Expert Systems With Applications 198 (2022) 116924

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Golden jackal optimization: A novel nature-inspired optimizer for engineering applications

Nitish Chopra [a,*], Muhammad Mohsin Ansari [b]

[a] Assistant Professor, Department of Electrical Engineering, St. Soldier Group of Institutions, IKGPTU, Jalandhar, India
[b] Head of Department in Electrical Engineering at Sind Institute of Management & Technology, Karachi, Pakistan

ARTICLE INFO

Keywords:
Nature-inspired algorithm
Golden Jackal optimization
Metaheuristic
Constrained problems
Optimization algorithm
GJO

ABSTRACT

A new nature-inspired optimization method, named the Golden Jackal Optimization (GJO) algorithm is proposed, which aims to provide an alternative optimization method for solving real-world engineering problems. GJO is inspired by the collaborative hunting behaviour of the golden jackals (Canis aureus). The three elementary steps of algorithm are prey searching, enclosing, and pouncing, which are mathematically modelled and applied. The ability of proposed algorithm is assessed, by comparing with different state of the art metaheuristics, on benchmark functions. The proposed algorithm is further tested for solving seven different engineering design problems and introduces a real implementation of the proposed method in the field of electrical engineering. The results of the classical engineering design problems and real implementation verify that the proposed algorithm is appropriate for tackling challenging problems with unidentified search spaces.

1. Introduction

Usually, multifaceted problems like those of engineering, have some significant control variables whose values have a direct influence on the output of the system. Consequently, the precise setting and receiving a correct solution necessitate a skilled designer though it may be tedious and time-consuming. Hence, it is sensible to adapt them as an optimization model that can be solved by smart algorithms (Liu, Wu, Xiao, Wang, & Zhang, 2018). So, numerous methods have been recommended which are usually classified into two main groups: traditional and nature-inspired optimization algorithms. The traditional methods are suitable for locating the optimal solution of differentiable and continuous functions. These methods apply differential calculus for finding the optimal solution. Gradient-based optimization, Response Surface Methods, Simplex Methods (Wehrens & Buydens, 2000), and Quadratic programming (Steffan & Heydt, 2012), etc. are some examples of traditional methods. Due to non-continuity or non-differentiability of objective functions of real-world applications, these methods fail to assure optimal solutions.

Nature-inspired optimization algorithms (Metaheuristics) are progressively becoming famous for handling hard optimization problems due to their simplicity, flexibility, derivative-free method, and evading local-optima. These methods make use of basic mathematical representations derived from nature, are simpler, and easier to implement for solving real-world difficulties. Meta-heuristics generally assume problems as black boxes, hence the flexibility to solve diverse problems without any distinct variations in the algorithm structure. In metaheuristic algorithms, optimization procedure begins with arbitrary solutions. The calculation of the derivative of objective space is unneeded to locate the optimal solution, which makes metaheuristics very appropriate for actual problems with unidentified derivative info. Metaheuristics have capabilities of avoiding local optima owing to their stochastic nature which lets them evade inactivity in local solutions and exploration of the whole search space broadly. Numerous metaheuristic algorithms have been presented in literature and classified into three subcategories: swarm-based (Krause, Cordeiro, Parpinelli, & Lopes, 2013), physics-based (Geem, Kim, & Loganathan, 2001) and evolution-based (Mühlenbein, Gorges-Schleuter, & Krämer, 1988).

Swarm-based metaheuristics are inspired by the social traits of species alike self-organization and labor division (Ab Wahab, Nefti-Meziani, & Atyabi, 2015). One excellent example is Particle Swarm Optimization (PSO) (Fearn, 2014) which is motivated by the flocking behaviour of birds. In PSO, every agent in the population is updated both by its individual best and global best agent. Other methods of swarm-based metaheuristics include Ant Colony Optimization (ACO) (Dorigo, Maniezzo, & Colorni, 1996), Glow-worm Swarm Optimization (GSO)

* Corresponding author at: Department of Electrical Engineering, St. Soldier Group of Institutions, Jalandhar, India.
E-mail address: nitishchopra22@gmail.com (N. Chopra).

https://doi.org/10.1016/j.eswa.2022.116924
Received 20 July 2020; Received in revised form 5 March 2022; Accepted 15 March 2022
Available online 18 March 2022

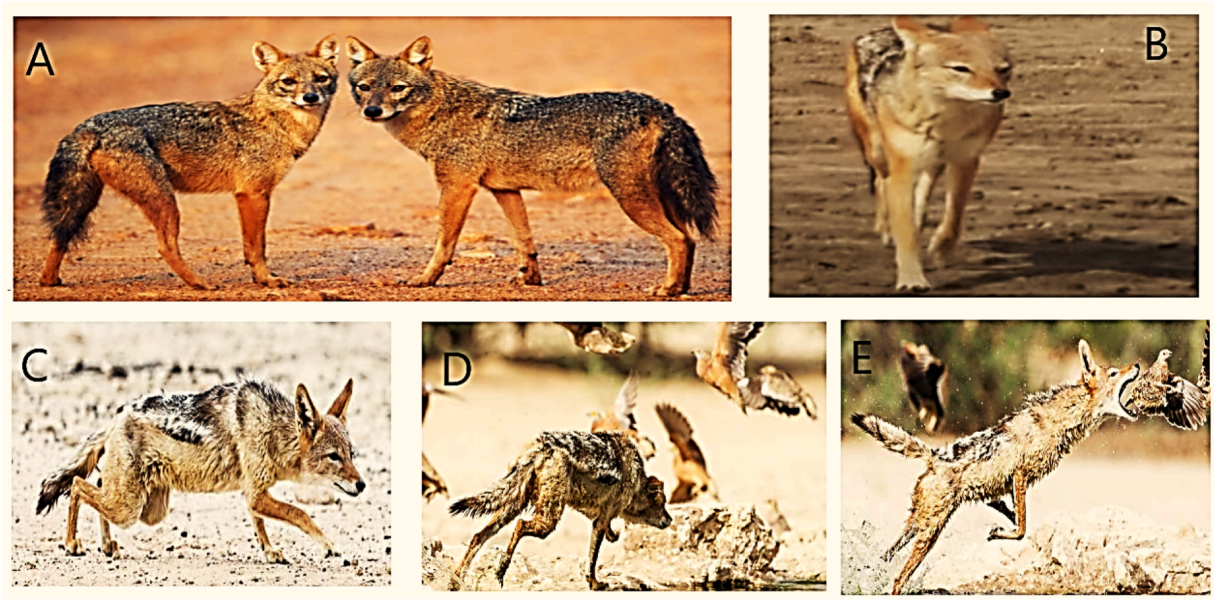

Fig. 1. A) Pair of Golden Jackal B) Golden Jackal searching for prey C) Stalking and enclosing of prey D) &E) Pouncing on prey.

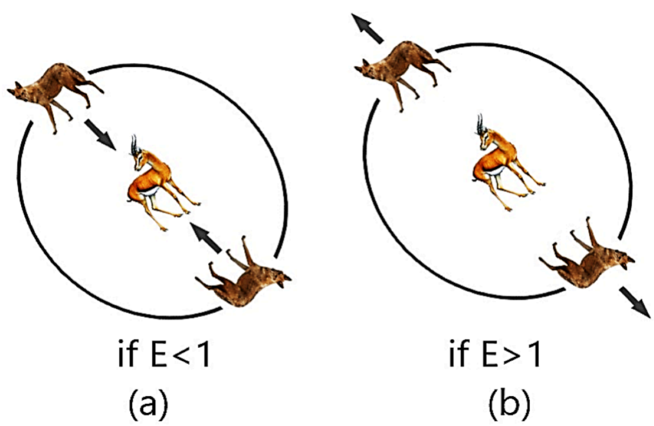

Fig. 2. Attacking vs searching for prey.

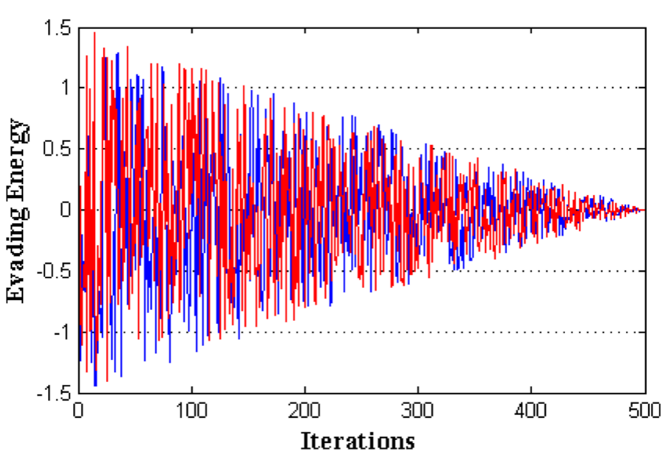

Fig. 3. Dynamic behaviour of evading energy with iterations over 2 runs.

(Krishnanand & Ghose, 2009), Grey Wolf Optimization (GWO) (Mirjalili, Mirjalili, & Lewis, 2014), Shark Smell Optimization (SSO) (Abedinia, Amjady, & Ghasemi, 2016), Firefly Algorithm (FA) (Yang, 2010), Cuckoo Search (CS) (Yang & Deb, 2009), Ant Lion Optimizer (ALO) (Mirjalili, 2015a), Tree Seed Algorithm (TSA) (Kiran, 2015), Sine Cosine Algorithm (SCA) (Mirjalili, 2016); Social Spider Optimization (Cuevas, Cienfuegos, Zaldívar, & Pérez-Cisneros, 2013), Birds Mating Optimizer (Askarzadeh, 2014), Grasshopper Optimization (Saremi, Mirjalili, & Lewis, 2017), Artificial Bee Colony algorithm (Akay & Karaboga, 2012), Moth Flame Optimization (Mirjalili, 2015c), Harris Hawks Optimization (Heidari et al., 2019), Marine Predator Algorithm (Faramarzi, Heidarinejad, Mirjalili, & Gandomi, 2020), Artificial Gorilla Troops Optimizer (Abdollahzadeh, Soleimanian Gharehchopogh, & Mirjalili, 2021), African Vultures Optimization algorithm (Abdollahzadeh, Gharehchopogh, & Mirjalili, 2021), Elephant Clan Optimization (Jafari, Salajegheh, & Salajegheh, 2021), Honey Badger Algorithm (Hashim, Houssein, Hussain, Mabrouk, & Al-Atabany, 2022).

Physics-based optimization algorithms characteristically imitate the laws of physics. Most famous algorithms are Simulated Annealing(SA) (Kirkpatrick, Gelatt, & Vecchi, 1983), Gravitational Search Algorithm (GSA) (Rashedi, Nezamabadi-pour, & Saryazdi, 2009), Gravitational Emulation Local Search (GELS) (Hosseinabadi, Siar, Shamshirband, Shojafar, & Mohd Hairul, 2014), Big-Bang Big-Crunch (BBBC) (Genç, Eksin, & Erol, 2010), Black Hole (BH) algorithm, (Hatamlou, 2013) Galaxy-based Search Algorithm (GbSA) (Hosseini, 2011), Charged System Search (CSS) (Kaveh & Talatahari, 2010), Ray Optimization (RO) algorithm (Kaveh & Khayatazad, 2012), Optics Inspired Optimization (OIO) (Husseinzadeh Kashan, 2015), Vortex Search algorithm (Doḷan & Ölmez, 2015), Multi-Verse Optimizer (MVO) (Mirjalili, Mirjalili, & Hatamlou, 2016), and Artificial Electric Field algorithm(Anita & Yadav, 2019).

