Expert Systems With Applications 166 (2021) 114107

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Red fox optimization algorithm

Dawid Połap, Marcin Woźniak [*]

Faculty of Applied Mathematics, Silesian University of Technology, Kaszubska 23, 44-100 Gliwice, Poland

ARTICLE INFO

MSC:
80M50
90C59

Keywords:
Optimization
Meta-heuristic

ABSTRACT

Fox is very popular in various regions of the Globe, where representatives of this kind can be found in Europe, Asia, North America, and even in some arctic regions. The way this predator lives and hunts is very peculiar. It is active all year round, traversing the lands in hunting both for different domestic and wild animals. In his strategy fox is using various tricks to distract prey while creeping what makes him a very efficient predator. The territorial habits and family relations between young and adult made the fox easily adaptable to various conditions and therefore helped him to survive in a changing environment.

In this article we propose a mathematical model of red fox habits, searching for food, hunting, and developing population while escaping from hunters. Described model is based on local and global optimization method with a reproduction mechanism. The novel model developed for optimization purposes we name the Red Fox Optimization Algorithm (RFO). The proposed method was subjected to benchmark tests using 22 test functions and 7 classic engineering optimization problems. Experimental results are compared to other meta-heuristic algorithms to show potential advantages.

1. Introduction

Meta-heuristic approaches turned out to be efficient in various optimization purposes due to high precision, speed of optimization, and low computational complexity. These algorithms proved value in complex problem solving, where solution space is not always well described and the problem demands solving complex mathematical models of life phenomena. Gao et al. (2016) presented the application of these methods in dynamic routing problem, Davari et al. (2016) discussed the application of heuristics in the budget planning of medical preventive health with some constraints and Chamaani et al. (2011) proposed optimization of time and frequency domain for antennas by the use of heuristic methods. Medical applications of these algorithms show efficiency in sequential data processing, which is very useful for various gene modeling operations (Biswas & Acharyya, 2016). Moreover, protein structure modeling and assignments can be solved by the application of devoted derivatives of heuristics as discussed in Brown et al. (2016). Similarly, medical expertise can be based on results of heuristic methods. Holm et al. (2016) presented catheter positioning implemented in prostate brachytherapy, while Ogiela and Krzyworzeka (2016) discussed the application of the heuristic approach to detection of threats over mammograms. A devoted version of the heuristic approach gave excellent results in medical data analysis for classification (Mohapatra et al., 2015) and radiotherapy planning (Kalantzis et al., 2016). Recently heuristic methods are reported to support text mining in real-time applications (Mosa, 2019) and spectral clustering (Janani & Vijayarani, 2019). Medical applications report these methods in protein-encoding (Gonzalez-Sanchez et al., 2019). Also in engineering meta-heuristics were applied in various tasks. Zhang et al. (2019) proposed robot localization using particle filtering, while Mirghasemi et al. (2019) applied heuristic methods to reduce noise in images. An extensive study of these methods concerning latest achievements and a variety of possible applications and theoretical backgrounds was presented in Das et al. (2016). The number of possibilities for potential applications is huge since heuristic methods give good accuracy of results with ease of implementation.

1.1. Related heuristic methods

Recent years showed that nature is an excellent developer of devoted behaviors that optimize actions in situations that cause various problems to performers. We can see astonishing ways animals hunt, communicate, breed, or travel in search for food. Similarly, interesting phenomena from the world of plants and elements of nature inspire the development of science. These behaviors were recently modeled in various optimization algorithms, which summary is presented in Table 1.

Among the state of the art of meta-heuristic algorithms, we have many well known classic models, like Simulated Annealing (SA) presented in Van Laarhoven and Aarts (1987), where the concept of

[*] Corresponding author.
E-mail addresses: Dawid.Polap@polsl.pl (D. Połap), Marcin.Wozniak@polsl.pl (M. Woźniak).

https://doi.org/10.1016/j.eswa.2020.114107
Received 11 November 2019; Received in revised form 9 September 2020; Accepted 7 October 2020
Available online 9 October 2020

Table 1
A summary of Nature-Inspired Algorithms, for which behavior of animals or plants was an inspiration for the optimization technique.

Author(s)	Method	Inspiration from nature	Year
Kennedy and Eberhart (1995)	Particle Swarm Optimization	Flock of fish, birds, etc.	1995
Dorigo and Di Caro (1999)	Ant Colony Optimization	Ants in a colony	1999
Abbass (2001)	Marriage in Honey Bees Optimization	Honey bees	2001
Li (2003)	Artificial Fish Swarm Algorithm	Fish swarm	2003
Martin and Stephen (2006)	Termite Algorithm	Termites in a colony	2006
Karaboga and Basturk (2007)	Artificial Bee Colony	Bees in a colony	2007
Pinto et al. (2007)	Wasp Swarm Algorithm	Wasps	2007
Mucherino and Seref (2007)	Monkey Search	Monkeys	2007
Lu and Zhou (2008)	Bee Collecting Pollen Algorithm	Bees	2008
Yang and Deb (2009)	Cuckoo Search Algorithm	Cuckoos	2009
Shiqin et al. (2009)	Dolphin Partner Optimization	Dolphins	2009
Yang (2010b)	Bat-Inspired Algorithm	Bats	2010
Yang (2010a)	Firefly Algorithm	Fireflies	2010
Oftadeh et al. (2010)	Hunting Search	Animals hunting	2010
Gandomi and Alavi (2012)	Krill Herd	Krill	2012
Pan (2012)	Fruit Fly Optimization Algorithm	Fruit flies	2012
Yang (2012)	Flower Pollination Algorithm	Flower pollination	2012
Askarzadeh and Rezazadeh (2013)	Bird Mating Optimizer	Birds	2013
Kaveh and Farhoudi (2013)	Dolphin Echolocation	Dolphins	2013
Mirjalili et al. (2014)	Gray Wolf Optimizer	Pack of wolves	2014
Mirjalili (2015b)	Moth-Flame Optimization Algorithm	Moths	2015
Mirjalili (2015a)	Ant Lion Optimizer	Ant lions	2015
Yazdani and Jolai (2016)	Lion Optimization Algorithm	Lions	2016
Mirjalili (2016)	Dragon-Fly Algorithm	Dragon flies	2016
Mirjalili and Lewis (2016)	Whale Optimization Algorithm	Whales	2016
Sharma et al. (2016)	Ageist Spider Monkey Optimization Algorithm	Monkeys	2016
Mirjalili et al. (2017)	Salp Swarm Algorithm	Salp	2017
Jain et al. (2018)	Squirrel Search Algorithm	Squirrels	2018
Heidari et al. (2019)	Harris Hawks Optimization	Hawks	2019
Li et al. (2020)	Slime Mould Algorithm	Mould	2020

burning iron and other metals was used to develop one of the first optimization strategies. In Holland (1992) was proposed Genetic Algorithm (GA), in which the evolution of genes was modeled into the optimization method. Differential Evolution (DE) proposed in Storn and Price (1997) used a concept of evolving differences in a process of adaptation to the given conditions. The general idea of nature-inspired swarm intelligence is to develop a model of communication and life interactions between individuals of selected animal species as an optimization strategy. For the state of the art here we can encounter Particle Swarm Optimization (PSO) presented in Kennedy and Eberhart (1995), which was one of the first among these algorithms. Ant Colony Optimization Algorithm (ACO) proposed in Dorigo and Di Caro (1999) was based on a model of ants searching for food and communicating between the swarm members where the best locations are. Artificial Bee Colony Algorithm (ABCA) presented in Karaboga and Basturk (2007) proposed a model of bees searching for the nectar flowers to produce honey for their colony. Cuckoo Search Algorithm (CSA) was proposed in Yang and Deb (2009), where the devoted stochastic theory was applied to simulate the way cuckoos search for nests of other birds to toss their eggs. The echolocation abilities of bats were modeled in the Bat Algorithm (BA) presented in Yang (2010b), where a devoted model of motion by using echolocation was implemented into an optimization algorithm. In Firefly Algorithm (FA) presented in Yang (2010a), communication between fireflies using flashing to exchange information and make the swarm move toward one direction was modeled to represent searching for optimal solutions. Flower Pollination Algorithm (FPA) proposed in Yang (2012), simulates the motion of pollen with the wind to search the space of solutions for the optimal point. In recent years were proposed many other nature-inspired algorithms based on special features or devoted behavior of animals while hunting, breeding, pairing, or other life activities seen in nature. Water waves that are braking at shore move in a peculiar way, which is possible to model into the Water Wave Optimization Algorithm (WWO) proposed in Zheng (2015). Ant Lion Optimizer (ALO) presented in Mirjalili (2015a) is based on a model of ant lions hunting ants in sand traps, while Whale Optimizer (WO) proposed in Mirjalili and Lewis (2016) maps humpback krill hunting activities. Even the way the flame attracts moths can inspire a method possible to be used in optimization problems. This approach was presented in Mirjalili (2015b) as Moth-Flame Optimization Algorithm (MFO). The hunting of dragonflies has some features that can be interesting for optimization problems what was discussed as Dragon-Fly Algorithm (DA) in Mirjalili (2016). The behavior of spider monkey was used to develop mathematical model of optimization algorithm discussed in Sharma et al. (2016). The latest algorithms were inspired by Harris Hawks (HHO) in Heidari et al. (2019), and by Slime Mould (SM) in Li et al. (2020).

All these research show that behaviors from nature can be efficiently mimicked by the use of mathematical approaches into solutions that serve as optimization techniques. Presented in this article red fox, as a predator is always depicted as a cunning and clever animal. Red fox is present from Europe to America hunting not only wild animals but also farmed animals like poultry, rabbits, etc. Therefore one of the most important opponents is human, who hunts for individuals which threaten farm. In this complex symbiosis, only the best individuals will have an offspring which will take over hunting territories or start new herd in a different hunting area. These relations, between predator and prey, between hunters and foxes, between members of the fox heard, were mimicked in the proposed model of the novel optimization algorithm. In the proposed model we assume that the domain for optimization is similar to lands and forests, in which fox is searching for possible occasions to hunt. Similarly to optimization problems fox does not know where to find food. The fox search domain can be roughed and therefore very difficult to penetrate, so fox needs a specific strategy to maximize the chance of success. Fox has developed a very efficient mechanism of hunting, for which we have distinguished two phases. The first one is implemented as a global search, the other as a local search. In the model, we propose to adopt travel through the lands and forests in search of hunting subdomains as a global search. Hunting for the prey is adopted as a local search for the optimum. Individuals in the search population exchange information about the domain during the global search phase (to exchange information where possible hunting can be beneficial), while during the local search each of them depends on his search abilities. Therefore we have a fast comparison between various and remote locations in the domain, but when it comes to precision of calculations each individual works separately in surrounding. Additionally to these two phases, the proposed model introduces a mechanism to control population. We propose a mechanism which simulates leaving the herd and establishing a new one or being killed fox by hunters. We propose a dynamic control of the population, in which we select the best individuals (an alpha couple) to reproduce and replace the weakest one in each iteration. This model helps to prevent the algorithm from blocking in the subspaces of the local extrema. Each part of the proposed model represents a different aspect of hunting and development which through the years helped the fox to succeed in various and changing conditions.

2. Red Fox Optimization Algorithm (RFO)

The population of red foxes consists both of those leaving on well-defined territories and those that lead a nomadic life. Each herd is shearing a single territory under the hierarchy of alpha couple. When the young grow up, they may leave the herd and set their herd if the chances to take control over another territory are large. Otherwise, they remain in the family and one day inherit the hunting area from their parents (Larivière & Pasitschniak-Arts, 1996). The red fox is an efficient hunter of small animals, both domestic and wild. The fox takes any chance for food while traversing through the territory, creeps up to the prey hiding until he comes close enough to effectively attack. The conceptual model of fox hunting attitudes is presented in Fig. 1. In our algorithm exploration of territories in search of food when the fox spots the prey in the distance was modeled as a global search. In the second phase, traversing through the habitat to get as close as possible to the prey before the attack was modeled as a local search.

2.1. Basic premise for the algorithm

In following iterations the population of individuals contains of a constant number of foxes. Each one of them is represented as a point $\bar{x} = (x_0, x_1, \ldots, x_{n-1})$ of n coordinates. To distinguish each fox $\bar{x}^i$ in iteration t, we introduce notation $\left(\bar{x}^i_j\right)^t$, where i is the number of the fox in the population and j represents coordinate according to dimensions of the solution space. We assume that foxes move in solution space using given equations in search of optimum values for the criterion function.

Let $f \in \mathbb{R}^n$ be the criterion function of n variables according to the solution space dimensions, and let the notation $(\bar{x})^{(i)} = \left[(x_0)^{(i)}, (x_1)^{(i)}, \ldots, (x_{n-1})^{(i)}\right]$ denotes each point in the space $\langle a, b\rangle^n$ where $a, b \in \mathbb{R}$. Then $(\bar{x})^{(i)}$ is the optimal solution if function $f\left((\bar{x})^{(i)}\right)$ value is a global minimum or maximum on $\langle a, b\rangle$.

2.2. In search for food — global search phase

In a herd, each fox must play a role, important for the survival of all the family members. If there is no food in the local habitat, or for the exploration of other territories, members of the herd travel to remote destinations. The information they get in this exploration they share with family to help with survival and development.

Exploration of the surrounding lands we model according to the fitness of all individuals. In the proposed way we assume that the best individual has explored the most interesting lands and can share with a family this information. Therefore first we sort population according to fitness condition, and for $(\bar{x}^{best})^t$ we calculate the square of the euclidean distance to each individual in the population as

$$d((\bar{x}^i)^t, (\bar{x}^{best})^t) = \sqrt{\|(\bar{x}^i)^t - (\bar{x}^{best})^t\|}, \tag{1}$$

and we move individuals in the population toward the best one

$$(\bar{x}^i)^t = (\bar{x}^i)^t + \alpha \operatorname{sign}((\bar{x}^{best})^t - (\bar{x}^i)^t), \tag{2}$$

where $\alpha \in (0, d((\bar{x}^i)^t, (\bar{x}^{best})^t))$ is randomly selected scaling hyperparameter set once in iteration for all individuals in the population. If the fitness values for new positions (after move toward the best location) are better the individuals stay in new positions, otherwise they return to previous positions. This represents that after exploration family members return home and show the others where to hunt. The family members go to directions shown by the explores. If there was a chance for the food they stay for hunting, otherwise return home with "empty hands". These operations represent proposed global search performed in each iteration of RFO, which sample presentation is shown in Fig. 2. Additionally, we assume that in remote lands fox does not know where to hide or escape, which makes him vulnerable to danger. Therefore we model possible killing of the worst fitted individuals in the population or reward the best individuals by reproduction according to the model in Section 2.4.

2.3. Traversing through the local habitat — local search phase

Red fox traverses its territory in search of potential prey. When a possible one is spotted, the fox starts to approach quietly as close as possible trying not to be noticed. Therefore he is circling and deceiving around the prey to convince it that he is not interested in hunting. However, when approaching at a close distance he moves as fast as possible to attack by surprise.

In RFO we have modeled observation and movement to deceive victims while hunting into a local search phase. To model a possibility of a fox being noticed while approaching closer to the prey we implemented random value $\mu \in \langle 0, 1\rangle$ set once in the iteration for all individuals in the population, which defines the action of the fox as

$$\begin{cases} \text{Move closer} & \text{if } \mu > 0.75 \\ \text{Stay and disguise} & \text{if } \mu \le 0.75 \end{cases}. \tag{3}$$

If μ parameter shows to move the population in this iteration we use a modified Cochleoid equation to visualize the movement of each individual. For this movement fox observation radius is represented by two parameters: $a \in \langle 0, 0.2\rangle$ is a scaling parameter set once in the iteration for all individuals in the population to model randomly changing distance from the prey during fox approaching, and $\phi_0 \in \langle 0, 2\pi\rangle$ is selected for all individuals at the beginning of the algorithm to model fox observation angle. Using them we define vision radius of the hunting fox

$$r = \begin{cases} a\dfrac{\sin(\phi_0)}{\phi_0} & \text{if } \phi_0 \neq 0 \\ \theta & \text{if } \phi_0 = 0 \end{cases}, \tag{4}$$

where θ is a random value between 0 and 1 set once at the beginning of the algorithm which is interpreted as the influence of adverse weather conditions such as fog, rain, etc. For the population of individuals, we model movements using the following system of equations for spatial coordinates

$$\begin{cases} x_0^{new} = ar \cdot \cos(\phi_1) + x_0^{actual} \\ x_1^{new} = ar \cdot \sin(\phi_1) + ar \cdot \cos(\phi_2) + x_1^{actual} \\ x_2^{new} = ar \cdot \sin(\phi_1) + ar \cdot \sin(\phi_2) + ar \cdot \cos(\phi_3) + x_2^{actual} \\ \ldots \\ x_{n-2}^{new} = ar \cdot \sum_{k=1}^{n-2} \sin(\phi_k) + ar \cdot \cos(\phi_{n-1}) + x_{n-2}^{actual} \\ x_{n-1}^{new} = ar \cdot \sin(\phi_1) + ar \cdot \sin(\phi_2) + \cdots + ar \cdot \sin(\phi_{n-1}) + x_{n-1}^{actual} \end{cases}, \tag{5}$$

wherein each angular value is randomized for each point according to $\phi_1, \phi_2, \ldots \phi_{n-1} \in \langle 0, 2\pi)$. This model represents the behavior of a fox after he notices a victim and tries to approach as close as possible to attack and if fails when discovered he tries to approach another one similarly. In the proposed RFO algorithm, this attitude is modeled as a local search phase. A sample graph of the modeled circling the prey while hunting is shown in Fig. 2.

