A fitness dependent salp swarm algorithm

1st Danilo Pelusi
Faculty of Communication Sciences
University of Teramo
Teramo, Italy
dpelusi@unite.it

2nd Raffaele Mascella
Faculty of Communication Sciences
University of Teramo
Teramo, Italy
rmascella@unite.it

3rd Luca Tallini
Faculty of Communication Sciences
University of Teramo
Teramo, Italy
ltallini@unite.it

***Abstract*—Salp Swarm Algorithm (SSA) is a novel swarm technique using to optimize design problems. SSA is inspired by the swarming behavior of salp observed in the deep area of sea. In spite of its application versatility, SSA suffers from mediocre convergence rate and limited exploratory capabilities. In this paper, a Fitness Dependent Salp Swarm Algorithm (FDSSA) is proposed. The novelties of the proposed approach are the definition of a fitness coefficient able to enhance the exploration, the introduction of a novel mathematical model to describe the trajectory of salps and a mutation mechanism to increase the convergence speed. The designed algorithm is tested on unimodal and multimodal benchmark functions and then compared with well-known heuristic algorithms. The results show the superiority of FDSSA with respect to the comparison algorithms in terms of optimization and convergence performances according to their computational complexity.**

***Index Terms*—Evolutionary Computation, Swarm Intelligence, Salp Swarm Algorithm, Optimization Algorithms, Mutation.**

I. INTRODUCTION

The optimization techniques are applied to solve designing problems on different areas. However, there is no an universal optimization algorithm able to solve all problems [1]. Because the application of exhaustive search techniques leads to huge computational costs, several nature-inspired optimization algorithms have been proposed in literature. The feature of these algorithms in solving the problems is the powerful and robust search ability. Among nature-inspired algorithms, the swarm search-based algorithms work with a population-based technique. At each iteration, the current solutions are produced according to historical information obtained by previous generations. Some of well-known swarm algorithms are: Particle Swarm Optimization (PSO) algorithm [2], Bat Algorithm (BA) [3], Artificial Bee Colony (ABC) algorithm [4], Moth-Flame Optimization (MFO) algorithm [5], Gray Wolf Optimizer (GWO) algorithm [6]. PSO is inspired by the behavior of social organisms in groups and it moves the population around looking for a potential solution. The echolocation behavior of the microbat is the main features of BA. ABC is a population-based algorithm that simulates the foraging behaviors of honey bees. MFO is based on the transverse orientation of moths with respect the moon. The hunting mechanism of gray wolves is reproduced in GWO.

Among population-based algorithms, some heuristic optimization algorithms are inspired by Physics. The Multi-Verse Optimizer (MVO) is a population-based algorithm inspired by the multiverse theory (white hole, black hole, and wormhole) [7]. In order to search a perfect state of a sound harmony, the frequency, timbre and amplitude are optimized by the Harmony Search (HS) algorithm [8].

A novel population-based optimization algorithm is the Salp Swarm Algorithm [11] which mimics the predatory behavior of salp swarm. SSA has a simple inspiration, few controlling parameters and adaptive exploratory behavior. This makes SSA a search technique widely used to solve a huge variety of optimization problems. A mutated SSA was utilized to assign the photovoltaic and shunt capacitors in the distribution systems [12]. Tubishat et al. [13] proposed an improved version of SSA to solve feature selection problems and select the optimal subset of features in wrapper-mode. An enhanced SSA was designed for improving the peak power point tracking and fault-ride through ability enhancement of a grid-tied permanent magnet synchronous generator [14]. In order to tackle feature selection problems more efficiently, a binary SSA with crossover scheme was proposed [15]. Ateya et al. [16] developed a chaotic SSA to get the optimal number of controllers and the best allocations of switches with the available controllers for large scale software-defined networking enabled networks. A novel improved-salp swarm optimization technique for optimizing gain parameters of type-II fuzzy PID controller was proposed by Sahu et al. [17]. A quantum-behaved and wavelet mutation SSA shown excellent solutions on constrained engineering problems [18]. Ma et al. [19] designed a comprehensive improved SSA for solving redundant container deployment. To solve the problem of worse segmentation effects for segmenting images according to the pixel, a suitable SSA was proposed [20]. The Simplified Salp Swarm Algorithm (SSSA) [21] used a random search radius to optimize the leader search range.