Evolution-based algorithms are enthused by the ideas of evolution in biology. Genetic algorithm (GA) (Holland, 1992), is a well-known and extensively used evolution-based algorithms. GA is a derivative-free method and modifies the initial population by imitating the natural selection of fittest. It may offer effective solutions and evade local optima. With its fame, numerous other evolution-based algorithms, like evolutionary programming (EP) (Yao, Liu, & Lin, 1999), Evolutionary Strategies (ES) (Beyer & Schwefel, 2002), Differential Evolution (DE) (Rocca, Oliveri, & Massa, 2011), Memetic Algorithm (MA) (Moscato, Mendes, & Berretta, 2007), Biogeography Based Optimization (BBO) (Simon, 2008), Bacterial Foraging Optimization (BFO) (Passino, 2002), Artificial Algae Algorithm (AAA) (Uymaz, Tezel, & Yel, 2015), Weed Colonization Algorithm (Mehrabian & Lucas, 2006), Monkey King Evolutionary (MKE) (Meng & Pan, 2016), etc. have been proposed.

```
Inputs: The population size N and maximum number of iterations T
Outputs: The location of prey and its fitness value
Initialize the random prey population Yi (i = 1, 2, . . ., N)
while ( t < T )
        Calculate the fitness values of preys
        Y1 = best prey (Male Jackal position)
        Y2 = second best prey (Female Jackal Position)
        for (each prey)
                Update the evading energy "E" using Eq. (6), Eq. (7) and Eq. (8)
                Update "rl" using Eq. (9) and Eq. (10)
                if (|E| >= 1)                                        (Exploration phase)
                 Update the prey position using Eq. (4), Eq. (5) and Eq. (11)
                 if (|E| < 1)                                        (Exploitation phase)
                 Update the prey position using Eq. (12), Eq. (13) and Eq. (11)
        end for
        t=t+1
end while
return Y1
```

Fig. 4. Pseudo code of the GJO algorithm.

Swarm-based metaheuristic algorithms exhibit two explicit behaviours, exploration, and exploitation (Alba & Dorronsoro, 2005). In the exploration stage, the search is carried out in broad variable space for better solutions employing stochastic operators to arbitrarily and globally explore the variable space. Though, exploitation restrain the search to a minor region located in the exploration stage to enhance the solution. Discovering an appropriate equilibrium amid these two is a problematic task owing to the stochastic trait of metaheuristics.

The "No Free Lunch" (NFL) theorem (Wolpert & Macready, 1997) has rationally demonstrated that no *meta*-heuristic is finest for solving every optimization problem. It can be said that a specific *meta*-heuristic might demonstrate very auspicious outcomes on a category of problems, but that algorithm might perform poorly on a diverse category of problems. So, it motivates our attempt to develop an efficient nature-inspired swarm-based optimizing algorithm for solving certain real-world problems. This work proposes a novel swarm-based method with inspiration from the hunting behaviour of golden jackal pair.

The rest of the paper is systematized as follows: Section 2 presents the proposed GJO algorithm. The results and discussion of benchmark functions and engineering problems are presented in Sections 3-4, respectively. Finally, Section 5 presents some conclusions and suggests few opportunities for research in the future.

2. Golden Jackal Optimization (GJO)

In this section the motivation of the proposed method is first discussed. Then, the mathematical model is provided.

2.1. Inspiration

The golden jackal (Canis aureus), is a mid-sized, terrestrial predator belonging to the Canidae family. They are found in North and East Africa, the Middle East, Europe, Southeast Asia, and Central Asia. They range from 3,500 m in the Bale Mountain range of Ethiopia to sea level in Eritrea (Admasu, Thirgood, Bekele, & Karen Laurenson, 2004). The golden jackal is about 70 to 85 cm in body length with standing height nearby 40 cm and tail length around 25 cm. Its. The fur is usually rough brown-tipped and pale gold to yellow, varying with region and season. The diminutive body and lengthy legs of jackal permit it to gallop over excessive distances for hunting prey (Moehlman & Hayssen, 2018). Golden jackals devour both animal and plant food. They are opportunistic hunters with a much diverse diet, consisting of rodents, young gazelles, ground birds, hares, reptiles, frogs, fruit, fish, and insects (Ivory, 1999).

Golden jackals are characteristically monogamous and dwell in copulated pairs. Jackal families consist of one or two grownup individuals known as "helpers." They are jackals former progeny, who after attaining adulthood live with the parents for about a year, looking after the subsequent child (Moehlman, 1983). Young ones are nurtured around eight weeks and then deterred. The young are fed by vomiting and commence eating solid food near three months. Protection and food are provided by both parents. The family is strengthened by helpers in many ways. Grown-ups use "predator bark" and "rumble growl" for warning the pups to take shelter, and a solitary adult can efficaciously drive off big predators. Food is also brought by helpers to a lactating mother which improves the upbringing of the pups indirectly by permitting the parents to devote additional time hunting alone or in pairs. Golden jackals use an extensive collection of howls for locating one another. A pair displays the bond amid them by howling together (Ivory, 1999).

Golden jackal pair (as shown if Fig. 1A) hunt and relax together. The entirety of their conduct is exceptionally synchronized. Family groups of up to 4–5 individuals (Heptner & Naumov, 1998) and groups up to 10 individuals have been recorded (Macdonald, 1979). Foraging families hold regions of a few square kilometres consistently, bits of which are set apart with pee, either by the male or the female jackal, to avert intruders. Cooperative foraging imperative to the jackals, allowing them to hunt a lot bigger prey in territories where they are available (Macdonald, 1979).Mated pairs hunting cooperatively have a high kill percentage than individuals (Wyman, 1967). When hunting in pairs or packs, jackals run parallel to their prey and surpass it in unity. When hunting birds or aquatic rodents, they will run along both sides of narrow rivers or streams and drive their prey from one jackal to another (Heptner & Naumov, 1998). They may be assisted by helpers in the hunt.

The foremost stages of golden jackal pair hunting (shown in Fig. 1) are as follows:

1) Searching, and proceed towards the prey.
2) Enclosing, and irritating the prey until it stops moving.
3) Pouncing towards the prey.

Table 1
Results of Unimodal benchmark functions.

Function	Algorithm Indices	GWO	BBO	GSA	PSO	TLBO	MVO	ALO	GJO
F1	**Best**	1.22E-23	0.001007	1.57E-09	1.63E-10	2.59E-43	0.014775	1.23E-06	2.83E-46
	Mean	3.37E-21	0.982942	7.21E-09	3.42E-08	1.03E-41	0.089075	9.89E-06	6.3E-41
	Worst	2.97E-20	9.359812	1.53E-08	2.09E-07	5.1E-41	0.278805	2.72E-05	6.4E-40
	SD	6.45E-21	1.998751	3.78E-09	5.35E-08	1.47E-41	0.05165	7.74E-06	1.51E-40
	P	3.02E-11	3.02E-11	3.02E-11	3.02E-11	0.830255	3.02E-11	3.02E-11	NA
F2	**Best**	3.75E-14	0.004685	0.000166	1.5E-05	1.38E-22	0.045632	0.000728	2.28E-25
	Mean	6.12E-13	0.212176	0.000242	0.000118	1.52E-21	0.110854	5.084022	2.23E-23
	Worst	2.99E-12	0.917104	0.000445	0.000756	7.83E-21	0.248201	33.87285	2.17E-22
	SD	7.08E-13	0.23216	5.91E-05	0.000144	1.45E-21	0.047319	7.181469	4.3E-23
	P	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.34E-11	3.02E-11	3.02E-11	NA
F3	**Best**	8.64E-11	0.000935	0.018573	0.002669	2.13E-19	0.167993	0.700749	3.69E-26
	Mean	2.4E-08	14.59694	7.109032	0.024382	2.99E-17	0.782433	294.3761	1.51E-20
	Worst	3.38E-07	136.0699	37.29439	0.140772	1.94E-16	1.836216	1551.355	1.6E-19
	SD	6.89E-08	28.80539	10.01559	0.02894	4.9E-17	0.448154	353.9731	3.53E-20
	P	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	NA
F4	**Best**	5.52E-08	0.007632	2.92E-05	0.000706	1.77E-18	0.119071	0.009392	4.93E-18
	Mean	1.08E-06	0.310127	6.25E-05	0.010106	7.68E-18	0.235037	3.269271	1.28E-15
	Worst	8.25E-06	1.331161	8.59E-05	0.037458	3.51E-17	0.400626	14.68888	4.66E-15
	SD	1.64E-06	0.332363	1.56E-05	0.008421	7.04E-18	0.07207	3.629238	1.27E-15
	P	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.82E-10	3.02E-11	3.02E-11	NA
F5	**Best**	6.02E + 00	1.13E-02	6.05E + 00	1.06E-01	3.05E + 00	4.24E + 00	1.67E-01	6.01E + 00
	Mean	7.06E + 00	5.60E + 00	6.64E + 00	1.73E + 01	4.64E + 00	2.87E + 02	1.63E + 02	4.23E + 00
	Worst	9.62E + 00	3.49E + 01	7.56E + 00	2.07E + 02	5.83E + 00	2.33E + 03	1.63E + 03	8.09E + 00
	SD	8.56E-01	6.83E + 00	3.56E-01	3.96E + 01	7.50E-01	5.79E + 02	3.49E + 02	4.31E-01
	P	2.38E-03	3.51E-02	9.53E-07	1.00E-03	3.02E-11	8.10E-10	3.16E-05	NA
F6	**Best**	9.79E-06	7.78E-06	2.46E-09	6.09E-10	8.76E-17	2.48E-02	2.11E-04	4.75E-05
	Mean	9.16E-03	1.98E + 00	7.47E-09	2.18E-08	8.16E-12	9.68E-02	9.05E-02	1.91E-01
	Worst	2.52E-01	1.25E + 01	2.07E-08	1.93E-07	2.36E-10	2.92E-01	2.78E-01	2.96E-01
	SD	4.61E-02	2.73E + 00	3.98E-09	3.77E-08	4.30E-11	5.25E-02	4.01E-02	3.81E-02
	P	8.89E-10	2.39E-04	3.02E-11	3.02E-11	3.02E-11	2.58E-01	2.34E-01	NA
F7	**Best**	0.000235	0.000153	0.002855	0.00479	0.000341	0.001328	0.015997	6.73E-05
	Mean	0.001378	0.005125	0.015827	0.01813	0.001714	0.007099	0.066562	0.000777
	Worst	0.00469	0.027525	0.036778	0.035549	0.003162	0.023837	0.175286	0.002856
	SD	0.001066	0.007062	0.007916	0.007877	0.000719	0.005166	0.036846	0.000657
	P	0.001174	4.64E-05	3.34E-11	3.02E-11	8.88E-06	9.92E-11	3.02E-11	NA

In this work, this hunting strategy of a golden jackal pair is mathematically modelled for designing GJO and performing optimization.

2.2. Mathematical model and algorithm

This subsection demonstrates the development process of the GJO algorithm as a simple and effective metaheuristic optimization method.

2.2.1. Search space formulation

Alike many other metaheuristics, GJO is a population-based method, in which the initial solution is uniformly distributed over the search space as the first trial:

$$Y_0 = Y_{min} + rand(Y_{max} - Y_{min}) \tag{1}$$

Where Y_{max} and Y_{min} are the upper and lower bound for variables and "rand" is a uniform random vector in the range of 0 to 1.

The initialization creates the initial matrix *Prey* of which first and second fittest is jackal pair. The *Prey* is shown as follows:

$$Prey = \begin{bmatrix} Y_{1,1} & Y_{1,2} & \cdots & Y_{1,d} \\ Y_{2,1} & Y_{2,1} & \cdots & Y_{2,d} \\ \vdots & \vdots & \vdots & \vdots \\ Y_{n,1} & Y_{n,2} & \cdots & Y_{n,d} \end{bmatrix} \tag{2}$$

Y_{ij} denotes the *j-th* dimension of *i-th* prey. There is a total of "n" preys, and "d" variables. The prey position refers to the parameters of a specific solution. A fitness (objective) function is applied for estimating the fitness value of each prey during optimization and subsequent matrix collect the fitness value of all preys:

$$F_{OA} = \begin{bmatrix} f(Y_{1,1}; Y_{1,2}; \cdots; Y_{1,d}) \\ f(Y_{2,1}; Y_{2,1}; \cdots; Y_{2,d}) \\ \vdots \\ f(Y_{n,1}; Y_{n,2}; \cdots; Y_{n,d}) \end{bmatrix} \tag{3}$$

where F_{OA} is the matrix for saving the fitness of each prey, Y_{ij} shows the value of j-th dimension of i-th prey, n is the number of preys, and f is the objective function. The fittest one is called Male Jackal and the second fittest is called Female Jackal. The jackal pair acquire corresponding prey position.

