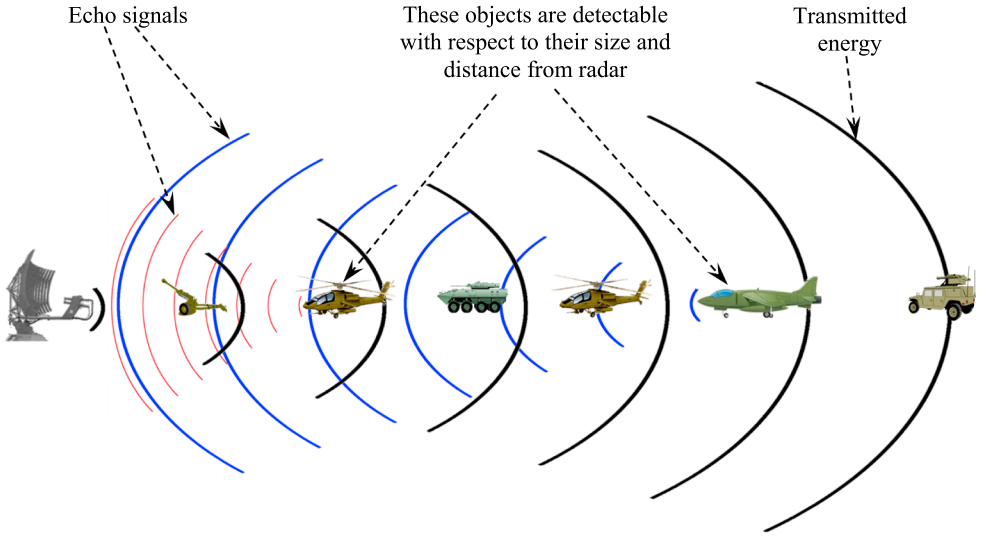


Fig. 5. Objects with different distance and size may or may not be detected by the artificial radar.

$$R^t_{\max_{ik}} = R^t_{\max}(ideal) \times \frac{f(\vec{P}_i^t) - f^t_{\min}}{f^t_{\max} - f^t_{\min}} \times \sqrt[4]{\sigma(\vec{P}_k^t)}, \quad \forall\, k = 1, ..,N_B, \quad k \neq i \tag{12}$$

Since $\sigma(\vec{P}_k^t)$ is the $\vec{P}_k^t$'s ability for being detected by $\vec{P}_i^t$, it is sensible to set $\sigma(\vec{P}_k^t) \equiv \left(\frac{f^t_{\max} - f(\vec{P}_k^t)}{f^t_{\max} - f^t_{\min}}\right)$. Hence, $\sigma(\vec{P}_k^t)$ becomes smaller as $f(\vec{P}_k^t)$ gets larger. We can further set $\sigma(\vec{P}_k^t) \equiv e^{-A\left(\frac{f(\vec{P}_k^t) - f^t_{\min}}{f(\vec{P}_i^t) - f^t_{\min}}\right)}$, which indicates that as $\vec{P}_k^t$ becomes worse than $\vec{P}_i^t$, its grace will dramatically decreases for being detected by $\vec{P}_i^t$. So we get $\sigma(\vec{P}_k^t) = \left(\frac{f^t_{\max} - f(\vec{P}_k^t)}{f^t_{\max} - f^t_{\min}}\right) e^{-A\left(\frac{f(\vec{P}_k^t) - f^t_{\min}}{f(\vec{P}_i^t) - f^t_{\min}}\right)}$. From (12), we arrive at:

$$R^t_{\max_{ik}} = (f(\vec{P}_i^t) - f^t_{\min}) \times \sqrt[4]{\left(\frac{f^t_{\max} - f(\vec{P}_k^t)}{f^t_{\max} - f^t_{\min}}\right) e^{-A\left(\frac{f(\vec{P}_k^t) - f^t_{\min}}{f(\vec{P}_i^t) - f^t_{\min}}\right)}}, \quad i \neq k \tag{13}$$

Let $u(\vec{X}) = \left(\frac{f(\vec{X}) - f^t_{\min}}{f^t_{\max} - f^t_{\min}}\right)$ be a normalization of $f(\vec{X})$. We write:

$$R^t_{\max_{ik}} = (f^t_{\max} - f^t_{\min}) \times u(\vec{P}_i^t) \times \sqrt[4]{(1 - u(\vec{P}_k^t)) e^{-A\frac{u(\vec{P}_k^t)}{u(\vec{P}_i^t)}}}, \quad i \neq k, \quad A > 0 \tag{14}$$

Now we can decide whether the kth individual is detectable by the ith one or not, using Definition 1.

Remark 1. If $f(\vec{P}_i^t) = f^t_{\min}$ then $R^t_{\max_{ik}} = 0,\ \forall\, k = 1, ..,N_B,\ i \neq k$. That is, no individual can be detected by the best individual in the population.

Remark 2. If $f(\vec{P}_k^t) = f^t_{\min}$ then $R^t_{\max_{ik}} = (f(\vec{P}_i^t) - f^t_{\min}) = R^t_{ik}$. That is, the best individual is detectable by any other individual.

Remark 3. If $f(\vec{P}_k^t) = f^t_{\max}$ then $R^t_{\max_{ik}} = 0$. That is, the worst individual is not detectable by any other individual. No individual moves towards the worst one.

Among detectable individuals by $\vec{P}_i^t$, one is selected for generation of a new solution in the Finish step. Here we use from the roulette wheel selection mechanism (as used in Karaboga and Akay (2009)) to select among detectable individuals. Hereafter, we address the selected individual as $\vec{P}_k^t$.

3.2. Finish step

The aim in the Finish step, is to produce an individual solution based on $\vec{P}_i^t$ and with the assistance of $\vec{P}_k^t$. Now we assume that $\vec{P}_i^t$ impersonates the position of a projectile launcher and $\vec{P}_k^t$ indicates the position of a target facility. The new solution generated by $\vec{P}_i^t$ and $\vec{P}_k^t$ is introduced by $\vec{E}_i^t$ which is treated as the explosion position of the artificial projectile launched from $\vec{P}_i^t$ towards $\vec{P}_k^t$. The extraction of $\vec{E}_i^t$ is based on the use of motion equations of Section 2. The following axioms are considered:

- The launched artificial projectile will be exploded after a certain duration.
- Before the launch, the approximate explosion time is obtained to by (2) which neglects air or wind drag.
- Right after its launch, the projectile is exposed to air drag and wind forces.

Let $[\vec{P}_i^t, u(\vec{P}_i^t)]_{1\times(n+1)}$ and $[\vec{P}_k^t, u(\vec{P}_k^t)]_{1\times(n+1)}$ be the positions of the artificial launcher and target on the transformed function surface in the $\vec{P} - u(\vec{P})$ coordinate system, respectively. From Fig. 6 the inclination β^t_{ik} is calculated as follows:

$$\beta_{ik}^t = \arctan \frac{u(\vec{P}_k^t) - u(\vec{P}_i^t)}{\|\vec{P}_k^t - \vec{P}_i^t\|} \tag{15}$$

Let α^t_{ik} be the launch angle. For a given value of α^t_{ik}, the value of v^t_{0ik} (the projectile initial velocity) is obtained based on (3) as follows:

$$v^t_{0ik} = \begin{cases} \sqrt{\frac{AB^t_{ik} \times g \times \cos^2 \beta^t_{ik}}{2\cos\alpha^t_{ik} \times \sin(\alpha^t_{ik} - \beta^t_{ik})}}, & f(\vec{P}_i^t) < f(\vec{P}_k^t) \\ \sqrt{\frac{AB^t_{ik} \times g \times \cos^2 |\beta^t_{ik}|}{2\cos\alpha^t_{ik} \times \sin(\alpha^t_{ik} + |\beta^t_{ik}|)}}, & f(\vec{P}_i^t) \geqslant f(\vec{P}_k^t) \end{cases} \tag{16}$$

$$AB^t_{ik} = \|[\vec{P}_k^t, u(\vec{P}_k^t)]_{1\times(n+1)} - [\vec{P}_i^t, u(\vec{P}_i^t)]_{1\times(n+1)}\|, \tag{17}$$

$$\alpha^t_{ik} = \begin{cases} \text{rand}(\beta^t_{ik}, \pi/2), & f(\vec{P}_i^t) < f(\vec{P}_k^t) \\ \text{rand}(0, \pi/2), & f(\vec{P}_i^t) \geqslant f(\vec{P}_k^t) \end{cases} \tag{18}$$

Once the value of v^t_{0ik} was determined by (16), the projectile's flight time (T^t_{ik}) can be calculated based on (2) as follows:

$$T^t_{ik} = \begin{cases} \frac{2v^t_{0ik}\sin(\alpha^t_{ik} - \beta^t_{ik})}{g\cos\beta^t_{ik}}, & f(\vec{P}_i^t) < f(\vec{P}_k^t) \\ \frac{2v^t_{0ik}\sin(\alpha^t_{ik} + |\beta^t_{ik}|)}{g\cos|\beta^t_{ik}|}, & f(\vec{P}_i^t) \geqslant f(\vec{P}_k^t) \end{cases} \tag{19}$$

Suppose that the mass of projectile is m^t_{ik} and the wind vector is $\vec{w}^t_{ik} = [w^t_{x_{ik}}, 0, w^t_{z_{ik}}]_{1\times3}$. Under the third axiom, the launched projectile is exposed to an unexpected wind and air drag forces which cause its motion trajectory becomes non-parabolic (see Fig. 7). So, depending on the magnitude of the forces, the projectile will traverse a non-parabolic path and is exploded at a certain time (T^t_{ik}) somewhere near or far from the target point $\vec{P}_k^t$.

The motion equations (i.e., (4) and (6)) essentially work in 2D search spaces ($n = 2$). We can theoretically generalize them such that they work in higher (say n) dimension search spaces, too. Let $\vec{E}_i^t$ be the explosion position of the artificial projectile launched from $\vec{P}_i^t$ toward $\vec{P}_k^t$, which is essentially a new solution. Following (4) and (6), we are now ready to propose the equations of the Finish step to generate $\vec{E}_i^t$ as follows:

$$x(T^t_{ik}) = w^t_{x_{ik}} T^t_{ik} + m^t_{ik}\left(v^t_{0ik}\cos\alpha^t_{ik} - w^t_{x_{ik}}\right)\left(1 - e^{-T^t_{ik}/m^t_{ik}}\right) \tag{20}$$

$$z(T^t_{ik}) = w^t_{z_{ik}} T^t_{ik} - m^t_{ik} w^t_{z_{ik}}\left(1 - e^{-T^t_{ik}/m^t_{ik}}\right) \tag{21}$$

$$\vec{E}_i^t = \vec{P}_i^t + x(T^t_{ik})\frac{(\vec{P}_k^t - \vec{P}_i^t)}{\|\vec{P}_k^t - \vec{P}_i^t\|} + z(T^t_{ik})\frac{(\vec{Z}^t_{ik\perp})}{\|\vec{Z}^t_{ik\perp}\|} \tag{22}$$

$x(T^t_{ik})$ and $z(T^t_{ik})$ are the projected explosion position on the so called x-axis and z-axis respectively. While the x-axis lies on the hyper-line connecting $\vec{P}_i^t$ and $\vec{P}_k^t$, the z-axis (for an n-dimensional search) is an imaginary axis perpendicular to x-axis. Suppose that the z-axis is

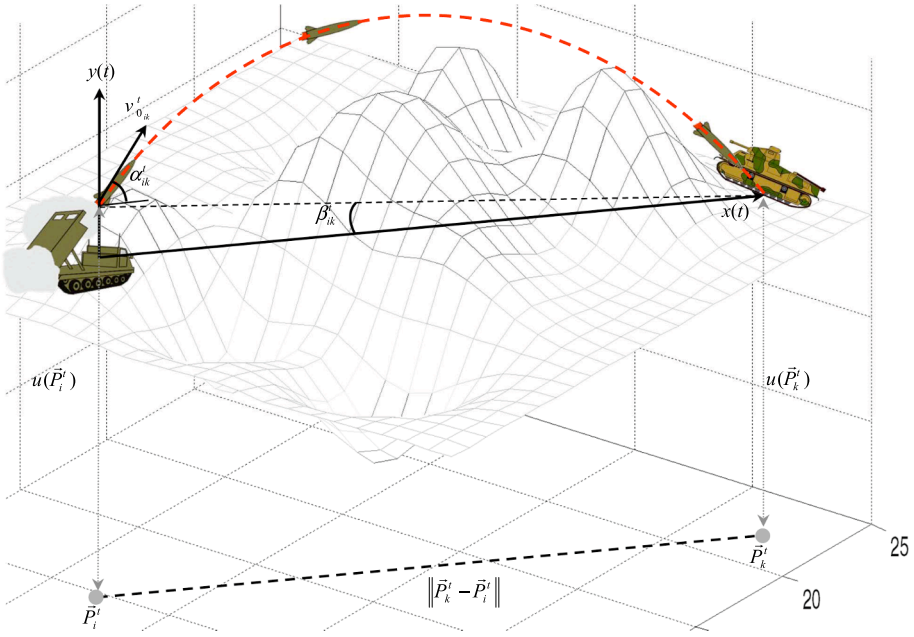


Fig. 6. Parabolic trajectory of the projectile in the absence of air or wind drag forces.

coextensive with $\vec{Z}^t_{ik\perp}$ which is perpendicular to the hyper-line connecting $\vec{P}^t_i$ and $\vec{P}^t_k$. We can write $(\vec{P}^t_k - \vec{P}^t_i)\cdot\vec{Z}^{t'}_{ik\perp} = 0$. One solution for this equation is $\vec{Z}^t_{ik\perp} = \left[1,\ 1,\ \ldots,1,\ 1 - \sum_{d=1}^{n} \frac{(p^t_{kd} - p^t_{id})}{(p^t_{kq} - p^t_{iq})},\ 1,\ \ldots,1\right]_{1\times n}$, whose qth element (selected randomly from $\{1, 2, \ldots, n\}$) is equal to $1 - \sum_{d=1}^{n} \frac{(p^t_{kd} - p^t_{id})}{(p^t_{kq} - p^t_{iq})}$.

$\vec{E}^t_i$ is a new candidate solution resultant from the Finish step. To end this section, it only remains to determine proper values for m^t_{ik} and $\vec{w}^t_{ik} = [w^t_{x_{ik}}, 0, w^t_{z_{ik}}]_{1\times 3}$.

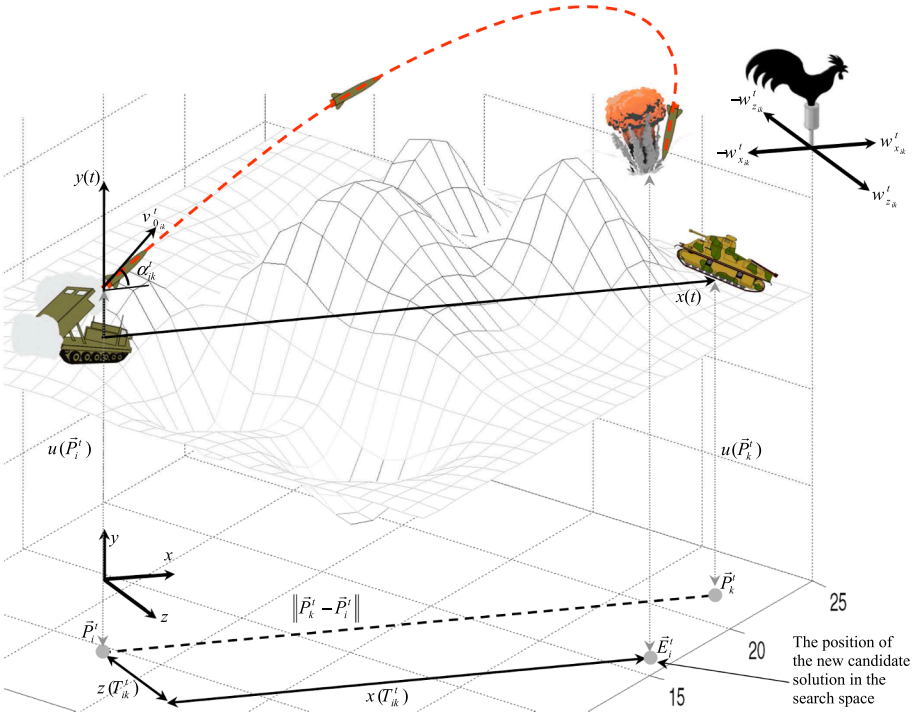


Fig. 7. Depending on the magnitude of the wind and air drag forces, the projectile traverses a non-parabolic path.

3.2.1. Determining the value of m_{ik}^t

From the theory of projectile motion we know that for large values of m, the projectile trajectory tends to parabolic trajectory of Fig. 2. A projectile with a small mass m is more affected by the air drag and wind forces. Since $\vec{P}_k^t$ is the selected individual, it is reasonable to move from $\vec{P}_i^t$ toward $\vec{P}_k^t$ if $f(\vec{P}_k^t) < f(\vec{P}_i^t)$ (or $u(\vec{P}_k^t) < u(\vec{P}_i^t)$) and move outward $\vec{P}_k^t$, otherwise. If we place:

$$m_{ik}^t = \max\left(m_{\min}, \min\left(\frac{u(\vec{P}_i^t)}{u(\vec{P}_k^t)}, \ m_{\max}\right)\right) \tag{23}$$

we can connect moving toward or outward decisions to fitness values which is quite appropriate to our purpose. $m_{\min}$ and $m_{\max}$ are constants and their value is set equal to 0.01 and 100 respectively, to do not allow the mass value becomes zero or infinity. A greater (smaller) value for $u(\vec{P}_i^t)/u(\vec{P}_k^t)$ imposes a greater (smaller) mass value which enforces the artificial projectile approaches to (retreat from) $\vec{P}_k^t$.