The main defect of SSA is that it suffers from a problem in exploitation which leads to a slow convergence rate [22]. Moreover, SSA has limited exploratory capacities with stagnation to local optimum. In order to overcome these limitations, some revised versions of SSA have been proposed. Rizk-Allah et al. [25] proposed a new binary version of SSA based on the improved Arctan transformation. A new control parameter was added to SSA to tune the current optimal solution [23]. Sayed et al. [24] proposed a chaos-induced SSA where chaotic variables are used to replace the random variables.

The challenge is to improve the exploration phase with

978-1-7281-6929-3/20/$31.00 ©2020 IEEE

a suitable time-independent parameter. Here, the idea is to exploit the information which comes from the computed fitness function values. In this way, the salps position depends on fitness change and the problem of slow convergence rate can be overcome.

In this paper, a Fitness Depended Salp Swarm Algorithm (FDSSA) is designed. The contributions of the proposed approach are: the introduction of a suitable fitness coefficient which enhances the exploration phase; the definition of a novel mathematical model to replicate the salps trajectory; the application of mutation mechanism to escape from local optimum and enhancing the population diversity.

The rest of the paper is organized as follows. Section II contains the description of SSA mathematical model. The proposed algorithm is described in Section III. Section IV illustrates the FDSSA results. The conclusions are contained in Section V.

II. Mathematical model of SSA

SSA is a population-based optimization method in which the target of salp swarm is a food search in the search space. Each salp updates its location in the search space whether it is the first salp in the chain (called leader) or a follower. The leader moves towards the target food source, whereas each follower can move towards other salps.

The presence of food in the space is represented by $F = (F_1, ..., F_d)$, where d is the spacial dimension. Moreover, the position of i-th salp in the j-th dimension at iteration t is referred as $X_j^i(t)$. Let n be the number of salps, the population is generated between lower bound lb and upper bound ub of search space through the equation

$$X_j^i(t) = lb_i + (ub_i - lb_i)R \tag{1}$$

with $i = 1, 2, ..., n$ and $j = 1, 2, ..., d$. The quantity R is a random number between 0 and 1.

The position of the leader is updated by

$$X_j^1 = \begin{cases} F_j \quad + \quad c_1((ub_j - lb_j)c_2 + lb_j), & c_3 \geq 0.5 \\ F_j \quad - \quad c_1((ub_j - lb_j)c_2 + lb_j), & c_3 < 0.5 \end{cases} \tag{2}$$

where X_j^1 shows the leader position in the j-th dimension, F_j is the food source position in the j-th dimension; ub_j and lb_j are upper bound and lower bound at j-th dimension respectively. The coefficients c_2 and c_3 are random numbers uniformly generated in $[0, 1]$. The balance between exploration and exploitation is assured by c_1 defined as

$$c_1 = 2\exp^{(-4t/T)^2} \tag{3}$$

where t is the current iteration and T is the max number of iterations. To update the followers position x, the Newton's law of motion is used:

$$x = \frac{1}{2}at^2 + v_0 t \tag{4}$$

where

$$a = \frac{(v_{final} - v_0)}{\Delta t}$$

is the acceleration of the follower. Moreover, v_{final} and v_0 are referred as final and initial speed of the follower, respectively. Because the follower follows the movement of the previous salp close to itself, it follows that

$$v_{final} = \frac{(X_j^{i-1}(t) - X_j^i(t))}{\Delta t}$$

Because the time in optimization is iteration, it follows that $\Delta t = (t + 1) - t = 1$. Considering $v_0 = 0$, the equation(4) can be rewritten as

$$x = \frac{1}{2}(X_j^{i-1}(t) - X_j^i(t)) \tag{5}$$

Therefore, the position of i-th follower in the dimension j at iteration $t + 1$ is $X_j^i(t + 1) = X_j^i(t) + x$, thus

$$X_j^i(t + 1) = \frac{1}{2}(X_j^i(t) + X_j^{i-1}(t)) \tag{6}$$

with $i \geq 2$ and $j \in [1, d]$.