2.2.2. Exploration stage or searching the prey

In this portion, the exploration strategy of GJO is proposed. As the jackal's nature is, they know how to perceive and follow the prey, but sporadically the prey cannot be caught easily and escape. Hence, the jackals wait and search for other prey. Hunting is led by Male jackal. Female jackal follows Male jackal.

$$Y_1(t) = Y_M(t) - E.|Y_M(t) - rl.Prey(t)| \tag{4}$$

Table 2
Results of multimodal and fixed-dimension multimodal benchmark functions.

Function	Algorithm Indices	GWO	BBO	GSA	PSO	TLBO	MVO	ALO	GJO
F8	**Best**	- 3149.32	- 4189.83	- 2117.29	- 2707.46	- 3909.15	- 3301.19	- 3439.58	- 2938.98
	Mean	- 2608.03	- 4185.11	- 1581.16	- 1971.04	- 3345.15	- 2786.67	- 2328.89	- 2298.22
	Worst	- 1793.28	- 4150.53	- 1050.51	- 1352.63	- 2748.21	- 2311.07	- 1805.89	- 1599.64
	SD	299.6133	9.37028	263.1334	339.8655	300.3953	259.2604	561.9509	288.6697
	P	7.74E-06	3.02E-11	1.25E-07	0.093341	4.98E-11	6.53E-08	0.290465	NA
F9	**Best**	0	0.000187	0.994961	2.992063	0.013259	10.00551	7.95967	0
	Mean	2.653841	0.97572	7.429027	8.659233	5.500317	20.87308	23.74631	0.604591
	Worst	9.140608	8.038128	14.92438	16.24605	14.22896	46.81747	49.74783	18.13774
	SD	2.834879	1.906529	3.404116	3.173189	3.437944	8.673715	11.01983	3.311483
	P	1.54E-10	4.56E-11	4.56E-11	4.56E-11	4.56E-11	6.55E-12	5.26E-12	NA
F10	**Best**	2.08E-12	0.016312	8.09E-05	1.07E-05	4.44E-15	0.092456	0.000557	4.44E-15
	Mean	2E-11	1.023696	0.000122	0.000151	4.9E-15	0.474091	1.389097	4.8E-15
	Worst	1E-10	3.163091	0.000188	0.000444	7.54E-15	2.047521	5.191245	7.99E-15
	SD	2.13E-11	1.067849	2.7E-05	0.000115	1.9E-15	0.581476	1.345589	1.08E-15
	P	1.25E-11	1.25E-11	1.25E-11	1.25E-11	0.000285	1.25E-11	1.25E-11	0
F11	**Best**	0	0.025154	2.16691	0.061535	0	0.273358	0.055238	0
	Mean	0.02985	0.705495	5.603493	0.939436	0.016522	0.514509	0.179803	0.01321
	Worst	0.101945	1.092374	11.33721	3.007362	0.096573	0.828937	0.324966	0.173643
	SD	0.026988	0.394948	2.647036	0.760557	0.023714	0.149401	0.075028	0.039472
	P	3.6E-07	2.73E-11	3.16E-12	4.16E-11	2.37E-06	3.75E-11	1.7E-09	NA
F12	**Best**	2.87E-06	0.000103	4.14E-11	2.98E-12	1.95E-17	0.001015	0.647629	6.26E-05
	Mean	0.009702	0.091372	0.041468	6.07E-10	2.19E-13	0.14556	5.684028	0.038206
	Worst	0.060857	0.524723	0.311007	5.82E-09	3.28E-12	1.091909	12.12193	0.078017
	SD	0.014151	0.145887	0.107529	1.12E-09	6.75E-13	0.26703	2.673696	0.018507
	P	4.11E-07	0.239837	1.11E-06	3.02E-11	3.02E-11	0.190718	3.02E-11	NA
F13	**Best**	1.55E-05	3.76E-04	2.60E-10	9.93E-11	2.00E-16	5.37E-03	3.26E-06	1.60E-04
	Mean	4.57E-02	7.13E-02	6.35E-04	5.27E-06	2.20E-03	2.04E-02	8.22E-03	1.30E-01
	Worst	2.90E-01	6.08E-01	1.10E-02	9.80E-05	1.10E-02	6.20E-02	7.27E-02	4.09E-01
	SD	7.95E-02	1.13E-01	2.45E-03	1.90E-05	4.47E-03	1.25E-02	1.62E-02	9.59E-02
	P	7.22E-06	0.006972	8.15E-11	3.02E-11	5.57E-10	9.51E-06	2.6E-08	NA
F14	**Best**	9.98E-01	9.98E-01	9.98E-01	9.98E-01	9.98E-01	9.98E-01	9.98E-01	9.98E-01
	Mean	5.32E + 00	1.20E + 00	7.37E + 00	2.61E + 00	1.10E + 00	1.13E + 00	4.39E + 00	3.23E + 00
	Worst	1.27E + 01	6.90E + 00	1.64E + 01	7.87E + 00	1.34E + 01	3.97E + 00	2.02E + 01	1.08E + 01
	SD	3.84E + 00	1.08E + 00	4.59E + 00	2.25E + 00	1.04E + 00	5.66E-01	4.29E + 00	2.70E + 00
	P	0.864994	7.22E-06	0.599689	2.64E-05	7.45E-05	2.61E-10	0.025075	NA
F15	**Best**	3.38E-04	3.71E-04	9.23E-04	3.43E-04	3.07E-04	5.26E-04	6.27E-04	3.13E-04
	Mean	2.65E-03	1.46E-03	3.42E-03	8.74E-04	1.76E-03	3.55E-03	2.66E-03	1.40E-03
	Worst	2.10E-02	4.59E-03	1.40E-02	1.35E-03	2.04E-02	2.04E-02	2.11E-02	2.04E-02
	SD	6.08E-03	8.13E-04	3.24E-03	1.91E-04	5.06E-03	6.28E-03	3.67E-03	1.47E-04
	P	1.76E-01	1.68E-04	1.11E-06	1.78E-04	2.88E-06	1.00E-03	3.57E-06	NA
F16	**Best**	- 1.03163	- 1.03021	- 1.03163	- 1.03163	- 1.03163	- 1.03163	- 1.03163	- 1.03163
	Mean	- 1.03163	- 0.87194	- 1.03163	- 1.03163	- 1.03163	- 1.03163	- 1.03163	- 1.03163
	Worst	- 1.03163	- 0.11366	- 1.03163	- 1.03163	- 1.03163	- 1.03162	- 1.03163	- 1.03162
	SD	1.28E-07	0.200259	1.56E-10	5.38E-16	5.61E-16	1.35E-06	3.17E-13	1.63E-06
	P	2.67E-09	3.02E-11	3.02E-11	1.14E-11	1.41E-11	0.297272	3.02E-11	NA
F17	**Best**	3.98E-01	5.83E-01	3.98E-01	3.98E-01	3.98E-01	4.09E-01	3.98E-01	3.98E-01
	Mean	3.98E-01	2.30E+00	3.98E-01	3.98E-01	3.98E-01	1.26E+00	3.98E-01	3.98E-01
	Worst	4.04E-01	1.07E+01	3.98E-01	3.98E-01	3.98E-01	6.07E+00	3.98E-01	4.08E-01
	SD	1.25E-03	2.35E+00	6.41E-11	0.00E+00	0.00E+00	1.22E+00	3.23E-13	1.88E-03
	P	2.15E-06	3.02E-11	3.02E-11	1.21E-12	1.21E-12	4.08E-11	3.02E-11	NA
F18	**Best**	3.00E+00	3.13E+00	3.00E+00	3.00E+00	3.00E+00	3.12E+00	3.00E+00	3.00E+00
	Mean	3.00E+00	2.21E+01	3.00E+00	3.00E+00	3.00E+00	3.55E+01	3.00E+00	3.00E+00
	Worst	3.00E+00	3.28E+01	3.00E+00	3.00E+00	3.00E+00	1.73E+02	3.00E+00	3.00E+00
	SD	2.38E-04	1.11E+01	6.93E-09	2.95E-15	1.46E-15	4.19E+01	1.13E-12	3.41E-05
	P	4.43E-03	3.02E-11	3.02E-11	2.82E-11	2.55E-11	3.02E-11	3.02E-11	NA
F19	**Best**	- 3.86278	- 3.84579	- 3.86278	- 3.86278	- 3.86278	- 3.82446	- 3.86278	- 3.86278
	Mean	- 3.86099	- 3.68881	- 3.86278	- 3.86278	- 3.86278	- 3.42549	- 3.85884	- 3.85894
	Worst	- 3.85489	- 3.15973	- 3.86278	- 3.86278	- 3.86278	- 2.62099	- 3.76278	- 3.85458

(*continued on next page*)

Table 2 (*continued*)

Function	Algorithm Indices	GWO	BBO	GSA	PSO	TLBO	MVO	ALO	GJO
	SD	0.002632	0.180471	1.54E-09	2.4E-15	2.65E-15	0.305416	0.002947	0.002875
	P	0.003034	3.02E-11	3.02E-11	1.01E-11	4.08E-12	3.02E-11	3.02E-11	NA
F20	**Best**	-3.32198	-3.23394	-3.322	-3.322	-3.322	-2.69782	-3.322	-3.322
	Mean	-3.26011	-2.88637	-3.322	-3.26255	-3.30524	-1.4287	-3.27578	-3.04444
	Worst	-3.05545	-1.96212	-3.322	-3.2031	-3.2031	-0.52679	-3.17066	-2.92164
	SD	0.08224	0.302937	2.16E-08	0.060463	0.041042	0.528007	0.062016	0.162718
	P	4.44E-07	0.006669	3.02E-11	3.08E-08	3.39E-10	6.07E-11	2.19E-08	NA
F21	**Best**	-10.1523	-10.1532	-10.1532	-10.1532	-10.1532	-10.1522	-10.1532	-10.1532
	Mean	-9.13073	-10.0144	-6.08176	-7.81357	-9.74742	-5.38122	-5.61447	-8.08247
	Worst	-2.52743	-7.17381	-2.63047	-2.63047	-4.17695	-2.62946	-2.63047	-2.62382
	SD	2.354604	0.538354	3.657423	3.215402	1.412273	3.305277	2.743847	2.259888
	P	5.46E-06	3.83E-05	0.935192	0.004953	1.04E-07	3.02E-04	0.217017	NA
F22	**Best**	-10.4011	-10.4028	-10.4029	-10.4029	-10.4029	-10.4023	-10.4029	-10.4025
	Mean	-9.17392	-10.2292	-9.72607	-8.28119	-10.0143	-7.43045	-6.46303	-9.44915
	Worst	-2.74979	-7.88914	-2.84452	-2.75193	-4.32538	-1.83746	-1.83759	-2.73083
	SD	2.523095	0.468476	2.073151	3.121549	1.479902	3.523388	3.39604	2.252838
	P	0.000377	0.035132	6.52E-08	0.00857	4.89E-09	0.009357	0.455284	NA
F23	**Best**	-10.5351	-10.5363	-10.5364	-10.5364	-10.5364	-10.5349	-10.5364	-10.5359
	Mean	-9.89337	-10.4083	-9.67808	-9.66644	-10.0897	-7.1673	-5.4446	-9.18821
	Worst	-2.42134	-9.74295	-3.83543	-2.87114	-3.83543	-1.8593	-1.67655	-1.95922
	SD	1.964786	0.230943	2.014163	2.295351	1.700093	3.749052	3.495033	2.826944
	P	7.09E-08	0.00137	1.86E-06	2.91E-07	3.59E-09	0.00731	0.085	NA

$$Y_2(t) = Y_{FM}(t) - E.|Y_{FM}(t) - rl.Prey(t)| \quad (5)$$

where t indicates the current iteration, Prey(t) is the position vector of the prey, and $Y_M(t)$ and $Y_{FM}(t)$ indicates the position of the male and female jackal. $Y_1(t)$ and $Y_2(t)$ are updated positions of male and female jackal corresponding to the prey.

E is Evading Energy of prey and is calculated as:

$$E = E_1 * E_0 \quad (6)$$

E_1 indicates the decreasing energy of the prey and E_0 denotes the initial state of its energy.

$$E_0 = 2 * r - 1 \quad (7)$$

Where "r" is an arbitrary number between 0 and 1.

$$E_1 = c_1 * (1 - (t/T)) \quad (8)$$

"T" denotes the maximum number of iterations, c_1 is constant value equal to 1.5 and "t" is the current iteration. E_1 is linearly decreased from 1.5 to 0 throughout iterations.