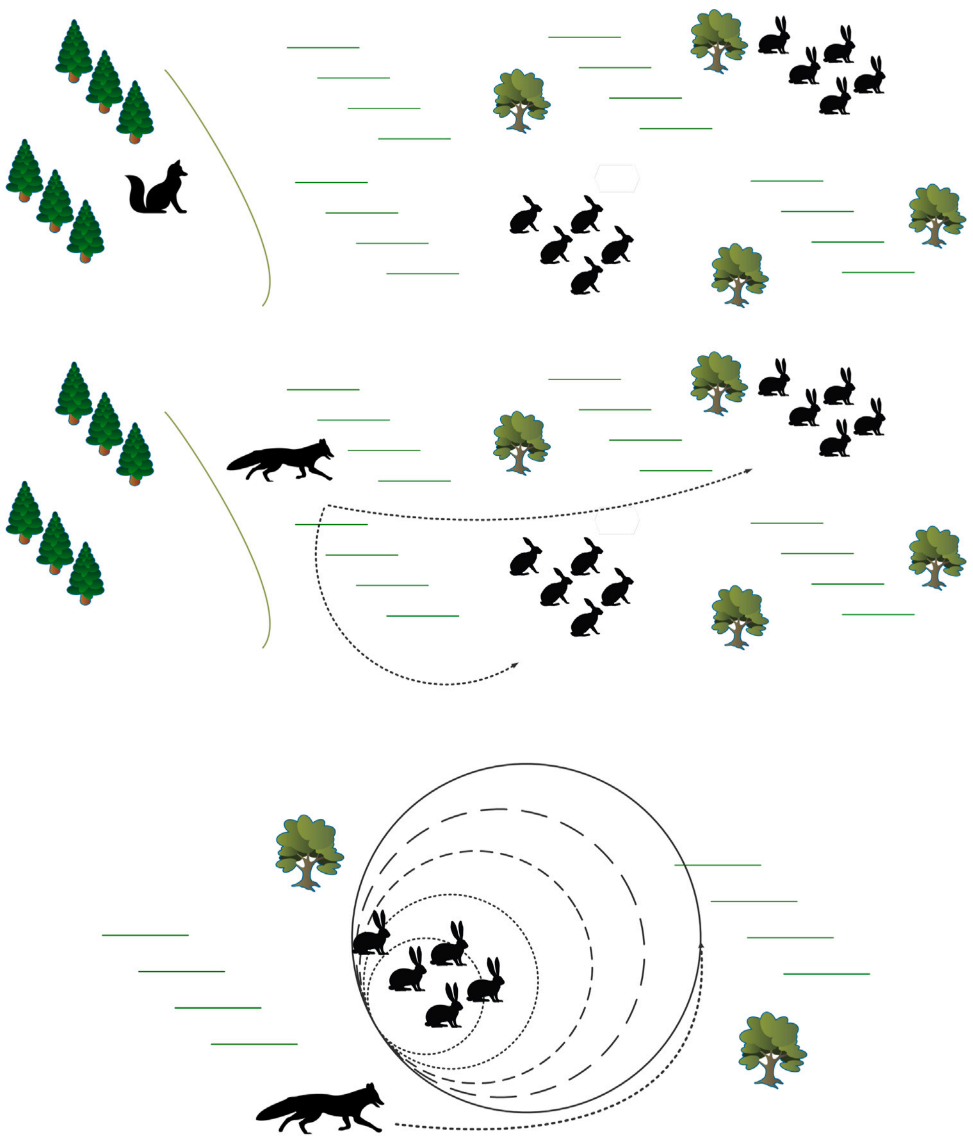


Fig. 1. Sample presentation of red fox natural hunting behavior which we modeled into optimization strategy. The fox in his search domain starts to search for locations of small animals. When he finds possible prey within his reach, the fox starts to move closer without notice. Approaching in the vicinity, he remains imperceptible deceiving the prey until the final act of hunting, when from the close distance fox attacks the prey.

2.4. Reproduction and leaving the herd

In nature, the red fox must face many dangers. There may be no food in the local habitat area, therefore it would be necessary to move elsewhere. Another danger comes from humans, who can hunt down the fox if damages in a population of domestic animals will be big. However, in nature, not all foxes die or migrate. Most of them are very smart and can escape and reproduce, giving new lines to the fox herds. The model of reproduction and leaving local habitat is presented in Fig. 3.

To model this behavior in each iteration we select 5% of the worst individuals in the population according to the value of criterion function, we used this value as our subjective assumption to simulate small changes in the herd. Since these are the worst fitted ones, we assume that these foxes either moved elsewhere or were shot down by hunters. To make the number of individuals in the population constant we replace them with new individuals using a model of habitat territory established by the alpha couple. In each t iteration of the RFO algorithm, we select two best individuals $(\overline{x}^{(1)})^t$ and $(\overline{x}^{(2)})^t$ to represent the alpha couple, for whom we calculate center of the habitat

$$(habitat^{(center)})^t = \frac{(\overline{x}^{(1)})^t + (\overline{x}^{(2)})^t}{2} \tag{6}$$

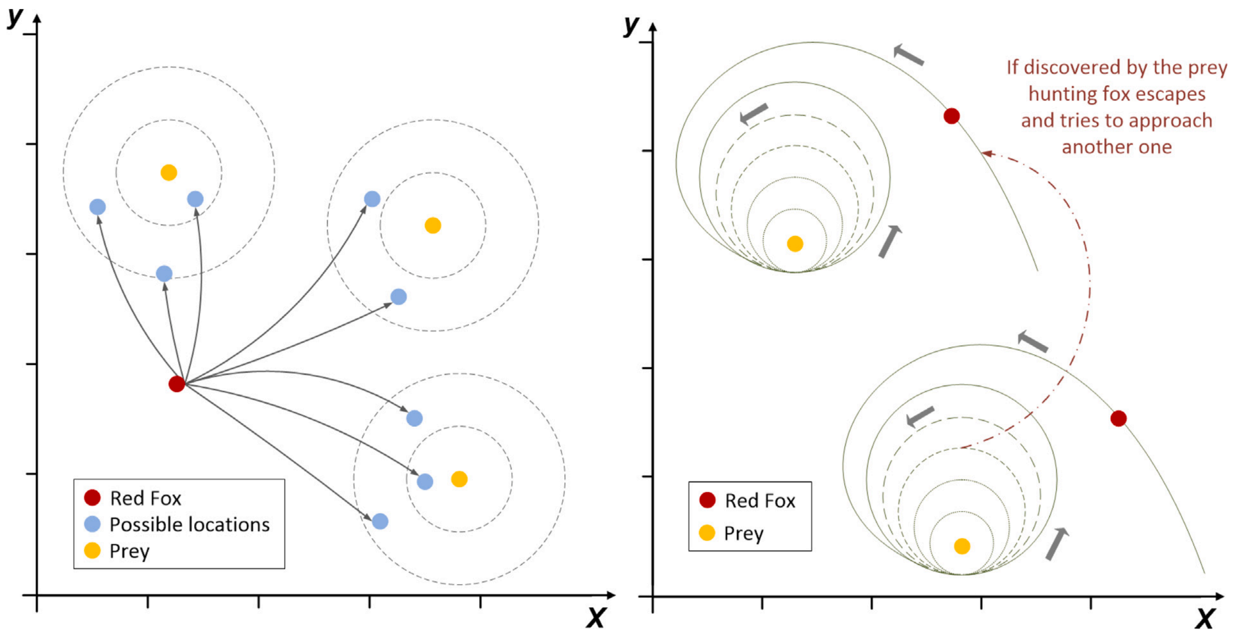
and the habitat as the square of the euclidean distance between the alpha couple

$$(habitat^{(diameter)})^t = \sqrt{\|(\overline{x}^{(1)})^t - (\overline{x}^{(2)})^t\|}. \tag{7}$$

For each iteration we take at random parameter $\kappa \in \langle 0, 1\rangle$ which defines replacements in the iteration in accordance with

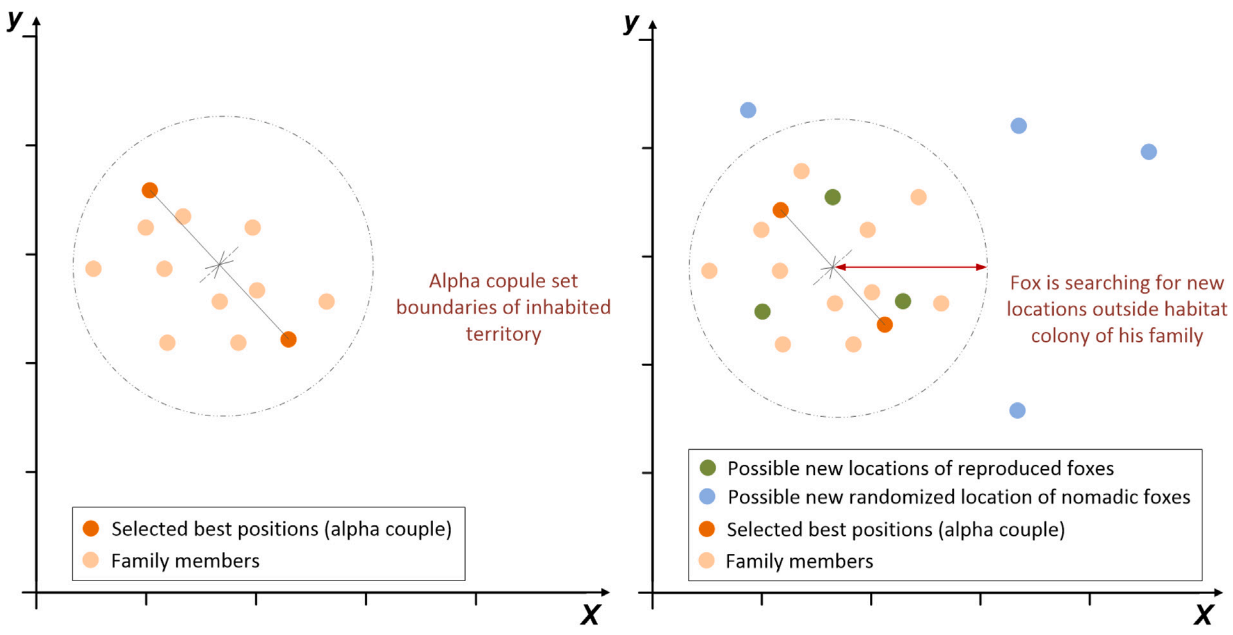
$$\begin{cases} \text{New nomadic individual} & \text{if } \kappa \geq 0.45 \\ \text{Reproduction of the alpha couple} & \text{if } \kappa < 0.45 \end{cases}. \tag{8}$$

In the first case, we assume that new family members leave the habitat as nomadic foxes and go outside the area in search of food and the possibility to reproduce in their herd. We select positions at random within the search space, but outside the habitat. In the second case, we assume that new individuals come from the alpha couple, therefore


y
x
Red Fox
Possible locations
Prey
If discovered by the prey hunting fox escapes and tries to approach another one

(a) RFO global search (b) RFO local search

Fig. 2. (a) RFO global search: Possible following hunting positions of red fox to start circling and deceiving around the prey. During traverse in the habitat, each individual looks for possible prey and moves toward it. (b) RFO local search: Visualization of red fox circling and deceiving around the prey to hunt it down. The individual tries to approach the prey as close as possible, however if noticed tries to attack the other one in a similar way.


y
x
Alpha copule set boundaries of inhabited territory
Selected best positions (alpha couple)
Family members
Fox is searching for new locations outside habitat colony of his family
Possible new locations of reproduced foxes
Possible new randomized location of nomadic foxes

(a) RFO reproduction phase 1 (b) RFO reproduction phase 2

Fig. 3. (a) RFO reproduction phase 1: In the population, alpha couple that have established the family, set boundaries of the inhabited territory where all members of the family live and hunt. (b) RFO reproduction phase 2: Alpha couple have the right to reproduce and enlarge the family, while all the others work for prosperity. If some members want to set their own rules, there comes a time to leave the habitat to search for new place outside family territory.

we reproduce two best individuals $(\overline{x}^{(1)})^t$ and $(\overline{x}^{(2)})^t$ combined in a new individual $(\overline{x}^{(reproduced)})^t$ as

$$(\overline{x}^{(reproduced)})^t = \kappa \frac{(\overline{x}^{(1)})^t + (\overline{x}^{(2)})^t}{2}. \tag{9}$$

We have implemented our optimization algorithm as presented in Algorithm 1.

2.5. Note on the convergence

Let us now present a discussion on the convergence of the proposed method. To do this we first introduce some definitions to remind some mathematical aspects of the topology of the solution space.

Definition 1. The sequence of the points $(\overline{x})^{(i)} = \left[(x_0)^{(i)}, (x_1)^{(i)}, \dots, (x_k)^{(i)}, \dots, (x_{n-1})^{(i)}\right]$ in iteration i of the optimization method resulting

Algorithm 1 Red Fox Optimization Algorithm

```
1: Start,
2: Define parameters of the algorithm: fitness function $f(\cdot)$, size of search space solution $\langle a,b \rangle$, number of iterations $T$, the maximum size of the population $n$, fox observation angle $\phi_0$, weather conditions $\theta$,
3: Generate population consisting of $n$ foxes at random within search space,
4: $t := 0$,
5: while $t \le T$ do
6:   Define coefficients for iteration: fox approaching change $a$, scaling parameter $\alpha$,
7:   for each fox in current population do
8:     Sort individuals according to the fitness function,
9:     Select $(\bar{x}^{best})^t$,
10:    Calculate reallocation of individuals according to Eq. (2),
11:    if reallocation is better than the previous position then
12:      Move the fox,
13:    else
14:      Return the fox to previous position,
15:    end if
16:    Choose parameter $\mu$ value to define noticing the hunting fox,
17:    if fox is not noticed then
18:      Calculate fox observation radius $r$ according to Eq. (4),
19:      Calculate reallocation according to Eq. (5),
20:    else
21:      Fox stays at his position to disguise,
22:    end if
23:  end for
24:  Sort population according to the fitness function,
25:  Worst foxes leave the heard or get killed by hunters,
26:  New foxes are replaced in the population using Eq. (8) as a nomadic fox outside the habitat or are reproduced from the alpha couple inside the herd Eq. (9),
27:  $t++$,
28: end while
29: Return the fittest fox $(\bar{x})^{best}$,
30: Stop.
```

in the solution space $\mathbf{X} = \langle a,b \rangle^n$ where $a,b \in \mathbb{R}$ of the optimization task will be called convergent to a certain value of $(\bar{x})^{(*)}$ if for each neighborhood V_* there will be an index n_0 that $\forall k > n_0\, (x_k)^{(i)} \in V_*$. Then the boundary $(x)^{(*)} \in (\bar{x})^{(*)}$ of this sequence in iteration i will be called an optimum in the sense of optimization task.

Definition 2. The sequence of the points $(\bar{x})^{(i)} = \left[(x_0)^{(i)}, (x_1)^{(i)}, \dots, (x_{n-1})^{(i)}\right]$ of the solution space $\mathbf{X}$ is a Cauchy sequence in the sense of the metric function $\|\cdot\|$ if $\forall \epsilon > 0, \exists n_0 \in \mathbb{N}$ that $\forall k,l > n_0$ we have $\|(x_k)^{(i)}, (x_l)^{(i)}\| < \epsilon$.

For the purpose of our research we assume that if in the solution space $\mathbf{X}$ for optimization task we can find a Cauchy sequence of points which in the sense of $\|\cdot\|$ function is convergent to some point of this space and this point is the best in the sense of our optimization task, we can say that the proposed method is convergent in the sense of the distance between points in the following iterations of the proposed method. Let us now define conditions for metric function over the distance between points.

Definition 3. Let $(\bar{x})^{(i)} = \left[(x_0)^{(i)}, (x_1)^{(i)}, \dots, (x_{n-1})^{(i)}\right]$ be the sequence of points of the solution space $\mathbf{X}$. Metric function $\|(x_k)^{(i)}, (x_l)^{(i)}\| \longrightarrow [0, +\infty)$ over the distance between points returned from the optimization must satisfy conditions:

(a) $\|(\bar{x})^{(i)}\| = 0 \Longrightarrow (x_k)^{(i)} = 0$,
(b) $\forall \alpha \in R, \|\alpha(\bar{x})^{(i)}\| = \alpha \|(\bar{x})^{(i)}\|$,
(c) $\forall (\bar{x})^{(i)}, (\bar{x})^{(j)} \in \mathbf{X}, \|(\bar{x})^{(i)} + (\bar{x})^{(j)}\| \le \|(\bar{x})^{(i)}\| + \|(\bar{x})^{(j)}\|$.

All sequences of points $(\bar{x})^{(i)} = \left[(x_0)^{(i)}, (x_1)^{(i)}, \dots, (x_{n-1})^{(i)}\right]$ returned by the optimization algorithm will be limited by some value $(x)^{(*)}$, which will be the best solution in a given iteration. If for the optimization task we define a metric function $\|\cdot\|$ in the sense of the distance between the best solution in a given iteration and this global optimum $(x)^{(*)}$ of the fitness function in each iteration i, the solution space $\mathbf{X}$ will be a normed and complete space.

Theorem 1. *The solution space* $\mathbf{X}$ *of to the given optimization problem is a normed and complete space for sequences of solutions in the sense of the norm* $\|(\bar{x})^{(i)}\| = |\sup((\bar{x})^{(i)}, (x)^{(*)})|$.