The search process of SSA is shown in Algorithm 1.

Finally, the main features of SSA are the following. SSA saves the best solution obtained so far and assigns it to the food source variable. SSA updates the position of leading salp with respect to the food source only. In this way, the leader always explores and exploits the space around it. Moreover, gradual movements of the followers prevent SSA from premature convergence.

Algorithm 1: Salp Swarm Algorithm

Input: n,d,T
Output: F
begin
  Initialize the population of salps with the equation (1);
  while *end condition is not satisfied* **do**
    Evaluate the fitness of each search agent;
    F=the best search agent;
    Compute the coefficient c_1 through (3);
    for *each salp* i *from* 1 *to* n **do**
      if $i == 1$ **then**
        The position of leading salp is updated using equation (2);
      end
      else
        The position of i-th follower is updated using equation (5);
      end
    end
    Amend the salps based on the upper and lower bounds of variables;
  end
end

III. Proposed algorithm

One of the defects of SSA is the limited exploration capability. In order to improve the SSA exploration phase, a fitness dependent coefficient is defined. Let f_{best} be the best value of fitness so far, the fitness weight of i-th leader w_i is defined as

$$w_i = \begin{cases} \left|\frac{f_{best}}{fit_i}\right| & fit_i \neq 0 \\ 1 & fit_i = 0 \end{cases} \tag{7}$$

where fit_i is the current fitness value of i-th leader. Because the coefficient c_1 defined in (3) affects the exploration, it is defined as

$$c_1 = 4 w_i \exp^{(-4t/T)^2} \tag{8}$$

In this way, the randomization of the algorithm is increased through the fitness values of leader. On the other hand, if the number of leaders increases then the algorithm randomization degree is increased. However, this fact causes a decreasing of the algorithm stability. Therefore, to balance randomness and stability of the algorithm, the first $n/2$ salps are selected as leaders whereas the latter $n/2$ salps are the followers.

In deep oceans, salps often form a swarm called salp chain [26]. Observing the salp swarm trajectory [27], [28], note that it is similar to a spiral shape. In view of the above, the proposed approach matches each follower to a leader and the followers spin around the matched leader by using a spiral shaped path. Thus, the position of i-th follower is defined by

$$X_j^i(t+1) = |X_j^l(t) - X_j^i(t)| e^t \cos(2\pi t) + X_j^l(t) \tag{9}$$

where $X_j^l(t)$ is the position of the l-th leader, with $i \in [n/2+1, n]$ and $l \in [1, n/2]$. Note that, the magnitude of the spiral line depends on the distance $|X_j^l(t) - X_j^i(t)|$ between follower and leader.

In order to improve the convergence rate of SSA, a mutation mechanism is proposed. In fact, a suitable mutation technique assures of escaping from local optimum and enhancing the diversity in population. The proposed mutation scheme is defined just for followers which are mutated by using the differential evolution mutation [29]. Among the various mutation strategies [30], the DE/best/2 has been used in this work. The idea is to consider the best position so far X_{best} as target vector and two perturbation difference vectors. Thus, the position of the $i\text{-}th$ follower (with $i \in [n/2+1, n]$) is mutated by means of the equation

$$X_j^i(t+1) = X_{best} + R(X_j^{i_1}(t) - X_j^{i_2}(t)) + R(X_j^{i_3}(t) - X_j^{i_4}(t)) \tag{10}$$

where R is a random number in $[0,1]$, i_1, i_2, i_3 and i_4 are integer random numbers in $[n/2+1, n]$ not equal to i. Therefore, if a follower is not selected as the best salp, it is mutated by (10) and added to the population for next generation.