"$|Y(t) - rl.Prey(t)|$" in equation (4) and (5) computes the distance between jackal and prey. This distance gets subtracted or added to current position of jackal depending upon evading energy of prey.

"rl" in Eq. (4) and Eq. (5) is a vector of random numbers based on Lévy distribution representing the Lévy movement. The multiplication of "rl" and *Prey* simulates the movement of prey in Lévy manner and is calculated as (Faramarzi et al., 2020):

$$rl = 0.05 * LF(y) \quad (9)$$

LF is the levy flight function, which is calculated using.

$$LF(y) = 0.01 \times (\mu \times \sigma) \Big/ \left(\left|v^{(1/\beta)}\right|\right); \sigma = \left(\frac{\Gamma(1+\beta) \times \sin(\pi\beta/2)}{\Gamma\left(\frac{1+\beta}{2}\right) \times \beta \times \left(2^{\frac{\beta-1}{2}}\right)}\right)^{1/\beta} \quad (10)$$

where u, v are random values inside (0,1), β is a default constant set to 1.5.Finally, the jackal positions are updated by taking the mean of Eq. (4) & Eq. (5).

$$Y(t+1) = \frac{Y_1(t) + Y_2(t)}{2} \quad (11)$$

2.2.3. Exploitation stage or enclosing and Pouncing the prey

When the prey is harassed by jackals its evading energy decreases and then the jackal pair enclose the prey detected in the previous stage. After enclosing, they pounce on prey and devour it. This behaviour of both male and female jackal hunting together is mathematically modelled as follows:

$$Y_1(t) = Y_M(t) - E.|rl.Y_M(t) - Prey(t)| \quad (12)$$

$$Y_2(t) = Y_{FM}(t) - E.|rl.Y_{FM}(t) - Prey(t)| \quad (13)$$

where t indicates the current iteration, Prey(t) is the position vector of the prey, and $Y_M(t)$ and $Y_{FM}(t)$ indicates the position of the male and female jackal. $Y_1(t)$ and $Y_2(t)$ are updated positions of male and female jackal corresponding to the prey. Prey's Evading Energy "E" is calculated as per Eq. (6). Finally, the jackal positions are updated as per Eq. (11).

The function of "rl" in Eq. (12) and Eq. (13) is to provide arbitrary behaviour in the exploitation stage, favouring exploration and local optima avoidance. "rl" is calculated as per Eq. (9). This element helps in evading local optima sluggishness, particularly in the concluding iterations.

The element can be well-thought-out as the consequence of hindrances to moving towards the prey. Usually, the difficulties in nature occur in the chasing paths of jackals preventing their suitable and rapid move towards the prey. This is the purpose of "rl" in the exploitation stage.

2.2.4. Switching from exploration to exploitation

In the GJO algorithm, the escaping energy of the prey is used for Switching from exploration to exploitation. The prey's energy declines significantly throughout evading behaviour. Considering this, the prey's evading energy is modelled as per Eq. (6). Initial energy E0 arbitrarily deviates from -1 to 1 at every iteration. When E0 value reduces from 0 to -1, the prey is physically waning, though when E0 value rises from

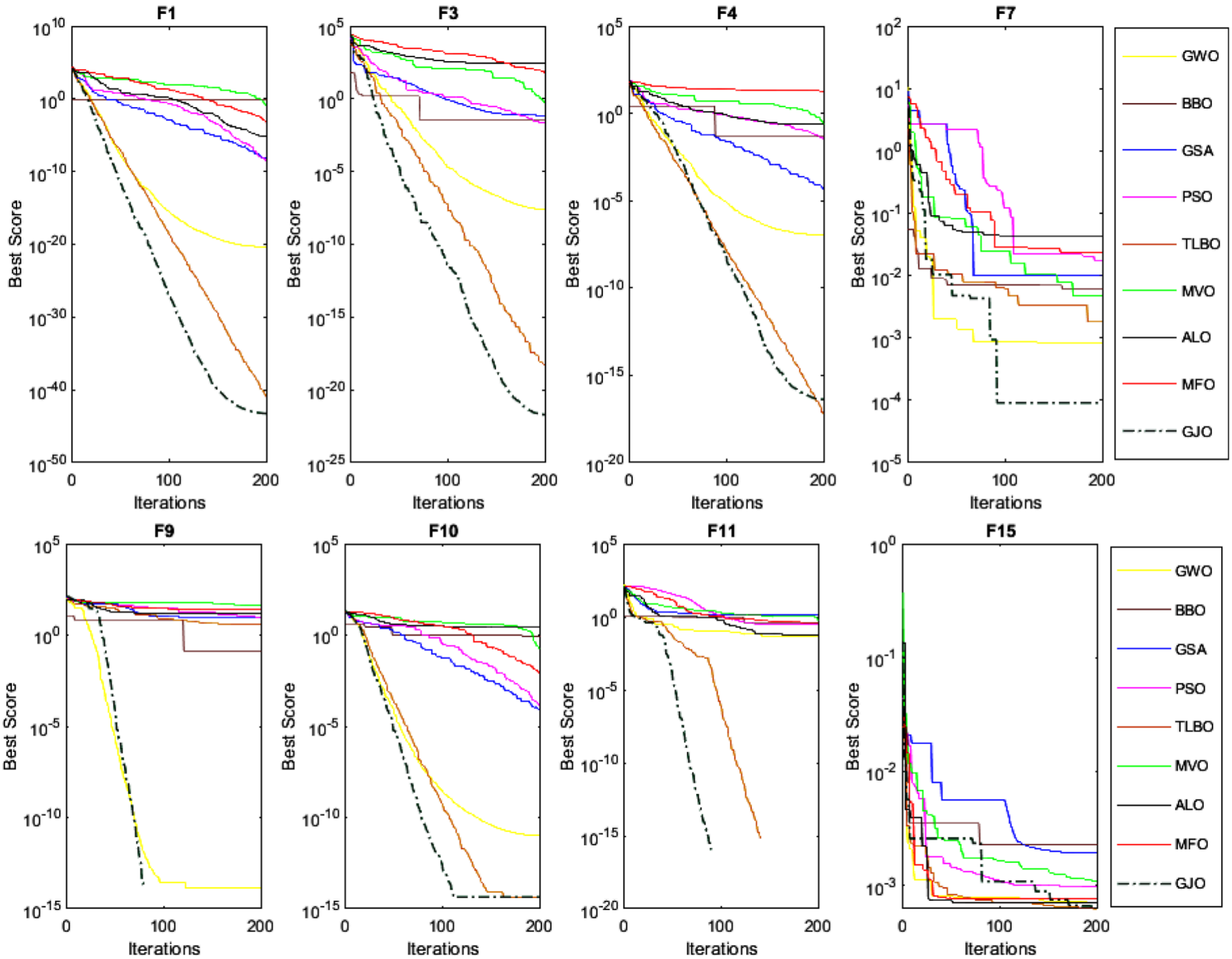

Fig. 5. Convergence curves of GJO algorithm in comparison to other algorithms on benchmark functions.

Table 3
The parameter settings of different algorithms.

Algorithm	Parameter values
PSO	$V_{max} = 6$, adaptive inertia weights: Wmax = 0.9 & $W_{min} = 0.2$, $C_1 = C_2 = 2$
GWO	Convergence constant a = [2,0]
BBO	Mutation probability = 0.05, Max immigration (I) and Max emigration (E) = 1, Habitat modification probability = 1, number of best solutions to keep from one generation to next = 4
GSA	initial gravitational constant $G_0 = 100$, alpha = 10,
TLBO	Teaching factor T = [1,2]
MVO	Wormhole Existence Probability $WEP_{Max} = 1$; $WEP_{Min} = 0.2$;

0 to 1, it indicates an improvement in the strength of prey.

The altering evading energy E declines during the iteration process as shown in Fig. 3. When $|E| > 1$, the jackal pairs search different sections for exploring prey, and when $|E| < 1$, GJO attacks the prey and performs exploitation as shown in Fig. 2.

To conclude, the search process in GJO begins with the creation of an arbitrary population of prey (candidate solutions). During iterations, the prey's likely position is estimated by male and female jackal hunting pair. Every candidate in the population updates its distance from the jackal pair. The E1 parameter is reduced from 1.5 to 0 for emphasizing exploration and exploitation, respectively. Golden Jackal hunting pair deviate from prey when $E > 1$ and congregate to the prey when $E < 1$. Lastly, the GJO algorithm is finished by the fulfilment of an end criterion. The pseudo-code of the GJO algorithm is presented in Fig. 4.

Table 4
The obtained optimal values in unimodal and multimodal benchmark test functions where population and C1 are fixed at 30 and 1.5. The iterations are varied from 100 to 1000.

Iteration	Function							
	F1	F2	F3	F7	F9	F10	F11	F15
100	6.94E-15	6.28E-09	3.27E-09	3.01E-04	2.56E-13	3.20E-08	1.12E-12	4.56E-04
300	9.83E-47	1.25E-26	1.70E-28	8.57E-05	0	4.44E-15	0	3.07E-04
500	7.64E-81	2.50E-44	2.50E-47	3.05E-05	0	4.44E-15	0	3.07E-04
1000	2.37E-160	5.72E-88	4.22E-92	3.99E-06	0	8.88E-16	0	3.07E-04

Table 5
The obtained optimal values in unimodal and multimodal benchmark test functions where iterations and C1 are fixed at 200 and 1.5. The population is varied from 30 to 100.

n	Function							
	F1	F2	F3	F7	F9	F10	F11	F15
30	6.00E-31	1.40E-17	2.95E-20	2.48E-04	0	1.51E-14	0	3.53E-04
50	5.74E-36	1.75E-20	4.47E-21	5.18E-05	0	7.99E-15	0	3.33E-04
80	5.43E-40	3.34E-22	2.79E-24	3.58E-05	0	4.44E-15	0	3.07E-04
100	1.33E-42	1.79E-23	2.51E-26	1.76E-05	0	4.44E-15	0	3.07E-04

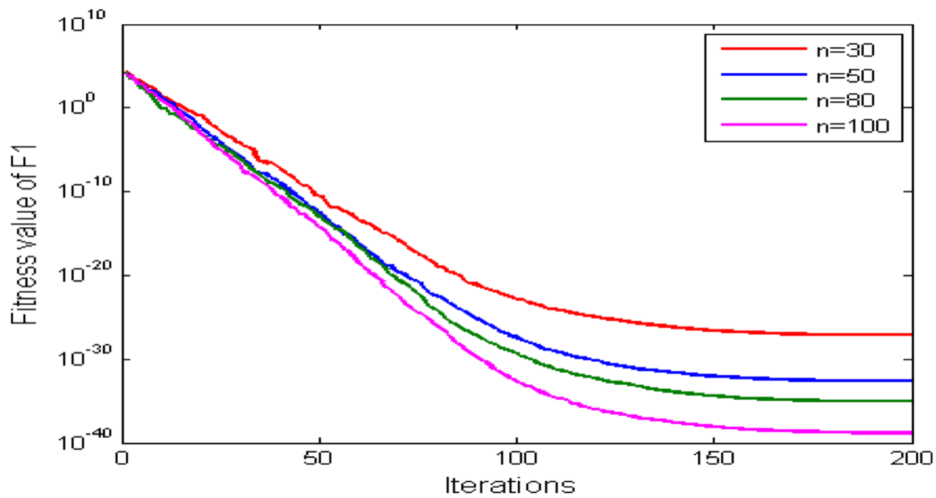

Fig. 6. Convergence curve of the GJO algorithm with different values of population for solving F1 benchmark function.

Table 6
The obtained optimal values in unimodal and multimodal benchmark test functions where iterations and population are fixed at 200 and 30. The C_1 is varied from 1 to 2.