3.2.2. Determining the wind vector $\vec{w}_{ik}^t$

Once again, it is rational to connect the wind effect to the fitness values. If $f(\vec{P}_k^t) < f(\vec{P}_i^t)$, it is reasonable that the wind direction on x-axis (an axis passing through $\vec{P}_i^t$ and $\vec{P}_k^t$) be toward $\vec{P}_k^t$ to push the artificial projectile near $\vec{P}_k^t$ (such a situation is seen in Fig. 7), vice versa. As a conclusion we define:

$$w_{x_{ik}}^t = \begin{cases} -\, m_{ik}^t \times v_{0_{ik}}^t \times \text{rand}(0,\,1), & f(\vec{P}_i^t) < f(\vec{P}_k^t) \\ \ \ m_{ik}^t \times v_{0_{ik}}^t \times \text{rand}(0,\,1), & f(\vec{P}_i^t) \geqslant f(\vec{P}_k^t) \end{cases} \tag{24}$$

If $w_{z_{ik}}^t = 0$ (i.e., no wind element in the z direction), the explosion position will be on the line segment between $\vec{P}_i^t$ and $\vec{P}_k^t$. However, if $w_{z_{ik}}^t \neq 0$, the explosion place deviates from the line segment between $\vec{P}_i^t$ and $\vec{P}_k^t$ (see Fig. 7). Large values of $w_{z_{ik}}^t$ cause that the algorithm performs diversification task well, but at the expense of poor intensification, vice versa. Therefore, the value of $w_{z_{ik}}^t$ must be large enough at the start of the search and becomes smaller gradually. A suitable form for $w_{z_{ik}}^t$ will be as follows:

$$w_{z_{ik}}^t = \begin{cases} -\, m_{\min} \times v_{0_{ik}}^t \times \text{rand}(0,\,1), & \text{rand}(0,\,1) \geqslant 0.5 \\ \ \ m_{\min} \times v_{0_{ik}}^t \times \text{rand}(0,\,1), & \text{rand}(0,\,1) < 0.5 \end{cases} \tag{25}$$

3.3. Exploit step

The Exploit phase in the F3EA targeting process contains information gathering from the target area, sensitive target site conquer and exploitation, battle damage assessment etc. The Exploit step of the F3EA algorithm tries to utilize the result of the Finish step. The $\vec{E}_i^t$ solution produced by (22) differs from $\vec{P}_i^t$ in all dimensions and this may results in the premature convergence. Let c_i^t denotes the number of changes made in $\vec{P}_i^t$. We use from truncated geometric distribution (Husseinzadeh Kashan & Karimi, 2010; Husseinzadeh Kashan, 2014, 2015a) to simulate c_i^t as follows.

$$c_i^t = \left\lfloor \frac{\ln(1 - (1 - (1 - q_i^t)^n)\ \text{rand}(0,\,1))}{\ln(1 - q_i^t)} \right\rfloor, \ c_i^t \in \{1,\,2,\,\ldots,n\} \tag{26}$$

where $q_i^t < 1, \ \ q_i^t \neq 0$. Let us set $\vec{U}_i^t \leftarrow \vec{P}_i^t$. Then, c_i^t number of dimensions are selected randomly from $\vec{E}_i^t$ and their value are assigned to their relevant dimensions in $\vec{U}_i^t$. We then treat $\vec{U}_i^t$ as the offspring output of the Exploit step and perform the Analyze step.

3.4. Analyze step

In the Analyze step, the new solutions generated by one of Fix or Exploit steps are entered into population or update the best solution found so far, if they provide better function values. After execution of Exploit step, selection is carried out between $\vec{U}_i^t$ and $\vec{P}_i^t$. If $f(\vec{U}_i^t)$ was better than $f(\vec{P}_i^t)$, then the ith individual is replaced with $\vec{U}_i^t$. In the similar way, the new solutions generated during a single run of the Fix step, replace the global best solution ($\vec{G}_{best}^t$) if any of them yields a better function value.

3.5. Fix step

The Fix step, which constitutes a local search phase, artificially models the aiming process toward targets. This step is modeled as a one dimensional optimization problem to scan the function surface on a given path to find the precise location of a local optimum via line search techniques. Fig. 8 depicts a conceptual schematic of the Fix step.

To destroy the military tank, the projectile launch angle must be high enough to traverses over the highest peak (local optimum). This means that the aiming should be in such a way that it considers the highest peak position on the projectile trajectory. Both of the blue and black trajectories in Fig. 8 yield an unsuccessful launch.

The problem of scanning the function surface on a given path to find the local optimum can be implement as a one dimensional optimization problem. Let $\vec{G}_0^t = [g_{01}^t,\ldots,g_{0n}^t]$ be the location of the artificial launcher within search space and $\vec{S}^t = [s_1^t,\ldots,s_n^t]$ be a given search direction. The Fix step includes finding the location of the deepest valley on the function surface along the line passing through $\vec{G}_0^t$, and coincide with $\vec{S}^t$. The counterpart optimization problem is as follows:

$$\begin{gathered} \min\ f(\vec{G}_0^t + \lambda\vec{S}^t) \\ \text{s. t.} \\ \lambda_{\min} \leqslant \lambda \leqslant \lambda_{\max} \\ \lambda_{\min} = \max_{d=1,\ldots,n}\{((x_{\min,d} - g_{0d}^t)/s_d^t)|_{s_d^t>0}, ((x_{\max,d} - g_{0d}^t)/s_d^t)|_{s_d^t<0}\} \\ \lambda_{\max} = \min_{d=1,\ldots,n}\{((x_{\max,d} - g_{0d}^t)/s_d^t)|_{s_d^t>0}, ((x_{\min,d} - g_{0d}^t)/s_d^t)|_{s_d^t<0}\} \end{gathered} \tag{27}$$

where $\lambda_{\min}$ and $\lambda_{\max}$ are determined such that $\vec{X}_{\min} \leqslant \vec{G}_0^t + \lambda\vec{S}^t \leqslant \vec{X}_{\max}$. Elements $s_d^t, \ \forall\, d = 1,\ldots,d$ are computed consecutively from the following single variable optimization problems, where $\vec{G}_d^t = \vec{G}_{d-1}^t + s_d^t\vec{e}_d$.

$$\begin{gathered} \min\ f(\vec{G}_{d-1}^t + s_d^t\vec{e}_d) \\ \text{s. t.} \\ \gamma_d^t(x_{\min,d} - g_{d-1,d}^t) \leqslant s_d^t \leqslant \gamma_d^t(x_{\max,d} - g_{d-1,d}^t) \end{gathered} \tag{28}$$

where γ_d^t is chosen randomly equal to 0.05 or 1 to preserve local or global explorations, respectively. $\vec{e}_d$ is the dth axis unit vector.

Given that λ^* is optimum for an instance of (27), $\vec{G}_0^t + \lambda^*\vec{S}^t$ is the location of a locally optimum solution resultant from the Fix step. In our implementation we set $\vec{G}_0^t \leftarrow \vec{G}_{best}^t$. Once a new iteration of the F3EA algorithm is started, the Fix step may be activated.

Any single variable search technique (e.g., golden section method or quadratic fit method) can be used to solve (27) and (28). In our implementation of the F3EA algorithm, we use from *fminbnd* function which is available in optimization toolbox of Matlab software. Users

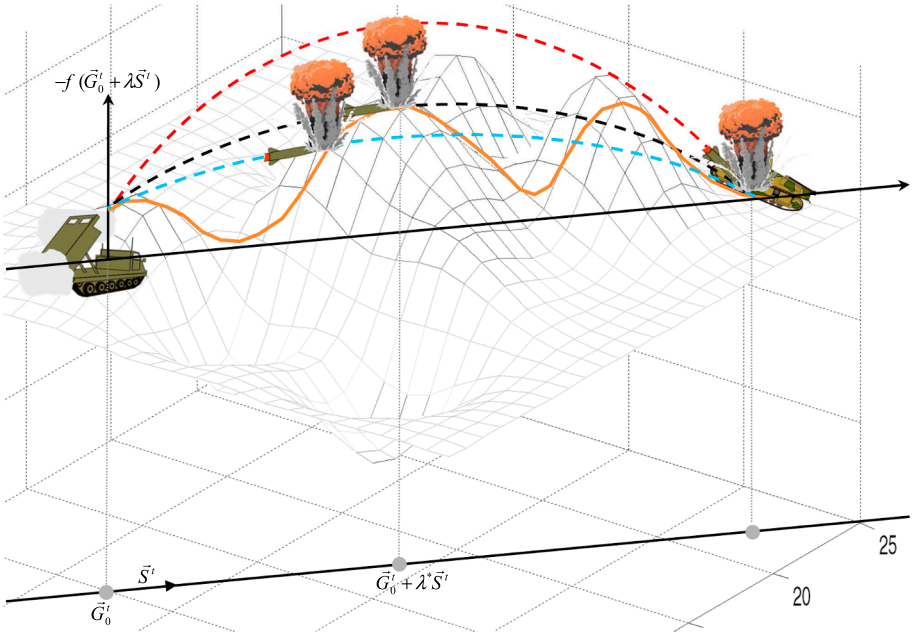


Fig. 8. Conceptual schematic of the Fix step.

may also use their own method to solve (27) and (28). To search for λ^* via *fminbnd*, the number of function evaluations allowed (i.e., *counter*) is an input parameter and is set equal to 10. The very detailed flowchart of the proposed F3EA algorithm implemented in this research is depicted in Fig. 9. A numerical example to show manually the step-wise procedure for generating a new solution by F3EA algorithm is given in Appendix A.

4. Computational experiments

Numerical optimization is a challenging attempt in many scientific areas. Here, we analyze the effectiveness and efficiency of the F3EA algorithm on various engineering optimization problems and test functions, and do comparisons against different algorithms. Our tests are composed of three categories:

1. The first category of experiments deals with nonlinear engineering optimization problems, wherein we test and compare the performance of F3EA algorithm with other rivals on pressure vessel design problem, speed reducer design problem, centrifugal pump design problem, 37-bar planar truss design problem, gear design problem, and the parameter estimation problem for Frequency-Modulated (FM) Sound Waves.
2. The second category of experiments is conducted on nonlinear benchmark functions. In the first experiment, 23 functions which have widely been investigated in other researches are considered. Our aim behind conducting such an experiment is to identify the ability of F3EA algorithm in finding the global optimum value and to evaluate its convergence quality. In the second experiment, we consider a set of 14 harder functions and evaluate the algorithm after performing a long run.
3. Using such experimental frameworks, we empirically show that different steps of the F3EA algorithm add values during optimization process. We also conduct a sensitivity analysis on the important parameters of the algorithm.

Before doing the the aforementioned experimental framework, we first analyze schematically the convergence behavior of the algorithm on the two dimensional minimum-mass tubular column design problem of Fig. 10. In this problem the design parameters are the mean radius of the tube (R) and the wall thickness (t). The nonlinear mathematical model of the problem is as follows:

Minimize $f(\vec{X}) = 2 \times 5 \times 7850Rt$

subject to:

$$g_1(\vec{X}) = \frac{50000}{2\times 250\times 10^6\pi Rt}\left[1 + \frac{2\times 0.02\times(R+0.5t)}{R}\sec\left(\frac{5\sqrt{2}}{R}\sqrt{\frac{50000}{2\times 210\times 10^9\pi Rt}}\right)\right] - 1 \leqslant 0$$

$$g_2(\vec{X}) = 1 - \frac{210\times 10^9\pi^3R^3t}{4\times 50000\times 25} \leqslant 0$$

$$g_3(\vec{X}) = \frac{0.002R}{0.25}\left[\sec\left(5\sqrt{\frac{50000}{210\times 10^9\pi R^3t}}\right) - 1\right] - 1 \leqslant 0$$

$$g_4(\vec{X}) = \frac{R}{50t} - 1 \leqslant 0$$

$$0.01 \leqslant R \leqslant 1,\quad 0.005 \leqslant t \leqslant 0.2.$$

The number of individuals were chosen equal to 100 to provide a schematic comparison between the convergence behavior of F3EA algorithm and weighted superposition attraction (WSA) method (Baykasoglu & Akpinar, 2015) which is a recently proposed swarm intelligence algorithm. In Fig. 11 the contour plot of objective function on the search domain together with the position of individuals in different iterations has been scattered to indicate how fast the population converges to the feasible global optimum point. The blue dot points in each piece of the figure indicate the feasible individuals while the red asterisk points indicate infeasible individuals.

From Fig. 11 it is clear that F3EA can truly converges to the global optimum very fast. Almost all individuals show tendency to move toward feasible global optimum from the very early iterations. Such plots has been depicted for WSA algorithm in Fig. 12. As can be seen the convergence quality of this algorithm is not competitive with that of F3EA. While with 14,374 function evaluations, F3EA almost finishes its

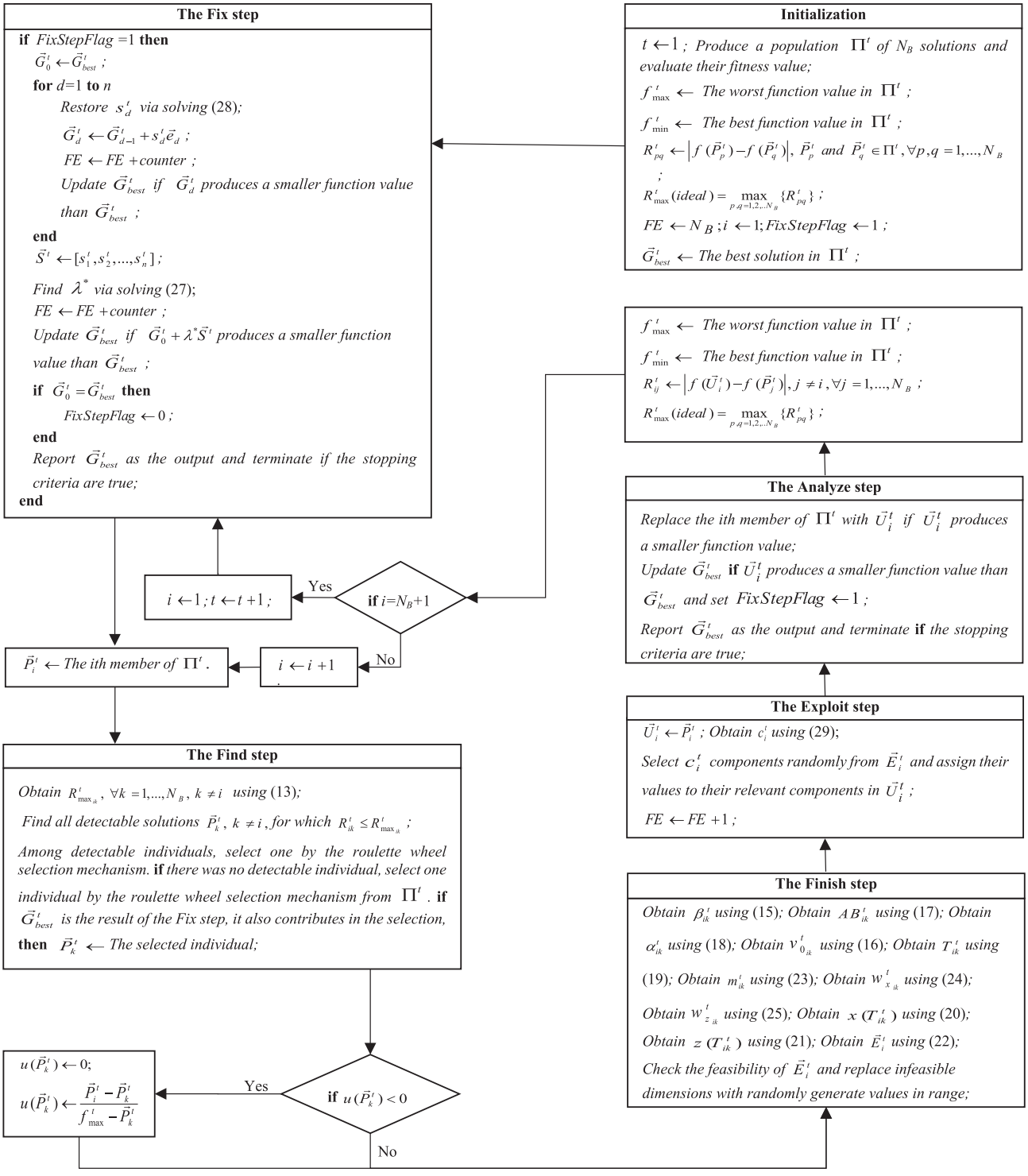


Fig. 9. Very detailed and ready to implement the flowchart of the F3EA meta-heuristic algorithm.

search, since all individuals converge to the global optimum with an acceptable precision, WSA fails to exhibit such a convergence quality even after 50,000 function evaluations. Indeed, such a convergent behavior is resulted by cleverly designed operators, which not only try to be a true model of their inspiration sources, but also help individuals bear the necessary information related to the optimization problem.