Proof. Normalization of the space:

(a) $\|(\bar{x})^{(i)}\| = 0 \Longrightarrow |\sup((\bar{x})^{(i)}, (x)^{(*)})| = 0 \Longrightarrow \exists (x_k)^{(i)} = (x)^{(*)} \Longrightarrow (x)^{(*)} \in (\bar{x})^{(i)}$,
(b) $\forall \alpha \in R, \|\alpha(\bar{x})^{(i)}\| = |\sup(\alpha(\bar{x})^{(i)}, \alpha(x)^{(*)})| = |\alpha \sup((\bar{x})^{(i)}, (x)^{(*)})| = |\alpha| \cdot |\sup((\bar{x})^{(i)}, (x)^{(*)})| = |\alpha| \cdot \|(\bar{x})^{(i)}\|$,
(c) $\forall (\bar{x})^{(i)}, (\bar{x})^{(j)} \in \mathbf{X}, \|(\bar{x})^{(i)} + (\bar{x})^{(j)}\| = |\sup((\bar{x})^{(i)} + (\bar{x})^{(j)}, (x)^{(*)})| = |\sup((\bar{x})^{(i)}, (x)^{(*)}) + \sup((\bar{x})^{(j)}, (x)^{(*)})|$
$\le |\sup((\bar{x})^{(i)}, (x)^{(*)})| + |\sup((\bar{x})^{(j)}, (x)^{(*)})| = \|(\bar{x})^{(i)}\| + \|(\bar{x})^{(j)}\|$.

Completeness of the space:
Let $\{(\bar{x})^{(i)}_n\}^{\infty}_{n=0}$ for $(\bar{x})^{(i)} \in \mathbf{X}$ be a sequence of sequences of possible solutions in the solution space $\mathbf{X}$. Since this space is bounded in the sense of the norm, $\forall \epsilon > 0\, \exists n_0 \in \mathbb{N}$, that $\forall n,m > n_0$ we have $\|(\bar{x})^{(i)}_n, (\bar{x})^{(i)}_m\| < \epsilon \Longrightarrow |\sup((\bar{x})^{(i)}_n, (\bar{x})^{(i)}_m; (x)^{(*)})| < \epsilon \Longrightarrow |sup((\bar{x})^{(i)}_n, (x)^{(*)})| < \epsilon$ and $|\sup((\bar{x})^{(i)}_m, (x)^{(*)})| < \epsilon$ what defines a Cauchy sequence. We can say that differences between sequences of the sequence $\{(\bar{x})^{(i)}_n\}^{\infty}_{n=0}$ are at maximum the differences of supremums of these sequences, which are bounded for the solution space $\mathbf{X}$ for a bounded optimum problem. Since we assume that evaluated optimization problems will have boundaries we can say that each of these sequences will be limited by a value to which they are convergent in the following iterations of the algorithm in the sense of $\|(\bar{x})^{(i)}\|$. So in the solution space $\mathbf{X}$ the sequence of sequences $\{(\bar{x})^{(i)}_n\}^{\infty}_{n=0}$ is a Cauchy sequence where $\lim_{n\to\infty}(\bar{x})^{(i)}_n = (\bar{x})^{(*)}$. Thus also for each sequence $\lim_{n\to\infty}(\bar{x})^{(i)} = (x)^{(*)}$, since $m \longrightarrow \infty$ we have $|\sup(x_n)^{(i)}, (x_m)^{(i)}; (x)^{(*)}| < \epsilon$. So in general for the solution space $\mathbf{X}$ we can say $\{(\bar{x})^{(i)}_n\}^{\infty}_{n=0}$ fulfills $\exists n_0 \in \mathbb{N}$ that $\forall n > n_0 |\sup((x_n)^{(i)}, (x)^{(*)})| < \epsilon$ so for $\epsilon > 0\, \|(\bar{x})^{(i)}_n, (\bar{x})^{(*)}\| < \epsilon \Longrightarrow |\sup((\bar{x})^{(i)}_n, (\bar{x})^{(*)})| < \epsilon \Longrightarrow (\bar{x})^{(i)}_n \xrightarrow{n\to\infty} (x)^{(*)}$. At the same time each sequence is limited because $|\sup((\bar{x})^{(i)})| = |\sup((\bar{x})^{(i)} + (\bar{x})^{(i)}_n - (\bar{x})^{(i)}_n)| \le |\sup((\bar{x})^{(i)} - (\bar{x})^{(i)}_n)| + |\sup((\bar{x})^{(i)}_n)| \le \epsilon + \max(\{(\bar{x})^{(i)}_n\}^{\infty}_{n=0}) = \epsilon + M = M'$. Therefore we can say about sequences returned from the proposed optimization method is bounded in the sense of Cauchy, and about the solution space $\mathbf{X}$ to be complete and normed. □

So the sequences of the results shall converge to the optimum returned as the best result $(x)^{(*)}$ in each iteration i. Optimization is repeated in the following iterations and the best result is improved in the following iterations, what composes an ordered sequence $(\bar{x})^{(*)}$ of these values.

2.6. Time complexity

The time complexity of RFO can be analyzed using procedure in Algorithm 1. Let us mark population size as n, problem dimension as D, and the number of iterations as T. In each iteration of RFO, all individuals are sorted what gives $O((n \times D)^2)$ operations in the worst case. Here comes conclusion that RFO time complexity can be reduced by fast sorting algorithm, especially for high dimensional task. Then, the best fox is selected which is constant because of the sorting population. Then, reallocation is performed for each fox, so there will be $O(n \times D \times k)$ operations, where k is a constant value which depends on satisfying condition and replacing values of a given fox. In the next lines, there is a possible reallocation based on choosing μ parameter. The worst case will be when each fox is reallocated, then time complexity for calculating observation will be $O(n \times D)$ operations, and reallocation $O(n \times D)$. In line 24–27, there will be a sorting operation again, so $O((n \times D)^2)$, and deleting the worst individuals and create new in the worst case will be $O((n \times D)^2)$ and it is true when all foxes will be deleted and recreated again. Line 27 is constant. Omitting the low-order terms, the computational time can be defined as $O(3 \times T \times n^2 \times D^2)$ operations in pessimistic case.

2.7. Discussion on differences and similarities to other heuristic approaches

In Table 1, we present many algorithms inspired by natural phenomena models from the last years. There are many different analogies to the nature, behavior of animals, or physical phenomena. The main reason for creating so many algorithms is modeling a solution which can be stable on average results and capable to solve complex problems from engineering and optimization sciences. It means, that on a specific number of iterations and the size of the population, the average values returned by the heuristic algorithm will be similar in the range of these algorithms and will fit given criterion or fitness condition. When modeling such algorithms, it is necessary to analyze two operations: local and global search methods. Based on these two procedures, we can control convergence and fitness function value in each iteration to avoid situation when some individuals stuck in local extrema. For example, in PSO, all particles are moving toward the best one. In GA there is a reproduction of genotypes based on selected best one. The result of this will help to reproduce a new individual somewhere between the parents. In RFO we propose a movement based on searching solution space between each fox and the best individual in the iteration. This movement is based on random value with the sign chosen by calculating the difference between the best and current fox. This allows to search solution space in a new, easy way. Most important difference of RFO is modeling the local search method and adding the mechanism of reproduction and leaving the herd. This is similar to deleting the worst cases in GA and other heuristic solutions. However, in RFO we manipulate the worst values by deleting them and replacing with random new solution, but also create a new one based on top rated solutions. Despite similarity to other models, the most important novelty of RFO is the new local/global searching and the mechanism of reproduction/leaving the herd. On the other hand, this proposal has potential flaw which is sorting mechanism performed twice in each iteration. It can influence complexity of RFO in high dimensional spaces and might be considered a drawback in some cases. However, as mentioned in Section 2.6, choosing fast sorting algorithm can much improve RFO. We can also modify RFO by deleting one sorting operation, for instance on the beginning of each iteration (it must be noted, that in this case, sorting should be performed once before the whole algorithm to find the best individual).

3. Benchmark tests

Proposed RFO method was evaluated in benchmark tests using 16 sample test functions in 2D presented in Table 2 and additional 6 functions in 20D to extend our benchmark tests also for a multi-dimensional domain presented in Table 3. For the benchmark tests we have chosen various examples, some of them are smooth surfaces of spherical shapes (i.e. Rotated Hyper-Ellipsoid or Sphere) and the other is ruffled (i.e. Rastragin, W/Wavy or Yang). Both types are good for optimization tests since in the first case an algorithm can easily stuck and for the second type, the algorithm must search among rapidly changing values.

In benchmark tests, we have compared the proposed RFO to a set of meta-heuristic algorithms, swarm intelligence algorithms, and nature-inspired heuristics. For the benchmark test we have taken classic meta-heuristic approaches like genetic algorithm — GA (Holland, 1992), simulated annealing — SA (Van Laarhoven & Aarts, 1987) and differential evolution — DE (Storn & Price, 1997). We also compared RFO to classic swarm intelligence, like particle swarm — PSO (Kennedy & Eberhart, 1995) which is one of the best heuristics based on the concept of swarm intelligence. We also compared RFO to ant colony — AACA (Dorigo & Di Caro, 1999) and bee colony — ABCA (Toksari, 2006). On the other hand, we also compared it to heuristics inspired by natural phenomena, like water wave — WWO (Zheng, 2015). Additionally, we have selected heuristics that use a nature-inspired search concept based on animals, insects, or plants strategies developed in their origin environments. One of the first and widely reported nature-inspired heuristics are mimicking habits of the bat — BA (Yang, 2010b), cuckoo — CSA (Yang & Deb, 2009), and firefly — FA (Yang, 2010a). Among the latest nature inspired heuristics we have taken method based on flower pollination — FPA (Yang, 2012), method based on dragonfly — DA (Mirjalili, 2016), moth — MFO (Mirjalili, 2015b), ant lion — ALO (Mirjalili, 2015a) and whale — WO (Mirjalili & Lewis, 2016). Each of the algorithms was run 100 times, and all compared heuristics have 100 individuals in population and 100 iterations. Tables 4–7 present average of all results among all runs for each one of the methods together with standard deviation values. The RFO results in Figs. 4–5 are presented for optimization accuracy and we show the best result from the last iteration of the algorithm over the function minima landscape, average optimization trajectory that represents averaged movement of the individuals in the following iterations, average fitness in the population, and average adaptation with convergence during the following iterations. Figs. 6–7 present a comparison of convergence for the analyzed heuristics in 20D, while Fig. 8 presents a comparison of standard deviations for these benchmark tests results.

Analyzing charts with results after the last iteration from the best optimization we can see that for all presented test functions individuals are located very close to the maximum/minimum. The best of them almost covers these points. Trajectories presenting average movements are a little different for spherical and ruffled shapes. However mainly for all of the test functions, up to 20 iteration we can see the highest changes in this value. That means the population is located in a surrounding of the maxima/minima in the first 20 iterations. After 20 iteration individuals are improving the precision of the final result, where for some of the test function like Sphere, in the last 10 iterations individuals move even less. Average fitness shows the average difference between the analytical solution and the results computed by the tested algorithm. For our tests, we have defined

$$fit = \frac{\sum_{k=0}^{individuals} |f(x^k) - f(x^{ideal})|}{individuals}, \tag{10}$$

where the number of $individuals$ was equal to 100 for all compared methods, (x^k) represents position of the individual k for compared method, and (x^{ideal}) represents analytical solution in which test function $f(\cdot)$ has maxima/minima. The charts in Figs. 4 and 5 show that up to 20 iteration there are the most significant changes in the adaptation of individuals to a given criterion. In the following iterations, RFO is adjusting the results to the expected value, so the changes are lower. That shows the high potential of the proposed algorithm. Csendes, Mishra, and Schumer Stegilitz are functions for which RFO has reached a very high precision in 2D, while for other the result was not so good. Convergence coefficient represents convergence to the maxima/minima defined in analytical way, what we have defined as

$$conv = \frac{|f(x^{best}) - f(x^{ideal})|}{|fit^{(t-1)}|}, \tag{11}$$

where (x^{best}) represents the best individual, (x^{ideal}) represents analytical solution in which test function $f(\cdot)$ has maxima/minima, and $fit^{(t-1)}$ is average fitness calculated in iteration $t-1$. Analysis of convergence

Table 2
Applied classic benchmark test functions used for calculations in 2D.

Function name	Function $f(\cdot)$	Range	f_{min}	Solution $\overline{x}$
Brown	$f_1(\overline{x}) = \sum_{i=1}^{n-1}(x_i^2)^{(x_{i+1}^2+1)} + (x_{i+1}^2)^{(x_i^2+1)}$	$\langle -1, 4\rangle$	0	$(0,\ldots,0)$
Csendes	$f_2(\overline{x}) = \sum_{i=1}^{n} x_i^6\left(2+\sin\left(\frac{1}{x_i}\right)\right)$	$\langle -1, 1\rangle$	0	$(0,\ldots,0)$
Griewank	$f_3(\overline{x}) = \sum_{i=1}^{n}\frac{x_i^2}{4000} - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{(i)}}\right)$	$\langle -10, 10\rangle$	0	$(0,\ldots,0)$
Powell	$f_4(\overline{x}) = \sum_{i=1}^{n/4}((x_{4i-3}+10x_{4i-2})^2 + 5(x_{4i-1}-x_{4i})^2 + (x_{4i-2}-2x_{4i-1})^4 + 10(x_{4i-3}-x_{4i})^4)$	$\langle -100, 100\rangle$	0	$(0,\ldots,0)$
Powell Sum	$f_5(\overline{x}) = \sum_{i=1}^{n}\lvert x_i\rvert^{i+1}$	$\langle -1, 1\rangle$	0	$(0,\ldots,0)$
Rastragin	$f_6(\overline{x}) = 10n + \sum_{i=1}^{n}[x_i^2 - 10\cos(2\pi x_i)]$	$\langle -10, 10\rangle$	0	$(0,\ldots,0)$
Rotated Hyper–Ellipsoid	$f_7(\overline{x}) = \sum_{i=1}^{n}\sum_{j=1}^{i} x_j^2$	$\langle -100, 100\rangle$	0	*$(0,\ldots,0)$
Mishra no. 11	$f_8(\overline{x}) = \left[\frac{1}{n}\sum_{i=1}^{n}\lvert x_i\rvert - \left(\lvert x_i\rvert\right)^{1/n}\right]^2$	$\langle -10, 10\rangle$	0	$(0,\ldots,0)$
Salomon	$f_9(\overline{x}) = 1 - \cos\left(\sum_{i=1}^{n}x_i^2\right) + 0.1\sqrt{\sum_{i=1}^{n}x_i^2}$	$\langle -100, 100\rangle$	0	$(0,\ldots,0)$
Sargan	$f_{10}(\overline{x}) = \sum_{i=1}^{n}\left(x_i^2 + 0.4\sum_{j\neq i}x_i x_j\right)$	$\langle -100, 100\rangle$	0	$(0,\ldots,0)$
Schumer Steiglitz	$f_{11}(\overline{x}) = \sum_{i=1}^{n}x_i^4$	$\langle -10, 10\rangle$	0	$(0,\ldots,0)$
Sphere	$f_{12}(\overline{x}) = \sum_{i=1}^{n}x_i^2$	$\langle -10, 10\rangle$	0	$(0,\ldots,0)$
W/Wavy	$f_{13}(\overline{x}) = 1 - \frac{1}{n}\sum_{i=1}^{n}\cos(10x_i)\exp(-0.5x_i^2)$	$\langle -\pi, \pi\rangle$	0	$(0,\ldots,0)$
Weierstrass	$f_{14}(\overline{x}) = \sum_{i=1}^{n}([x_i+0.5])^2$	$\langle -30, 30\rangle$	0	$(-\frac{1}{2},\ldots,-\frac{1}{2})$
Yang	$f_{15}(\overline{x}) = \left(\sum_{i=1}^{n}\lvert x_i\rvert\right)\exp\left(-\sum_{i=1}^{n}\sin(x_i^2)\right)$	$\langle -2\pi, 2\pi\rangle$	0	$(0,\ldots,0)$
Zakharov	$f_{16}(\overline{x}) = \sum_{i=1}^{n}x_i^2 + (0.5ix_i)^2 + \left(\sum_{j=1}^{n}0.5jx_j\right)^4$	$\langle -10, 10\rangle$	0	$(0,\ldots,0)$

Table 3
Applied classic benchmark test functions used for calculations in 20D.