C_1	Function							
	F1	F2	F3	F7	F9	F10	F11	F15
1	1.94E-24	1.65E-14	5.21E-15	2.33E-04	0	3.28E-13	0	3.08E-04
1.5	1.06E-31	1.33E-17	2.72E-19	3.74E-05	0	1.87E-14	0	3.10E-04
2	1.15E-32	2.93E-19	3.26E-18	5.86E-05	0	7.99E-15	0	3.20E-04

Some points worth noticing in the GJO optimization procedure are:

a) The proposed Male and female jackal pair assists GJO for saving the best solutions found during iterations.

b) The arbitrary parameters “E” and “rl” assist jackal pair to search randomly.

c) The proposed foraging strategy allows candidate solutions in locating prey’s feasible position.

d) The adaptive values of “E1” and “E” assure exploration and exploitation.

e) Only parameter "rl" and one optional parameter "C_1 " need to be adjusted in GJO.

3. Results & discussion

The GJO algorithm is tested on 23 benchmark functions (Mirjalili et al., 2014) given in Tables A1–A3 in Appendix A, where Range is the bound of the function’s search space, D specifies function’s dimensionality, and Fmin is optimum. The used benchmark functions are minimization functions classified into three categories: unimodal, multimodal and fixed-dimension multimodal functions (Mirjalili et al.,

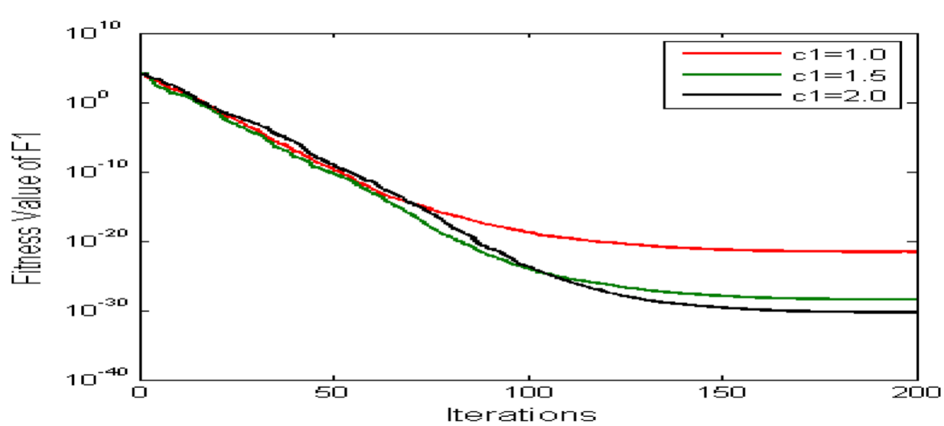

Fig. 7. Convergence curve of the GJO algorithm with different values of parameter C_1 for solving F1 benchmark function.

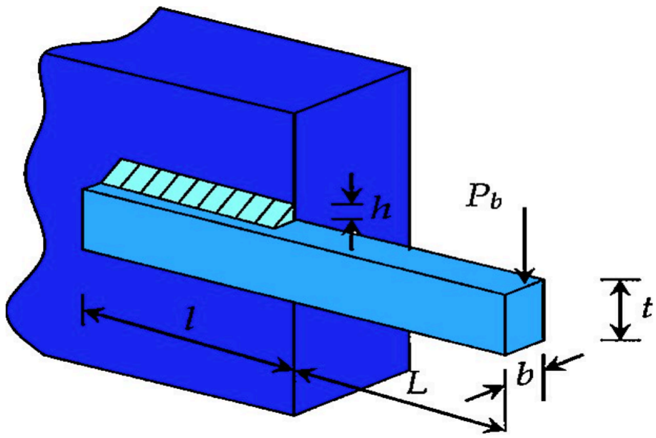

Fig. 8. The welded beam design example.

Table 7
Comparison results of the welded beam design problem.

Algorithm	Optimal values for variables				Optimal Cost
	h	l	t	b	
GJO	0.20562	3.4719	9.0392	0.20572	1.72522
MVO (Mirjalili et al., 2016)	0.205463	3.473193	9.044502	0.205695	1.72645
GSA (Mirjalili et al., 2014)	0.182129	3.856979	10.0000	0.202376	1.87995
GWO (Mirjalili et al., 2014)	0.205676	3.478377	9.03681	0.205778	1.72624
CPSO (He & Wang, 2007)	0.202369	3.544214	9.048210	0.205723	1.73148
GA (Coello and Carlos, 2000)	0.1829	4.0483	9.3666	0.2059	1.82420
GA (Deb, 2000)	0.2489	6.1730	8.1789	0.2533	2.43312
Coello (Coello and Carlos, 2002)	0.208800	3.420500	8.997500	0.2100	1.74831
Random (Ragsdell & Phillips, 1976)	0.4575	4.7313	5.0853	0.6600	4.11856
Simplex (Ragsdell & Phillips, 1976)	0.2792	5.6256	7.7512	0.2796	2.53073
David (Ragsdell & Phillips, 1976)	0.2434	6.2552	8.2915	0.2444	2.38411
APPROX (Ragsdell & Phillips, 1976)	0.2444	6.2189	8.2915	0.2444	2.38154

2014).

The GJO algorithm was run along with other algorithms for 30 times on every benchmark function with population size fixed at 30 for all algorithms. The maximum iteration count was set at 200. The parameter settings of other algorithms are given in Table 3. The results statistics (average and standard deviation) are stated in Tables 1–2. For result validation, the GJO algorithm is compared to popular metaheuristics like PSO (Fearn, 2014), GWO (Mirjalili et al., 2014), ALO (Mirjalili, 2015a), GSA (Rashedi et al., 2009), MVO (Mirjalili et al., 2016), BBO (Simon, 2008) and Teaching–Learning Based Optimization TLBO (Rao, Savsani, & Vakharia, 2011). The standard deviation and mean only compare the algorithm's overall performance, whereas a statistical test evaluates every run's result and verify that the outcomes are statistically substantial. In this work, Wilcoxon rank-sum test (Derrac, García, Molina, & Herrera, 2011; Mirjalili, 2015a) is used which is a non-parametric assessment in statistics for verifying if two sets of solutions are dissimilar statistically substantial or not. This test gives a parameter "p" whose value governs the significance level of two algorithms. An algorithm is statistically substantial if it has a p-value smaller than 0.05. The p values of GJO in comparison with other algorithms for each benchmark function are given in Tables 1–2.

3.1. Exploitation analysis

As per the results in Table 1, GJO delivers very viable results. This algorithm outperforms all others in F1, F2, F3, and F7. GJO performance was second best to that of the TLBO algorithm in F4 and better than all other mentioned algorithms. In F5, GJO gives better average results than other metaheuristics. Considering the suitability of unimodal functions for benchmarking exploitation, these results demonstrate the superiority of GJO for exploiting the optimal. The statistical significance of results is shown by the p-values<0.05 in Table 1. This is owing to the proposed exploitation operators discussed previously.

3.2. Exploration analysis

Unlike unimodal functions, multimodal functions consist of multiple local optima which makes them appropriate for testing the exploration capability of an algorithm. As per the results in Table 2, GJO performs well on multimodal functions too. In F9, GJO provides better mean value and BBO gives better standard deviation in results compared to other algorithms. GJO outperforms all other algorithms in F10, F11, and F15 benchmarks. In F16, F17, F18, F19, and F21 functions, GJO find the global solution and performance comparable to all other metaheuristics on F20, F22, and F23. These outcomes demonstrate the excellence of the GJO algorithm in terms of exploration. The convergence curve of GJO in comparison to other algorithms is shown in Fig. 5.

3.3. Run time complexity

The run time complexity of GJO depends on the three processes: initialization, fitness evaluation, and updating of golden jackals.

Note that with n jackals, the computational complexity of the initialization process is O(n). The computational complexity of the updating mechanism is $O(t \times n) + O(t \times n \times d)$, which is composed of searching for the best location and updating the location vector of all jackals, where "t" is the maximum number of iterations and d is the dimension of specific problems. Therefore, Run Time Complexity of the GJO is $O(n\times (t + td + 1))$.

3.4. Parameter analysis

The proposed GJO algorithm employs three parameters i.e., n (i.e., population), maximum number of iterations and parameter C_1.

Maximum number of iterations: GJO algorithm was run for diverse number of iteration processes. The values of maximum iteration used in this work are 100, 300, 500, and 1000. Table 4 show the variations of iterations on various benchmark test functions. The results show that GJO converges towards the optimal solution when the number of iterations is increased.

Population(n): GJO algorithm was run for different values of population (i.e., 30, 50, 80, 100). Table 5 and Fig. 6 show the variations of different number of search agents on benchmark test functions. It is analysed from Fig. 6 that the value of fitness function decreases when population number increases.

Parameter C_1: GJO algorithm was run for different values of C_1 (i.e., 1,1.5 and 2). Table 6 and Fig. 7 show the variations of C_1 on benchmark test functions. It is analysed from Fig. 7 that the value of fitness function decreases when C_1 increases.

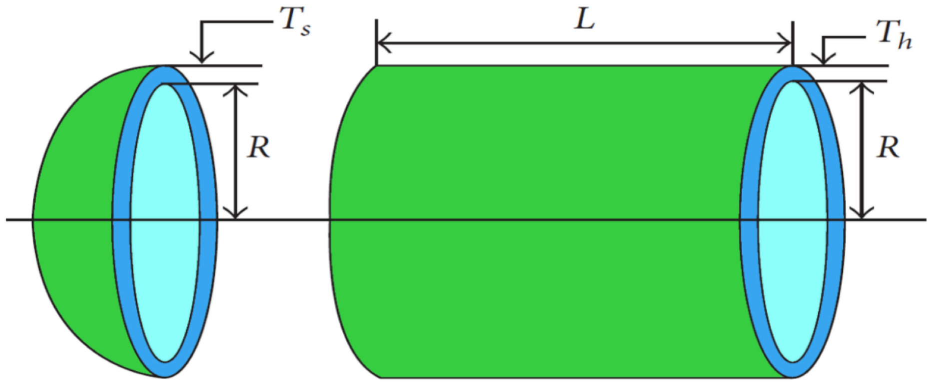

Fig. 9. The Pressure vessel design example.

Table 8
Comparison results of the pressure vessel design problem.

Algorithm	Optimal values for variables				Optimal Cost
	Ts	Th	R	L	
GJO	0.7782955	0.3848046	40.32187	200	5887.071123
MFO (Mirjalili, 2015c)	0.8125	0.4375	42.098445	176.636596	6059.7143
MVO (Mirjalili et al., 2016)	0.8125	0.4375	42.0907382	176.738690	6060.8066
GWO (Mirjalili et al., 2014)	0.812500	0.434500	42.089181	176.758731	6051.5639
GSA(Mirjalili et al., 2014)	1.1250	0.6250	55.988659	84.4542025	8538.8359
PSO (He & Wang, 2007)	0.8125	0.4375	42.091266	176.746500	6061.0777
GA (Coello and Carlos, 2000)	0.8125	0.4345	40.323900	200.000000	6288.7445
GA (Coello et al., 2002)	0.8125	0.4375	42.097398	176.654050	6059.9463
GA (Deb, 1997)	0.9375	0.5000	48.329000	112.679000	6410.3811
ES (Mezura-Montes & Coello Coello, 2008)	0.8125	0.4375	42.098087	176.640518	6059.7456
DE (Li et al., 2007)	0.8125	0.4375	42.098411	176.637690	6059.7340
ACO (Kaveh & Talatahari, 2010)	0.8125	0.4375	42.103624	176.572656	6059.0888
LM (Kannan & Kramer, 1993)	1.1250	0.6250	58.291000	43.6900000	7198.0428
Branch-bound (Sandgren, 1990)	1.1250	0.6250	47.700000	117.701000	8129.1036

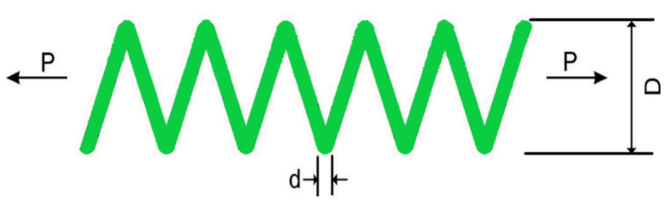

Fig. 10. Tension/compression spring design problem.

4. GJO for solving engineering problems

This section assesses the performance of GJO in real-world problems using constrained engineering benchmarks. The engineering problem includes welded beam design, pressure vessel design, tension/compression spring design weight minimization of a speed reducer, gear train design, three bar truss design and their mathematical formulation can be found in appendix B. GJO is also applied to solve practical engineering example of economic load dispatch in an electrical power system. A simple method of the death penalty is employed here to convert the constrained problems into the unconstrained ones.