4.1. Experiments with engineering optimization problems

Complex real word engineering problems, provide a good yardstick to assess the performance of F3EA algorithm versus other optimization algorithms. Here our test bed includes the following problems

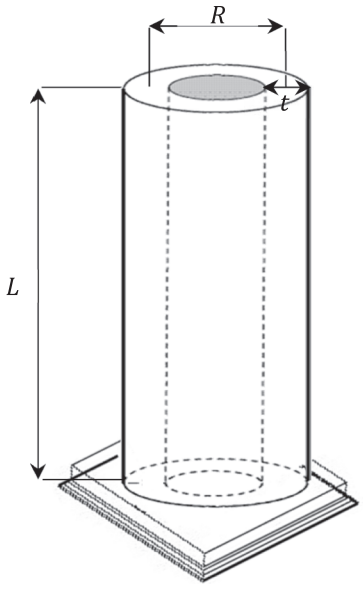


Fig. 10. Tubular column and its design parameter.

- pressure vessel design problem,
- Simply supported 37-bar truss design problem,
- speed reducer design problem,
- centrifugal pump design problem,
- gear design problem,
- parameter estimation problem for Frequency-Modulated (FM) Sound Waves.

Among these problems the last three problems are unconstrained engineering optimization problems, while the first three ones own hard constraints. To handle design constraints and reflect any constraint violation, the objective function is penalized (Gen & Cheng, 2000; Parouha & Das, 2015). The penalty term is $\sum_{\forall j} 10^{+4} cv_j$, where cv_j is the constraint violation associated to constraint j. We compare the performance of F3EA algorithm against a huge set of 39 comparator algorithms. The list of these algorithms is as follows:

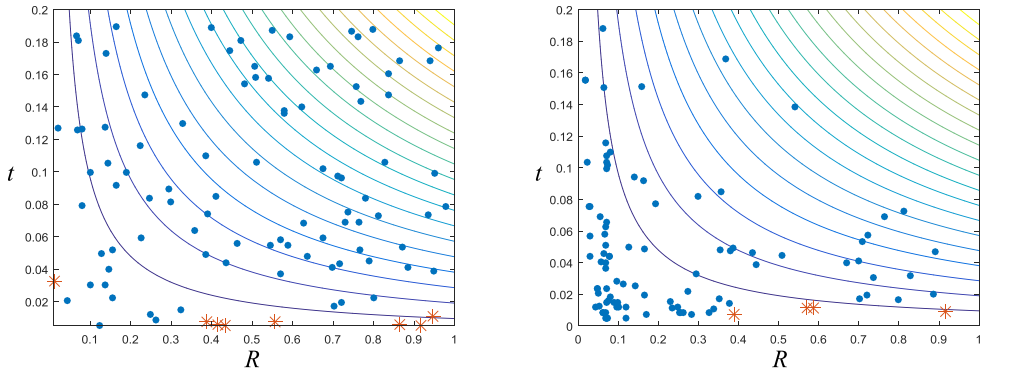
- LCA: League Championship Algorithm (Husseinzadeh Kashan, 2011),
- IPSO: Improved particle swarm optimizer (He, Prempain, & Wu, 2004),
- PSRE: Particle Swarm optimization algorithm and Receptor Editing (Yildiz, 2008),
- COPSO: Constrained Optimization Particle Swarm Optimization (Aguirre, Munoz Zavala, Villa Diharce, & Botello Rionda, 2007),
- MBFOA: Modified Bacterial Foraging Algorithm (Mezura-Montes & Hernández-Ocaña, 2009),
- RSPSO: Ranking Selection-based Particle Swarm Optimizer (Wang & Yin, 2008),
- ABC: Artificial Bee Colony (Akay & Karaboga, 2012),
- SES: Simple Evolutionary Algorithm (Mezura-Montes, Coello, & Landa-Becerra, 2003),
- εPSO: ε Particle Swarm Optimization (Takahama & Sakai, 2006),
- CPSO-DD: Constrained Particle Swarm Optimization (PSO) algorithm-stagnation Detection and Dispersion mechanism (Worasucheep, 2008),
- UPSOm: Unified Particle Swarm Optimization (Parsopoulos & Vrahatis, 2005),
- CPSO: Co-evolutionary Particle Swarm Optimization (He & Wang, 2007),
- CDE: Co-evolutionary Differential Evolution (Huang, Wang, & He, 2007),
- CSA: Crow Search Algorithm (Askarzadeh, 2016),
- Co-evolutionary penalty (Coello, 2000),
- NHGA: Niche Hybrid Genetic Algorithm (Lingyun, Mei, Guangming, & Guang, 2005),
- PSO: Particle Swarm Optimization (Gomes, 2011),
- CSS: Charged System Search (Kaveh & Zolghadr, 2011),
- E-CSS: Enhanced-Charged System Search (Kaveh & Zolghadr, 2011),
- DPSO: Democratic Particle Swarm Optimization (Kaveh & Zolghadr, 2014),
- DE: Differential Evolution (Kaveh, Jafari, & Farhoudi, 2015),
- RO: Ray Optimization (Kaveh & Khayatazad, 2013),
- OMGSA: Orthogonal Multi-Gravitational Search Algorithm (Khatibinia & Naseralavi, 2014),
- SGA: Search Group Algorithm (Goncalves, Lopez, & Miguel, 2015),
- Sic-PSO: SImple Constrained Particle Swarm Optimizer (Cagnina, Esquivel, & Coello, 2008),
- DES: Differential Evolution Strategy (Kim, Chong, Park, & Lowther, 2007),
- WSA: Weighted Superposition Attraction (Baykasoglu & Akpinar, 2015),
- SLPSO: Self-Learning Particle Swarm Optimizer, APSO: Adaptive Particle Swarm Optimization, CLPSO: Comprehensive Learning Particle Swarm Optimizer, CPSOH: Cooperative Particle Swarm Optimizer, FIPS: Fully Informed Particle Swarm, SPSO: Standard Particle Swarm Optimization, JADE: Adaptive Differential Evolution with optional external archive, HRCGA: Real-Coded Genetic Algorithm, FPSO: Frankenstein's Particle Swarm Optimizer, TRIBES-D: An adaptive particle swarm optimization algorithm, APrMF: A Probabilistic Memetic Framework, G-CMA-ES Covariance Matrix Adaptation Evolution Strategy (Li, Yang, & Nguyen, 2012).

Since we have used a large number of comparator algorithms in our experiments, we were not able to tune all of them. So to have a fair comparisons and also have a rigorous evaluation of F3EA algorithm, we used from the results reported in the literature. The replication length is 30 runs for F3EA algorithm on each problem and best, mean, worst, and standard deviation (stdev) statistics are reported accordingly. It is worth to mention that the performance of F3EA algorithm on constrained optimization problems was always feasible. For all engineering optimization problems we set $q_i^t = 0.3$ and $A = 1$. Later in Section 4.2 we perform a sensitivity analysis on the F3EA control parameters.

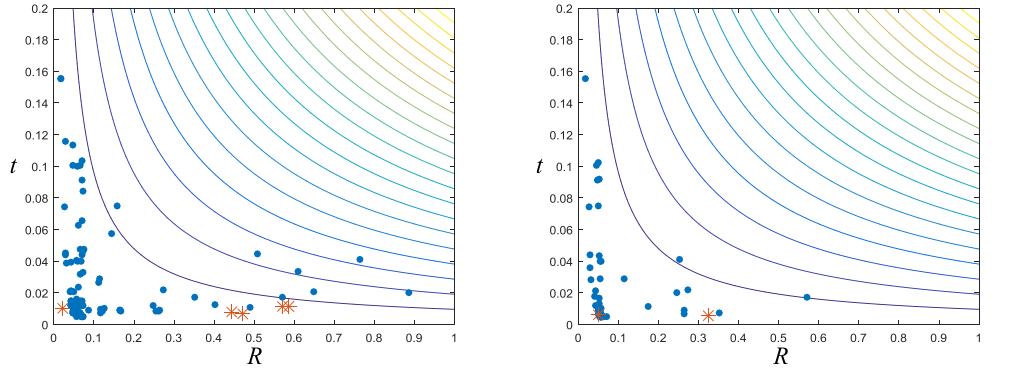
4.1.1. Pressure vessel design problem

The objective in this problem is to minimize the total design costs of a cylindrical vessel, which is capped by hemispherical heads. The design parameters are: shell thickness, head thickness, inner radius, and the cylindrical section length. The first two parameters are integer multiples of 0.0625, while the last two parameters are continuous. The mathematical statement of the problem can be found in Husseinzadeh Kashan (2011).

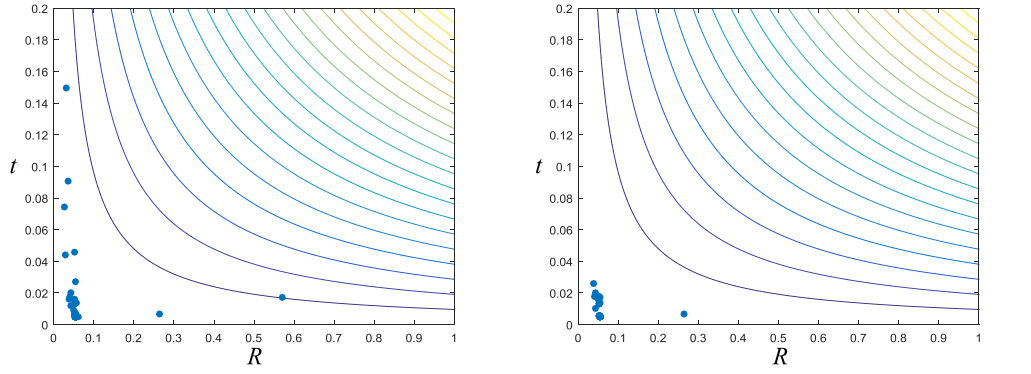
To run F3EA algorithm, the population size (N_B) is set equal to 150. Results, which have been summarized in Table 1, indicate that the performance of F3EA on this challenging design optimization problem is incredibly well. F3EA is the most stable algorithm among 15 rivals which reports the smallest standard deviation value and is the most dependable algorithm in terms of the mean performance. With 30,000



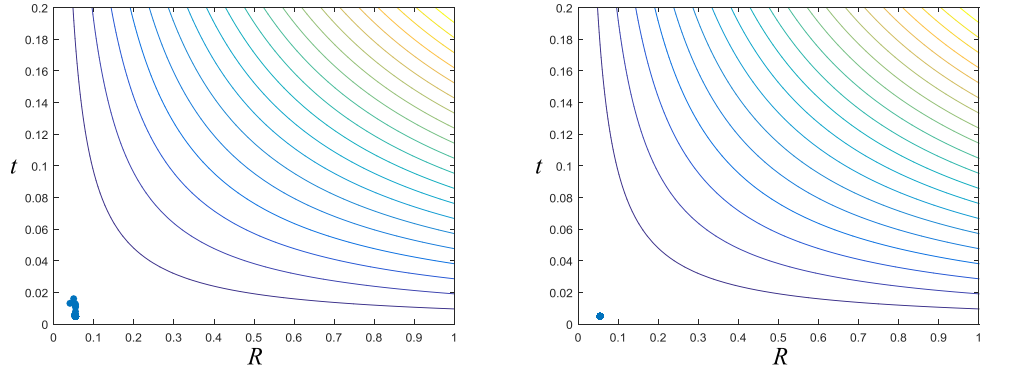
Individuals of 0. iteration (FE=100) Individuals of 2. iteration (FE=350)



Individuals of 4. iteration (FE=598) Individuals of 8. iteration (FE=1116)



Individuals of 15. iteration (FE=1959) Individuals of 30. iteration (FE=3707)



Individuals of 60. iteration (FE=7394) Individuals of 120. iteration (FE=14374)

Fig. 11. Behavior of F3EA on a minimum-mass tubular column problem.

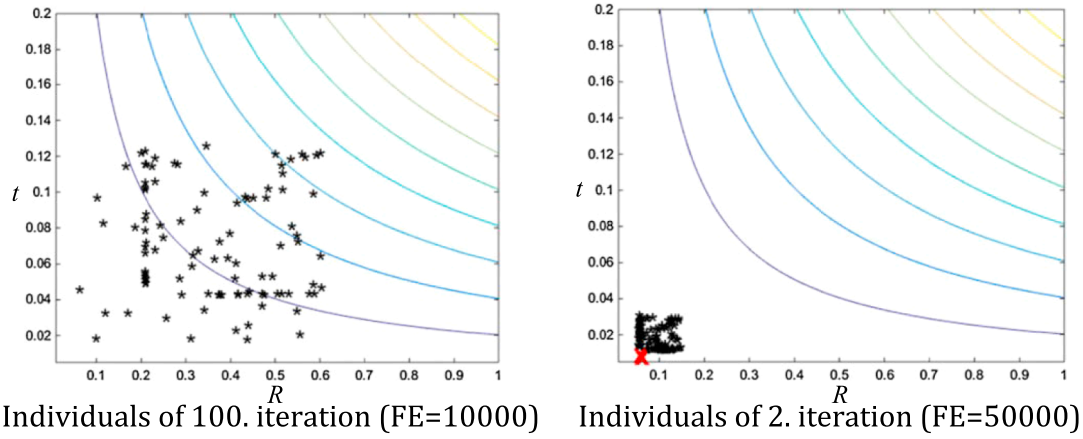


Fig. 12. Behavior of the WSA on a minimum-mass tubular column problem (Baykasoglu & Akpinar, 2015).

evaluations it finds a design very close to the optimal design. Moreover, its worst statistics is the smallest among those rivals which count the same number of function evaluations. These are LCA, COPSO and RSPSO which seems to compete in the best way with F3EA. When the number of evaluations is increased up to 50,000, F3EA's statistics improve. Though they may not show a radical change in the value since it seems that 30,000 evaluations is enough. While, algorithms such as CDE, CSA and Co-evolutionary penalty count a huge number of function evaluations before getting stop, their results are still poor. Here, F3EA reports the best mean and stdev values. In terms of the min value, it is placed second since its gap with the best value reported by CPSO-DD (that is place first here) is about 2E−06. In terms of the max value, the same situation is held. Following our analysis, we came at the point to conclude that F3EA is one of the most effective and efficient algorithms for optimizing the design of a pressure vessel.

Table 2
Statistical comparison of the results on a simply supported 37-bar truss design problem (the best known objective value = NA).

Algorithm	Best	Mean	Stdev	#eval
F3EA	360.279	361.079	0.351	20,000
NHGA	368.84	378.826	9.0325	14,038
Wang, Zhang, and Jiang (2004)	366.5	NA	NA	NA
PSO	377.20	381.2	4.26	NA
CSS	362.84	366.77	3.742	NA
E-CSS	362.38	365.75	3.461	NA
DPSO	360.40	362.21	Nfmnc.hg,1.68	NA
DE	361.03	NA	NA	NA
RO	364.04	NA	NA	NA
OMGSA	359.97	361.96	1.868	NA
F3EA	359.930	360.454	0.325	50,000
SGA	359.93	360.2	0.26	50,000

4.1.2. Simply supported 37-bar planar truss structure

The aim in this problem is to optimize the size and shape of a simply supported 37-bar planar truss structure (Jalili, Husseinzadeh Kashan, & Hosseinzadeh, 2017). A non-structural mass of 10 kg is attached to free nodes at the lower chord of the structure. Young's modulus is 2.1×10^{11} N/m^2 and material density of truss members is 7800 kg/m^3. The cross-sectional areas of all members except than those of the lower chord are considered as design variables. The constant rectangular cross-sectional areas for all members of the lower chord are specified as 4×10^{-3} m^2. The y-coordinates of upper nodes are considered as layout variables where, their vertical position can vary between ± 1.5 m. Moreover, the structure is subjected to frequency constraints $\omega_1 \geq 20$ Hz, $\omega_2 \geq 40$ Hz, $\omega_3 \geq 60$ Hz.

The optimization results obtained by F3EA algorithm have been summarized in Table 2 together with the results of other rivals taken from literature. N_B is set equal to 50. Obviously, F3EA performs well on this hard problem. It truly converges to the optimum and is able to always give a feasible design.

With 20,000 design evaluations, F3EA performs better than many

Table 1
Statistical comparison of the results on a pressure vessel design problem (the best known objective value = 6059.714).