Function name	Function $f(\cdot)$	Range	f_{min}	Solution $\overline{x}$
Dixon-Price	$f_{17}(\overline{x}) = (x_1-1)^2 + \sum_{i=1}^{n}i(2x_i^2 - x_{i-1})^2$	$\langle -10, 10\rangle$	0	$\left(2^{-\frac{2^1-2}{2^1}},\ldots,2^{-\frac{2^n-2}{2^n}}\right)$
Rosenbrock	$f_{18}(\overline{x}) = \sum_{i=1}^{n-1}\left(100(x_{i+1}-x_i^2)^2 + (x_i-1)^2\right)$	$\langle -100, 100\rangle$	0	$(1,\ldots,1)$
Schwefel	$f_{19}(\overline{x}) = 418.9829n - \sum_{i=1}^{n}x_i\sin(\sqrt{\lvert x_i\rvert})$	$\langle -500, 500\rangle$	0	$(420.97,\ldots,420.97)$
Shubert	$f_{20}(\overline{x}) = \sum_{i=1}^{n}\sum_{j=1}^{5}j\sin((j+1)x_i)+j$	$\langle -10, 10\rangle$	−24.06	$(5.79,\ldots,5.79)$
Sum squares	$f_{21}(\overline{x}) = \sum_{i=1}^{n}ix_i^2$	$\langle -10, 10\rangle$	0	$(0,\ldots,0)$
Salomon	$f_{22}(\overline{x}) = 1 - \cos\left(2\pi\sqrt{\sum_{i=1}^{n}x_i^2}\right) + 0.1\sum_{i=1}^{n}x_i^2$	$\langle -100, 100\rangle$	0	$(0,\ldots,0)$

coefficient in Figs. 4 and 5 also show that first 20 iterations are very important in RFO, since during these the individuals in the algorithm concentrate on the surrounding of the maxima/minima. For the next iterations RFO improves precision. Analyzing charts in Figs. 4 and 5 we see that for Yang and Rastragin functions the results of this measurement are worse in comparison to other test functions. This can be caused by the function surface type in which we can notice several changes among maxima and minima. In comparison, for spherical test functions, we did not notice this behavior. Analyzing results in Tables 4 and 6 we can see that the results do not differ much between tested algorithms. On the other hand for each of test functions in 2D from Table 4 we can find the best result. For f_1 function the best results were achieved for CSA, DA, PSO and RFO; for f_2 function BA, GA, PSO, FPA and RFO reached lowest averaged results; for f_3 function CSA, DA, GA, PSO and RFO were the best; for f_4 function FA, FPA, GA, PSO and RFO; for f_5 function DA outperformed all other methods; for f_6 function FA, PSO and RFO were the best; for f_7 function BA, GA, PSO were reaching the lowest values; for f_8 function BA, CSA and MFO were the best; for f_9 function CSA, MFO, PSO and RFO; for f_{10} function FA, MFO, RFO; for f_{11} function CSA, FPA were the best; for f_{12} function DA, MFO, GA and RFO; for f_{13} function DA, FA, FPA, MFO and RFO reached the lowest results; for f_{14} function FPA, GA, PSO and RFO outperformed other methods; for f_{15} CSA, MFO and WWO; for f_{16} function FA, GA, PSO and RFO were the best. Similarly for test functions in 20D from Table 6 we can also the best results. For f_{17} function MFO, PSO GA, DE, and RFO were the best; for f_{18} function GA, PSO, DE, SA, and RFO achieved the lowest values; for f_{19} function MFO, GA, DE and RFO outperformed other methods; for f_{20} function CSA, DA, MFO, GA and RFO reached close to the optimum; for f_{21} function CSA, MFO, GA, PSO, AACA, ABCA, and RFO; for f_{22} function CSA, PSO and RFO were the best. In general, RFO was among the best methods for 17 functions among 22, where for these functions in 20D RFO was among the best

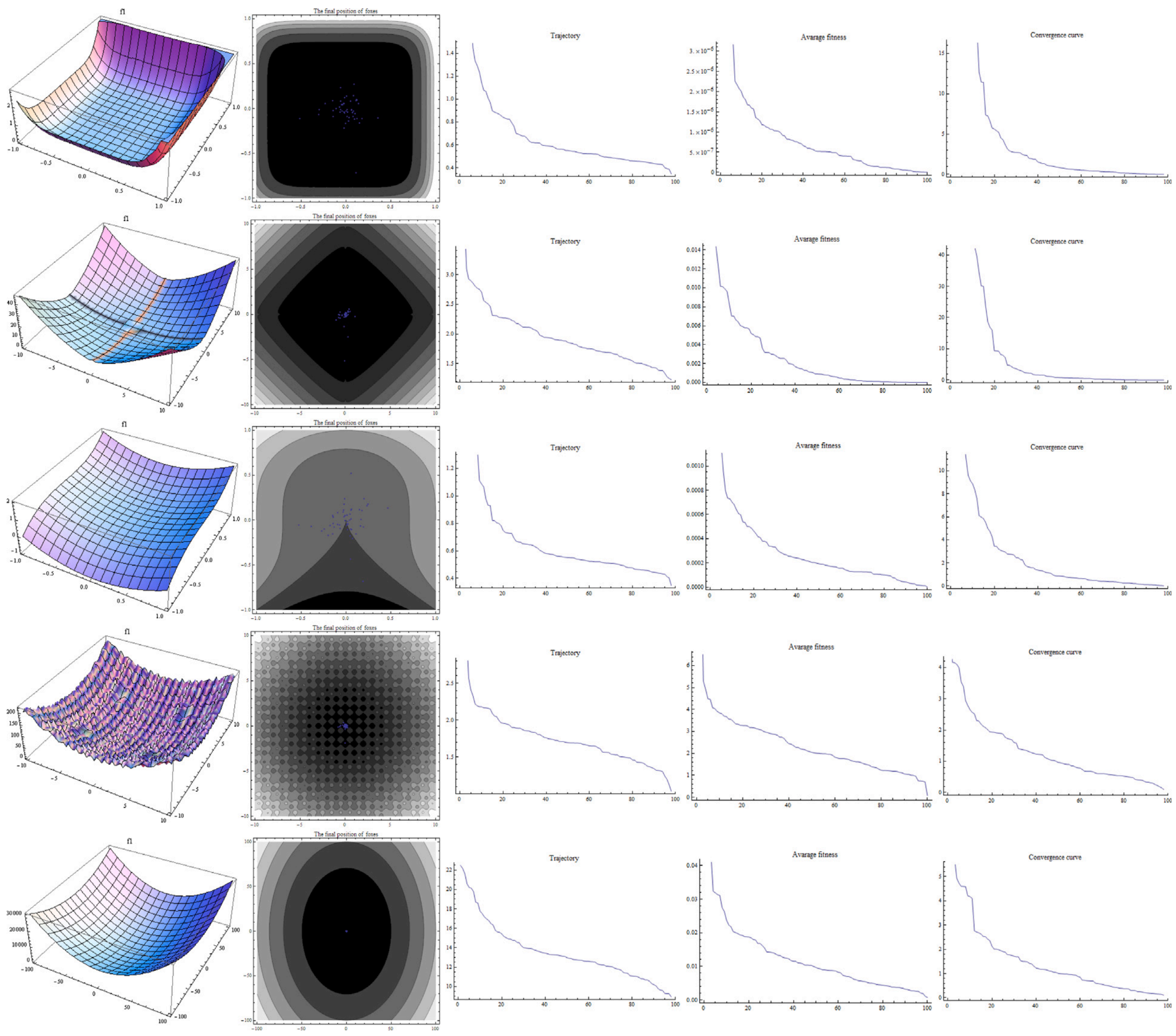


Fig. 4. Sample benchmark tests results for selected test functions from Table 2. In the following rows results for Csendes, Mishra, PowellSum, Rastragin, Rotated Hyper-Ellipsoid: we can see a chart presenting the function, positions of the located points after last iteration, optimization trajectory that represents average movement of individuals, average fitness in the population, and convergence rate in the population.

methods for each of the applied test functions. This examination gave a result of 77.27% of experiments for the proposed RFO to be among the best methods. Other algorithms which were among the best: PSO 13 times, GA 12 times, MFO 10 times, and CSA 9 times, what gave respectively 59%, 54.5%, 45.45%, 40.1%. When it comes to standard deviation presented in Tables 5 and 7 we see that the situation repeats. RFO in comparison to other methods for most of the test functions achieved very low results, which classifies it among the best methods.

3.1. *Conclusions from optimization of test functions*

Proposed model of red fox living and hunting behaviors gave good results in benchmark tests using test functions from Table 2 in 2D and test functions from Table 3 in 20D. Implemented the composition of global and local search methods with reproduction enabled to achieve precise results in comparison to other heuristic algorithms. The proposed meta-heuristic algorithm converges to the maxima/minima of optimized functions with the following iterations. We can see in Figs. 4–5 that RFO after each iteration reaches better convergence, and therefore more individuals in the population are located in the direct surrounding of maxima/minima. For the proposed RFO this convergence was visible in all iterations, however, most spectacular improvements are for the first 20 iterations. That means, in these first 20 iterations RFO is efficiently localizing maxima/minima in the search space while in the following iterations RFO is correcting results of optimization with each relocation of individuals. In Fig. 6 we can see a comparison for 2D test functions and in Fig. 7 a comparison for 20D test functions. We used a radar type chart to present the averaged optimal solutions concerning ideal value, so when the closer to the center the value is located the better is the algorithm. Analyzing these charts we can conclude that there were some test functions that were more difficult for examined algorithms, but also there were some much easier for optimization. Functions $f_2, f_3, f_{14}, f_{16}, f_{18}$ were less problematic, while functions $f_4, f_5, f_8, f_{15}, f_{20}$ made some problems to achieve lower results. For most of the test functions RFO, PSO, GA, CSA, and MFO are the most efficient algorithms. Comparing the results from Fig. 8 we can see that performance of RFO is similar for most of the algorithms. The chart presents a comparison of std. deviations for used functions, so we can compare the algorithms for any significant differences in results. The results are visible in concentration for BA, CSA, FPA, MFO, RFO for 2D functions and MFO, RFO, PSO, GA for 20D functions. These algorithms returned results which std. deviations do not differ much. There are no visible influences i.e. by random factors or evident trends in these results. In this comparison, RFO was among the best methods.

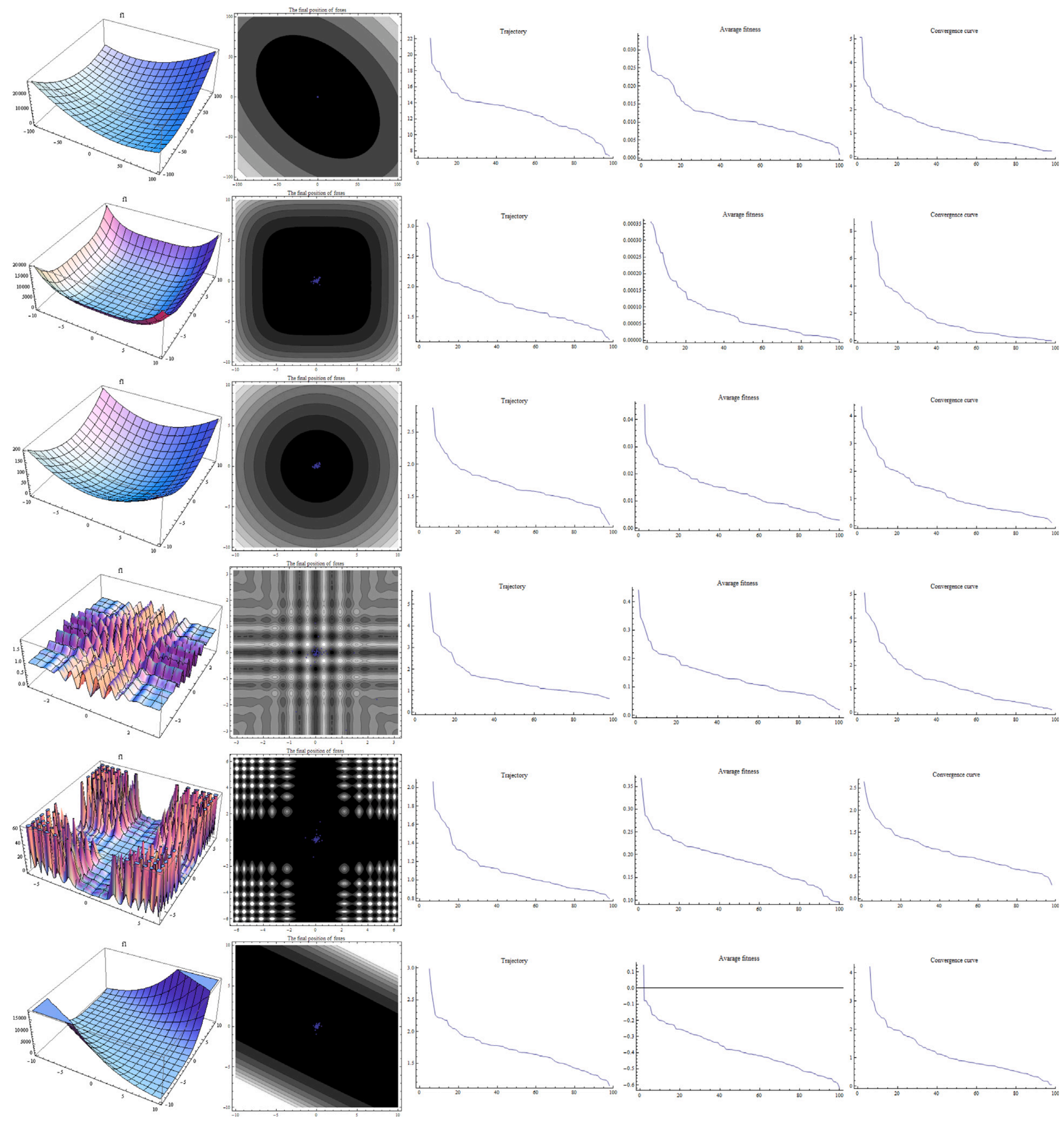


Fig. 5. Sample benchmark tests results for selected test functions from Table 2. In the following rows results for Sargan, Schumer Stegilitz, Sphere, W/Wavy, Yang, Zakharov: we can see a chart presenting the function, positions of the located points after last iteration, optimization trajectory that represents average movement of individuals, average fitness in the population, and convergence rate in the population.

To give additional results for the comparison between examined algorithms we have also calculated statistical tests, which results are presented in Table 8 for 2D results and Table 9 for 20D results. For the ANOVA test, we see that in 2D DA, FPA, GA, and AACA and for 20D ALO reached the highest p-values, which means that for these methods the ANOVA test discovered similarities in results to the proposed RFO. From the Friedmann test, we conclude that the results of PSO are the most related to the results of the proposed RFO. Kruskal–Wallis test confirmed both ANOVA and Friedmann results presenting results of DA, FPA, GA, AACA, FA, and PSO as the most related to the results of RFO in 2D and results of BA, MFO, WWO, GA, SA as the most related to the results of RFO in 20D. Mann–Whitney test presented results of DA, FPA, GA, PSO, AACA as most related to the results of RFO in 2D and results of BA, DA, MFO, WWO, GA, DE, WO as most related to the results of RFO in 20D. We select the most often repeated methods as these which have the most confirmed similarities in results of optimization to the results of the proposed RFO. Therefore we conclude that results of RFO are the most related in 2D to results of DA, FPA, GA, AACA, PSO, and in 20D to results of MFO, GA, PSO, DA. We can see that results of our method are related to the results of well-known heuristic algorithms like PSO, GA, AACA, and the latest nature-inspired algorithms like DA, FPA, MFO.

4. RFO efficiency in classic engineering optimization problems

Optimization algorithms are commonly used in engineering problems to position and balance objects' operation characteristics for

Table 4
Obtained optimal solutions for applied benchmark test functions in 2D in domain $\times 10^{-5}$, averaged for performed benchmark tests.

Function	BA	CSA	DA	FA	FPA	MFO	RFO	WWO
f_1	0.02236	0.01223	0.01238	0.01829	0.01597	0.01646	0.01209	0.02009
f_2	0.00975	0.01999	0.01228	0.01382	**0.00948**	0.01739	0.01157	0.01727
f_3	0.015	0.01166	0.01158	0.01849	0.01272	0.01772	0.011	0.01245
f_4	0.01426	0.01604	0.01384	0.01251	0.01223	0.01402	0.01159	0.01646
f_5	0.01488	0.01544	**0.01152**	0.01762	0.0144	0.01354	0.01474	0.01326
f_6	0.01504	0.01855	0.01651	0.01221	0.01436	0.01653	0.01374	0.0174
f_7	0.01277	0.01378	0.01549	0.01409	0.01977	0.01381	0.01507	0.01324
f_8	0.00961	0.00932	0.0137	0.01317	0.01402	**0.00766**	0.01391	0.01309
f_9	0.01496	**0.01389**	0.01795	0.01845	0.01517	0.01458	0.01417	0.01562
f_{10}	0.0141	0.01429	0.01774	**0.00918**	0.01374	0.01285	0.01188	0.01402
f_{11}	0.01919	0.01078	0.01335	0.01838	**0.00875**	0.01998	0.01912	0.01954
f_{12}	0.01273	0.01558	0.0119	0.01418	0.0128	**0.00962**	0.01299	0.01706
f_{13}	0.01821	0.01708	0.01212	**0.01062**	0.01372	0.01285	0.0143	0.01862
f_{14}	0.01406	0.01259	0.01281	0.01461	0.00972	0.01331	0.00891	**0.00781**
f_{15}	0.01484	0.01358	0.01545	0.01679	0.01917	**0.01339**	0.01429	0.01189
f_{16}	0.01252	0.01488	0.0182	0.01088	0.01404	0.01224	0.01016	0.01598
Function	GA	DE	PSO	AACA	ABCA	SA	ALO	WO
f_1	0.01243	0.01291	**0.01125**	0.01347	0.01403	0.01452	0.01572	0.01638
f_2	0.01103	0.01271	0.01131	0.01315	0.01357	0.01298	0.01733	0.01521
f_3	0.01043	0.01122	**0.01012**	0.01246	0.01274	0.01355	0.01511	0.01495
f_4	0.01206	0.01281	**0.01154**	0.0128	0.01244	0.01411	0.01549	0.01683
f_5	0.01410	0.01501	0.01465	0.01566	0.0151	0.01523	0.01411	0.01622
f_6	0.01421	0.01521	**0.01362**	0.01455	0.01503	0.01677	0.01499	0.01721
f_7	**0.01204**	0.01481	0.01223	0.01455	0.01487	0.01615	0.01733	0.01611
f_8	0.01432	0.01521	0.01431	0.0146	0.01437	0.01522	0.01551	0.01455
f_9	0.01502	0.01572	0.01454	0.01513	0.01503	0.01581	0.01768	0.01722
f_{10}	0.01234	0.01341	0.01201	0.0145	0.01397	0.01294	0.01561	0.01433
f_{11}	0.0192	0.02117	0.01931	0.01877	0.01913	0.02145	0.02211	0.02345
f_{12}	0.01182	0.01234	0.01261	0.01355	0.01362	0.01433	0.01233	0.01576
f_{13}	0.01357	0.01395	0.01322	0.01402	0.01395	0.01621	0.01655	0.01899
f_{14}	0.00935	0.01331	0.00971	0.00987	0.011	0.0113	0.01211	0.01267
f_{15}	0.01457	0.01496	0.01411	0.0153	0.01526	0.01526	0.01452	0.01739
f_{16}	0.01033	0.01135	**0.01008**	0.01188	0.01198	0.01244	0.01352	0.01501

Table 5
Standard deviation for obtained results for applied benchmark test functions in 2D in domain $\times 10^{-5}$.