4.1. Welded beam design

The first example to evaluate the performance of GJO in the engineering domain is a well-known welded beam design shown in Fig. 8. The objective is to find the best design variables for minimizing the entire manufacturing price of a welded beam subject to Bending stress (θ), Shear stress (τ), the beam end deflection (δ), the bar's buckling load (Pc) and other constraints The four variables in this problem are the thickness of weld (h), bar length (l), height (t), and thickness (b). The mathematical formulation (Mirjalili et al., 2014) is given in appendix B. The results of this problem by different algorithms are given in Table 7.

Table 9
Comparison results of tension/compression spring design problem.

Algorithm	Optimal values for variables			Optimum weight
	d	D	N	
GJO	0.0515793	0.354055	11.4484	0.01266752
MFO (Mirjalili, 2015c)	0.051994457	0.36410932	10.868421862	0.0126669
GWO (Mirjalili et al., 2014)	0.05169	0.356737	11.28885	0.012666
GSA (Mirjalili et al., 2014)	0.050276	0.323680	13.525410	0.0127022
PSO (He & Wang, 2007)	0.051728	0.357644	11.244543	0.0126747
ES (Mezura-Montes & Coello Coello, 2008)	0.051989	0.363965	10.890522	0.0126810
GA (Carlos A Coello Coello, 2000)	0.051480	0.351661	11.632201	0.0127048
HS (Lu, Gu, Zhang, & Jin, 2013)	0.051154	0.349871	12.076432	0.0126706
DE (Li et al., 2007)	0.051609	0.354714	11.410831	0.0126702

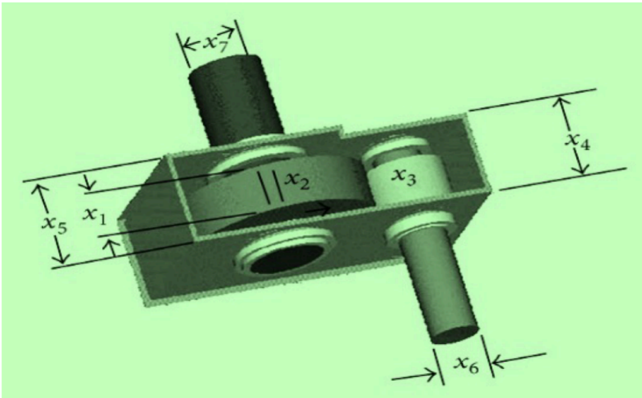

Fig. 11. Speed reducer design.

4.2. Pressure vessel design

The goal is to obtain a design for a pressure vessel with the least fabrication cost (Mirjalili et al., 2014). Fig. 9 shows the pressure vessel and parameters involved in the design. The four variables in this problem are shell thickness (Ts), head Thickness (Th), Inner radius (R), and length of the cylindrical section neglecting head(L). Problem formulation in Appendix B shows that this problem is highly constrained, so the results Table 8 evidence the merits of GJO in solving such problems.

4.3. Tension/compression spring design

The objective is again the minimization of the fabrication cost of spring with three structural parameters (Mirjalili et al., 2014): wire diameter (d), mean coil diameter (D), and the number of active coils (N). Fig. 10 shows the spring and its parameters. The results of GJO and its comparison with another algorithm like MFO (Mirjalili, 2015b); GWO (Mirjalili et al., 2014); GSA (Mirjalili et al., 2014); PSO (He & Wang, 2007); ES (Mezura-Montes & Coello Coello, 2008); GA (Coello Coello, 2000); HS (Mahdavi et al., n.d.) and DE (Li et al., n.d.) is given in Table 9.

4.4. Weight minimization of a speed reducer

It includes the designing of a speed reducer for small aircraft engine and is a difficult benchmark since it has seven design variables (x_1 to x_7) (Dhiman & Kumar, 2017). As revealed in Fig. 11 the design parameters are the face width (x_1), module of teeth (x_2), count of teeth in the pinion (x_3), first shaft's length between bearings (x_4), second shaft length between bearings (x_5), first shaft's diameter (x_6), and second shaft's diameter (x_7). The results of GJO and its comparison with another algorithms is given in Table 10. The mathematical formulation of this problem is given in appendix B.

4.5. Gear train design problem

The key objective of this problem is to minimization of the ratio of gears for the preparation of the compound gear train as shown in Fig. 12. The aim is to find the optimum number of teeth for four gears of a train for minimalizing the gear ratio (Mirjalili, 2015b). The results of this problem by different algorithms are given in Table 11. The mathematical formulation of this problem is given in appendix B.

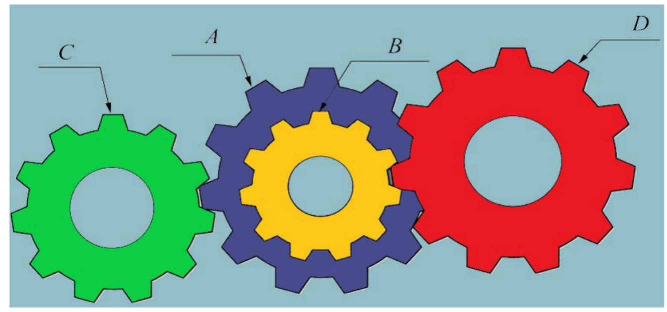

Fig. 12. Gear train design problem.

Table 11
Comparison results of Gear train design problem.

Algorithm	Optimal values for variables				Fmin
	n_A	n_B	n_C	n_D	
GJO	56.7889	18.5239	19.8299	44.8316	1.7754e-19
ALO (Mirjalili, 2015b)	49	19	16	43	2.7009e-012
CS (Gandomi et al., 2013)	43	16	19	49	2.7009e-012
MBA (Sadollah et al., 2013)	43	16	19	49	2.7009e-012
ISA (Gandomi, 2014)	N/A	N/A	N/A	N/A	2.7009e-012
GA (Wu & Chow, 1995)	N/A	N/A	N/A	N/A	2.33e-07
ALM (Kannan & Kramer, 1994)	33	15	13	41	2.1469e-08

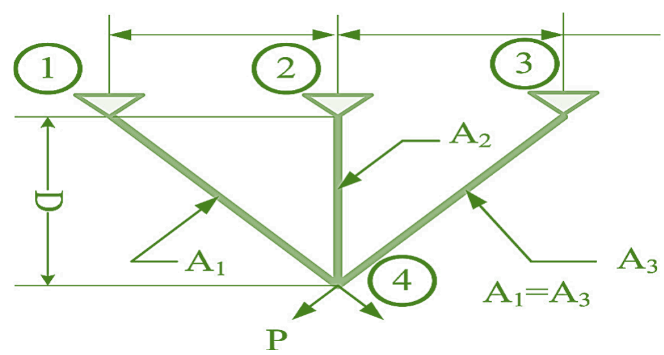

Fig. 13. Three bar truss design problem.

Table 10
Comparison results of speed reducer design problem.

Algorithm	Optimal values for variables							Optimum weight
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	
GJO	3.500003	0.7	17	7.321686	7.72122	3.35025	5.28665	2994.80495
MFPA (Meng, Pauline, Kiong, Wahab, & Jafferi, 2017)	3.5	0.7	17	7.3	7.8005	3.35021	5.28668	2996.219
SHO (Dhiman & Kumar, 2017)	–	–	–	–	–	–	–	2998.550
CS (Gandomi, Yang, & Alavi, 2013)	3.5015	0.7	17	7.6050	7.8181	3.3520	5.2875	3000.981
EA (Mezura-Montes, Coello Coello, & Landa-Becerra, 2003)	3.506163	0.700831	17	7.46018	7.962143	3.3629	5.3090	3025.005
(Akhtar, Tai, & Ray, 2002)	3.506122	0.700006	17	7.549126	7.85933	3.365576	5.289773	3008.08

Table 12
Comparison results for three bar truss design problem.

Algorithm	Optimal values for variables		Optimum Weight	Max. eval.
	Y_1	Y_2		
GJO	0.788657163482708	0.408299125193296	263.8958439	4000
ALO (Mirjalili, 2015a)	0.788662816000317	0.408283133832901	263.8958434	14,000
DEDS (Zhang, Luo, & Wang, 2008)	0.78867513	0.40824828	263.8958434	15,000
PSO-DE (Liu, Cai, & Wang, 2010)	0.7886751	0.4082482	263.8958433	17,600
MBA (Sadollah, Bahreininejad, Eskandar, & Hamdi, 2013)	0.7885650	0.4085597	263.8958522	20,000
CS (Gandomi et al., 2013)	0.78867	0.40902	263.9716	15,000

Table.13
The results obtained by the GJO for economic load dispatch.

Unit	GJO	Unit	GJO	Unit	GJO	Unit	GJO
Pg1	113.9999	Pg11	168.8007	Pg21	523.306	Pg31	190
Pg2	111.4377	Pg12	168.8071	Pg22	523.3182	Pg32	190
Pg3	97.6254	Pg13	125.0023	Pg23	523.3457	Pg33	190
Pg4	179.7438	Pg14	394.2794	Pg24	523.3426	Pg34	200
Pg5	94.41467	Pg15	394.2794	Pg25	523.3216	Pg35	200
Pg6	139.9999	Pg16	304.5198	Pg26	523.3049	Pg36	166.8621
Pg7	300	Pg17	489.2944	Pg27	10.51422	Pg37	91.53331
Pg8	284.687	Pg18	489.2851	Pg28	11.61251	Pg38	109.9944
Pg9	284.9294	Pg19	511.288	Pg29	10.60077	Pg39	90.78915
Pg10	130.0002	Pg20	511.3568	Pg30	93.10038	Pg40	511.3032
Total power output (MW)		10,500		**Total fuel cost ($/h)**			
						121586.8532	

Table 14
Comparison of results obtained for economic load dispatch with other algorithms.

Method	Minimum cost ($/h)	Average cost ($/h)	Maximum cost ($/h)
HGPSO (Ling et al., 2008)	124797.13	126855.70	NA
SPSO (Ling et al., 2008)	124350.40	126074.40	NA
PSO (Victoire & Jeyakumar, 2004)	123930.45	124154.49	NA
CEP (Sinha et al., 2003)	123488.29	124793.48	126902.89
HGAPSO (Ling et al., 2008)	122780.00	124575.70	NA
FEP (Sinha et al., 2003)	122679.71	124119.37	127245.59
MFEP (Sinha et al., 2003)	122647.57	123489.74	124356.47
IFEP (Sinha et al., 2003)	122624.35	123382.00	125740.63
TM (D. Liu & Cai, 2005)	122477.78	123078.21	124693.81
EP-SQP (Victoire & Jeyakumar, 2004)	122323.97	122379.63	NA
MPSO (Park et al., n.d.)	122252.26	NA	NA
ESO (Pereira-Neto, Unsihuay, & Saavedra, 2005)	122122.16	122524.07	123143.07
HPSOM (Ling et al., 2008)	122112.40	124350.87	NA
PSO-SQP (Victoire & Jeyakumar, 2004)	122094.67	122245.25	NA
GA_MU (Chiang, 2007)	122000.2837	NA	NA
Improved GA (Ling & Leung, n.d.,)	121915.93	122811.41	123334.00
HPSOWM (Ling et al., 2008)	121915.30	122844.40	NA
IGAMU (Chiang, 2007)	121819.25	NA	NA
HDE(Wang et al., 2007)	121813.26	122705.66	NA
PSO(Selvakumar & Thanushkodi, 2008)	121735.4736	122513.9175	123467.4086
APSO(1) (Selvakumar & Thanushkodi, 2008)	121704.7391	122221.3697	122995.0976
ST-HDE (Wang, Chiou, & Liu, 2007)	121698.51	122304.30	NA
NPSO-LRS (Selvakumar & Thanushkodi, 2007)	121664.43	122209.31	122981.59
APSO(2) (Selvakumar & Thanushkodi, 2008)	121663.5222	122153.6730	122912.3958
GJO	121586.8532	122146.9161	123095.462

4.6. Three-bar truss design problem

The objective in truss design is minimizing the burden of the bar structures. The restraints of this problem are stress, buckling and deflection constraints of each bar. The complete structure of the truss is showed in Fig. 13.The results of this problem by different algorithms are given in Table 12. The mathematical formulation of this problem is given in appendix B.