Algorithm 2: Fitness Dependent Salp Swarm Algorithm

Input: n,d,T
Output: f_{best}
begin
  Initialize the population of salps with the equation (1);
  Evaluate the fitness of salps using objective function;
  while $t < T$ **do**
    for *each salp* i *from* 1 *to* n **do**
      if $i < n/2 + 1$ **then**
        Calculate the fitness weight with (7);
        Compute the coefficient c_1 through (8);
        The position of i-th leader is updated using equation (2);
        Evaluate the fitness fit_i of leader using objective function;
        if $fit_i < f_{best}$ **then**
          $f_{best} = fit_i$;
        end
      end
      else
        The position of i-th follower is updated using equation (9);
        Evaluate the fitness fit_i of follower using objective function;
        if $fit_i < f_{best}$ **then**
          $f_{best} = fit_i$;
        end
        else
          Apply mutation on the i-th follower for the next generation using equation (10)
        end
      end
    end
    $t = t + 1$;
  end
end

The steps of the proposed FDSSA are resumed in the Algorithm 2.

The time complexity of SSA depends on the number of iteration T, the number of search agents n and the dimension d of search space. SSA utilizes random algorithm to generate random number. Because there are many different algorithms to generate random numbers, the proposed computational complexity analysis does not consider the cost of such algorithms. To obtain a strict complexity analysis, the big-O notation is used [9], [10]. Following the approaches of [11], [28], [19], the time complexity of FDSSA is the same of SSA, i.e. $O(T(nd + nC_f))$, where C_f is the cost of objective function.

IV. Experimental results

The proposed algorithm is compared with the well-known heuristic algorithms PSO, MVO, BA, HS, ABC, SSA and SSSA. This choice depends on their common population-based features. Moreover, the source code sharing of these algorithms provides a better comparison among algorithms. All the algorithms are tested on the benchmark functions of Tables I and II. In particular, Table I shows the unimodal benchmark functions, whereas the functions in Table II are multimodal functions with variable dimension. Generally, the test functions are divided in unimodal and multi-modal functions. Because the unimodal benchmark functions have one global optimum and no local optima, they are used to test the exploitation level and the convergence of the algorithm. On the other hand, multimodal benchmark functions have a massive number of local optima, and they are used to test the local optima avoidance and exploration levels.

In order to achieve meaningful statistical results, the heuristic algorithms must run on a problem at least 10 times. The metrics of their performances are evaluated on average and standard deviation of the best obtained solution in the last iteration. Here, average and standard deviation are calculated on 30 runs. Moreover, the number of agents is equal to 30 and the number of iteration is 1000 (i.e. $n = 30$ and $T = 1000$) for all the considered algorithms. This fact assures the same swarm size for PSO, MVO, BA, HS, ABC, SSA and SSSA. The Table III shows the parameters settings for the involved algorithms.

Statistical tests are essential to check significant improvements by a proposed algorithm over existing methods. For a suitable comparison among algorithms, the Friedman rank test [31] is applied on the mean solutions obtained by FDSSA and competitor algorithms.

The simulation results are shown in Table IV. Note that, FDSSA achieves the best solutions with respect to the competitor algorithms on the test functions $F1$, $F3$ and $F5$. For the functions $F4$, $F6$, $F11$, $F12$ and $F13$, FDSSA is the second-best algorithm. On remaining functions, the proposed algorithm achieves competitive results. Except for PSO, the Friedman rank test shows that FDSSA is the best algorithm (see Table IV). On the other hand, PSO has a greater computational complexity than FDSSA. In fact, the computational complexity of PSO is $O(T(n^2d + n^2C_f))$, versus $O(T(nd + nC_f))$ of FDSSA.