Function	BA	CSA	DA	FA	FPA	MFO	RFO	WWO
f_1	0.02482	0.02822	0.03452	0.03892	0.02618	**0.02249**	0.02391	0.03127
f_2	0.31562	0.32417	0.37132	0.31251	**0.29137**	0.3103	0.32171	0.33179
f_3	0.00437	0.00456	0.00435	0.00431	0.00367	0.00352	**0.00351**	0.00376
f_4	0.41207	0.41562	0.43129	0.42944	0.43589	**0.40146**	0.43017	0.44126
f_5	0.45136	0.44517	0.46312	0.43217	0.42621	0.42201	0.43152	0.42623
f_6	0.00482	0.00478	0.00483	0.00486	0.00364	0.00352	0.00372	0.00371
f_7	0.00449	0.00451	0.00492	0.00441	0.00353	0.00268	0.00298	0.00310
f_8	0.48139	0.45478	0.40238	0.44056	0.46707	0.43123	**0.40125**	0.48015
f_9	0.21301	0.24019	0.22812	0.21563	0.22059	0.21067	0.22131	0.22831
f_{10}	0.33299	0.41083	0.33166	0.34817	0.35123	0.32573	0.33071	0.41408
f_{11}	0.34922	0.33458	0.36522	0.35128	0.34017	0.31983	0.31982	0.41072
f_{12}	0.00496	0.00431	0.00474	0.00423	0.00373	0.00313	0.00322	0.00357
f_{13}	0.00473	0.00572	0.00533	0.00481	0.00395	0.00365	**0.00317**	0.00377
f_{14}	0.47243	0.50731	0.39455	0.43735	0.39450	0.37561	0.36547	0.41092
f_{15}	0.00463	0.00491	0.00514	0.00498	0.00393	0.00352	**0.00341**	0.00395
f_{16}	0.00498	0.00442	0.00489	0.00435	0.00372	0.00351	0.00344	0.00378
Function	GA	DE	PSO	AACA	ABCA	SA	ALO	WO
f_1	0.02311	0.02914	0.03172	0.02819	0.02782	0.032341	0.03422	0.03432
f_2	0.32873	0.33526	0.32344	0.31552	0.31443	0.31424	0.32857	0.33412
f_3	0.00402	0.00493	0.00413	0.00392	0.00373	0.00443	0.00483	0.00459
f_4	0.41571	0.44135	0.42165	0.42005	0.41334	0.41315	0.45037	0.44416
f_5	0.41312	0.45371	0.42113	0.42416	0.41023	**0.40924**	0.43588	0.42739
f_6	0.00418	0.00436	0.00376	0.00369	**0.00344**	**0.00334**	0.00458	0.00477
f_7	**0.00219**	0.00411	0.00315	0.00442	0.00371	0.00412	0.00458	0.00413
f_8	0.42312	0.41372	0.41529	0.41674	0.42114	0.42153	0.43578	0.44173
f_9	**0.21002**	0.22316	0.22331	0.24215	0.23728	0.23115	0.24311	0.24781
f_{10}	0.33112	0.36112	0.33562	0.34004	0.33352	**0.32378**	0.34351	0.36125
f_{11}	**0.31412**	0.32517	0.33376	0.33265	0.33557	0.31558	0.35112	0.34516
f_{12}	0.00256	0.00355	**0.00236**	0.00433	0.00471	0.00276	0.00342	0.00348
f_{13}	0.00361	0.00352	0.00355	0.00424	0.00377	0.00355	0.00406	0.00427
f_{14}	0.41317	**0.3216**	0.33261	0.42778	0.42516	0.40051	0.41378	0.42031
f_{15}	0.00351	0.00411	0.00411	0.00424	0.00345	0.00386	0.00427	0.00435
f_{16}	**0.00341**	0.00532	0.00369	0.00419	0.00412	0.00378	0.00471	0.00452

Table 6
Obtained optimal solutions for applied benchmark test functions in 20D in domain $\times 10^{-5}$ except for f_{20}, averaged for performed benchmark tests.

Function	BA	CSA	DA	FA	FPA	MFO	RFO	WWO
f_{17}	0.02342	0.01831	0.02231	0.02419	0.01345	0.00866	0.00929	0.0129
f_{18}	0.00718	0.00461	0.00711	0.00711	0.00636	0.00413	0.00371	0.00517
f_{19}	0.01128	0.01166	0.01031	0.01019	0.01112	0.00925	0.009102	0.01157
f_{20}	−24.0519	−24.0523	−24.0523	−24.05212	−24.0522	−24.05242	−24.05241	−24.05226
f_{21}	0.01051	0.00732	0.00823	0.01033	0.0091	0.00721	0.00754	0.01106
f_{22}	0.01225	0.00685	0.01233	0.01041	0.00924	0.00754	0.00704	0.01091
Function	**GA**	**DE**	**PSO**	**AACA**	**ABCA**	**SA**	**ALO**	**WO**
f_{17}	0.00914	0.00913	**0.00811**	0.0191	0.00972	0.00929	0.01931	0.01402
f_{18}	**0.0013**	0.00379	0.00312	0.0051	0.00421	0.00291	0.007014	0.00624
f_{19}	**0.00841**	0.00921	0.00951	0.0101	0.01202	0.01047	0.01119	0.01602
f_{20}	**−24.05261**	−24.05191	−24.05191	−24.0521	−24.0518	−24.05172	−24.05193	−24.0522
f_{21}	0.00759	0.00902	**0.00709**	0.00712	0.00729	0.01082	0.01021	0.01257
f_{22}	0.00779	0.01041	**0.00671**	0.01019	0.00839	0.01431	0.01349	0.01471

Table 7
Standard deviation for obtained results for applied benchmark test functions in 20D in domain $\times 10^{-5}$ except for f_{20}.

Function	BA	CSA	DA	FA	FPA	MFO	RFO	WWO
f_{17}	0.01312	0.01498	0.02317	0.04113	0.03518	0.00108	0.00193	0.03274
f_{18}	0.00218	0.00142	0.03621	0.03831	0.02418	0.00181	0.00287	0.03216
f_{19}	0.01576	0.01887	0.03451	0.02318	0.03661	0.00126	0.00116	0.03413
f_{20}	0.03198	0.01287	0.04197	0.05128	0.05128	0.00288	**0.00246**	0.04158
f_{21}	0.02186	0.00277	0.00233	0.02817	0.02319	0.00319	0.00278	0.02318
f_{22}	0.01971	0.00228	0.03574	0.03447	0.03118	0.00328	0.00299	0.02218
Function	**GA**	**DE**	**PSO**	**AACA**	**ABCA**	**SA**	**ALO**	**WO**
f_{17}	0.00116	0.00351	**0.00101**	0.01012	0.01021	0.02319	0.02245	0.03314
f_{18}	**0.00101**	0.00419	0.00210	0.01131	0.01310	0.00311	0.02311	0.03415
f_{19}	**0.00110**	0.02336	0.00121	0.01245	0.00921	0.02316	0.02231	0.02812
f_{20}	0.03182	0.02181	0.03281	0.02973	0.03121	0.05314	0.03217	0.05346
f_{21}	0.00129	0.02133	0.00248	**0.00103**	0.00224	0.02615	0.03181	0.02112
f_{22}	0.00131	0.01911	**0.00111**	0.01013	0.00178	0.01141	0.02314	0.02671

Table 8
Results for statistical tests comparison of the 2D results for the significance level 0.05.

ANOVA	BA	CSA	DA	FA	FPA	MFO	RFO	WWO
F-ratio	0.13084	0.58499	0.00087	0.66849	0.03078	0.49447	–	0.51571
p-value	0.71879	0.447265	0.976635	0.416709	0.861301	0.484573	–	0.475377
	GA	**DE**	**PSO**	**AACA**	**ABCA**	**SA**	**ALO**	**WO**
F-ratio	0.02913	0.50707	0.12202	0.05674	0.07957	0.81557	1.15382	1.54777
p-value	0.865045	0.479081	0.728041	0.812516	0.778824	0.369974	0.286913	0.218148
Friedman	**BA**	**CSA**	**DA**	**FA**	**FPA**	**MFO**	**RFO**	**WWO**
χ_r^2-ratio	1.125	1.125	1.125	1.125	2	1.125	–	0.5
p-value	0.28884	0.28884	0.28884	0.28885	0.1573	0.28885	–	0.4795
	GA	**DE**	**PSO**	**AACA**	**ABCA**	**SA**	**ALO**	**WO**
χ_r^2-ratio	1.125	4.5	0.125	0.5	3.125	8	4.5	8
p-value	0.28885	0.03389	0.72367	0.4795	0.0771	0.00468	0.03389	0.00468
Kruskal–Wallis	**BA**	**CSA**	**DA**	**FA**	**FPA**	**MFO**	**RFO**	**WWO**
H-ratio	0.1623	0.4068	0.0218	0.349	0.0379	0.2603	–	0.4689
p-value	0.68708	0.52361	0.88258	0.55466	0.84563	0.60989	–	0.49348
	GA	**DE**	**PSO**	**AACA**	**ABCA**	**SA**	**ALO**	**WO**
H-ratio	0.0305	0.349	0	0.0055	0.2146	0.8093	1.3334	1.8029
p-value	0.86143	0.55466	1	0.94113	0.64319	0.36832	0.2482	0.17936
Mann–Whitney	**BA**	**CSA**	**DA**	**FA**	**FPA**	**MFO**	**RFO**	**WWO**
Z-ratio	−0.3961	−0.63108	−0.14099	−0.58408	−0.18798	−0.50352	–	−0.67807
U-ratio	482	464.5	501	468	497.5	474	–	461
p-value	0.68916	0.5287	0.88866	0.56192	0.8493	0.61708	–	0.4965
	GA	**DE**	**PSO**	**AACA**	**ABCA**	**SA**	**ALO**	**WO**
Z-ratio	0.16784	−0.58408	0.00671	−0.06714	−0.45652	−0.89291	−1.14802	−1.336
U-ratio	499	468	512	506.5	477.5	445	426	412
p-value	0.86502	0.56192	0.99202	0.9442	0.64552	0.37346	0.25014	0.18024

minimal work cost. Therefore in this section, we present optimization results for some classical examples, where RFO was compared to other meta-heuristics and golden standard results from analytical methods.

4.1. Three bar truss problem

In this optimization, we consider a steel construction that is acting with loadings according to the scheme presented in Fig. 9. In the model,

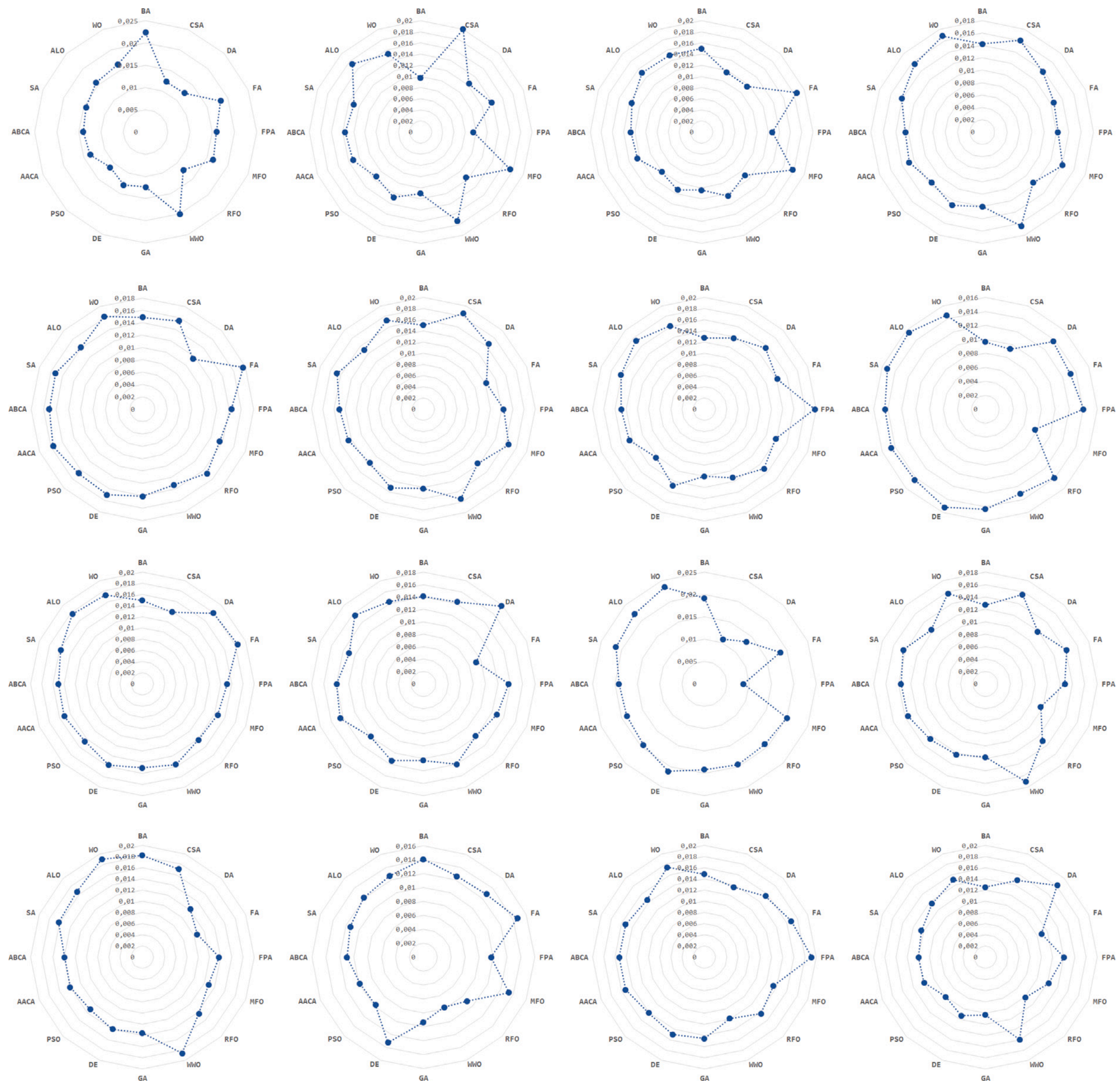


Fig. 6. Comparison of the benchmark tests results for test functions in 2D in domain $\times 10^{-5}$ from Table 4, where in the first row we see results for functions f_1-f_4, in the second row for functions f_5-f_8, in the third row for functions f_9-f_{12} and in the last row for functions f_{13}-f_{16}.

we assume that construction topology consists of elongated elements that are composed in the shape of a triangle to hold the loadings. Elements x_1 and x_3 are of the same length since these are sides of the triangle, while element x_2 is a central element placed in the middle of this construction. To the top of the triangle, we can attach loadings that cause the existence of stress constraints g_1, g_2, and g_3 that guarantee bounding and functionality. The optimization of these construction variables is a continuous mathematical problem and cross-sectional design physical problem. To solve the problem we must optimize a total weight of this construction

$$f_w(\overline{x}) = l(2\sqrt{2}x_1 + x_2) \tag{12}$$

calculating lengths of elements for minimal weight of the structure, where we assume $x_1, x_2 \in \langle 0, 1\rangle$, for which stress constraints are modeled using the following equations

$$\begin{cases} g_1(\overline{x}) = \dfrac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \leq 0 \\ g_2(\overline{x}) = \dfrac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \leq 0 \\ g_3(\overline{x}) = \dfrac{1}{\sqrt{2}x_2 + x_1} P - \sigma \leq 0 \end{cases}, \tag{13}$$

where other construction parameters are l = 100 cm, P = 2 kN/cm^2, and σ = 2 kN/cm^2. Optimization results from different meta-heuristic algorithms are presented in Table 14.

4.2. Welded beam design problem

In this optimization we consider a construction composed of two components — beam and weld presented in Fig. 9. Model variables represent dimensions of welded part, which during optimization shall be optimized to minimize the cost of the weld and material of the beam,

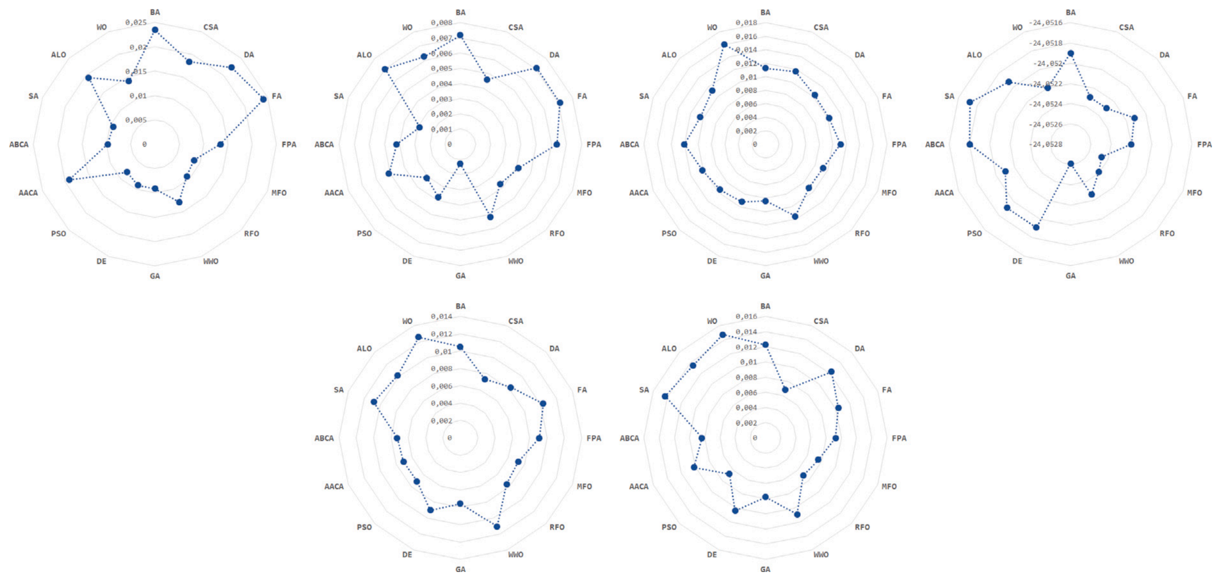


Fig. 7. Comparison of the benchmark tests results for test functions in 20D in domain $\times 10^{-5}$ from Table 6, where in the first row we see results for functions f_{17}-f_{20} and in the last row for functions f_{21}-f_{22}.