4.7. Economic load dispatch (ELD)

ELD is one of the basic engineering problems in power system planning. The main purpose of this problem is the optimal allocation of required power among available thermal units to minimize the total fuel cost while satisfying load demand, and all the power units diverse operating constraints (Sinha, Chakrabarti, & Chattopadhyay, 2003). The generator cost is formulated by quadratic functions, and the total fuel cost F(PG) in ($/h) can be expressed as:

$$F(P_g) = \sum_{i=1}^{n} \left(a_i P_{gi}^2 + b_i P_{gi} + c_i \right) \tag{14}$$

And after including valve point loading effects the equation (14) is modified as below:

$$F(P_g) = \sum_{i=1}^{n} \left(a_i P_{gi}^2 + b_i P_{gi} + c_i \right) + \left| d_i sin \left(e_i \left(P_{gi}^{min} - P_{gi} \right) \right) \right| \tag{15}$$

where fuel-cost coefficients of the i^{th} unit are a_i, b_i, and c_i, and d_i & e_i are the fuel-cost coefficients of the i^{th} unit with valve-point effects.

The proposed approach is applied to find minimum total fuel cost of the ELD problem with valve point effect neglecting power losses for a 40 thermal unit power system, which is considered with load demand of test system at value 10500 MW. The system data is given in Ref. (Sinha et al., 2003) and generator power output obtained by the GJO algorithm are shown in Table 13 and results compared with other methods are given in Table 14. The GJO was run with 5000 iterations and a population size of 200 for 30 times.

5. Conclusion

This work proposed an original population-based optimization algorithm motivated by golden jackals. The hunting strategy of golden jackals is imitated in the proposed method. To benchmark the performance of the proposed algorithm in terms of exploration and exploitation, twenty-three test functions were used. The outcomes exhibited that GJO was proficient to deliver very competitive results in comparison to famous *meta*-heuristics like GWO, PSO, MVO, GSA, BBO, TLBO, and ALO. To start with, the outcomes on the unimodal functions presented the superiority of the GJO algorithm in exploitation. Second, the exploration behaviour of GJO was established by the outcomes of multimodal functions.

Furthermore, the engineering design problems solved by GJO algorithm demonstrate that algorithm has good performance in unidentified, difficult search spaces. Finally, the GJO algorithm was used to solve a real problem in electrical engineering i.e., economic load dispatch problem. The dimensionality of the problem considered was quite high. The results on this problem exhibited an extensive enhancement in comparison to other metaheuristics, revealing the application of the proposed algorithm for solving real problems. These problems prove that GJO can demonstrate high performance in both constrained and unconstrained problems.

Numerous research directions can be suggested for studies in the future with the proposed algorithm. Solving dissimilar optimization problems in diverse fields can be done. Further, modifying this algorithm for solving multi-objective problems can be a good contribution.

CRediT authorship contribution statement

Nitish Chopra: Conceptualization, Methodology, Formal analysis, Validation, Software, Writing – original draft, Writing – review & editing. **Muhammad Mohsin Ansari:** Formal analysis, Validation, Software, 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 would like to clarify that this research work was self-funded and neither funds nor support were taken from any organization.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.eswa.2022.116924.

References:

Ivory, A. (1999). *Canis aureus*. On-Line.

Ab Wahab, M. N., Nefti-Meziani, S., & Atyabi, A. (2015). A comprehensive review of swarm optimization algorithms. *PLoS One, 10*(5). https://doi.org/10.1371/journal.pone.0122827

Abdollahzadeh, B., Gharehchopogh, F. S., & Mirjalili, S. (2021). African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems. *Computers and Industrial Engineering, 158*, Article 107408. https://doi.org/10.1016/j.cie.2021.107408

Abdollahzadeh, B., Soleimanian Gharehchopogh, F., & Mirjalili, S. (2021). Artificial gorilla troops optimizer: A new nature-inspired metaheuristic algorithm for global optimization problems. *International Journal of Intelligent Systems, 36*(10), 5887–5958. https://doi.org/10.1002/int.22535

Abedinia, O., Amjady, N., & Ghasemi, A. (2016). A new metaheuristic algorithm based on shark smell optimization. *Complexity, 21*(5), 97–116. https://doi.org/10.1002/cplx.21634

Admasu, E., Thirgood, S. J., Bekele, A., & Karen Laurenson, M. (2004). Spatial ecology of golden jackal in farmland in the Ethiopian Highlands. *African Journal of Ecology, 42*(2), 144–152. https://doi.org/10.1111/j.1365-2028.2004.00497.x

Akay, B., & Karaboga, D. (2012). Artificial bee colony algorithm for large-scale problems and engineering design optimization. *Journal of Intelligent Manufacturing, 23*(4), 1001–1014. https://doi.org/10.1007/s10845-010-0393-4

Akhtar, S., Tai, K., & Ray, T. (2002). A socio-behavioral simulation model for engineering design optimization. *Engineering Optimization, 34*(4), 341–354. https://doi.org/10.1080/03052150212723

Alba, E., & Dorronsoro, B. (2005). The exploration/exploitation tradeoff in dynamic cellular genetic algorithms. *IEEE Transactions on Evolutionary Computation, 9*(2), 126–142. https://doi.org/10.1109/TEVC.2005.843751

Anita, & Yadav, A. (2019). AEFA: Artificial electric field algorithm for global optimization. *Swarm and Evolutionary Computation, 48*, 93–108. 10.1016/j.swevo.2019.03.013.

Askarzadeh, A. (2014). Bird mating optimizer: An optimization algorithm inspired by bird mating strategies. *Communications in Nonlinear Science and Numerical Simulation, 19*(4), 1213–1228. https://doi.org/10.1016/j.cnsns.2013.08.027

Beyer, H.-G., & Schwefel, H.-P. (2002). Evolution strategies – A comprehensive introduction. *Natural Computing, 1*(1), 3–52. https://doi.org/10.1023/A:1015059928466

Chiang, C. L. (2007). Genetic-based algorithm for power economic load dispatch. *IET Generation, Transmission and Distribution, 1*(2), 261–269. https://doi.org/10.1049/iet-gtd:20060130

Coello Coello, C. A. (2000). Constraint-handling using an evolutionary multiobjective optimization technique. *Civil Engineering and Environmental Systems, 17*(4), 319–346. https://doi.org/10.1080/02630250008970288

Coello, C., & Carlos, A. (2000). Use of a self-adaptive penalty approach for engineering optimization problems. *Computers in Industry, 41*(2), 113–127. https://doi.org/10.1016/S0166-3615(99)00046-9

Coello, C., & Carlos, A. (2002). Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering, 191*(11–12), 1245–1287. https://doi.org/10.1016/S0045-7825(01)00323-1

Coello, C., Carlos, A., & Montes, E. M. (2002). Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. *Advanced Engineering Informatics, 16*(3), 193–203. https://doi.org/10.1016/S1474-0346(02)00011-3

Cuevas, E., Cienfuegos, M., Zaldívar, D., & Pérez-Cisneros, M. (2013). A swarm optimization algorithm inspired in the behavior of the social-spider. *Expert Systems with Applications, 40*(16), 6374–6384. https://doi.org/10.1016/j.eswa.2013.05.041

Deb, K. (1997). GeneAS: A Robust Optimal Design Technique for Mechanical Component Design. In *Evolutionary Algorithms in Engineering Applications* (pp. 497–514). Springer Berlin Heidelberg. 10.1007/978-3-662-03423-1_27.

Deb, K. (2000). An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering, 186*(2–4), 311–338. https://doi.org/10.1016/S0045-7825(99)00389-8

Derrac, J., García, S., Molina, D., & Herrera, F. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation, 1*(1), 3–18. https://doi.org/10.1016/j.swevo.2011.02.002

Dhiman, G., & Kumar, V. (2017). Spotted hyena optimizer: A novel bio-inspired based metaheuristic technique for engineering applications. *Advances in Engineering Software, 114*, 48–70. https://doi.org/10.1016/j.advengsoft.2017.05.014

Dorigo, M., Maniezzo, V., & Colorni, A. (1996). Ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 26*(1), 29–41. https://doi.org/10.1109/3477.484436

Dołan, B., & Ölmez, T. (2015). A new metaheuristic for numerical function optimization: Vortex Search algorithm. *Information Sciences, 293*, 125–145. https://doi.org/10.1016/j.ins.2014.08.053

Faramarzi, A., Heidarinejad, M., Mirjalili, S., & Gandomi, A. H. (2020). Marine Predators Algorithm: A nature-inspired metaheuristic. *Expert Systems with Applications, 152*. https://doi.org/10.1016/j.eswa.2020.113377

Fearn, T. (2014). Particle swarm optimisation. *NIR News, 25*(1), 27. https://doi.org/10.1255/nirn.1421

Gandomi, A. H. (2014). Interior search algorithm (ISA): A novel approach for global optimization. *ISA Transactions, 53*(4), 1168–1183. https://doi.org/10.1016/j.isatra.2014.03.018

Gandomi, A. H., Yang, X. S., & Alavi, A. H. (2013). Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems. *Engineering with Computers, 29*(1), 17–35. https://doi.org/10.1007/s00366-011-0241-y

Geem, Z. W., Kim, J. H., & Loganathan, G. V. (2001). A new heuristic optimization algorithm: Harmony search. *SIMULATION, 76*(2), 60–68. https://doi.org/10.1177/003754970107600201

Genç, H. M., Eksin, I., & Erol, O. K. (2010). Big bang - Big crunch optimization algorithm hybridized with local directional moves and application to target motion analysis problem. *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics*, 881–887. 10.1109/ICSMC.2010.5641871.

Hashim, F. A., Houssein, E. H., Hussain, K., Mabrouk, M. S., & Al-Atabany, W. (2022). Honey Badger Algorithm: New metaheuristic algorithm for solving optimization problems. *Mathematics and Computers in Simulation, 192*, 84–110. https://doi.org/10.1016/j.matcom.2021.08.013

Hatamlou, A. (2013). Black hole: A new heuristic optimization approach for data clustering. *Information Sciences, 222*, 175–184. https://doi.org/10.1016/j.ins.2012.08.023

He, Q., & Wang, L. (2007). An effective co-evolutionary particle swarm optimization for constrained engineering design problems. *Engineering Applications of Artificial Intelligence, 20*(1), 89–99. https://doi.org/10.1016/j.engappai.2006.03.003

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. https://doi.org/10.1016/j.future.2019.02.028

Heptner, V., & Naumov, N. (1998). Mammals of the Soviet Union: Sirenia and Carnivora (Sea Cows, Wolves and Bears). *Science Publishers*, *II*(Part 1a), 151–153.

Holland, J. H. (1992). Genetic algorithms. *Scientific American, 267*(1), 66–72. https://doi.org/10.1038/scientificamerican0792-66

Hosseinabadi, A. A. R., Siar, H., Shamshirband, S., Shojafar, M., & Mohd Hairul, M. H. N. (2014). Using the gravitational emulation local search algorithm to solve the multi-objective flexible dynamic job shop scheduling problem in Small and Medium Enterprises. *Annals of Operations Research, 229*(1), 451–474. https://doi.org/10.1007/s10479-014-1770-8

Hosseini, H. S. (2011). Principal components analysis by the galaxy-based search algorithm: A novel metaheuristic for continuous optimisation. *International Journal of Computational Science and Engineering, 6*(1/2), 132. https://doi.org/10.1504/ijcse.2011.041221

Husseinzadeh Kashan, A. (2015). A new metaheuristic for optimization: Optics inspired optimization (OIO). *Computers and Operations Research, 55*, 99–125. https://doi.org/10.1016/j.cor.2014.10.011

Jafari, M., Salajegheh, E., & Salajegheh, J. (2021). Elephant clan optimization: A nature-inspired metaheuristic algorithm for the optimal design of structures[Formula presented]. *Applied Soft Computing, 113*, Article 107892. https://doi.org/10.1016/j.asoc.2021.107892

Kannan, B. K., & Kramer, S. N. (1994). An augmented lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. *Journal of Mechanical Design, Transactions of the ASME, 116*(2), 405–411. https://doi.org/10.1115/1.2919393

Kannan, B. K., & Kramer, S. N. (1993). Augmented Lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. *American Society of Mechanical Engineers, Design Engineering Division (Publication) DE, 65*(pt 2), 103–112. https://asmedigitalcollection.asme.org/mechanicaldesign/article-abstract/116/2/405/454458.