Algorithm	Best	Mean	Worst	Stdev	#eval
F3EA	6059.714410	6065.535905	6090.590368	11.121030	30,000
LCA	6059.728799	6068.553765	6090.675140	12.676393	30,000
IPSO	6059.7143	6289.92881	NA	305.78	30,000
PSRE	6059.7144	6097.446	6156.57	35.781	30,000
COPSO	6059.714335	6071.013366	NA	15.101157	30,000
MBFOA	6059.945	6107.340	NA	8.20E+1	30,000
RSPSO	6059.7143	6066.2032	6100.31956	13.3035	30,000
ABC	6059.714736	6245.308144	NA	2.05E+02	30,000
SES	6059.714355	6355.343115	6846.628418	2.56E+02	36,000
F3EA	6059.714337	6062.168228	6090.527075	7.931293	50,000
LCA	6059.714353	6064.921613	6090.648391	10.442833	50,000
εPSO	6059.7143	6136.7744	6410.0868	112.3306	50,000
CPSO-DD	6059.714335	6072.546102	6090.526430	15.193979	50,000
UPSOm	6544.27	9032.55	NA	995.573	100,000
CPSO	6061.0777	6147.1332	6363.8041	86.4545	200,000
CDE	6059.7340	6085.2303	6371.0455	43.0130	204,800
CSA	6059.71436343	6342.49910551	7332.84162110	384.94541634	250,000
Co-evolutionary penalty	6288.7445	6293.84323196	6308.14965192	7.41328537	900,000

Table 3
Statistical comparison of the results on a speed reducer design problem (the best known objective value = 2996.348165).

Algorithm	Best	Mean	Worst	Stdev	#eval
F3EA	2996.348165	2996.348165	2996.348168	5.08871E−07	30,000
SiC-PSO	2996.348165	2996.4085	NA	NA	24,000
COPSO	2996.372448	2996.408525	NA	2.867E−02	30,000
MBFOA	2999.264	3014.759	NA	11.0	30,000
DES	2996.348	2996.348	NA	7.54E−06	36,000
SES	3025.005127	3088.777816	3226.248291	47.36189	36,000
WSA	2996.348222	2996.348232	2996.348243	9.10931E−06	500,000

Table 4
Gap percent between the PSO and F3EA on a centrifugal pump design problem.

	$p = 1$	$p = 2$	$p = 10$	$p = 100$
$W = 0$	1302.62%	–	–	–
$w = 0.1$	3.59%	0.12%	0.04%	0.86%
$w = 0.2$	1.69%	0.12%	0.36%	1.39%
$w = 0.3$	0.29%	0.71%	0.04%	0.11%
$w = 0.4$	0.56%	0.39%	0.67%	1.04%
$w = 0.5$	0.47%	0.02%	0.11%	0.94%
$w = 0.6$	0.82%	0.51%	0.09%	0.31%
$w = 0.7$	0.04%	0.05%	1.17%	0.19%
$w = 0.8$	0.12%	0.11%	1.71%	0.1%
$w = 0.9$	0.11%	0.65%	1.66%	0.12%
$W = 1$	0.14%	–	–	–

rivals in all statistics. The only exception is the best value of 359.97, obtained by OMGSA, which is better than 360.279 obtained by F3EA algorithm. The very low standard deviation indicates that the algorithm always converges toward the optimum design. When the number of structural analysis is increased to 50,000, improvement is attainable in the best and mean performance of F3EA algorithm. However, this improvement is not enough for F3EA to beat SGA. While the best performance of SGA is similar to F3EA, its mean performance is a little bit better. From comparisons with other rivals it can be concluded that F3EA algorithm is effective and more dependable than many other methods in obtaining a better structural design.

4.1.3. Speed reducer design problem

The objective in this problem is to minimize the weight of a speed reducer such that the design constraints being enforced. There are seven design parameters in this problem such as the face width, the teeth module, the number of teeth in the pinion, the length of the first shaft, the length of the second shaft, the first shaft diameter, and the second shaft diameter. The constraints are on the bending stress of the gear teeth, surface stress, transverse defections of the shafts and stresses in the shaft. The mathematical formulation of this problem can be found in Husseinzadeh Kashan (2011).

To run F3EA algorithm, the population size (N_B) is set equal to 15. Results are tabulated in Table 3. As can be seen, F3EA exhibits an acceptable performance compared with 6 rivals. The standard deviation reported by F3EA is very low that indicates it finds stably the true optimum. DES is also able to exhibit a very good competitive performance. However, it evaluates 6000 functions more than F3EA. SiC-PSO is also able to find the global optimum. However, its mean performance is not as competitive as that of F3EA or DES algorithms. COPSO is also not able to show a competitive performance. The speed reducer design problem can be a challenging problem for algorithms such as SES or MBFOA since their performance lies far away from the optimum. Even counting a huge number of function evaluations is not enough for an algorithm like WSA to obtain the true optimum with an acceptable precision. As a conclusion, F3EA can be a dependable algorithm for solving the speed reducer design optimization problem with relatively a small amount of function evaluations.

4.1.4. The centrifugal pump design optimization problem

To verify the effectiveness of the proposed F3EA algorithm on another real-world application, we select an engineering optimization problem, which is the centrifugal pump design optimization problem. Centrifugal pumps are turbo pumps which are used generally for lifting fluids via rotating impeller to increase the fluid pressure. η (that should be maximized) and *NPSHr* (that should be minimized) are the two most significant parameters of such pumps. The polynomial models of η and *NPSHr* can be found in Nourbakhsh, Safikhani, and Derakhshan (2011).

β_{1Hub}, $\beta_{1Shroud}$, γ_{mid} and β_2 are the design parameters. The engineering design of a centrifugal pump calls for simultaneous optimization of η and *NPSHr* with regard to design parameters. The compromise programming counterpart of the multi-objective design optimization problem can be given by:

Table 5
Statistical comparison of the results on a gear train design problem (the best known objective value = 2.700857E−12).

Algorithm	Best	Mean	Worst	Stdev	#eval
F3EA	2.700857E−12	2.26679E−10	1.36165E−09	3.97161E−10	10,000
CSA	2.700857E−12	2.05932E−09	3.18473E−08	5.059779E−09	10,000
F3EA	2.700857E−12	6.63503E−11	6.60209E−10	1.23193E−10	20,000
SLPSO	2.70E−12	2.22E−09	6.19E−09	9.83E−09	20,000
APSO	2.70E−12	1.59E−09	1.31E−08	1.44E−08	20,000
CLPSO	2.79E−12	1.99E−10	1.36E−09	2.22E−09	20,000
CPSOH	1.54E−10	2.80E−07	2.02E−06	2.33E−06	20,000
FIPS	8.88E−10	3.27E−08	8.9E−07	8.77E−07	20,000
SPSO	2.70E−12	1.39E−08	2.56E−07	2.57E−07	20,000
JADE	2.70E−12	2.10E−10	1.36E−09	2.59E−09	20,000
HRCGA	2.70E−12	1.53E−10	1.18E−09	1.88E−09	20,000
FPSO	2.057E−09	1.57E−05	0.00017	0.00018	20,000
TRIBES−D	9.64E−12	3.62E−06	7.17E−05	1.26E−05	20,000
APrMF	2.70E−12	1.01E−09	2.36E−09	6.84E−10	20,000
G−CMA−ES	2.70E−12	2.44E−02	7.32E−01	1.31E−01	20,000
ABC	2.700857E−12	3.641339E−10	NA	5.52581E−10	30,000
UPSOm	2.700857E−12	3.80562E−08	NA	1.09631E−07	100,000

Table 6
Statistical comparison of the results on the parameter estimation for an FM sound waves synthesis (the best objective value = 0).

Algorithm	Best	Mean	Worst	Stdev	#eval
F3EA	1.9641E−14	2.735721	12.542048	5.058143	30,000
SLPSO	0	4.18	13.79	26.99	30,000
APSO	0	11.33	34.22	41.13	30,000
CLPSO	0.007	3.82	14.08	23.53	30,000
CPSOH	3.54	27.08	42.52	60.61	30,000
FIPS	0	5.93	15.11	25.75	30,000
SPSO	0	9.88	18.27	33.85	30,000
JADE	0	7.55	13.92	26.18	30,000
HRCGA	0	8.41	17.59	32.54	30,000
FPSO	0	5.22	15.82	28.31	30,000
TRIBES-D	2.22	14.68	22.24	4.57	30,000
APrMF	0.063	0.54	0.94	0.22	30,000
G-CMA-ES	3.326	38.75	55.09	16.77	30,000

$$\text{Minimize } \left(w \left| \frac{(\eta^* - \eta)}{(\eta^* - \eta_*)} \right|^p + (1-w) \left| \frac{(NPSHr^* - NPSHr)}{(NPSHr^* - NPSHr_*)} \right|^p \right)^{\frac{1}{p}}$$

$$\text{subject to:}$$

$$0^\circ \leqslant \beta_{1Hub} \leqslant 30^\circ$$

$$30^\circ \leqslant \gamma_{mid} \leqslant 70^\circ$$

$$40^\circ \leqslant \beta_2 \leqslant 60^\circ$$

$$60^\circ \leqslant \beta_{1Shroud} \leqslant 89^\circ$$

where A^* and A_* are the upper bound and lower bounds of the criterion A. w is a weight coefficient. As a comparator algorithm, we use from PSO algorithm with the following recommended parameters: *swarm size* = 20, $V_{min} = -4$, $V_{max} = 4$, $c_1 = c_2 = 1.49445$ and *maximum iterations* = 50. The inertia weight is also reduced from 1 to 0, linearly. To have a fair comparison, for the F3EA algorithm, *NB* is set equal to 20 and FE_{max} is set equal to 1000. Results under different levels of w and p are summarized in Table 4. In this table, the gap percent between the performance of PSO and F3EA are reported in which a positive gap percent indicates the better performance of F3EA compared to PSO. From the results, it can be seen that on all compromise optimization problem instances, obtained based on different levels of w and p, the F3EA algorithm performs better than PSO. When the objective is only minimization of *NPSHr* (w = 0) there is a very large gap value, indicating that F3EA performs far better than PSO. However, for greater values of w the gap percent may be negligible.

4.1.5. The gear train design optimization problem

The aim in this problem is to optimize the gear ratio of a compound gear train composed of four gears. The target gear ratio is 1/6.931 and the design should be in such a way that gear ratio being as close as possible to the target ratio. The number of teeth of each gear is the design variable which must be an integer between 12 and 60. The mathematical statement of this function is as $f(\vec{X}) = (1/6.931 - x_A x_B / x_D x_F)^2$, $x_A, x_B, x_D, x_F \in [12, 60]$. Except the boundary constraints, there are no further constraints. We set $N_B = 200$ and summarized the results in Table 5. Only 10,000 function evaluations is enough for F3EA to shows its potency. Except the best values which are the same and optimal, on other statistics F3EA outperforms CSA. When 20,000 function evaluations is allowed, F3EA improves its performance and beats all 14 rivals in terms of mean, worst and stdev criteria. Though, there are several algorithms which perform competitive, in terms of the absolute performance they are inferior to F3EA algorithm.

4.1.6. The parameter estimation of an FM synthesizer

Frequency-Modulated (FM) sound wave synthesis has an important role in modern music systems. In the parameter estimation of an FM synthesizer the aim is to generate a sound similar to target sound. This problem is a highly complex with strong epistasis multimodal problem. There are 6 design parameters a_1, ω_1, a_2, ω_2, a_3, ω_3 which are defined in range $[-6.4, 6.35]$. The function is introduced as $f(\vec{X}) = \sum_{t=0}^{100} \left(a_1 \sin(\frac{2\pi}{100} t\omega_1 + a_2 \sin(\frac{2\pi}{100} t\omega_2 + a_3 \sin(\frac{2\pi}{100} t\omega_3))) - \sin(\frac{\pi}{10} t - 1.5 \sin(\frac{9.6\pi}{100} t + 2 \sin(\frac{9.8\pi}{100} t))) \right)^2$. We set $N_B = 75$ and summarized the results in Table 6. While, none of 13 algorithms in this table is able to find the global optimum of this challenging function for all of the 30 runs, it can be seen that the performance of F3EA is acceptable and better than most of comparator algorithms. The mean and worst performance of F3EA is the second best among 12 rivals. Here, APrMF is the best of all algorithms. However, in terms of min value, APrMF performs weak but F3EA performs very close to optimal. G-CMA-ES, CPSOH and TRIBES-D are the weakest algorithms and other rivals' performance lies within the continuum.

4.2. The effect of F3EA algorithm control parameters

In this section we investigate the effect that various setting for the two control parameters of the F3EA algorithm may have on its performance. These parameters are parameter A (used in the find step) and

Table 7
Statistical results obtained by the F3EA algorithm under different levels of q_i^t.

	Statistics	$q_i^t = -1$	$q_i^t = -0.3$	$q_i^t = 0.3$	$q_i^t = 0.999$	q_i^t: Dynamic
Pressure vessel design	Best	6059.714337	6059.714359	6059.714410	6060.254789	6059.938983
	Mean	6068.992864	6066.929342	6065.535905	6173.562694	6074.186724
	Worst	6090.537456	6090.530115	6090.590368	6443.268042	6091.901012
	Stdev	13.696685	13.241330	11.121030	112.048571	14.313663
37-bar truss design	Best	361.309554	361.135856	360.279246	362.269554	360.505352
	Mean	364.438135	363.390618	361.079542	364.290330	361.623822
	Worst	372.363329	369.272526	361.957553	367.172075	363.259992
	Stdev	2.302944	1.971115	0.351197	1.186362	0.658118
Speed reducer design	Best	2996.348165	2996.348165	2996.348165	2996.348165	2996.348165
	Mean	2996.348168	2996.348165	2996.348165	2996.348165	2996.348177
	Worst	2996.348244	2996.348172	2996.348168	2996.348177	2996.348480
	Stdev	1.44219E−05	1.31418E−06	5.08871E−07	2.12102E−06	5.73039E−05
Gear train design	Best	2.70086E−12	2.70086E−12	2.70086E−12	2.70086E−12	2.70086E−12
	Mean	4.21283E−11	1.49297E−10	2.26679E−10	1.01089E−09	2.25079E−10
	Worst	1.54505E−10	1.36165E−09	1.36165E−09	2.35764E−09	1.36165E−09
	Stdev	4.63034E−11	3.36186E−10	3.97161E−10	8.52210E−10	3.72385E−10
Parameter estimation for FM sound waves	Best	1.14430E−13	1.67026E−14	1.96410E−14	3.72549E−12	7.14834E−14
	Mean	13.164721	8.745440	2.735721	8.823814	6.379750
	Worst	25.107337	21.430257	12.542048	18.396648	15.373278
	Stdev	5.865482	7.262202	5.058143	6.901888	6.230510

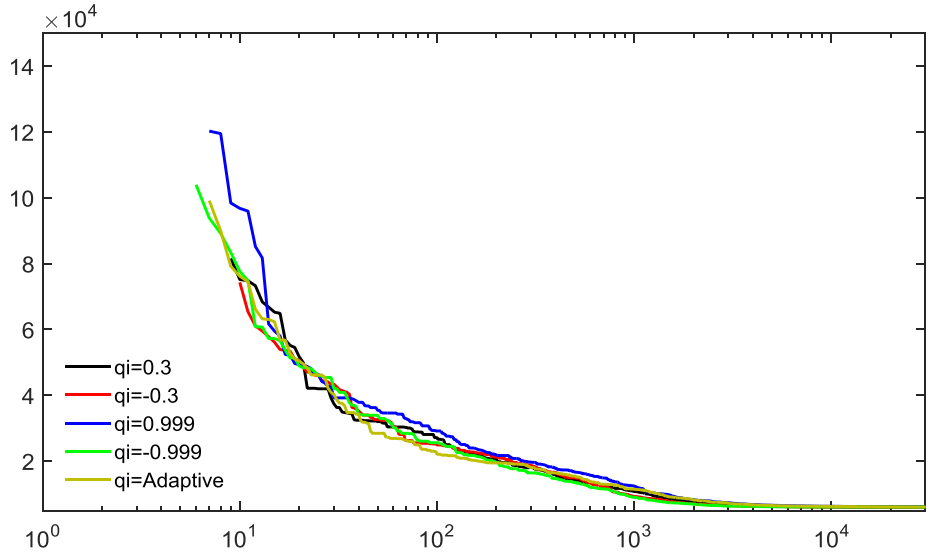


Fig. 13. Evolution of the mean of the best function values under different levels of q_i^t for a pressure vessel design problem.

parameter q_i^t (used in the exploit step). At a fixed level of A we investigate the effect of q_i^t and vice versa. Later in Section 4.2.2 we will demonstrate that the performance of F3EA algorithm is not very sensitive to the exact setting of A and it may be excluded from set of control parameters. Therefore, the number of input parameters of F3EA algorithm can be reduced to two, e.g. the population size (N_B) and q_i^t. Before proceeding to the next section it is worth to note that like for all population based algorithms, the population size (N_B) is also a control parameter for the F3EA. It is obvious that large or small population sizes have effect on the behaviour of the algorithm. So, it is not expected to have always the best ever performance by the algorithm under a fixed value of the population size. We considered different values for population sizes to achieve the best performance on engineering design optimization problems. Because each engineering design problem has its own characteristics. However, we used variable population size for different problem sets. For example for all of 23 test problems considered in Experiment 1 in Section 4.3.1 or all of 14 test problems considered in Experiment 2 in Section 4.3.2 the population size is constant.

4.2.1. *Effect of* q_i^t

The parameter q_i^t used in (26) controls the amount of genes in the parent solutions ($\vec{P}_i^t$) which are replaced by the offspring ($\vec{E}_i^t$) genes. To understand how different setting of q_i^t alter the performance of F3EA algorithm, five levels are considered e.g., 0.999, 0.3, -0.3, -1 and dynamic update of q_i^t (Lin & Gen, 2008). By dynamic update we mean that $q_i^t = \max(-1, -2(FE/FE_{\max})^2 + 1)$, where FE is the total number of functions evaluated so far and FE_{max} is the maximum allowed evaluations. Based on such a scheme, at the start of the search the value of q_i^t is initialized at 0.999, which enforces very few number of changes. The value of q_i^t decreases to -1 at the end of the search to impose changes almost in all dimensions. Results are summarized in Table 7. Figs. 13–17 demonstrate the progress of the mean of best function values among 30 runs obtained by F3EA algorithm under various setting for q_i^t at $A = 1$ for engineering design problems.