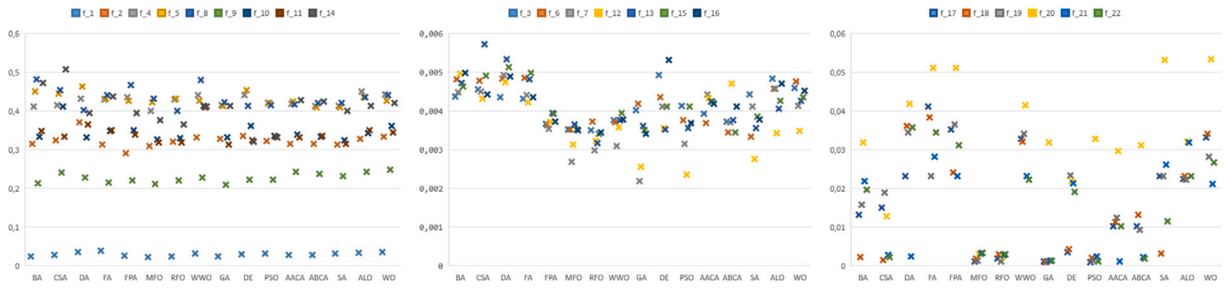
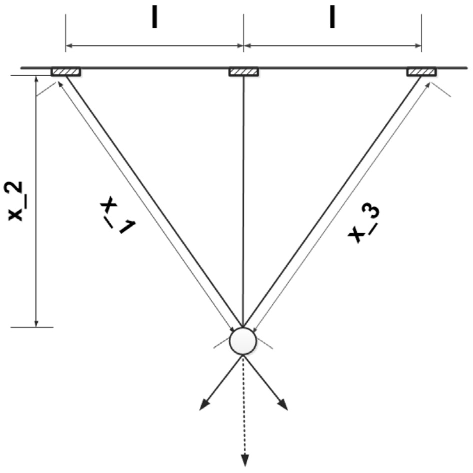


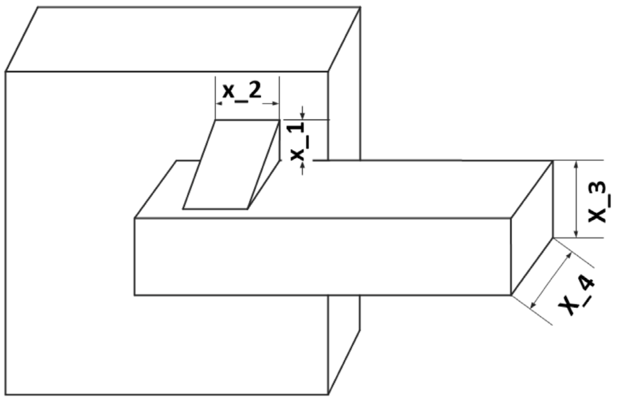
Fig. 8. Benchmark tests results for test functions in 2D in domain $\times 10^{-5}$ from Table 5 on the left and middle, while for test functions in 20D from Table 7 on the right.

Table 9
Results for statistical tests comparison of the 20D results for the significance level 0.05.

ANOVA	BA	CSA	DA	FA	FPA	MFO	RFO	WWO
F-ratio p-value	0.99631 0.329058	1.3665 0.254925	1.09048 0.307705	0.25828 0.616361	1.56846 0.223581	0.5485 0.466762	– –	0.5059 0.484393
	GA	DE	PSO	AACA	ABCA	SA	ALO	WO
F-ratio p-value	0.59395 0.449094	0.49428 0.489399	0.58815 0.451289	2.15775 0.156008	1.56519 0.224049	0.41876 0.524253	0.00173 0.96716	1.02899 0.321423
Friedman	BA	CSA	DA	FA	FPA	MFO	RFO	WWO
χ_r^2-ratio p-value	0.3333 0.5637	0.3333 0.5638	0.3333 0.5639	0.3333 0.564	0 1	0 1	– –	0 1
	GA	DE	PSO	AACA	ABCA	SA	ALO	WO
χ_r^2-ratio p-value	0 1	0 1	3 0.08326	1.3333 0.24821	0 1	0.0833 0.77283	0.3333 0.5637	0.3333 0.5638
Kruskal–Wallis	BA	CSA	DA	FA	FPA	MFO	RFO	WWO
H-ratio p-value	0.0033 0.95396	0.0833 0.77283	0 1	0.0533 0.81736	0.0833 0.77283	0.0075 0.93099	– –	0.0033 0.95396
	GA	DE	PSO	AACA	ABCA	SA	ALO	WO
H-ratio p-value	0.0033 0.95396	0 1	0.0533 0.81736	0.4033 0.52537	0.03 0.86249	0.0075 0.93099	0.1633 0.68611	0 1
Mann–Whitney	BA	CSA	DA	FA	FPA	MFO	RFO	WWO
Z-ratio U-ratio p-value	−0.02887 71 0.97606	0.25981 67 0.79486	0.02887 72 0.97606	−0.20207 68 0.84148	0.25981 67 0.79486	0.05774 70.5 0.95216	– – –	0.02887 71 0.97606
	GA	DE	PSO	AACA	ABCA	SA	ALO	WO
Z-ratio U-ratio p-value	0.02887 71 0.97606	0.02887 72 0.97606	0.20207 68 0.84148	0.60622 61 0.54186	0.14434 69 0.88866	−0.05774 70.5 0.95216	−0.37528 65 0.70394	0.02887 72 0.97606

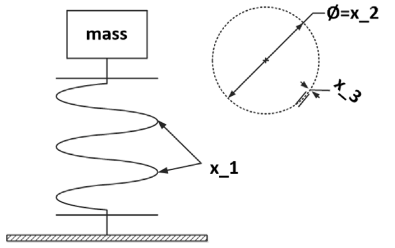


(a) model of tree bar truss

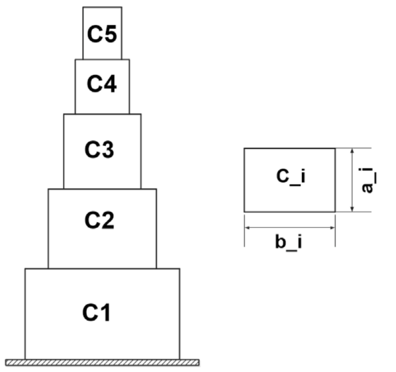


(b) model of welded beam

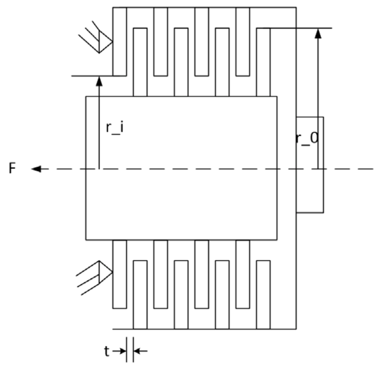
Fig. 9. Classic engineering optimization problems: (a) Scheme of tree bar truss model used to solve optimal design problem, (b) Scheme of welded beam model used to solve optimal design problem.



(a) model of compression spring

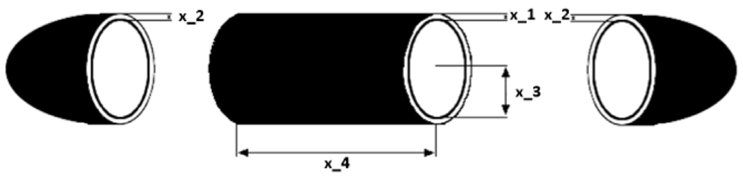


(b) model of cantilever beam

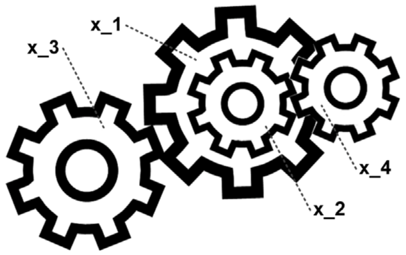


(c) model of multiplate clutch

Fig. 10. Classic engineering optimization problems: (a) Scheme of compression spring model used to solve optimal design problem, (b) Scheme of cantilever beam model used to solve optimal design problem, (c) Scheme of multiplate clutch break to solve optimal design problem.



(a) model of pressure vessel



(b) model of gear

Fig. 11. Classic engineering optimization problems: (a) Scheme of pressure vessel used to solve optimal design problem, (b) Scheme of gear train problem model used to solve optimal design problem.

however during optimization we must avoid deflection of the beam. To solve this optimization problem we must find optimum to optimize a total weight of this construction

$$f_v(\vec{x}) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14 + x_2), \tag{14}$$

where $x_1, x_4 \in \langle 0.1, 2\rangle$ and $x_2, x_3 \in \langle 0.1, 10\rangle$ and $x_1, \ldots, x_4$ are physical quantities of the weld like height, length, etc.

Additionally to the model of welded beam design, model functions d_1–d_7 should be considered for the optimization procedure as constrained conditions

$$\begin{cases} d_1(\vec{x}) = \tau(\vec{x}) - \tau_{max} \le 0 \\ d_2(\vec{x}) = \sigma(\vec{x}) - \sigma_{max} \le 0 \\ d_3(\vec{x}) = x_1 - x_4 \le 0 \\ d_4(\vec{x}) = 1.10471x_1^2 + 0.04811x_3x_4(14 + x_2) - 5 \le 0 \\ d_5(\vec{x}) = 0.125 - x_1 \le 0 \\ d_6(\vec{x}) = \delta(\vec{x}) - \delta_{max} \le 0 \\ d_7(\vec{x}) = P - P_c(\vec{x}) \le 0 \end{cases}, \tag{15}$$

which coefficients we model according to equations

$$\begin{cases} \tau(\vec{x}) = \sqrt{\tau_1^2 + 2\tau_1\tau_2\frac{x_2}{2R} + \tau_2^2} \\ \tau_1 = \frac{P}{\sqrt{2}x_1x_2}, \tau_2 = \frac{MR}{J} \\ R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1+x_3}{2}\right)^2}, M = P\left(L + \frac{x_2}{2}\right), J = 2\left(\sqrt{2}x_1x_2\left(\frac{x_2^2}{4} + \left(\frac{x_1+x_2}{2}\right)^2\right)\right) \\ \sigma(\vec{x}) = \frac{6PL}{x_4x_3^2}, \delta(\vec{x}) = \frac{6PL^3}{Ex_3^2x_4} \\ P_c(\vec{x}) = \frac{4.013E\sqrt{\left(\frac{x_3^2x_4^6}{36}\right)}}{L^2}\left(1 - \frac{x_3\sqrt{\frac{E}{4G}}}{2L}\right) \end{cases}. \tag{16}$$

In performed optimization we have used model parameters values: $P = 6000$ [lb], $L = 14$ [in.], $E = 30 \times 10^6$ [psi], $G = 12 \times 10^6$ [psi], $\tau_{max} = 13600$ [psi], $\sigma_{max} = 30000$ [psi] and $\gamma_{max} = 0.25$ [in.] The golden standard analytical results from the literature are presented in Table 10 while the obtained results from different meta-heuristic algorithms are presented in Table 15.

4.3. Compression spring problem

In this optimization, we consider a spring construction that is acting with tension/compression from the load according to the scheme presented in Fig. 10. The objective is to minimize spring volume weight that reacts with tension/compression caused by the load. Assumed design variables represent number of active spring coils x_1, winding diameter x_2, and wire diameter x_3. To solve the optimization problem we must optimize a volume of the spring defined according to

$$f_v(\vec{x}) = (x_3 + 2)x_2x_1^2, \tag{17}$$

calculating lengths of elements for minimal volume of the spring, where we assume $x_1 \in \langle 0.05, 2\rangle$, $x_2 \in \langle 0.25, 1.3\rangle$, $x_3 \in \langle 2, 15\rangle$, for which stress constraints are modeled using the following equations

$$\begin{cases} g_1(\vec{x}) = 1 - \frac{x_2^3x_3}{71785x_1^4} \leq 0 \\ g_2(\vec{x}) = \frac{4x_2^2 - x_1x_2}{12566(x_2x_1^2 - x_1^4)} - \frac{1}{5108x_1^2} \leq 0 \\ g_3(\vec{x}) = 1 - \frac{140.45x_1}{x_2^2x_3} \leq 0 \\ g_4(\vec{x}) = \frac{x_1 + x_2}{1.5} - 1 \leq 0 \end{cases} \tag{18}$$

The golden standard analytical results from the literature are presented in Table 11 and obtained results from different meta-heuristic algorithms are presented in Table 16.

4.4. Cantilever beam design problem

In this optimization, we consider a cantilever construction that is composed of 5 compartments presented in Fig. 10. The objective is to minimize the weight of all the sections, assumed that design variables represent the width and height dimensions of each compartment. To solve the optimization problem we must optimize total weight defined according to

$$f_w(\vec{c}) = 0.6224\sum_{i=1}^{5} x_i, \tag{19}$$

where $c_1, c_2, c_3, c_4, c_5 \in \langle 0.01, 100\rangle$ and the limit state function is defined as

$$g(\vec{c}) = \frac{61}{c_1^3} + \frac{27}{c_2^3} + \frac{19}{c_3^3} + \frac{7}{c_4^3} + \frac{1}{c_5^3} - 1 \leq 0. \tag{20}$$

The golden standard analytical results from the literature are presented in Table 12 and obtained results from different meta-heuristic algorithms are presented in Table 17.

4.5. Pressure vessel design problem

In this optimization we consider a gas storage container in a shape of compressed air tank with hemispherical cylinders at both ends as shown in Fig. 11. This kind of container can be used for liquid gas that must be keep under pressure to maintain the gas properties. In general this object must be optimized in construction weight to keep the maximum pressure of 1000 [psi] for the minimum volume of 750 [ft^3], what we model as

$$g_1(\vec{x}) = 0.6224x_1x_3x_4 + 1.7781x_3^2x_2 + 3.1661x_4x_1^2 + 19.84x_3x_1^2 \tag{21}$$

where x_1 is coating thickness of cylinder, x_2 is coating thickness of hemispherical cover, x_3 is radius of the cylinder without the shell, x_4 is length of the cylinder, and $x_1, x_2 \in \langle 0, 99\rangle$ and $x_3, x_4 \in \langle 10, 200\rangle$. Additionally we assume conditions c_1–c_4 for the dimensions of the cylinder

$$\begin{cases} c_1(\vec{x}) = -x_1 + 0.0193x_3 \leq 0 \\ c_2(\vec{x}) = 0.00954x_3 - x_3 \leq 0 \\ c_3(\vec{x}) = 129600 - \frac{4}{3}\pi x_3^3 - \pi x_3^2x_4 \leq 0 \\ c_4(\vec{x}) = x_2 - 240 \leq 0 \end{cases}. \tag{22}$$

The golden standard analytical results from the literature are presented in Table 13 and obtained results from different meta-heuristic algorithms are presented in Table 18.

4.6. Gear train problem

In this optimization we consider a construction of mechanical system in which four gear wheels are run, see Fig. 11. Calculations are done to optimize the cost of work by minimizing the gear ratio modeled by

$$f_w(\vec{x}) = \left(\frac{1}{6.931} - \frac{x_2x_3}{x_1x_4}\right)^2, \tag{23}$$

variables $x_1, x_2, x_3, x_4 \in \langle 12, 60\rangle$ are dimensions of the following gears in the system in Fig. 11. Obtained results from different meta-heuristic algorithms are presented in Table 19.

4.7. Multi-plate disc clutch brake

In this optimization, we consider finding the lowest possible weight for a multi-plate disc clutch brake (Heidari et al., 2019; Savsani & Savsani, 2016), depending on five different variables. Applied optimization model is presented in Fig. 10. The variables are the actuating force F, the outer r_i, inner radius r_o, the number of friction surfaces Z, and the thickness of the brake discs marked as t. The problem is defined as a function $f_{mp}(\cdot)$ dependent on eight conditions modeled for geometry and operating requirements. It is defined as

$$f_{mp}(\vec{x}) = \pi(r_o^2 - r_i^2)t(Z + 1)\rho, \tag{24}$$

with the following constraints

$$\begin{cases} g_1(\vec{x}) = r_0 - r_i - \Delta r \geq 0 \\ g_2(\vec{x}) = l_{max} - (Z + 1)(t + \delta) \geq 0 \\ g_3(\vec{x}) = P_{max} - P_{rz} \geq 0 \\ g_4(\vec{x}) = P_{max}v_{sr,max} - P_{rz}v_{sr} \geq 0 \\ g_5(\vec{x}) = v_{sr,max} - v_{sr} \geq 0 \\ g_6(\vec{x}) = T_{max} - T \geq 0 \\ g_7(\vec{x}) = M_h - sM_s \geq 0 \\ g_8(\vec{x}) = T \geq 0 \end{cases}, \tag{25}$$

Table 10
Literature gold standard results from classic/analytical approaches in solving welded beam optimization problem.