Kaveh, A., & Khayatazad, M. (2012). A new meta-heuristic method: Ray optimization. *Computers and Structures, 112–113*, 283–294. https://doi.org/10.1016/j.compstruc.2012.09.003

Kaveh, A., & Talatahari, S. (2010). A novel heuristic optimization method: Charged system search. *Acta Mechanica, 213*(3–4), 267–289. https://doi.org/10.1007/s00707-009-0270-4

Kaveh, A., & Talatahari, S. (2010). An improved ant colony optimization for constrained engineering design problems. *Engineering Computations (Swansea, Wales), 27*(1), 155–182. https://doi.org/10.1108/02644401011008577

Kiran, M. S. (2015). TSA: Tree-seed algorithm for continuous optimization. *Expert Systems with Applications, 42*(19), 6686–6698. https://doi.org/10.1016/j.eswa.2015.04.055

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science, 220*(4598), 671–680. https://doi.org/10.1126/science.220.4598.671

Krause, J., Cordeiro, J., Parpinelli, R. S., & Lopes, H. S. A. (2013). A Survey of Swarm Algorithms Applied to Discrete Optimization Problems. In *Swarm Intelligence and Bio-Inspired Computation* (pp. 169–191). Elsevier Inc. 10.1016/B978-0-12-405163-8.00007-7.

Krishnanand, K. N., & Ghose, D. (2009). Glowworm swarm optimization for simultaneous capture of multiple local optima of multimodal functions. *Swarm Intelligence, 3*(2), 87–124. https://doi.org/10.1007/s11721-008-0021-5

Li, L. J., Huang, Z. B., Liu, F., & Wu, Q. H. (2007). A heuristic particle swarm optimizer for optimization of pin connected structures. *Computers and Structures, 85*(7–8), 340–349. https://doi.org/10.1016/j.compstruc.2006.11.020

Ling, S. H., Iu, H. H. C., Chan, K. Y., Lam, H. K., Yeung, B. C. W., & Leung, F. H. (2008). Hybrid particle swarm optimization with wavelet mutation and its industrial applications. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 38*(3), 743–763. https://doi.org/10.1109/TSMCB.2008.921005

Ling, S. H., & Leung, F. H. F. (n.d.). An Improved Genetic Algorithm with Average-Bound Crossover and Wavelet Mutation Operations. In *Springer*.

Liu, D., & Cai, Y. (2005). Taguchi method for solving the economic dispatch problem with nonsmooth cost functions. *IEEE Transactions on Power Systems, 20*(4), 2006–2014. https://doi.org/10.1109/TPWRS.2005.857939

Liu, H., Cai, Z., & Wang, Y. (2010). Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization. *Applied Soft Computing Journal, 10*(2), 629–640. https://doi.org/10.1016/j.asoc.2009.08.031

Liu, Q., Wu, L., Xiao, W., Wang, F., & Zhang, L. (2018). A novel hybrid bat algorithm for solving continuous optimization problems. *Appl Soft Comput, 73*. https://doi.org/10.1016/j.asoc.2018.08.012

Lu, J., Gu, J., Zhang, S., & Jin, Z. (2013). An improved harmony search algorithm for continuous optimization problems. *Proceedings - International Conference on Natural Computation, 402–406*. https://doi.org/10.1109/ICNC.2013.6818009

Macdonald, D. W. (1979). The flexible social system of the golden jackal, Canis aureus. *Behavioral Ecology and Sociobiology, 5*(1), 17–38. https://doi.org/10.1007/BF00302692

Mehrabian, A. R., & Lucas, C. (2006). A novel numerical optimization algorithm inspired from weed colonization. *Ecological Informatics, 1*(4), 355–366. https://doi.org/10.1016/j.ecoinf.2006.07.003

Meng, O. K., Pauline, O., Kiong, S. C., Wahab, H. A., & Jafferi, N. (2017). Application of modified flower pollination algorithm on mechanical engineering design problem. *IOP Conference Series: Materials Science and Engineering, 166*(1), Article 012032. https://doi.org/10.1088/1757-899X/165/1/012032

Meng, Z., & Pan, J. S. (2016). Monkey King Evolution: A new memetic evolutionary algorithm and its application in vehicle fuel consumption optimization. *Knowledge-Based Systems, 97*, 144–157. https://doi.org/10.1016/j.knosys.2016.01.009

Mezura-Montes, E., & Coello Coello, C. A. (2008). An empirical study about the usefulness of evolution strategies to solve constrained optimization problems. *Article in International Journal of General Systems, 37*(4), 443–473. https://doi.org/10.1080/03081070701303470

Mezura-Montes, E., Coello Coello, C. A., & Landa-Becerra, R. (2003). Engineering Optimization Using a Simple Evolutionary Algorithm. In *Proceedings of the International Conference on Tools with Artificial Intelligence* (pp. 149–156). https://doi.org/10.1109/tai.2003.1250183

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

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

Mirjalili, S. (2016). SCA: A Sine Cosine Algorithm for solving optimization problems. *Knowledge-Based Systems, 96*, 120–133. https://doi.org/10.1016/j.knosys.2015.12.022

Mirjalili, S., Mirjalili, S. M., & Hatamlou, A. (2016). Multi-Verse Optimizer: A nature-inspired algorithm for global optimization. *Neural Computing and Applications, 27*(2), 495–513. https://doi.org/10.1007/s00521-015-1870-7

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software, 69*, 46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007

Moehlman, P. D. (1983). Socioecology of silverbacked and golden jackals (Canis mesomelas and Canis aureus). *Advances in the Study of Mammalian Behavior, 7*, 423–453.

Moehlman, P. D., & Hayssen, V. (2018). Canis aureus (Carnivore: Canidae). *Mammalian Species, 50*(957), 14–25. https://doi.org/10.1093/mspecies/sey002

Moscato, P., Mendes, A., & Berretta, R. (2007). Benchmarking a memetic algorithm for ordering microarray data. *BioSystems, 88*(1–2), 56–75. https://doi.org/10.1016/j.biosystems.2006.04.005

Mühlenbein, H., Gorges-Schleuter, M., & Krämer, O. (1988). Evolution algorithms in combinatorial optimization. *Parallel Computing, 7*(1), 65–85. https://doi.org/10.1016/0167-8191(88)90098-1

Park, J., Lee, K., … J. S.-I. T. on, & 2005, undefined. (n.d.). A particle swarm optimization for economic dispatch with nonsmooth cost functions. *Ieeexplore.Ieee.Org.*

Passino, K. M. (2002). Biomimicry of bacterial foraging for distributed optimization and control. *IEEE Control Systems, 22*(3), 52–67. https://doi.org/10.1109/MCS.2002.1004010

Pereira-Neto, A., Unsihuay, C., & Saavedra, O. R. (2005). Efficient evolutionary strategy optimisation procedure to solve the nonconvex economic dispatch problem with generator constraints. *IEE Proceedings: Generation, Transmission and Distribution, 152*(5), 653–660. https://doi.org/10.1049/ip-gtd:20045287

Ragsdell, K. M., & Phillips, D. T. (1976). Optimal design of a class of welded structures using geometric programming. *Journal of Manufacturing Science and Engineering, Transactions of the ASME, 98*(3), 1021–1025. https://doi.org/10.1115/1.3438995

Rao, R. V., Savsani, V. J., & Vakharia, D. P. (2011). Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems. *CAD Computer Aided Design, 43*(3), 303–315. https://doi.org/10.1016/j.cad.2010.12.015

Rashedi, E., Nezamabadi-pour, H., & Saryazdi, S. (2009). GSA: A gravitational search algorithm. *Information Sciences, 179*(13), 2232–2248. https://doi.org/10.1016/j.ins.2009.03.004

Rocca, P., Oliveri, G., & Massa, A. (2011). Differential evolution as applied to electromagnetics. *IEEE Antennas and Propagation Magazine, 53*(1), 38–49. https://doi.org/10.1109/MAP.2011.5773566

Sadollah, A., Bahreininejad, A., Eskandar, H., & Hamdi, M. (2013). Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems. *Applied Soft Computing Journal, 13*(5), 2592–2612. https://doi.org/10.1016/j.asoc.2012.11.026

Sandgren, E. (1990). Nonlinear integer and discrete programming in mechanical design optimization. *Journal of Mechanical Design, Transactions of the ASME, 112*(2), 223–229. https://doi.org/10.1115/1.2912596

Saremi, S., Mirjalili, S., & Lewis, A. (2017). Grasshopper Optimisation Algorithm: Theory and application. *Advances in Engineering Software, 105*, 30–47. https://doi.org/10.1016/j.advengsoft.2017.01.004

Selvakumar, A. I., & Thanushkodi, K. (2007). A new particle swarm optimization solution to nonconvex economic dispatch problems. *IEEE Transactions on Power Systems, 22*(1), 42–51. https://doi.org/10.1109/TPWRS.2006.889132

Selvakumar, A. I., & Thanushkodi, K. (2008). Anti-predatory particle swarm optimization: Solution to nonconvex economic dispatch problems. *Electric Power Systems Research, 78*(1), 2–10. https://doi.org/10.1016/j.epsr.2006.12.001

Simon, D. (2008). Biogeography-based optimization. *IEEE Transactions on Evolutionary Computation, 12*(6), 702–713. https://doi.org/10.1109/TEVC.2008.919004

Sinha, N., Chakrabarti, R., & Chattopadhyay, P. K. (2003). Evolutionary programming techniques for economic load dispatch. *IEEE Transactions on Evolutionary Computation, 7*(1), 83–94. https://doi.org/10.1109/TEVC.2002.806788

Steffan, N., & Heydt, G. T. (2012). Quadratic programming and related techniques for the calculation of locational marginal prices in distribution systems. *North American Power Symposium (NAPS), 2012*, 1–6. https://doi.org/10.1109/NAPS.2012.6336310

Uymaz, S. A., Tezel, G., & Yel, E. (2015). Artificial algae algorithm (AAA) for nonlinear global optimization. *Applied Soft Computing Journal, 31*, 153–171. https://doi.org/10.1016/j.asoc.2015.03.003

Victoire, T. A. A., & Jeyakumar, A. E. (2004). Hybrid PSO-SQP for economic dispatch with valve-point effect. *Electric Power Systems Research, 71*(1), 51–59. https://doi.org/10.1016/j.epsr.2003.12.017

Wang, S. K., Chiou, J. P., & Liu, C. W. (2007). Non-smooth/non-convex economic dispatch by a novel hybrid differential evolution algorithm. *IET Generation, Transmission and Distribution, 1*(5), 793–803. https://doi.org/10.1049/iet-gtd:20070183

Wehrens, R., & Buydens, L. M. C. (2000). Classical and Nonclassical Optimization Methods. In *Encyclopedia of Analytical Chemistry*. John Wiley & Sons, Ltd. 10.1002/9780470027318.a5203.

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Trans Evol Comput, 1*. https://doi.org/10.1109/4235.585893

Wu, S. J., & Chow, P. T. (1995). Genetic algorithms for nonlinear mixed discrete-integer optimization problems via meta-genetic parameter optimization. *Engineering Optimization, 24*(2), 137–159. https://doi.org/10.1080/03052159508941187

Wyman, J. (1967). The jackals of the Serengeti. *Animals*, 79–83. https://ci.nii.ac.jp/naid/20001148889/.

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

Yang, X. S., & Deb, S. (2009). Cuckoo search via Lévy flights. In *2009 World Congress on Nature and Biologically Inspired Computing*. https://doi.org/10.1109/NABIC.2009.5393690

Yao, X., Liu, Y., & Lin, G. (1999). Evolutionary programming made faster. *IEEE Transactions on Evolutionary Computation, 3*(2), 82–102. https://doi.org/10.1109/4235.771163

Zhang, M., Luo, W., & Wang, X. (2008). Differential evolution with dynamic stochastic selection for constrained optimization. *Information Sciences, 178*(15), 3043–3074. https://doi.org/10.1016/j.ins.2008.02.014