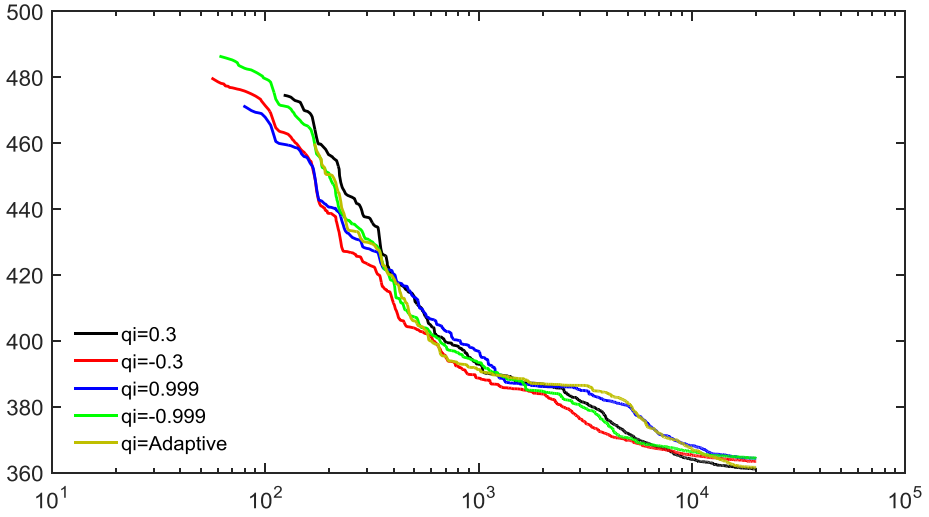


Fig. 14. Evolution of the mean of the best function values under different levels of q_i^t for a 37-bar truss design problem.

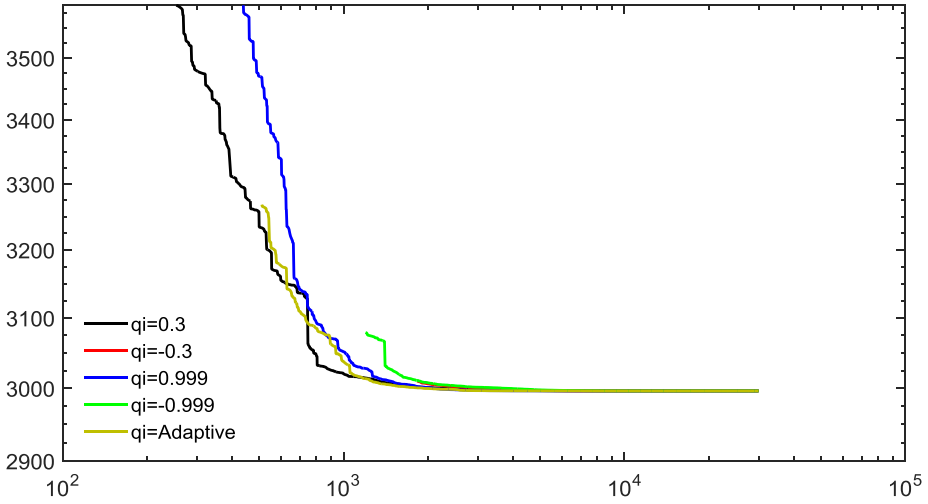


Fig. 15. Evolution of the mean of the best function values under different levels of q_i^t for a speed reducer design problem.

Back to Fig. 18, generally $q_i^t = 0.3$ provides a good result almost on all problems. There exist always an alternate setting which dominates $q_i^t = 0.999$ on each problem. So $q_i^t = 0.999$ would not be recommended here. A same pattern of mean performance on pressure vessel design, 37-bar truss design and parameter estimation for FM sound waves problems is seen by F3EA under different setting of q_i^t. That is, the best mean performance is achieved by $q_i^t = 0.3$, and $q_i^t = -0.3$ yields better mean than $q_i^t = -1$. On gear train design and speed reducer design problems almost all setting of q_i^t provide good results. When q_i^t is controlled dynamically, results are also acceptable. This indicates that a dynamic control, in place of exact setting, can provide acceptable results.

4.2.2. Effect of A

The parameter A used in (13) controls the selection pressure. To understand how different setting of A alter the performance of F3EA, four levels are considered e.g., 0.01, 1, 5 and 10. Results are summarized in Table 8. Figs. 19–23 demonstrate the progress of the mean of best function values among 30 runs obtained by F3EA algorithm under various setting for A at $q_i^t = 0.3$ for engineering design problems. From these figures one can conclude that the progress of the mean of best function values under different levels of A is rather the same. However, plots for $A = 10$ on Figs. 22 and 23 deviate from others indicating a premature convergence.

Fig. 24 demonstrate that $A = 1$ provides the best results for engineering design problems in overall. Generally, as the value of A increases the performance of F3EA algorithm is diminished. However, the reader should be noted that such a conclusion is just based on our sensitivity analysis on these five engineering design problems. Our preliminary experiments on other problems show that a values less than 5 seems to be suitable for A. Since the algorithm performs well under $A = 1$, hence parameter A could be fixed and being excluded from the sets of parameters.

4.3. Experiments with benchmark functions

4.3.1. Experiment 1

There are 23 functions in this experiment, by which we compare the effectiveness and efficiency of the F3EA algorithm on a collection of well-recognized algorithms (e.g., COIO, TLBO, ABC and CMA-ES).

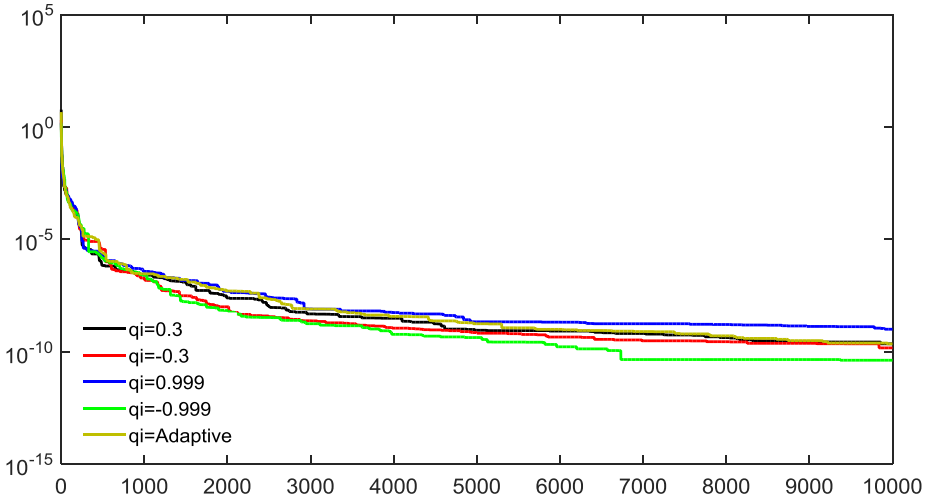


Fig. 16. Evolution of the mean of the best function values under different levels of q_i^t for a gear train design problem.

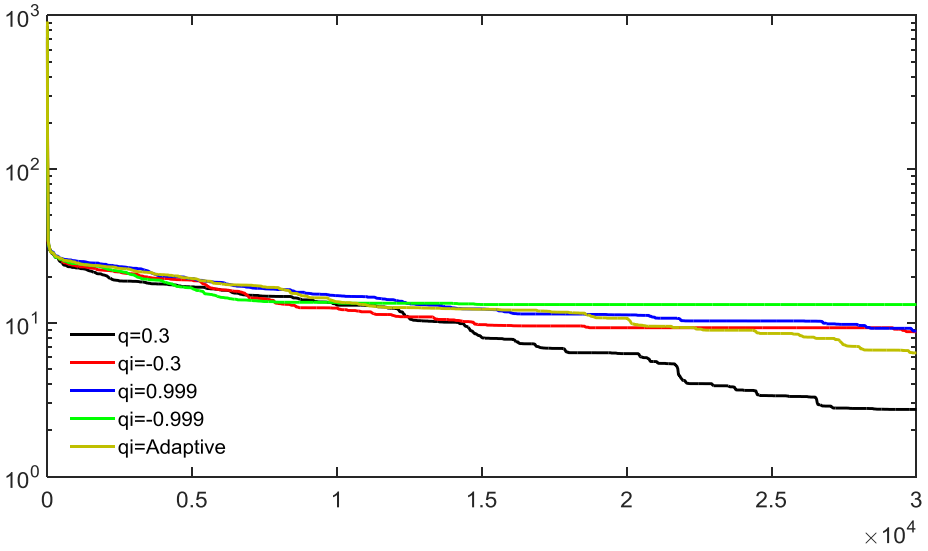


Fig. 17. Evolution of the mean of the best function values under different levels of q_i^t for the parameter estimation for an FM sound waves problem.

These functions have been used widely in previous researches such as Karaboga and Akay (2009), He, Wu, and Saunders (2009), Rao and Patel (2013) and Husseinzadeh Kashan (2015b) and range from unimodal (P1-P7) to multimodal (P8-P23) and from low-dimensional (P14-P23) to relatively high-dimensional (P1-P13). The mathematical statement of these functions can be found in Husseinzadeh Kashan (2015b). The termination criteria for all algorithms is either evaluation of 1E+5 solutions or getting the gap of 1E−3. Comparisons are made based on mean and standard deviation of the number of solutions evaluated among 50 replications, percentage of the successful runs (SR) and the significance test (SIG) (see Table 9).

Here the population size (N_B) is set to 50 and kept constant for the whole set of benchmark functions. We set $A = 1$ and use a dynamic update scheme for q_i^t. Results of the mean performance demonstrate that F3EA gets the optimum of 16 out of 17 functions quicker than CMA-ES and on one problem (P3) it performs slower. On 6 problems, it is not possible to conduct comparisons because of the inability of algorithms to find the global optimum. F3EA achieves optimum of 21 out of 23 functions quicker than ABC and only on 2 problems it fails to perform faster. Comparing with TLBO, the record is 19 out of 23 for F3EA and only 3 out of 23 for TLBO. On Quartic function, both algorithms never obtain the optimum. Finally, F3EA surpasses COIO on 16 out of 23 problems in terms of convergence speed.

In summary, on 15 out of 23 functions, F3EA algorithm exhibits the fastest average convergence toward the global minimum; on 7 out of 23 functions COIO is the fastest and only on one problem TLBO shows a rapid convergence. Albeit, the true number of function evaluations by TLBO, as presented in Rao and Patel (2013), is approximated and greater than those reported in Table 9.

The success rates (SR) are also reported in Table 9 for each algorithm on each function. We see that COIO is always talented in finding the minimum of all functions. F3EA is also able to exhibit a powerful performance of finding the global minimum of 21 out of 23 functions in all of 50 runs. Moreover, on P5, F3EA finds the true minimum in several runs. None of TLBO, ABC and CMA-ES are able to perform optimum in all run for at least one problem. CMA-ES, ABC and TLBO achieve 100% success rate for 8, 16 and 19 out of 23 functions, respectively.

We desist to report the best function values obtained by F3EA algorithm, since it finds the global optimum of all functions (except for P7). However, on Rosenbrock (P5) and Quartic (P7) functions, F3EA is failed to achieve 100% success rate. For these problems, we report the mean and standard deviation values in parenthesis. As can be seen, the performance of F3EA is better than ABC and TLBO on P7 and closely worse than ABC on P5. Later in Table 10 we show that F3EA performs incredibly well and far better than ABC on the challenging P5 function when more number of *FE*s is allowed.

Following Zhang, Luo, and Wang (2008), a statistical analysis is conducted between the mean performance of F3EA versus ABC, TLBO and COIO algorithms, by means of an approximate two-sample *t*-test, according to following statistics:

$$t_0 = \frac{\bar{y}_1 - \bar{y}_2}{\sqrt{S_1^2/n_1 + S_2^2/n_2}} \tag{29}$$

$$df = \left\lfloor 1/\left(\left(\left(\frac{S_1^2/n_1}{S_1^2/n_1+S_2^2/n_2}\right)^2/n_1\right) + \left(\left(\frac{S_2^2/n_2}{S_1^2/n_1+S_2^2/n_2}\right)^2/n_2\right)\right)\right\rfloor \tag{30}$$

where $\bar{y}_1$ and $\bar{y}_2$ are the mean values and S_1 and S_2 are the standard deviation obtained by two different algorithms. n_1 and n_2 are the number of independent runs. *df* is the degree of freedom. Results of significance test have been summarized in Table 9, wherein the signs "−", "+", and "∼" denote that the number of functions evaluated by the corresponding algorithm to reach required precision, is worse, better, and similar to that of the F3EA algorithm, respectively. From Table 9, it can be seen that F3EA performs faster than others on many functions. Our proposed algorithm performs significantly faster than ABC on 19 functions and performs a slower convergence only on 2 functions. Comparing to TLBO, the results are significant on 17 functions for the F3EA and significant only on 3 functions for TLBO. Finally, F3EA shows a faster convergence than COIO on 16 out of 23 functions. On seven functions, COIO exhibits a faster convergence.

In summary, from the results of Experiment 1, we see that the proposed F3EA algorithm is very efficient and effective and provides the best results faster than others.

4.3.2. Experiment 2

A set of 14 functions taken from Karaboga and Akay (2009) are the test bed of this experiment. It is worth to mention that, the set of

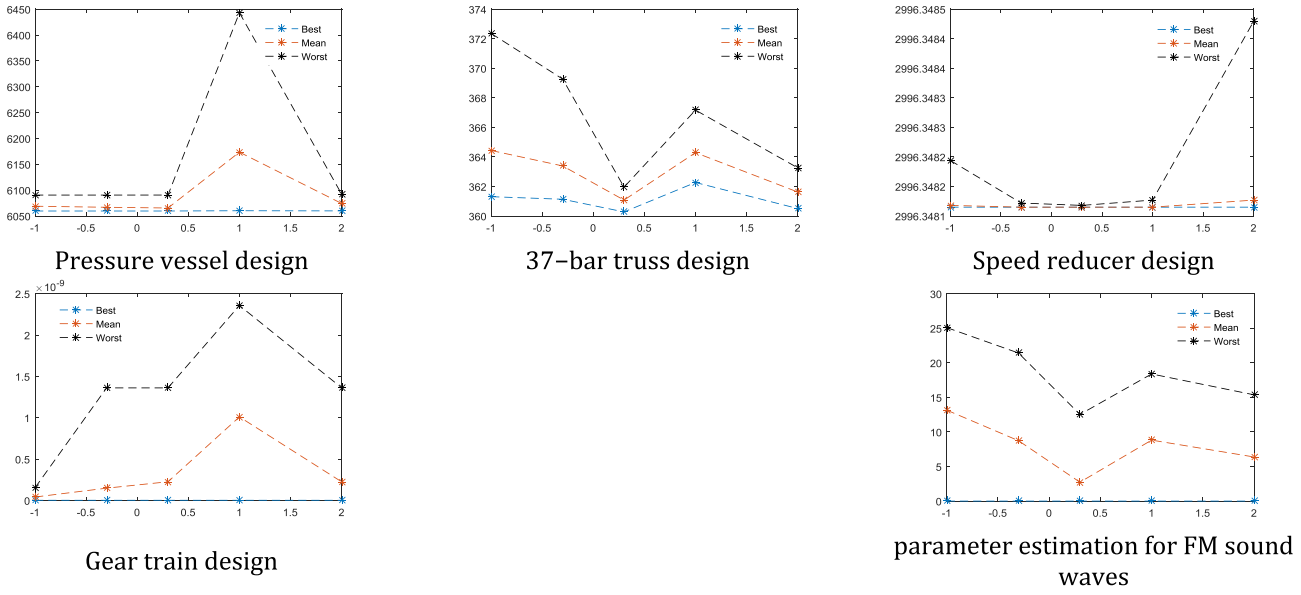


Fig. 18. Performance of F3EA under different setting for q_i^t ("2" on $\times$ axis is for adaptive update of q_i^t).

Table 8
Statistical results obtained by the F3EA algorithm under different levels of A.