	Method	x_1	x_2	x_3	x_4	Optimum
Siddall (1972)	Siddall	0.2444	6.2189	8.2915	0.2444	2.38154
Ragsdell and Phillips (1976)	Ragsdell	0.2455	6.1960	8.2730	0.2455	2.38594
Ragsdell and Phillips (1976)	Random	0.4575	4.7313	5.0853	0.6600	4.11856
Ragsdell and Phillips (1976)	Simplex	0.2792	5.6256	7.7512	0.2796	2.53073
Ragsdell and Phillips (1976)	David	0.2434	6.2552	8.2915	0.2444	2.38411
Ragsdell and Phillips (1976)	Approx	0.2444	6.2189	8.2915	0.2444	2.38154

Table 11
Literature gold standard results from classic/analytical approaches in solving compression spring optimization problem.

	Method	x_1	x_2	x_3	Optimum
Arora (2004)	Constraint correction	14.25	0.3159	0.05	0.0128334
Belegundu (1983)	Mathematical optimization	9.1854	0.39918	0.053396	0.0127303

Table 12
Literature gold standard results from classic/analytical approaches in solving cantilever beam design optimization problem.

	Method	c_1	c_2	c_3	c_4	c_5	Optimum
Chickermane and Gea (1996)	Generalized Convex Approximation	6.01	5.3	4.49	3.49	2.15	13.3442
Chickermane and Gea (1996)	Moving Asymptotes	6.01	5.3	4.49	3.49	2.15	13.3442

Table 13
Literature gold standard results from classic/analytical approaches in solving pressure vessel optimization problem.

	Method	x_1	x_2	x_3	x_4	Optimum
Kannan and Kramer (1994)	Lagrange multiplier	1.125	0.625	58.291	43.69	7198.0428
Sandgren (1990)	Branch-bound	1.125	0.625	47.7	117.701	8129.1036

where $M_h = 0.66\mu F Z \frac{r_o^3-r_i^2}{r_o^2-r_i^3}$, $P_{rz} = \frac{F}{\pi(r_o^2-r_i^2)}$, $\upsilon_{sr} = \frac{2\pi n(r_o^3-r_i^3)}{90(r_o^2-r_i^2)}$, $T = \frac{I_z \prod n}{30(M_h+M_f)}$ $\Delta r = 20$ [mm], $I_z = 55$ [kgmm^2], $P_{max} = 1$ [MPa], $F_{max} = 1000$ [N], $T_{max} = 15$ [s], $\mu = 0.5$, $s = 1.5$, $M_s = 40$ [Nm], $M_f = 3$ [Nm], $n = 250$ [rpm], $\upsilon_{sr,max} = 10$ [m/s], $l_{max} = 30$ [mm], $r_{i,min} = 60$, $r_{i,max} = 80$, $r_{o,max} = 110$, $t_{min} = 1.5$, $t_{max} = 3$, $F_{min} = 600$, $F_{max} = 1000$, $Z_{min} = 2$, $Z_{max} = 9$. Obtained results are presented in Table 20.

4.8. Conclusions from the optimization of engineering problems

Results of optimization for technical problems are presented in Tables 14–20. For the tree bar truss problem GA achieved the lowest value of the cost function, after were FPA, RFO, and AACA. For the welded beam problem CSA reached the lowest value of the cost function after were DA, MFO, and PSO. If we compare optimization results from Table 15 to literature results from Table 10 we can see that examined heuristics returned results most closely similar to Siddall, Ragsdell, and David methods. For the compression spring problem, PSO and MFO returned the lowest value of the cost function, after were CSA, RFO, AACA, ALO, SA, DA, FPA which all returned values very similar. Comparing results from Table 16 to the literature results from Table 11 we see that for most of the examined heuristics results are very similar to these from Mathematical optimization and Constraint correction. Mainly the results differ on one model variable, however this difference is not so big. Moreover, the cost function value is very similar between literature standards and examined heuristics.

For the cantilever beam design problem the lowest value of cost function was found by RFO, after were FPA, MFO, CSA. However, in this experiment, all the heuristics returned very similar results so we cannot easily define the best algorithm. If we compare them to the literature standards from Table 12 we can see that all heuristics returned results very similar to presented Generalized Convex Approximation and Moving Asymptotes. For the pressure vessel problem, the lowest value of cost function was found by RFO, after were CSA, PSO, and DA. Comparing optimization results from Table 18 to the literature standards from Table 13 we see that all heuristics reached lower cost function values than presented Lagrange multiplier and branch-bound methods. For the gear train problem, the best result was returned by PSO after were MFO, WWO, AACA, ABCA, and SA. In Table 20, the average results for the multi-plate clutch brake problem are presented. The best results were achieved by RFO, and WO — the difference between them was just 0.00101 which is very small. It must be noted that in general, the obtained result for all heuristic is very different. BA, MFO, WWO has reached a results which differ significantly from the best ones.

In these experiments proposed RFO was 5 times among the best methods, where our algorithm was the best 3 times. Among other examined heuristics MFO and CSA were the best since both 4 times returned the results with very good cost function value, and CSA won the competition for the welded beam problem. Classical heuristics PSO and AACA were 3 times among the best algorithms, where PSO was the best for gear train problem. So in general we cannot say the proposed RFO was clearly the best algorithm among examined heuristics. On the other hand, RFO succeeds in a similar number of experiments as well known classical algorithms PSO and AACA, which is a very good result.

5. Discussion on results and comparisons

From benchmark tests, we can draw conclusions. We have seen that the proposed RFO was efficient both for test functions and engineering problems. From the results, we cannot say that our method was the best in every competition. On the other hand from the analysis of the results, we can say that our method was many times among the best, moreover RFO has won in many cases.

In RFO we have proposed a composition of global search phase and local search phase based on the fox hunting model. The first one was introduced to efficiently search in entire model space, while the second one was defined to increase the precision of calculations. Similar two-phase models were introduced in MFO, CSA, and DA. These heuristics also achieved very good results in benchmark tests. In RFO the information was exchanged between individuals what made it converge to the optimum fast in the initial phase of the algorithm. This was visible for test functions, where we presented charts of convergence and trajectories of optimization. Trajectory represents an average movement of individuals in the population during the following iterations

Table 14
Optimal construction variables for the three-bar truss problem.

	BA	CSA	DA	FA	FPA	MFO	RFO	WWO
x_1	0.76255	0.8207	0.77312	0.78489	0.72599	0.76525	0.75356	0.82511
x_2	0.69453	0.5215	0.76265	0.66347	0.62842	0.58935	0.55373	0.52519
$f_w(\overline{x})$	285.13471	284.27901	294.93635	288.34741	268.18298	275.38038	268.51195	285.89535
$g_1(\overline{x})$	−0.11546	−0.13976	−0.16648	−0.14561	−0.00326	−0.06773	−0.02220	−0.15022
$g_2(\overline{x})$	−1.26175	−1.42328	−1.24659	−1.30625	−1.24187	−1.31873	−1.32373	−1.42586
$g_3(\overline{x})$	−0.85371	−0.71647	−0.91989	−0.83935	−0.76138	−0.74899	−0.69846	−0.72436
	GA	DE	PSO	AACA	ABCA	SA	ALO	WO
x_1	0.74313	0.79234	0.73351	0.75332	0.73891	0.81023	0.79458	0.83211
x_2	0.56021	0.57312	0.62198	0.55892	0.78415	0.63428	0.63208	0.63219
$f_w(\overline{x})$	**266.20990**	281.41959	269.66595	268.96307	287.41030	292.59565	287.94916	298.57524
$g_1(\overline{x})$	−0.00303	−0.11402	−0.01677	−0.02477	−0.10549	−0.18004	−0.14922	−0.21891
$g_2(\overline{x})$	−1.30564	−1.36180	−1.25660	−1.32031	−1.18782	−1.35152	−1.33372	−1.37755
$g_3(\overline{x})$	−0.69739	−0.75222	−0.76016	−0.70445	−0.91767	−0.82851	−0.81549	−0.84135

Table 15
The obtained optimal solution for the welded beam problem.

	BA	CSA	DA	FA	FPA	MFO	RFO	WWO
x_1	0.21834	0.19312	0.20485	0.23141	0.20158	0.21015	0.21846	0.22214
x_2	4.01215	3.85691	3.84712	4.04761	3.91024	3.49261	3.51024	3.67812
x_3	8.97261	8.89429	8.94681	8.84958	8.56842	8.98423	8.87254	8.84965
x_4	0.23257	0.21453	0.21431	0.24053	0.22641	0.21964	0.22491	0.23489
$f_v(\overline{x})$	2.01960	**1.79813**	1.82466	2.08763	1.84712	1.83105	1.86612	1.96842
$d_1(\overline{x})$	−75.7536	−89.0708	−83.8156	−71.4389	−86.0950	−88.3606	−85.1728	−80.6412
$d_2(\overline{x})$	−269.177	−296.975	−293.799	−267.556	−303.202	−284.287	−284.659	−273.976
$d_3(\overline{x})$	−0.01423	−0.02141	−0.00946	−0.00912	−0.02483	−0.00949	−0.00645	−0.01275
$d_4(\overline{x})$	−3.13902	−3.31956	−3.30732	−3.09265	−3.28350	−3.29054	−3.26621	−3.17756
$d_5(\overline{x})$	−0.09334	−0.06812	−0.07985	−0.10641	−0.07658	−0.08515	−0.09346	−0.09714
$d_6(\overline{x})$	−0.00175	−0.00194	−0.00191	−0.00174	−0.00198	−0.00185	−0.00185	−0.00178
$d_7(\overline{x})$	−2627.537	−732.350	−737.922	−3456.995	−1717.598	−1273.296	−1745.002	−2807.270
	GA	DE	PSO	AACA	ABCA	SA	ALO	WO
x_1	0.22958	0.23485	0.22485	0.22416	0.23152	0.22845	0.23551	0.23486
x_2	3.51231	3.59865	3.48015	3.51512	3.52481	3.51842	3.51482	3.52146
x_3	8.71948	8.72145	8.59438	8.71412	8.72145	8.65014	8.63148	8.62147
x_4	0.23102	0.24812	0.22846	0.23151	0.23958	0.23846	0.24512	0.24016
$f_v(\overline{x})$	1.90165	2.05143	1.84559	1.89509	1.97040	1.94132	1.99817	1.95994
$d_1(\overline{x})$	−81.7246	−78.2460	−84.7883	−83.7013	−80.7788	−82.3764	−80.0349	−80.1843
$d_2(\overline{x})$	−286.945	−267.049	−298.669	−286.690	−276.568	−282.467	−275.982	−282.336
$d_3(\overline{x})$	−0.00144	−0.01327	−0.00361	−0.00735	−0.00806	−0.01001	−0.00961	−0.0053
$d_4(\overline{x})$	−3.24462	−3.10690	−3.29292	−3.24451	−3.17910	−3.20386	−3.15591	−3.19369
$d_5(\overline{x})$	−0.10458	−0.10985	−0.09985	−0.09916	−0.10652	−0.10345	−0.11051	−0.10986
$d_6(\overline{x})$	−0.00187	−0.00174	−0.00195	−0.00187	−0.00180	−0.00184	−0.00180	−0.00184
$d_7(\overline{x})$	−2296.283	−4279.837	−1945.469	−2345.728	−3254.492	−3074.872	−3842.262	−3249.516

Table 16
Optimal construction variables for the compression spring problem.

	BA	CSA	DA	FA	FPA	MFO	RFO	WWO
x_1	0.05841	0.05143	0.05201	0.05424	0.05211	0.05224	0.05189	0.05257
x_2	0.41846	0.35412	0.36242	0.39212	0.37125	0.36951	0.36142	0.36994
x_3	12.01451	11.91451	12.01431	11.88131	11.81432	10.78154	11.58436	11.37447
$f_v(\overline{x})$	0.02000	0.01303	0.01373	0.01601	0.01392	0.01288	0.01321	0.01367
$g_1(\overline{x})$	−0.05362	−0.05348	−0.08881	−0.15294	−0.14206	−0.01744	−0.05084	−0.05036
$g_2(\overline{x})$	−0.01939	−0.03263	−0.03094	−0.02527	−0.02979	−0.02985	−0.03120	−0.02943
$g_3(\overline{x})$	−2.89937	−3.83460	−3.62898	−3.17002	−3.49470	−3.98415	−3.81624	−3.74314
$g_4(\overline{x})$	−0.68208	−0.72963	−0.72371	−0.70242	−0.71776	−0.71883	−0.72446	−0.71832
	GA	DE	PSO	AACA	ABCA	SA	ALO	WO
x_1	0.05312	0.05211	0.05071	0.05214	0.05121	0.05243	0.05203	0.05241
x_2	0.36912	0.368421	0.35924	0.35854	0.35924	0.36121	0.36124	0.36221
x_3	11.72141	11.69421	11.62233	11.63542	11.71252	11.6161	11.70211	11.69425
$f_v(\overline{x})$	0.01429	0.01370	**0.01258**	0.01329	0.01291	0.01351	0.01339	0.01362
$g_1(\overline{x})$	−0.03137	−0.10480	−0.13511	−0.01082	−0.09989	−0.00922	−0.04858	−0.02603
$g_2(\overline{x})$	−0.02892	−0.03012	−0.03292	−0.03124	−0.03229	−0.03059	−0.03106	−0.03050
$g_3(\overline{x})$	−3.67159	−3.61087	−3.74845	−3.89593	−3.75835	−3.85871	−3.78541	−3.79780
$g_4(\overline{x})$	−0.71850	−0.71964	−0.7267	−0.72621	−0.72636	−0.72424	−0.72448	−0.72358

of the algorithm, while average fitness and convergence rates in the population were defined in (10) and (11) respectively. These measures showed how RFO works in the search space to find optimal solutions. In the first 20–30 iterations changes in values were significant, while after these RFO localized most of the individuals in the surrounding of the optimum and started to work on the precision of the final results. An additional advantage was in the proposed mechanism of selection, where we modeled herd behaviors of foxes to simulate the selection

Table 17
Optimal construction variables for the cantilever beam problem.

	BA	CSA	DA	FA	FPA	MFO	RFO	WWO
c_1	6.01214	5.97145	6.01451	6.00422	6.00612	6.01815	6.00845	6.01421
c_2	5.39452	5.35642	5.39652	5.31245	5.31022	5.30334	5.30485	5.41214
c_3	4.44581	4.48254	4.49542	4.52142	4.49214	4.49323	4.49215	4.51243
c_4	3.51212	3.49214	3.49543	3.51213	3.49143	3.49683	3.49842	3.51024
c_5	2.21112	2.16421	2.21421	2.16412	2.15421	2.15227	2.14463	2.15843
$f_w(\overline{c})$	13.42872	13.36091	13.45385	13.39052	13.35304	13.35908	**13.34954**	13.44847
$g_1(\overline{c})$	−0.07700	−0.06386	−0.08266	−0.07230	−0.06403	−0.06566	−0.06345	−0.08120

	GA	DE	PSO	AACA	ABCA	SA	ALO	WO
c_1	6.01123	6.01245	6.00945	6.00846	6.01245	6.01253	6.00845	6.01452
c_2	5.45246	5.46215	5.45632	5.46012	5.46214	5.45962	5.45184	5.46215
c_3	4.48142	4.49012	4.49102	4.49521	4.512	4.4951	4.49852	4.49721
c_4	3.49654	3.51426	3.49215	3.49562	3.49021	3.49516	3.49423	4.48956
c_5	2.15483	2.16421	2.15481	2.16011	2.15026	2.15241	2.15426	2.15483
$f_w(\overline{c})$	13.44164	13.47072	13.44617	13.45598	13.46068	13.45306	13.44838	14.07761
$g_1(\overline{c})$	−0.07780	−0.08384	−0.07863	−0.08065	−0.08158	−0.08002	−0.07934	−0.16775

Table 18
Optimal construction variables for the pressure vessel design problem.

	BA	CSA	DA	FA	FPA	MFO	RFO	WWO
x_1	0.91245	0.82145	0.81782	0.84651	0.82451	0.82541	0.81425	0.84621
x_2	0.45685	0.44254	0.45231	0.51213	0.44891	0.51231	0.44521	0.46012
x_3	42.21142	42.31223	42.21492	42.25413	42.32151	42.31024	42.20231	42.32612
x_4	176.45821	176.54262	177.31542	176.56234	176.64852	176.61243	176.62145	176.63212
$f_w(\overline{c})$	6839.8955	6171.5471	6179.0417	6557.8214	6217.216	6422.4895	6113.3195	6405.018
$c_1(\overline{c})$	−0.09776	−0.00482	−0.00307	−0.03100	−0.00770	−0.00882	0.000254	−0.02931
$c_2(\overline{c})$	−41.80872	−41.90857	−41.81218	−41.85102	−41.91776	−41.90660	−41.79969	−41.92232
$c_4(\overline{c})$	−63.54179	−63.45738	−62.68458	−63.43766	−63.35148	−63.38757	−63.37855	−63.36788

	GA	DE	PSO	AACA	ABCA	SA	ALO	WO
x_1	0.82154	0.85301	0.81685	0.82645	0.82451	0.81612	0.85148	0.84263
x_2	0.44322	0.45321	0.44642	0.45016	0.44261	0.44115	0.45652	0.52103
x_3	42.452	42.32104	42.29842	42.62151	42.56213	42.10029	42.36521	42.35121
x_4	176.62315	176.62531	176.65314	176.54813	176.45982	176.7288	176.61026	176.60592
$f_w(\overline{c})$	6200.0828	6429.7482	6152.2335	6284.0087	6233.7566	**6098.67**	6436.9571	6577.9336
$c_1(\overline{c})$	−0.00221	−0.03621	−0.00049	−0.00385	−0.00306	−0.00358	−0.03383	−0.02525
$c_2(\overline{c})$	−42.04700	−41.91729	−41.89489	−42.2149	−42.15608	−41.69865	−41.96104	−41.94717
$c_4(\overline{c})$	−63.37685	−63.37469	−63.34686	−63.45187	−63.54018	−63.2712	−63.38974	−63.39408

Table 19
Optimal variables for the gear train design problem.