The improvements of FDSSA with respect to SSA are over the exploration phase. In fact, with the definition of the fitness weight w_i, the exploration is enhanced and the premature convergence is avoided. Remind that one of the main defects of SSA is the mediocre convergence rate. Figures 1, 2, 3, 4, 5, 6, 7 show the convergence trend of the algorithms. Note that, the proposed algorithm provides the best convergence speed. FDSSA convergence curves shows an excellent exploitation. This is thanks to the novel mathematical model for the followers positions and suitable mutant mechanism. In other terms, the proposed contributions improve exploration and convergence rate at the same computational complexity of SSA.

TABLE I
UNIMODAL FUNCTIONS

Unimodal functions	S
$F1(X) = \sum_{i=1}^{d} x_i^2$	$[-100; 100]^d$
$F2(X) = \sum_{i=1}^{d} \lvert x_i\rvert + \prod_{i=1}^{D} \lvert x_i\rvert$	$[-10; 10]^d$
$F3(X) = \sum_{i=1}^{d} \left(\sum_{j=1}^{i} x_j\right)^2$	$[-100; 100]^d$
$F4(X) = \max_i\{\lvert x_i\rvert, 1 \leq i \leq d\}$	$[-100; 100]^d$
$F5(X) = \sum_{i=1}^{d-1} \left[100\,(x_{i+1} - x_i)^2 + (x_i - 1)^2\right]$	$[-30; 30]^d$
$F6(X) = \sum_{i=1}^{d} ([x_i + 0.5])^2$	$[-100; 100]^d$
$F7(X) = \sum_{i=1}^{d} i \cdot x_i^4 + random[0, 1)$	$[-1.28; 1.28]^d$

TABLE II
MULTIMODAL FUNCTIONS WITH VARIABLE DIMENSION

Multimodal functions	S
$F8(X) = \sum_{i=1}^{d} -x_i \sin \sqrt{\lvert x_i\rvert}$	$[-500; 500]^d$
$F9(X) = \sum_{i=1}^{d} \left[x_i^2 - 10\cos(2\pi x_i + 10)\right]$	$[-5.12; 5.12]^d$
$F10(X) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{d} x_i^2}\right) + -\exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)\right) + 20 + e$	$[-32; 32]^d$
$F11(X) = \frac{1}{4000}\sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600; 600]^d$
$F12(X) = \frac{\pi}{d}\{10\sin(\pi y_1) + \sum_{i=1}^{d-1}(y_i - 1)^2[1+ +10\sin^2(\pi y_{i+1})] + (y_d - 1)^2\}+ +\sum_{i=1}^{d} u(x_i, 10, 100, 4)$, $y_i = 1 + \frac{x_i+1}{4}$, $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a \\ 0, & a < x_i < a \\ k(-x_i - a)^m, & x_i < -a \end{cases}$	$[-50; 50]^d$
$F13(X) = 0.1\{\sin^2(3\pi x_1) + \sum_{i=1}^{D}(x_i - 1)^2[1+ +\sin^2(3\pi x_1 + 1)] + (x_d - 1)^2[1 + \sin^2(2\pi x_d)]\}+ +\sum_{i=1}^{d} u(x_i, 5, 100, 4)$	$[-50; 50]^d$

TABLE III
PARAMETERS SETTINGS FOR INVOLVED ALGORITHMS

Algorithm	Parameters
PSO	$w_{min} = 0.4$, $w_{max} = 0.9$, $c_1 = 2$, $c_2 = 2$
MVO	$WEP_{min} = 0.2$, $WEP_{max} = 1$
BA	$f_{min} = 0$, $f_{max} = 2$, $\alpha = 0.5$, $\gamma = 0.5$, $r_0 = 0.001$
HS	$bw = 0.2$, $HMCR = 0.95$, $PAR = 0.3$
ABC	$L = \lfloor 0.6nd \rceil$, $a = 1$
SSA	$c_2, c_3 \in [0, 1]$
SSSA	$c_2, c_3 \in [0, 1]$
FDSSA	$c_2, c_3 \in [0, 1]$