	Statistics	$A = 0.01$	$A = 1$	$A = 5$	$A = 10$
Pressure vessel design	Best	6059.714729	6059.71441	6059.730909	6059.715336
	Mean	6067.014051	6065.535905	6073.005141	6079.746642
	Worst	6090.921717	6090.590368	6097.903236	6092.903754
	Stdev	11.87301924	11.12103048	12.97768857	14.95465177
37-bar truss design	Best	360.5322019	360.4532413	360.5914425	360.4879927
	Mean	361.1197147	361.0795423	361.8499024	362.9608297
	Worst	362.1774908	361.957553	363.8902818	368.1911299
	Stdev	0.401784857	0.351197165	0.835929801	1.78556847
Speed reducer design	Best	2996.348165	2996.348165	2996.348165	2996.348165
	Mean	2996.348165	2996.348165	2996.348165	2996.348165
	Worst	2996.348165	2996.348168	2996.348168	2996.348172
	Stdev	1.18338E−07	5.08871E−07	6.04351E−07	1.22008E−06
Gear train design	Best	2.70086E−12	2.70086E−12	2.70086E−12	2.70086E−12
	Mean	4.23815E−10	2.26679E−10	4.50089E−10	1.44497E−09
	Worst	2.35764E−09	1.36165E−09	2.35764E−09	1.82738E−08
	Stdev	6.10835E−10	3.97161E−10	6.87226E−10	3.30252E−09
Parameter estimation for FM sound waves	Best	4.51939E−15	1.9641E−14	3.14987E−15	1.14036E−14
	Mean	3.904867	2.735721	4.072101	10.906626
	Worst	12.553617	12.542048	14.813067	19.639355
	Stdev	5.403326	5.058143	5.987534	6.048209

functions available in Karaboga and Akay (2009) is composed of 50 functions that most of them either have been already considered in Experiment 1 or are relatively easy for existing algorithms. The set of 14 benchmark faunctions considered in this sections are the hardest functions in the set and some of them are really hard enough that many optimization algorithms are prone to get stuck in local optima. The mathematical statement of these functions can be found in Husseinzadeh Kashan (2015b).

To conduct a fair comparison, all algorithms are executed 30 times on each of 14 functions with FE_{max} = 500,000. Results provided by F3EA algorithm are compared against those of COIO, TLBO, ABC, DE, PSO and GA. For F3EA, N_B is set equal to 120. In Table 10, the mean and standard deviation of the best function values found in each run are reported. Values less than 10E−12 are treated as 0. From Table 10, it is seen that F3EA is the most effective algorithm for the reason that it demonstrates a performance exceeded or equal to those which are provided by rivals, on many functions. On 11 out of 14 functions, the performance of F3EA is ranked first. This record for COIO, TLBO, ABC, DE, PSO and GA is 8, 6, 3, 4, 4 and 0, respectively.

From the results of the significance test, we infer that on all functions, F3EA significantly performs better than GA. None of PSO, DE and ABC algorithms are able to show a better (significant or insignificant) result than F3EA. TLBO is able to perform significantly better than F3EA only on one function, and on the rest of 13 functions it is either

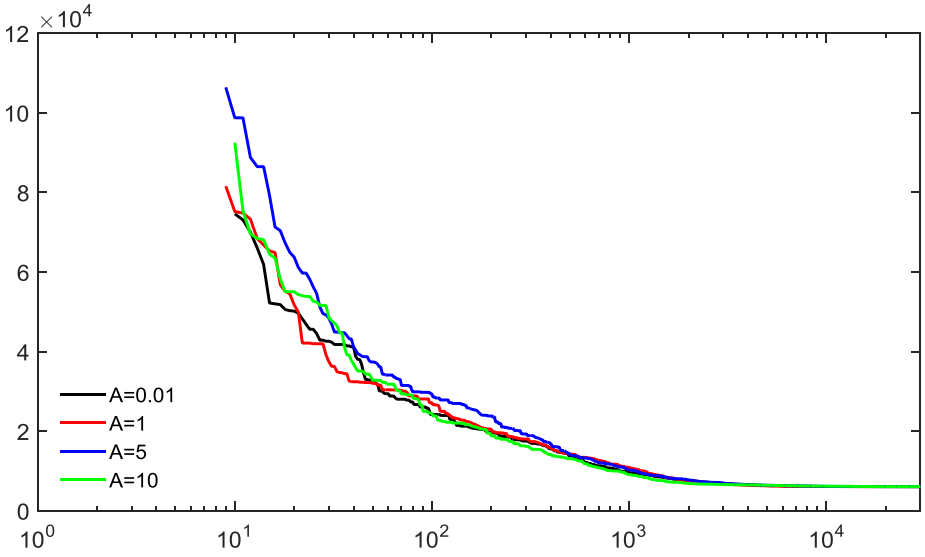


Fig. 19. Evolution of the mean of the best function values under different levels of *A* for a pressure vessel design problem.

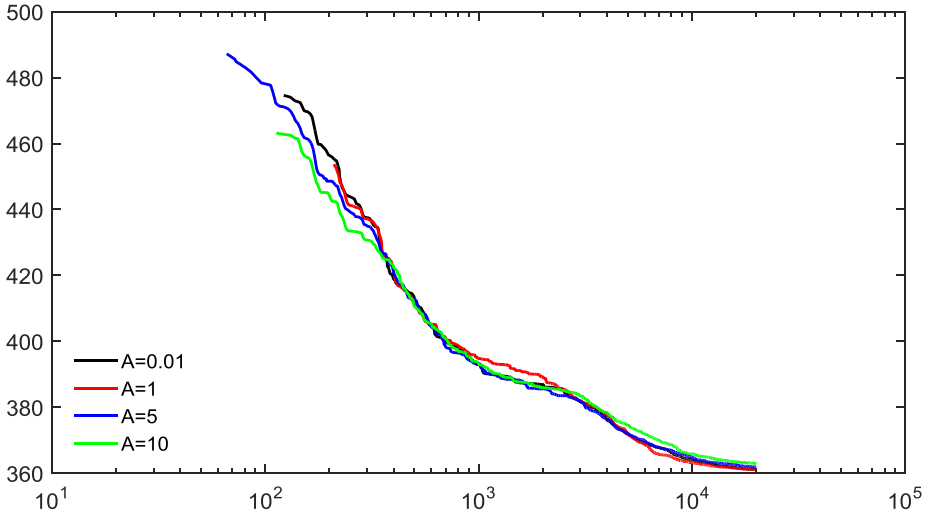


Fig. 20. Evolution of the mean of the best function values under different levels of *A* for a 37-bar truss design problem.

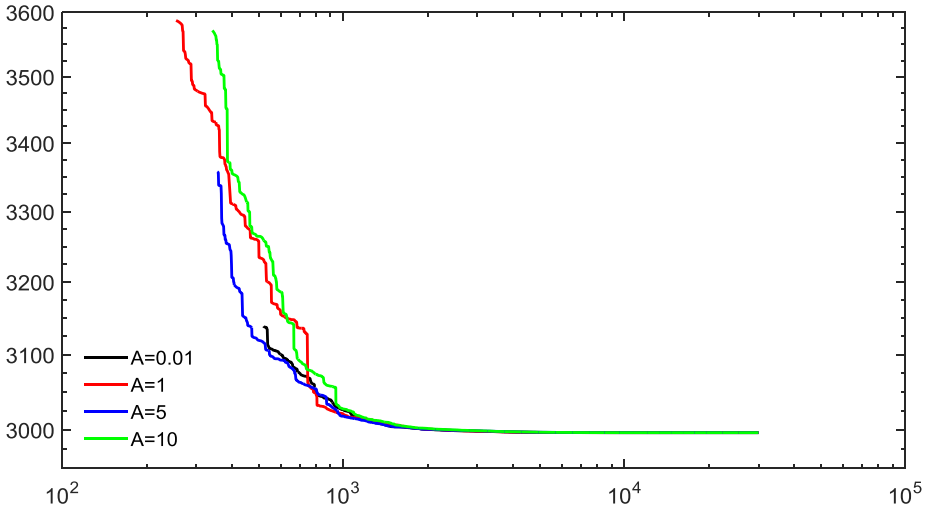


Fig. 21. Evolution of the mean of the best function values under different levels of *A* for a speed reducer design problem.

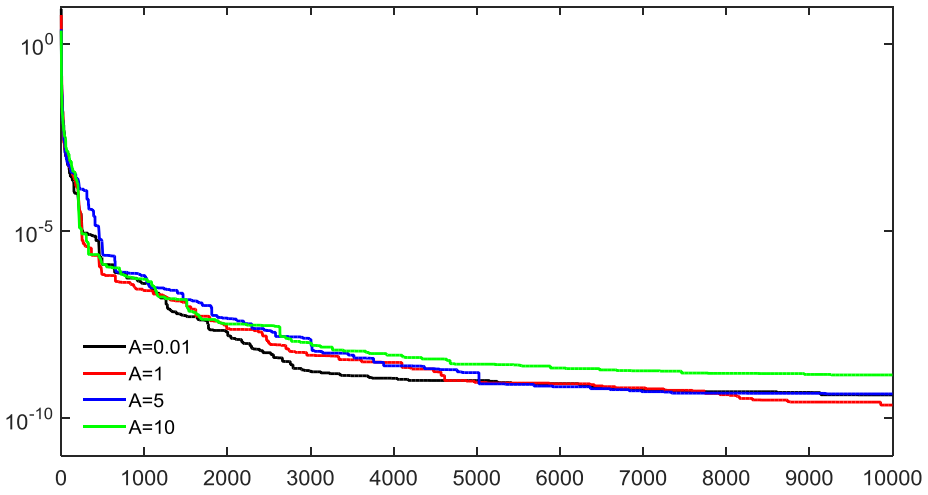


Fig. 22. Evolution of the mean of the best function values under different levels of *A* for a gear train design problem.

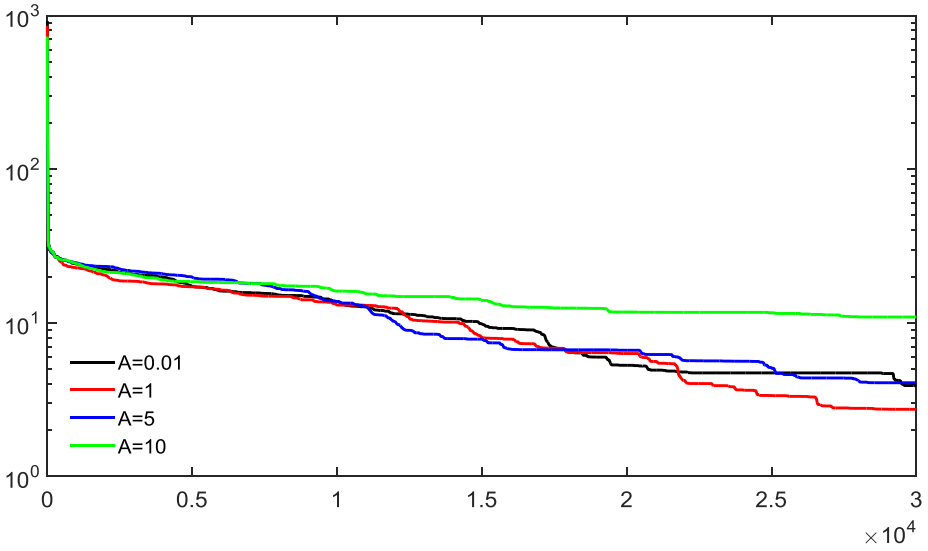


Fig. 23. Evolution of the mean of the best function values under different levels of *A* for the parameter estimation for an FM sound waves problem.

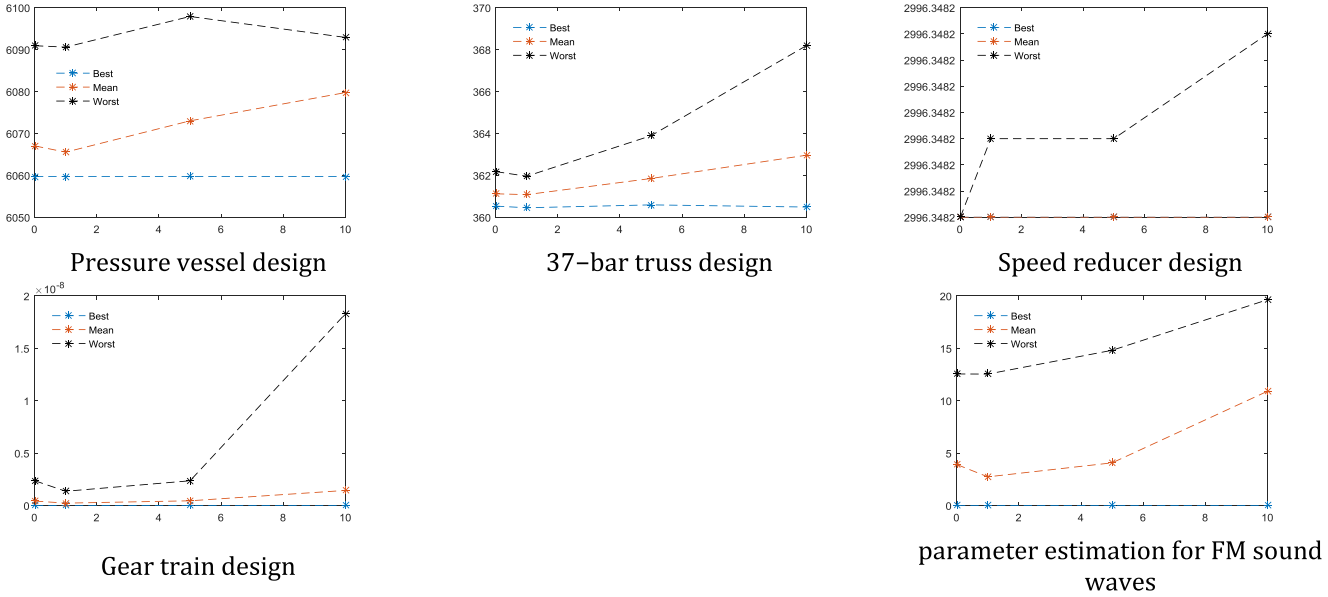


Fig. 24. Performance of the F3EA algorithm under different setting for *A*.

Table 9
Comparative results on unconstrained test functions.

Functions	CMA-ES		ABC				TLBO				COIO				F3EA		
	Mean	Sr	Mean	Stdev	Sr	Sig	Mean	Stdev	Sr	Sig	Mean	Stdev	Sr	Sig	Mean	Stdev	Sr
P1-Sphere	10,721	100	9264	1481	100	–	4648	148	100	–	2621	402	100	–	568	395	100
P2-Schwefel 2.22	12,145	100	12,991	673	100	–	7395	163	100	–	5221	529	100	–	602	380	100
P3-Schwefel 1.2	21,248	100	12,255	1390	100	+	12,218	1305	100	+	9901	1461	100	+	36,908	2641	100
P4-Schwefel 2.21	20,813	100	1E+5 (2.712)	0 (1.18)	0	–	9563	715	100	–	4270	478	100	–	3710	782	100
P5-Rosenbrock	55,821	90	1E+5 (0.936)	0 (1.76)	0	–	1E+5 (16.321)	0 (1.356)	0	–	11,950	5375	100	+	98,540 (1.187)	4556 (1.882)	14
P6-Step	2184	36	4853	1044	100	–	13,778	1491	100	–	1040	192	100	–	676	399	100
P7-Quartic	66,713	0	1E+5 (0.0906)	0 (0.0189)	0	~	1E+5 (0.00625)	0 (0.00386)	0	~	19,202	10,453	100	+	1E+5 (0.00341)	0 (0.00127)	0
P8-Schwefel 2.26	6621	0	64,632	23,897	86	~	1E+5	0	40	–	37,778	6027	100	+	59,045	3755	100
P9-Rastrigin	10,079	0	26,731	9311	100	–	34,317	13,866	100	–	4462	909	100	–	602	419	100
P10-Ackley	10,654	100	16,616	1201	100	–	3868	2634	100	–	4905	551	100	–	983	750	100
P11-Griewank	10,522	92	36,151	17,128	96	–	10,090	16,237	100	+	3635	453	100	+	23,960	15,733	100
P12-Penalized	13,981	88	73,440	2020	100	–	10,815	1430	100	–	2264	609	100	+	7525	9174	100
P13-Penalized 2	13,756	86	8454	1719	100	+	30,985	12,937	100	–	3884	1775	100	+	15,376	12,381	100
P14-Foxholes	540	0	1046	637	100	–	524	150	100	+	1359	1064	100	–	772	361	100
P15-Kowalik	13,434	88	6120	4564	100	–	2488	2700	100	~	4856	3031	100	–	2318	4518	100
P16-Six-Hump	619	100	342	109	100	–	447	175	100	–	974	274	100	–	264	171	100
P17-Branin	594	100	530	284	100	–	362	88	100	–	1203	358	100	–	233	95	100
P18-Goldstein-Price	2052	78	15,186	13,500	100	–	452	244	100	–	647	137	100	–	246	80	100
P19-Hartman 3	996	100	4747	16,011	100	–	547	135	100	–	996	274	100	–	128	57	100
P20-Hartman 6	2293	48	1583	457	100	–	24,847	29,465	96	–	4558	1853	100	–	1202	848	100
P21-Shekel 5	1246	40	6069	13,477	98	–	1245	114	100	~	3317	2572	100	–	1181	975	100
P22-Shekel 7	1267	48	7173	9022	100	–	1272	99	100	–	3062	734	100	–	1093	725	100
P23-Shekel 10	1275	52	15,392	24,413	96	–	1270	135	100	–	3150	1009	100	–	1052	730	100

significantly inferior to F3EA or performs the same as F3EA. Comparing against COIO, it is significantly superior over F3EA on only two functions. F3EA significantly outperforms GA, PSO, DE, ABC, TLBO and COIO on 14, 10, 10, 11, 6 and 7 out of 14 functions, respectively.