	BA	CSA	DA	FA	FPA	MFO	RFO	WWO
x_1	57.53212	55.46532	51.65482	51.98541	54.36485	48.32162	52.1581	57.26512
x_2	19.75461	16.23612	17.65231	25.65891	23.65232	18.62531	23.01251	23.56412
x_3	19.32561	22.31521	22.63215	14.65912	18.65231	20.4121	16.1401	15.65423
x_4	44.65231	45.92531	51.02643	46.95641	56.12413	56.14523	47.2105	46.25481
$f_w(\overline{x})$	0.085784	0.08209	0.08752	0.08902	0.08344	0.08088	0.08709	0.080396

	GA	DE	PSO	AACA	ABCA	SA	ALO	WO
x_1	53.24152	56.51231	51.24132	52.12413	52.14268	53.32151	53.45621	52.12412
x_2	23.1423	25.32142	21.1421	21.10214	22.01452	22.1012	22.32104	23.14621
x_3	16.32512	17.21423	14.56213	16.21413	16.54123	16.01241	16.54813	16.54123
x_4	48.54123	45.21351	46.21452	47.01245	46.1254	47.51241	47.15542	46.4812
$f_w(\overline{x})$	0.08436	0.09914	**0.07523**	0.0806	0.08742	0.08063	0.08457	0.09138

of individuals. Similarities between results of RFO and other heuristics were confirmed by the statistical test. The test has identified results of PSO, MFO, GA, AACA, CSA, and DA as most related to RFO results.

In general, we can say that the proposed RFO was efficient in optimization tasks. The results confirmed the high potential of the proposed algorithm. Although there are still some aspects which we want to concentrate on in future research. All heuristics in large depend on the initial population. Since for most of the algorithms we randomly select them in the search space this is a weak point of every method. The randomness of the initial selection can help in better results but on the other hand, can also make the optimization less efficient. Therefore proposition of a technique for pre-selection of the individuals for the initial population can help to increase precision and faster the convergence to the optimum. The other aspect of future research can be the parallelization of calculations. A division of tasks between multiple cores can help with the better search in the model space.

6. Final remarks

In this article we present a nature-inspired model of hunting and developing the population of a well-known animal — red fox. This mammal has adapted to various environmental conditions where he was able to benefit and develop in many destinations. We propose a model of the features that made the fox an efficient hunter. We called the developed optimization method a Red Fox Optimization Algorithm (RFO).

Presented novel meta-heuristic algorithm models behavior of red fox into optimization technique. In the proposed algorithm we have

Table 20
Optimal construction variables for the multi-plate disc clutch brake problem.

	BA	CSA	DA	FA	FPA	MFO	RFO	WWO
r_i	70	78	79	65	68	66	72	66
r_o	90	100	100	91	91	93	93	93
F	810	909	820	861	980	669	762	602
Z	3	2	2	2	2	2	2	2
t	1	1	1	1	1	1	1	1
$f_{mp}(\overline{x})$	0.31345	0.28773	0.2762	0.29802	0.2687	0.31543	**0.25359**	0.31543
$g_1(\overline{x})$	0	2	1	6	3	7	1	7
$g_2(\overline{x})$	20	22.5	22.5	22.5	22.5	22.5	22.5	22.5
$g_3(\overline{x})$	0.91939	0.92607	0.93053	0.9324	0.91466	0.95037	0.92996	0.95534
$g_4(\overline{x})$	9830.37109	9826.96428	9836.5515	9860.74163	9821.22546	9895.76691	9847.99318	9906.2058
$g_5(\overline{x})$	7895.76389	7659.30836	7647.33891	7940.10185	7905.24039	7899.75472	7829.59394	7899.75471
$g_6(\overline{x})$	14	14	14	14	14	14	14	14
$g_7(\overline{x})$	96669.1875	80439.81438	72929.43687	67042.035	77608.88302	53099.74982	62512.392	47775.82868
$g_8(\overline{x})$	0.01488	0.01788	0.01972	0.021445	0.01853	0.02071	0.023	0.03008

	GA	DE	PSO	AACA	ABCA	SA	ALO	WO
r_i	74	67	76	77	70	72	77	72
r_o	96	91	97	98	92	94	100	93
F	775	727	986	936	741	696	606	772
Z	2	2	2	2	2	2	2	2
t	1	1	1	1	1	1	1	1
$f_{mp}(\overline{x})$	0.2748	0.27862	0.26694	0.27002	0.26187	0.26833	0.29912	0.2546
$g_1(\overline{x})$	2	4	1	1	2	2	3	1
$g_2(\overline{x})$	22.5	22.5	22.5	22.5	22.5	22.5		22.5
$g_3(\overline{x})$	0.93401	0.934010.93894	0.91357	0.91888	0.93378	0.9393	0.95259	0.92904
$g_4(\overline{x})$	9852.4003	9872.81399	9803.40419	9813.39429	9858.79635	9867.41006	9889.59949	9845.99834
$g_5(\overline{x})$	7763.41699	7916.9346	7725.46628	7699.42667	7867.47051	7815.45114	7671.21594	7829.59394
$g_6(\overline{x})$	14	14	14	14	14	14	14	14
$g_7(\overline{x})$	65520.31765	57235.9743	83790.82844	81410.1888	59726.07556	57465.1547	53333.53017	63333.552
$g_8(\overline{x})$	0.021944	0.02517	0.16961	0.17664	0.02507	0.02501	0.02696	0.22701

modeled global search simulating the way red fox is searching for prey over the land and local search simulating the way red fox is disguising prey while hunting. Moreover, we have also modeled reproduction and leaving the herd.

Results of experimental benchmark tests have shown that the proposed RFO algorithm can precisely find maxima/minima of test functions. It is also efficient in considered optimization problems. Statistics of benchmark tests have shown fast convergence. RFO locates most of the individuals in the direct surrounding of the optimum after a short time, which gives RFO a good advantage to improve the final precision of results. Moreover, the proposed local search mechanism reduces computational complexity but also prevents the algorithm from being stacked during optimization.

CRediT authorship contribution statement

Dawid Połap: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data curation, Writing - original draft, Writing - review & editing. **Marcin Woźniak:** Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data curation, Writing - original draft, Writing - review & editing.

Declaration of competing interest

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

Acknowledgment

Authors acknowledge contribution to this project of the "Diamond Grant 2016" No. 0080/DIA/2016/45 funded by the Polish Ministry of Science and Higher Education.

References

Abbass, H. A. (2001). MBO: Marriage in honey bees optimization-a haplometrosis polygynous swarming approach. In *Evolutionary computation, 2001. proceedings of the 2001 congress on (vol. 1)* (pp. 207–214). IEEE.

Arora, J. S. (2004). *Introduction to optimum design*. Elsevier Academic Press.

Askarzadeh, A., & Rezazadeh, A. (2013). A new heuristic optimization algorithm for modeling of proton exchange membrane fuel cell: bird mating optimizer. *International Journal of Energy Research*, *37*(10), 1196–1204.

Belegundu, A. D. (1983). *A study of mathematical programming methods for structural optimization.*

Biswas, S., & Acharyya, S. (2016). Neural model of gene regulatory network: a survey on supportive meta-heuristics. *Theory in Biosciences*, *135*(1–2), 1–19. http://dx.doi.org/10.1007/s12064-016-0224-z.

Brown, P., Pullan, W., Yang, Y., & Zhou, Y. (2016). Fast and accurate non-sequential protein structure alignment using a new asymmetric linear sum assignment heuristic. *Bioinformatics*, *32*(3), 370–377. http://dx.doi.org/10.1093/bioinformatics/btv580.

Chamaani, S., Mirtaheri, S. A., & Abrishamian, M. S. (2011). Improvement of time and frequency domain performance of antipodal vivaldi antenna using multi-objective particle swarm optimization. *IEEE Transactions on Antennas and Propagation*, *59*(5), 1738–1742.

Chickermane, H., & Gea, H. (1996). Structural optimization using a new local approximation method. *International Journal for Numerical Methods in Engineering*, *39*(5), 829–846.

Das, S., Mullick, S. S., & Suganthan, P. N. (2016). Recent advances in differential evolution - An updated survey. *Swarm and Evolutionary Computation*, *27*, 1–30. http://dx.doi.org/10.1016/j.swevo.2016.01.004.

Davari, S., Kilic, K., & Naderi, S. (2016). A heuristic approach to solve the preventive health care problem with budget and congestion constraints. *Applied Mathematics and Computation*, *276*, 442–453. http://dx.doi.org/10.1016/j.amc.2015.11.073.

Dorigo, M., & Di Caro, G. (1999). Ant colony optimization: a new meta-heuristic. In *Evolutionary computation, 1999. CEC 99. proceedings of the 1999 congress on (vol. 2)* (pp. 1470–1477). IEEE.

Gandomi, A. H., & Alavi, A. H. (2012). Krill herd: a new bio-inspired optimization algorithm. *Communications in Nonlinear Science and Numerical Simulation*, *17*(12), 4831–4845.

Gao, S., Wang, Y., Cheng, J., Inazumi, Y., & Tang, Z. (2016). Ant colony optimization with clustering for solving the dynamic location routing problem. *Applied Mathematics and Computation*, *285*, 149–173. http://dx.doi.org/10.1016/j.amc.2016.03.035.

Gonzalez-Sanchez, B., Vega-Rodríguez, M. A., & Santander-Jiménez, S. (2019). Multi-objective memetic meta-heuristic algorithm for encoding the same protein with multiple genes. *Expert Systems with Applications*.

Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., & Chen, H. (2019). Harris hawks optimization: Algorithm and applications. *Future Generation Computer Systems*, *97*, 849–872.

Holland, J. H. (1992). Genetic algorithms. *Scientific American*, *267*(1), 66–73.

Holm, Å., Tedgren, Å. C., & Larsson, T. (2016). Heuristics for integrated optimization of catheter positioning and dwell time distribution in prostate HDR brachytherapy. *Annals of Operations Research*, *236*(2), 319–339. http://dx.doi.org/10.1007/s10479-013-1448-7.

Jain, M., Singh, V., & Rani, A. (2018). A novel nature-inspired algorithm for optimization: Squirrel search algorithm. *Swarm and Evolutionary Computation*.

Janani, R., & Vijayarani, S. (2019). Text document clustering using spectral clustering algorithm with particle swarm optimization. *Expert Systems with Applications*, *134*, 192–200.

Kalantzis, G., Shang, C., Lei, Y., & Leventouri, T. (2016). Investigations of a GPU-based levy-firefly algorithm for constrained optimization of radiation therapy treatment planning. *Swarm and Evolutionary Computation*, *26*, 191–201. http://dx.doi.org/10.1016/j.swevo.2015.09.006.

Kannan, B., & Kramer, S. N. (1994). An augmented Lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. *Journal of Mechanical Design*, *116*(2), 405–411.

Karaboga, D., & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *Journal of Global Optimization*, *39*(3), 459–471.

Kaveh, A., & Farhoudi, N. (2013). A new optimization method: dolphin echolocation. *Advances in Engineering Software*, *59*, 53–70.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of IEEE international conference on neural networks (vol. 4)* (pp. 1942–1948).

Larivière, S., & Pasitschniak-Arts, M. (1996). Vulpes vulpes. *Mammalian Species*, *537*, 1–11. http://dx.doi.org/10.2307/3504236.

Li, X. (2003). *A new intelligent optimization-artificial fish swarm algorithm* (Doctor thesis), China: Zhejiang University of Zhejiang.

Li, S., Chen, H., Wang, M., Heidari, A. A., & Mirjalili, S. (2020). Slime mould algorithm: A new method for stochastic optimization. *Future Generation Computer Systems*.

Lu, X., & Zhou, Y. (2008). A novel global convergence algorithm: bee collecting pollen algorithm. In *International conference on intelligent computing* (pp. 518–525). Springer.

Martin, R., & Stephen, W. (2006). Termite: A swarm intelligent routing algorithm for mobilewireless Ad-Hoc networks. In *Stigmergic optimization* (pp. 155–184). Springer.

Mirghasemi, S., Andreae, P., & Zhang, M. (2019). Domain-independent severely noisy image segmentation via adaptive wavelet shrinkage using particle swarm optimization and fuzzy C-means. *Expert Systems with Applications*, *133*, 126–150.

Mirjalili, S. (2015a). The ant lion optimizer. *Advances in Engineering Software*, *83*, 80–98. http://dx.doi.org/10.1016/j.advengsoft.2015.01.010.

Mirjalili, S. (2015b). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-Based Systems*, *89*, 228–249. http://dx.doi.org/10.1016/j.knosys.2015.07.006.

Mirjalili, S. (2016). Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. *Neural Computing and Applications*, *27*(4), 1053–1073. http://dx.doi.org/10.1007/s00521-015-1920-1.

Mirjalili, S., Gandomi, A. H., Mirjalili, S. Z., Saremi, S., Faris, H., & Mirjalili, S. M. (2017). Salp swarm algorithm: a bio-inspired optimizer for engineering design problems. *Advances in Engineering Software*, *114*, 163–191.

Mirjalili, S., & Lewis, A. (2016). The Whale Optimization Algorithm. *Advances in Engineering Software*, *95*, 51–67. http://dx.doi.org/10.1016/j.advengsoft.2016.01.008.

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software*, *69*, 46–61.

Mohapatra, P., Chakravarty, S., & Dash, P. K. (2015). An improved cuckoo search based extreme learning machine for medical data classification. *Swarm and Evolutionary Computation*, *24*, 25–49. http://dx.doi.org/10.1016/j.swevo.2015.05.003.

Mosa, M. A. (2019). Real-time data text mining based on Gravitational Search Algorithm. *Expert Systems with Applications*, *137*, 117–129.

Mucherino, A., & Seref, O. (2007). Monkey search: a novel metaheuristic search for global optimization. In *AIP conference proceedings (vol. 953) (no. 1)* (pp. 162–173). AIP.

Oftadeh, R., Mahjoob, M., & Shariatpanahi, M. (2010). A novel meta-heuristic optimization algorithm inspired by group hunting of animals: Hunting search. *Computers & Mathematics with Applications*, *60*(7), 2087–2098.

Ogiela, M. R., & Krzyworzeka, N. (2016). Heuristic approach for computer-aided lesion detection in mammograms. *Soft Computing*, *20*(10), 4193–4202. http://dx.doi.org/10.1007/s00500-016-2186-y.

Pan, W. T. (2012). A new fruit fly optimization algorithm: taking the financial distress model as an example. *Knowledge-Based Systems*, *26*, 69–74.

Pinto, P. C., Runkler, T. A., & Sousa, J. M. (2007). Wasp swarm algorithm for dynamic MAX-sat problems. In *International conference on adaptive and natural computing algorithms* (pp. 350–357). Springer.

Ragsdell, K., & Phillips, D. (1976). Optimal design of a class of welded structures using geometric programming. *Journal of Engineering for Industry*, *98*(3), 1021–1025.

Sandgren, E. (1990). Nonlinear integer and discrete programming in mechanical design optimization. *Journal of Mechanical Design*, *112*(2), 223–229.

Savsani, P., & Savsani, V. (2016). Passing vehicle search (PVS): A novel metaheuristic algorithm. *Applied Mathematical Modelling*, *40*(5–6), 3951–3978.

Sharma, A., Sharma, A., Panigrahi, B. K., Kiran, D., & Kumar, R. (2016). Ageist spider monkey optimization algorithm. *Swarm and Evolutionary Computation*, *28*, 58–77. http://dx.doi.org/10.1016/j.swevo.2016.01.002.

Shiqin, Y., Jianjun, J., & Guangxing, Y. (2009). A dolphin partner optimization. In *Intelligent systems, 2009. GCIS'09. WRI global congress on (vol. 1)* (pp. 124–128). IEEE.

Siddall, J. N. (1972). *Analytical decision-making in engineering design*. Prentice Hall.

Storn, R., & Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, *11*(4), 341–359.

Toksari, M. D. (2006). Ant colony optimization for finding the global minimum. *Applied Mathematics and Computation*, *176*(1), 308–316.

Van Laarhoven, P. J., & Aarts, E. H. (1987). Simulated annealing. In *Simulated annealing: theory and applications* (pp. 7–15). Springer.

Yang, X. S. (2010a). Firefly algorithm, stochastic test functions and design optimisation. *International Journal of Bio-Inspired Computation*, *2*(2), 78–84. http://dx.doi.org/10.1504/IJBIC.2010.032124.

Yang, X. S. (2010b). A new metaheuristic bat-inspired algorithm. In *Nature inspired cooperative strategies for optimization* (pp. 65–74). Springer, http://dx.doi.org/10.1007/978-3-642-12538-6_6.

Yang, X. S. (2012). Flower pollination algorithm for global optimization. In *International conference on unconventional computing and natural computation* (pp. 240–249). Springer, http://dx.doi.org/10.1007/978-3-642-32894-7_27.

Yang, X. S., & Deb, S. (2009). Cuckoo search via Lévy flights. In *Nature & biologically inspired computing, 2009. NaBIC 2009. World congress on* (pp. 210–214). http://dx.doi.org/10.1109/NABIC.2009.5393690.

Yazdani, M., & Jolai, F. (2016). Lion optimization algorithm (LOA): a nature-inspired metaheuristic algorithm. *Journal of Computational Design and Engineering*, *3*(1), 24–36.

Zhang, Q. b., Wang, P., & Chen, Z. h. (2019). An improved particle filter for mobile robot localization based on particle swarm optimization. *Expert Systems with Applications*.

Zheng, Y. J. (2015). Water wave optimization: a new nature-inspired metaheuristic. *Computers & Operations Research*, *55*, 1–11. http://dx.doi.org/10.1016/j.cor.2014.10.008.