On Powell (Q5) function, the performance of F3EA is the best and very close to the global optimum which is equal to 0. Here the performance of DE and TLBO is also acceptable. However, COIO is the best of all. It always stops at the global optimum. On Rosenbrock (P5) function, F3EA, COIO and TLBO perform very well. However, F3EA is significantly better than TLBO and COIO. On Dixon-Price (Q7) function, F3EA and ABC algorithms obtain the true optimum in all runs, but most of the rest of algorithms trap in local optima. The exceptional performance of F3EA is on Langerman 5 (Q11), Langerman 10 (Q12), FletcherPowell 5 (Q13) and FletcherPowell 10 (Q14) functions, for which it always finds the true minimum. Except for DE and COIO, which are only able to truly optimize Langerman 5 (Q11) function, no algorithm is able to reveal a performance as competitive as F3EA. On Trid 6 (Q2) and Trid 10 (Q3) functions, all algorithms except GA perform equally well. GA is the weakest among rivals. There are 3 functions on which the first rank is not devoted to F3EA. These are Powell (Q5), Perm (Q9) and PowerSUM (Q10) functions on which, the output of F3EA is superior over all rivals except TLBO or COIO, which perform negligibly better in average.

4.4. On the effectiveness of the Find and Fix steps

In this section we check that whether different steps of F3EA algorithm have a significant influence on its performance or not. To check the effectiveness of the Find step, we temporarily inactivate the Find step and select $\vec{P}_k^t$, $(\vec{P}_k^t \neq \vec{P}_i^t)$ randomly via conducting a roulette wheel mechanism among all individuals in the population. Results are compared with an algorithm, which employs the Find step. We also check the effectiveness of the find step under different levels of A. To make comparisons, we take 23 benchmark functions used in Experiment 1. To have a true comparison, we inactivate the Fix step of F3EA algorithm to measure the pure effect of Find step on the convergence and performance of the algorithm. Based on 50 independent runs, results are depicted in Table 11. From this table, the Find step has a positive effect on speeding up the convergence of the F3EA algorithm. On 10 out of 23 functions, F3EA-with Find step ($A = 1$), evaluates significantly a smaller number of functions than the F3EA-without Find step. Only on 4 functions, F3EA-with Find step ($A = 1$) evaluates significantly a larger number of functions than F3EA-without Find step. On the remaining functions, no significant difference is detectable between algorithms. However, from Table 11, it is seen that on 14 out of 23 functions, F3EA-with Find step ($A = 1$) evaluates a smaller mean number of functions than F3EA-without Find step; on 6 problems, the situation is reversed; and on 3 functions both algorithms count the same number of *FE*s. Results show that on P3, P4 and P5 functions, F3EA-with Find step ($A = 1$) performs better than F3EA-without Find step, and on P7 both algorithms show a similar mean value. Comparing the performance of F3EA algorithm under different values of A, results indicate that unlike engineering design problems, here $A = 5$ provides smaller mean values than A = 1. Hence, tuning of this parameter may seem required when solving optimization problems. From the above observations, it can be concluded that the Find step of the proposed F3EA algorithm has a positive influence on its performance.

To investigate the effectiveness of the Fix step of F3EA algorithm, we perform comparisons with and without Fix step. The mean and standard deviation of the best function values resulted from 50 replications on each of 23 functions are summarized in Table 12. Results indicate that on 21 out of 23 functions, the mean performance of F3EA algorithm that affords from the Fix step is just better than the F3EA algorithm, which does not employ the Fix step. On 2 out of 23 functions there is no significant difference between the two cases. These

Table 10
Comparative results on harder test functions.

Function	Best value		GA		PSO		DE		ABC		TLBO		COIO		F3EA
			Performance	Sig	Performance	Sig	Performance	Sig	Performance	Sig	Performance	Sig	Performance	Sig	
Q1-Colville	0	Mean	0.014938	−	0	~	0.0409122	−	0.0929674	−	0	~	0	~	0
		Stdev	0.007364		0		0.081979		0.066277		0		0		0
Q2-Trid 6	−50	Mean	−49.9999	−	−50	~	−50	~	−50	~	−50	~	−50	~	−50
		Stdev	2.25E−5		0		0		0		0		0		0
Q3-Trid 10	−210	Mean	−209.476	−	−210	~	−210	~	−210	~	−210	~	−209.8299	−	−210
		Stdev	0.193417		0		0		0		0		0.2553		0
Q4-Zakharov	0	Mean	0.013355	−	0	~	0	~	0.0002476	−	0	~	0	~	0
		Stdev	0.004532		0		0		0.000183		0		0		0
Q5-Powell	0	Mean	9.703771	−	0.00011004	−	2.17E−7	−	0.0031344	−	5.86E−8	−	0	+	7.15E−12
		Stdev	1.547983		0.000160		1.36E−7		0.000503		8.13E−8		0		1.96E−11
Q6-Rosenbrock	0	Mean	1.96E+5	−	15.088617	−	18.203938	−	0.0887707	−	1.62E−5	−	1.30E−5	−	6.13E−7
		Stdev	3.85E+4		24.170196		5.036187		0.077390		3.64E−5		2.93E−5		1.01E−6
Q7-Dixon-Price	0	Mean	1.22E+3	−	0.66666667	−	0.6666667	−	0	~	0.6666667	~	0.00980867	−	0
		Stdev	2.66E+2		1E−8		1E−9		0		0		0.00495085		0
Q8-Kowalik	0.0003075	Mean	0.005615	−	0.00049062	−	0.0004266	−	0.0004266	−	0.0003076	~	0.000307486	−	0.00030748
		Stdev	0.008171		0.000366		0.000273		6.04E−5		0		3.0418E−09		0
Q9-Perm	0	Mean	0.302671	−	0.03605158	−	0.0240069	−	0.0411052	−	6.77E−4	+	0.0096613	~	0.0028613
		Stdev	0.193254		0.048927		0.046032		0.023056		7.45E−4		0.0290010		0.002458
Q10-PowerSum	0	Mean	0.010405	−	11.3904479	−	0.0001425	−	0.0029468	−	7.43E−5	~	2.8563E−06	+	7.27E−5
		Stdev	0.009077		7.355800		0.000145		0.002289		1.11E−4		1.4924E−05		9.7E−5
Q11-Langerman 5	−1.5	Mean	−0.96842	−	−0.5048579	−	−1.499999	~	−0.938150	−	−0.939702	−	−1.499999	−	−1.49999922
		Stdev	0.287548		0.213626		0		0.000208		1.55E−5		4.7624E−08		0
Q12-Langerman 10	NA	Mean	−0.63644	−	−0.0025656	−	−1.0528	−	−0.4460925	−	−0.64906	−	−0.8456012	−	−1.5
		Stdev	0.374682		0.003523		0.302257		0.133958		1.73E−1		0.1636252		0
Q13-FletcherPowell 5	0	Mean	0.004303	−	1457.88344	−	5.988783	−	0.1735495	−	2.2038134	−	0	~	0
		Stdev	0.009469		1269.3623		7.334731		0.068175		4.39		0		0
Q14-FletcherPowell10	0	Mean	29.57348	−	1364.45555	−	781.55028	−	8.2334401	−	35.971004	−	3.6795318	−	1.08E−12
		Stdev	16.021078		1325.3796		1048.81348		8.092742		7.13E+1		4.4047507		5.19E−12

Table 11
Comparative results on the effectiveness of the Find step.

Function	F3EA with the find step												F3EA without the find step			
	A = 0.01			A = 1			A = 5			A = 10						
	Mean	Stdev	Sr	Mean	Stdev	Sr	Mean	Stdev	Sr	Mean	Stdev	Sr	Mean	Stdev	Sr	Sig*
P1	30,847	614	100	30,174	511	100	28,463	466	100	32,204	550	100	34,489	471	100	−
P2	29,886	729	100	29,716	708	100	29,081	900	100	30,288	902	100	33,060	946	100	−
P3	1E+05 (142.5)	0 (66.37)	0	1E+05 (126.5)	0 (63.4)	0	1E+05 (91.0)	0 (40.4)	0	1E+05 (10.0)	0 (9.3)	0	1E+05 (135.9)	0 (66.8)	0	~
P4	1E+05 (3.20)	0 (0.75)	0	1E+05 (3.05)	0 (0.71)	0	1E+05 (2.42)	0 (0.6)	0	1E+05 (2.83)	0 (0.63)	0	1E+05 (3.32)	0 (0.62)	0	~
P5	1E+05 (1.32)	0 (1.82)	0	1E+05 (1.11)	0 (1.36)	0	1E+05 (0.54)	0 (0.83)	0	1E+05 (9.97)	0 (22.63)	0	1E+05 (1.51)	0 (1.81)	0	~
P6	22,011	1013	100	21,348	1064	100	19,757	995	100	23,673	1143	100	25,162	961	100	−
P7	99,721 (0.002)	1973 (0.0009)	2	99,952 (0.002)	340 (0.0009)	2	1E+05 (0.003)	0 (0.001)	0	1E+05 (0.004)	0 (0.001)	0	99,489 (0.002)	1503 (0.0008)	12	+
P8	66,203	2390	100	63,543	2112	100	60,233	2319	100	64,509	2753	100	80,067	3839	100	−
P9	35,975	1125	100	35,051	1214	100	32,563	837	100	38,395	1247	100	42,043	1143	100	−
P10	37,529	601	100	36,664	747	100	34,147	690	100	37,254	1026	100	42,311	843	100	−
P11	41,124	1241	100	40,106	1213	100	36,369	1610	100	42,821	1232	100	51,631	2462	100	−
P12	26,428	934	100	25,679	632	100	24,632	800	100	26,818	916	100	30,478	971	100	−
P13	28,883	673	100	28,331	651	100	27,628	734	100	29,792	794	100	32,097	723	100	−
P14	794	382	100	874	359	100	834	349	100	778	287	100	985	323	100	~
P15	6421	7392	100	8120	7985	100	7263	7828	100	7131	7126	100	4461	2282	100	+
P16	757	232	100	736	249	100	742	294	100	500	128	100	796	180	100	~
P17	1120	393	100	1084	659	100	1350	1021	100	1454	2102	100	1096	380	100	~
P18	1767	2066	100	1489	1599	100	1044	961	100	923	1429	100	1023	641	100	+
P19	663	189	100	636	176	100	709	339	100	539	188	100	770	144	100	−
P20	3046	1199	100	3093	1845	100	3054	542	100	2983	568	100	3271	396	100	~
P21	3881	3070	100	3555	2088	100	3680	2077	100	3514	1866	100	3255	1111	100	~
P22	6199	13,844	100	4903	3592	100	5218	3332	100	4799	3717	100	3819	1113	100	+
P23	4087	2841	100	4094	2460	100	5196	3066	100	3934	2680	100	3836	1012	100	~

* The significant test has been conducted based on A = 1.

Table 12
Comparative results on the effectiveness of the Fix step.

Function	F3EA-with the Fix step			F3EA-without the Fix step			
	Mean	Stdev	Sr	Mean	Stdev	Sr	Sig
P1	568	395	100	30,174	511	100	−
P2	602	380	100	29,716	708	100	−
P3	36,908	2641	100	1E+05	0	0	−
P4	3710	782	100	1E+05	0	0	−
P5	98,540	4556	14	1E+05	0	0	−
P6	676	399	100	21,348	1064	100	−
P7	1E+5	0	0	99,952	340	2	~
P8	59,045	3755	100	63,543	2112	100	−
P9	602	419	100	35,051	1214	100	−
P10	983	750	100	36,664	747	100	−
P11	23,960	15,733	100	40,106	1213	100	−
P12	7525	9174	100	25,679	632	100	−
P13	15,376	12,381	100	28,331	651	100	−
P14	772	361	100	874	359	100	~
P15	2318	4518	100	8120	7985	100	−
P16	264	171	100	736	249	100	−
P17	233	95	100	1084	659	100	−
P18	246	80	100	1605	1759	100	−
P19	128	57	100	636	176	100	−
P20	1202	848	100	3093	1845	100	−
P21	1181	975	100	3555	2088	100	−
P22	1093	725	100	4903	3592	100	−
P23	1052	730	100	4094	2460	100	−

observations sound that the Fix step of F3EA algorithm has a positive influence on its performance.

5. Conclusion

As a new evolutionary algorithm, the F3EA meta-heuristic algorithm was introduced in this paper. It mimics the Find-Fix-Finish-Exploit-Analyze phases of the targeting process. Mathematical modelling of different steps of the algorithm have been established cleverly using physical theories governing the performance of battlefield facilities. To find out the merit of the proposed F3EA algorithm, extensive analysis have been carried out based on real-world engineering

optimization problems and 37 benchmark functions. Sensitivity analysis of the control parameters has been investigated. Moreover, the effectiveness of different steps of F3EA algorithm has been examined experimentally via temporarily inactivating each step and performing comparisons. Our comparisons have indicated that the F3EA algorithm is generally able to dominate the best of competitors and show a very fast convergence toward optimum. The proposed F3EA algorithm is in its embryonic stages, and it is the authors' hope that it stimulates forthcoming researches on evolving the theory and applications of F3EA algorithm.

Acknowledgement

This work is partly supported by the Grant-in-Aid for Scientific Research (C) of Japan Society of Promotion of Science (JSPS), No. 15K00357.

Appendix A. A numerical example

The step-wise procedure for generating a new solution by F3EA algorithm is given as follows. For demonstration of the procedure, the Rastrigin multimodal function is considered in 3 dimensions. We assume that $N_B = 4$ and $q_i^t = 0.3$

$$\Pi^t = \begin{bmatrix} \vec{P}_1^t \\ \vec{P}_2^t \\ \vec{P}_3^t \\ \vec{P}_4^t \end{bmatrix} = \begin{bmatrix} p_{11}^t & p_{12}^t & \cdots & p_{1n}^t \\ p_{21}^t & p_{22}^t & \cdots & p_{2n}^t \\ \vdots & \vdots & \vdots & \vdots \\ p_{N_B1}^t & p_{N_B2}^t & \cdots & p_{N_Bn}^t \end{bmatrix} = \begin{bmatrix} 1.5574 & 1.7873 & 1.5547 \\ -4.6428 & 2.5774 & -3.2881 \\ 1.9864 & 3.5567 & -2.2167 \\ 4.3399 & -1.0777 & -4.6816 \end{bmatrix}$$

$$\text{The corresponding function values} = \begin{bmatrix} f(\vec{P}_1^t) \\ f(\vec{P}_2^t) \\ f(\vec{P}_3^t) \\ f(\vec{P}_4^t) \end{bmatrix} = \begin{bmatrix} 54.4866 \\ 86.4596 \\ 48.8408 \\ 72.6013 \end{bmatrix} \Rightarrow \vec{G}_{best}^t = [1.9864 \quad 3.5567 \quad -2.2167]$$

We demonstrate the mechanism for $i = 1$.

- ***Find step***
  - $f_{max} = 86.4596$
  - $f_{\min} = 48.8408$
  - $R_{pq}^t = |f(\vec{P}_p^t) - f(\vec{P}_q^t)| = \begin{bmatrix} 0 & 31.9729 & 5.6454 & 18.1147 \\ 31.9729 & 0 & 37.6184 & 13.8582 \\ 5.6454 & 37.6184 & 0 & 23.7601 \\ 18.1147 & 13.8582 & 23.7601 & 0 \end{bmatrix}$
  - $R_{\max\,12}^t = 0$
  - $R_{\max\,13}^t = (54.4866 - 48.8408) \times \sqrt[4]{\left(\frac{86.4596 - 48.8408}{86.4596 - 48.8408}\right) e^{-\left(\frac{48.8408 - 48.8408}{54.4866 - 48.8408}\right)}} = 5.6454$
  - $R_{\max\,14}^t = (54.4866 - 48.8408) \times \sqrt[4]{\left(\frac{86.4596 - 72.6013)}{86.4596 - 48.8408}\right) e^{-\left(\frac{72.6013 - 48.8408}{54.4866 - 48.8408}\right)}} = 1.5357$
  - $R_{12}^t > R_{\max\,12}^t$
  - $R_{13}^t \leqslant R_{\max\,13}^t \Rightarrow k = 3$ and individual $\vec{P}_3^t$ is selected.
  - $R_{14}^t > R_{\max\,14}^t$
- ***Finish step***
  - $u(\vec{P}_3^t) = \left(\frac{\vec{P}_3^t - f_{\min}^t}{f_{\max}^t - f_{\min}^t}\right) = \left(\frac{48.8408 - 48.8408}{86.4596 - 48.8408}\right) = 0$
  - $u(\vec{P}_1^t) = \left(\frac{\vec{P}_1^t - f_{\min}^t}{f_{\max}^t - f_{\min}^t}\right) = \left(\frac{54.4866 - 48.8408}{86.4596 - 48.8408}\right) = 0.15$
  - $\beta_{13}^t = \arctan \frac{u(\vec{P}_3^t) - u(\vec{P}_1^t)}{\| \vec{P}_3^t - \vec{P}_1^t \|} = \arctan \frac{0 - 0.15}{\sqrt{(1.9864 - 1.5574)^2 + (3.5567 - 1.7873)^2 + (-2.2167 - 1.5547)^2}} = -0.0358$
  - $\overset{u(\vec{P}_3^t) < u(\vec{P}_1^t)}{\rightarrow} \alpha_{13}^t = \text{rand}(0, \pi/2) = 0.6507$
  - $AB_{13}^t = \|[\vec{P}_3^t, u(\vec{P}_3^t)]_{1\times4} - [\vec{P}_1^t, u(\vec{P}_1^t)]_{1\times4}\| =$
  - $\sqrt{(1.9864 - 1.5574)^2 + (3.5567 - 1.7873)^2 + (-2.2167 - 1.5547)^2 + (0 - 0.15)^2} = 4.1905$
  - $v_{013}^t = \sqrt{\frac{AB_{13}^t \times g \times \cos^2 |\beta_{13}^t|}{2\cos\alpha_{13}^t \times \sin(\alpha_{13}^t + |\beta_{13}^t|)}} = \sqrt{\frac{4.1905 \times 9.81 \times \cos^2 0.0358}{2\cos 0.6507 \times \sin(0.6507 + 0.0358)}} = 6.3798$
  - $T_{13}^t = \frac{2v_{013}^t \sin(\alpha_{13}^t + |\beta_{13}^t|)}{g\cos|\beta_{13}^t|} = \frac{2 \times 6.3798 \times \sin(0.6507 + 0.0358)}{9.81 \times \cos 0.0358} = 0.8250$
  - $m_{13}^t = \max\left(m_{\min}, \min\left(\frac{u(\vec{P}_1^t)}{u(\vec{P}_3^t)}, \ m_{\max}\right)\right) = \max\left(0.01, \min\left(\frac{0.15}{0}, 100\right)\right) = 100$
  - $w_{x_{13}}^t = m_{13}^t \times v_{013}^t \times \text{rand}(0, 1) = 100 \times 6.3798 \times 0.7728 = 493.0937$
  - $x(T_{13}^t) = w_{x_{13}}^t T_{13}^t + m_{13}^t (v_{013}^t \cos\alpha_{13}^t - w_{x_{13}}^t)(1 - e^{-T_{13}^t/m_{13}^t}) =$
    $493.0937 \times 0.8250 + 100 \times (6.3798 \times \cos 0.6507 - 493.0937) \times (1 - e^{-0.8250/100}) = 5.8442$
  - $\overset{\text{rand}(0,1) = 0.8229 \geqslant 0.5}{\rightarrow} w_{z_{13}}^t = -m_{\min} \times v_{013}^t \times \text{rand}(0, 1) = -0.01 \times 6.3798 \times 0.0420 = -0.0026$
  - $z(T_{13}^t) = w_{z_{13}}^t T_{13}^t - m_{13}^t w_{z_{13}}^t (1 - e^{-T_{13}^t/m_{13}^t}) = -0.0026 \times 0.8250 + 100 \times 0.0026 \times (1 - e^{-0.8250/100}) = -9.1120E - 6$

– $\overset{q=1}{\rightarrow} 1 - \sum_{d=1}^{3} \frac{(p_{3d}^t - p_{1d}^t)}{(p_{3q}^t - p_{1q}^t)} = \frac{(1.9864 - 1.5574)}{(1.9864 - 1.5574)} + \frac{(3.5567 - 1.7873)}{(1.9864 - 1.5574)} + \frac{(-2.2167 - 1.5547)}{(1.9864 - 1.5574)} = 4.6667$

– $\Rightarrow \vec{Z}_{13\perp}^t = [4.6667 \quad 1 \quad 1]$

– $\vec{E}_1^t = \vec{P}_1^t + x(T_{13}^t)\frac{(\vec{P}_3^t - \vec{P}_1^t)}{\|\vec{P}_3^t - \vec{P}_1^t\|} + z(T_{13}^t)\frac{(\vec{Z}_{13\perp}^t)}{\|\vec{Z}_{13\perp}^t\|} = [1.5574 \quad 1.7873 \quad 1.5547] -$
$5.8442 \times \frac{[1.9864 \quad 3.5567 \quad -2.2167] - [1.5574 \quad 1.7873 \quad 1.5547]}{\sqrt{(1.9864 - 1.5574)^2 + (3.5567 - 1.7873)^2 + (-2.2167 - 1.5547)^2}} - 9.1120E - 6 \times \frac{[4.6667 \quad 1 \quad 1]}{\sqrt{4.6667^2 + 1^2 + 1^2}}$

– $\Rightarrow \vec{E}_1^t = [2.1560 \quad 4.2565 \quad -3.7083]$

- ***Exploit step***

– $\vec{U}_1^t = [1.5574 \quad 1.7873 \quad 1.5547]$

– $c_1^t = \left\lceil \frac{\ln(1 - (1 - (1 - q_1^t)^n) \text{ rand}(0,1))}{\ln(1 - q_1^t)} \right\rceil = \left\lceil \frac{\ln(1 - (1 - (1 - 0.3)^n) \ 0.7078)}{\ln(1 - 0.3)} \right\rceil = 2$

– $c_1^t = 2$ indicates that two dimensions of $\vec{U}_1^t$ should be changed. Let us assume that the first and third dimensions of $\vec{E}_1^t$ being selected and their value being assigned to their relevant dimensions in $\vec{U}_1^t$. Therefore, $\vec{U}_1^t = [2.1560 \quad 1.7873 \quad -3.7083]$ and $f(\vec{U}_1^t) = 46.2934$.

- ***Analyze step***

– $f(\vec{U}_1^t) < f(\vec{P}_1^t)$ therefore, $\vec{U}_1^t$ enters into population and replaces $\vec{P}_1^t$.

$$\Pi^t = \begin{bmatrix} 2.1560 & 1.7873 & -3.7083 \\ -4.6428 & 2.5774 & -3.2881 \\ 1.9864 & 3.5567 & -2.2167 \\ 4.3399 & -1.0777 & -4.6816 \end{bmatrix}$$

– $f(\vec{U}_1^t) < \vec{G}_{best}^t \Rightarrow \vec{G}_{best}^t = [2.1560 \quad 1.7873 \quad -3.7083]$

- ***Fix step***

– $\vec{G}_0^t = \vec{G}_{best}^t = [2.1560 \quad 1.7873 \quad -3.7083]$

– $\gamma_1^t = 1$

– $\vec{G}_1^t = \vec{G}_0^t + s_1^t \vec{e}_1 = [2.1560 \quad 1.7873 \quad -3.7083] + s_1^t \times [1 \quad 0 \quad 0] = [2.1560 + s_1^t \quad 1.7873 \quad -3.7083]$

– Solving the following single variable problem we obtain $s_1^t = -2.1560$.

$$\min f([2.1560 + s_1^t \quad 1.7873 \quad -3.7083]) = (2.1560 + s_1^t)^2 - 10\cos(2\pi(2.1560 + s_1^t))$$
$$\text{s. t.}$$
$$(-5.12 - 2.1560) \leqslant s_1^t \leqslant (5.12 - 2.1560)$$

– $\vec{G}_1^t = [0 \quad 1.7873 \quad -3.7083] \Rightarrow f(\vec{G}_1^t) = 37.2139 < f(\vec{G}_{best}^t) \Rightarrow \vec{G}_{best}^t = [0 \quad 1.7873 \quad -3.7083]$

– $\vec{G}_2^t = \vec{G}_1^t + s_2^t \vec{e}_2 = [0 \quad 1.7873 \quad -3.7083] + s_2^t \times [0 \quad 1 \quad 0] = [0 \quad 1.7873 + s_2^t \quad -3.7083]$

– Solving the following single variable problem we obtain $s_2^t = -1.7873$.

$$\min f([0 \quad 1.7873 + s_2^t \quad -3.7083]) = (1.7873 + s_2^t)^2 - 10\cos(2\pi(1.7873 + s_2^t))$$
$$\text{s. t.}$$
$$(-5.12 - 1.7873) \leqslant s_2^t \leqslant (5.12 - 1.7873)$$

– $\vec{G}_2^t = [0 \quad 0 \quad -3.7083] \Rightarrow f(\vec{G}_2^t) = 26.3417 < f(\vec{G}_{best}^t) \Rightarrow \vec{G}_{best}^t = [0 \quad 0 \quad -3.7083]$

– $\vec{G}_3^t = \vec{G}_2^t + s_3^t \vec{e}_3 = [0 \quad 0 \quad -3.7083] + s_3^t \times [0 \quad 0 \quad 1] = [0 \quad 0 \quad -3.7083 + s_3^t]$

– Solving the following single variable problem we obtain $s_3^t = 3.7083$.

$$\min f([0 \quad 0 \quad -3.7083 + s_3^t]) = (-3.7083 + s_3^t)^2 - 10\cos(2\pi(-3.7083 + s_3^t))$$
$$\text{s. t.}$$
$$(-5.12 + 3.7083) \leqslant s_3^t \leqslant (5.12 + 3.7083)$$

– $\vec{G}_2^t = [0 \quad 0 \quad 0] \Rightarrow f(\vec{G}_2^t) = 0 < f(\vec{G}_{best}^t) \Rightarrow \vec{G}_{best}^t = [0 \quad 0 \quad 0]$

The algorithm achieves the optimal solution and hence stops.

References

Aguirre, Hernandez, Munoz Zavala, A., Villa Diharce, E., Botello Rionda, S. (2007). COPSO: Constrained optimization via PSO algorithm. Center for Research in Mathematics (CIMAT). Technical report No. I-07-04/22-02-2007.

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

Askarzadeh (2016). A novel meta-heuristic method for solving constrained engineering optimization problems: Crow search algorithm. *Computers and Structures, 169*, 1–12.

Atashpaz-Gargari, E., & Lucas, C. (2007). Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition. *Proceedings of IEEE congress on evolutionary computation* (pp. 4661–4666). .

Bansal, R. K. (1998). *Engineering mechanics and strength of materials*. Luxmi Publications.

Baykasoglu, & Akpinar, S. (2015). Weighted Superposition Attraction (WSA): A swarm intelligence algorithm for optimization problems – Part 2: Constrained optimization. *Applied Soft Computing*. https://doi.org/10.1016/j.asoc.2015.08.052.

Cagnina, L. C., Esquivel, S. C., & Coello, C. A. C. (2008). Solving engineering optimization problems with the simple constrained particle swarm optimizer. *Informatica, 32*, 319–326.

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

Fathollahi Fard, M., Hajiaghaei-Keshteli, M., & Tavakkoli-Moghaddam, R. (2018). The social engineering optimizer (SEO). *Engineering Applications of Artificial Intelligence* (in press).

Ferry, M. (2013). F3EA-a targeting paradigm for contemporary warfare. *Australian Army Journal, 1*, 49–64.
Gen, M., & Cheng, R. (2000). *Genetic algorithms and engineering optimization.* New York: Wiley.
Gomes, M. H. (2011). Truss optimization with dynamic constraints using a particle swarm algorithm. *Expert System with Applications, 38*, 957–968.
Goncalves, M. S., Lopez, R. H., & Miguel, L. F. F. (2015). Search group algorithm: A new meta-heuristic method for the optimization of truss structures. *Computers and Structures, 153*, 165–184.
He, S., Prempain, E., & Wu, Q. H. (2004). An Improved particle swarm optimizer for mechanical design optimization problems. *Engineering Optimization, 36*, 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*, 89–99.
He, S., Wu, Q. H., & Saunders, J. R. (2009). Group search optimizer: An optimization algorithm inspired by animal searching behavior. *IEEE Transactions on Evolutionary Computation, 13*, 973–990.
Huang, F., Wang, L., & He, Q. (2007). An effective co-evolutionary differential evolution for constrained optimization. *Applied Mathematics and Computation, 186*, 340–356.
Husseinzadeh Kashan, A. (2011). An efficient algorithm for constrained global optimization and application to mechanical engineering design: League championship algorithm (LCA). *Computer Aided Design, 43*, 1769–1792.
Husseinzadeh Kashan, A. (2014). League Championship Algorithm (LCA): An algorithm for global optimization inspired by sport championships. *Applied Soft Computing, 16*, 171–200.
Husseinzadeh Kashan, A. (2015a). An effective algorithm for constrained optimization based on optics inspired optimization (OIO). *Computer-Aided Design, 63*, 52–71.
Husseinzadeh Kashan, A. (2015b). A new metaheuristic for optimization: Optics inspired optimization (OIO). *Computers & Operations Research, 55*, 99–125.
Husseinzadeh Kashan, A., & Karimi, B. (2010). A new algorithm for constrained optimization inspired by the sport league championships. *IEEE world congress on computational intelligence (WCCI2010)* (pp. 487–494). .
Jalili, S., Husseinzadeh Kashan, A., & Hosseinzadeh, Y. (2017). League championship algorithms for optimum design of pin-jointed structures. *Journal of Computing in Civil Engineering, 31*, 1–17.
Karaboga, D., & Akay, B. (2009). A comparative study of Artificial Bee Colony algorithm. *Applied Mathematics and Computation, 214*, 08–132.
Kaveh, Jafari, L., & Farhoudi, N. (2015). Truss optimization with natural frequency constraints using a dolphin echolocation algorithm. *Asian Journal of Civil Engineering, 16*, 29–46.
Kaveh, & Khayatazad, M. (2013). Ray optimization for size and shape optimization of truss structures. *Computers and Structures, 117*, 82–94.
Kaveh, & Zolghadr, A. (2011). Shape and size optimization of truss structures with frequency constraints using enhanced charged system search algorithm. *Asian Journal of Civil Engineering, 12*, 487–509.
Kaveh, & Zolghadr, A. (2014). Democratic PSO for truss layout and size optimization with frequency constraints. *Computers and Structures, 130*, 10–21.
Khatibinia, M., & Naseralavi, S. (2014). Truss optimization on shape and sizing with frequency constraints based on orthogonal multi-gravitational search algorithm. *Journal of Sound and Vibration, 333*, 6349–6369.
Kim, H. K., Chong, J. K., Park, K. Y., & Lowther, D. A. (2007). Differential evolution strategy for constrained global optimization and application to practical engineering problems. *IEEE Transactions on Magnetics, 43*, 1565–1568.
Li, Yang, S., & Nguyen, T. T. (2012). A self-learning particle swarm optimizer for global optimization problems. *IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics, 42*, 627–646.
Lin, L., & Gen, M. (2008). Auto-tuning strategy for evolutionary algorithms: Balancing between exploration and exploitation. *Soft Computing, 13*, 157–168.
Lingyun, W., Mei, Z., Guangming, W., & Guang, M. (2005). Truss optimization on shape and sizing with frequency constraints based on genetic algorithm. *Computational Mechanics, 25*, 361–368.
Mezura-Montes, E., Coello, C. A. C., & Landa-Becerra, R. (2003). Engineering optimization using a simple evolutionary algorithm. *Proceedings of the fifteenth international conference on tools with artificial intelligence (ICTAI'2003)* (pp. 149–156). .
Mezura-Montes, E., & Hernández-Ocaña, B. (2009). Modified bacterial foraging optimization for engineering design. In C. H. Dagli (Ed.). *Proceedings of the artificial neural networks in engineering conference, ASME Press Series, Intelligent engineering systems through artificial neural networks* (pp. 357–364). .
Mohamed, W. (2015). An improved differential evolution algorithm with triangular mutation for global numerical optimization. *Computers & Industrial Engineering, 85*, 359–375.
Norton, K. A., & Omberg, A. C. (1947). The maximum range of a radar set. *Proceedings of the IRE* (pp. 927–931). .
Nourbakhsh, Safikhani, H., & Derakhshan, S. (2011). The comparison of multi-objective particle swarm optimization and NSGA II algorithm: Applications in centrifugal pumps. *Engineering Optimization, 43*, 1095–1113.
Parouha, R. P., & Das, K. N. (2015). An efficient hybrid technique for numerical optimization and applications. *Computers & Industrial Engineering, 85*, 359–375 83, 193–216.
Parsopoulos, K., & Vrahatis, M. (2005). Unified particle swarm optimization for solving constrained engineering optimization problems. *ICNC 2005: Advances in natural computation volume 3612/2005 of lecture notes in computer science* (pp. 582–591). Berlin/Heidelberg: Springer.
Rao, R., & Patel, V. (2013). Comparative performance of an elitist teaching-learning-based optimization algorithm for solving unconstrained optimization problems. *International Journal of Industrial Engineering Computations, 4*, 29–50.
Takahama, T., Sakai, S. (2006). Solving constrained optimization problems by the ε constrained particle swarm optimizer with adaptive velocity limit control. In Proceedings of the 2nd IEEE international conference on cybernetics & intelligent systems (CIS2006) (pp. 683–689).
Wang, J., & Yin, Z. (2008). A ranking selection-based particle swarm optimizer for engineering design optimization problems. *Structural and Multidiciplinary Optimization, 37*, 131–147.
Wang, D., Zhang, W. H., & Jiang, J. S. (2004). Truss optimization on shape and sizing with frequency constraints. *AIAA Journal, 42*, 1452–1456.
Worasucheep (2008). Solving constrained engineering optimization problems by the constrained PSO-DD. *Proceedings of ECTI-CON, 2008*, 5–8.
Xing, & Gao, W. G. (2013). *Innovative computational intelligence: A rough guide to 134 clever algorithms.* Springer International Publishing Switzerland.
Yildiz, R. (2008). A novel particle swarm optimization approach for product design and manufacturing. *International Journal of Advance Manufacturing Technology, 40*, 617–628.
Zhang, M., Luo, W., & Wang, X. (2008). Differential evolution with dynamic stochastic selection for constrained optimization. *Information Sciences, 178*, 3043–3074.