V. CONCLUSIONS

In this paper, a fitness dependent salp swarm algorithm is designed. The proposed scheme has been tested on unimodal and multimodal benchmark functions. To asses the algorithm performances, it has been compared with well-known heuristic algorithms. The results show that the fitness factor used to update the leader positions enhances the exploration. In this way, the premature convergence is avoided. Moreover, the convergence problems of SSA are overtaken thanks to the proposed mathematical model for the followers positions and its mutant mechanism. Thus, the exploitation phase is improved. In other terms, the results show the superiority of FDSSA with respect to the comparison algorithms in terms of optimization and convergence performances according to their computational complexity.

The future research task will focus on finding the best trade-off between exploration and exploitation by tuning the leader position. Moreover, FDSSA will be applied to other benchmark functions and engineering design problems.

ACKNOWLEDGMENT

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

REFERENCES

[1] D.H. Wolpert , W.G. Macready, . “No free lunch theorems for optimization”, IEEE Trans. Evol. Comput. 1, 67-82, 1997.

TABLE IV
COMPARISON RESULTS FOR THE SELECTED BENCHMARK FUNCTIONS

		PSO	MVO	BA	HS	ABC	SSA	SSSA	FDSSA
$F1$	mean	$1.666e-03$	$3.416e-03$	$3.598e+04$	$4.088e+03$	$4.342e-01$	$1.336e+02$	$1.718e-01$	$8.804e-11$
	std	$1.792e-03$	$1.495e-03$	$6.931e+03$	$1.082e+03$	$2.755e-01$	$9.851e+01$	$1.813e-01$	$1.523e-11$
$F2$	mean	$2.448e-14$	$1.792e-02$	$3.168e+06$	$8.107e-01$	$6.967e-07$	$4.443e-01$	$1.605e+06$	$8.161e-03$
	std	$3.306e-14$	$5.566e-03$	$1.091e+07$	$5.408e-01$	$3.966e-07$	$5.178e-01$	$4.054e+06$	$4.093e-02$
$F3$	mean	$5.774e-06$	$3.140e-02$	$9.084e+04$	$1.889e+03$	$1.188e+02$	$1.579e+01$	$1.498e+09$	$1.134e-09$
	std	$9.105e-06$	$2.661e-02$	$3.955e+04$	$7.757e+02$	$4.556e+01$	$2.384e+01$	$8.132e+08$	$6.632e-10$
$F4$	mean	$4.091e-06$	$4.507e-02$	$7.193e+01$	$1.927e+01$	$2.616e+00$	$2.059e+00$	$1.140e+07$	$1.610e-05$
	std	$6.020e-06$	$1.626e-02$	$8.482e+00$	$5.876e+00$	$5.954e-01$	$1.895e+00$	$3.619e+06$	$4.015e-06$
$F5$	mean	$1.286e+01$	$2.043e+02$	$9.224e+07$	$1.222e+04$	$7.403e+00$	$2.035e+02$	$1.815e+08$	$2.640e+00$
	std	$3.250e+01$	$4.888e+02$	$5.053e+07$	$1.113e+04$	$4.409e-01$	$3.207e+02$	$1.898e+08$	$6.989e+00$
$F6$	mean	$1.713e-22$	$3.999e-03$	$3.346e+04$	$2.033e+02$	$2.624e-06$	$3.349e-05$	$1.626e-01$	$5.630e-12$
	std	$4.879e-22$	$2.616e-03$	$9.148e+03$	$8.608e+01$	$3.265e-06$	$1.142e-04$	$1.514e-01$	$8.518e-13$
$F7$	mean	$3.553e-03$	$1.601e-03$	$7.584e+01$	$6.414e-02$	$9.502e-03$	$1.088e-02$	$1.686e+05$	$1.943e-03$
	std	$1.796e-03$	$9.372e-04$	$3.368e+01$	$2.566e-02$	$3.456e-03$	$8.743e-03$	$6.971e+04$	$1.480e-03$
$F8$	mean	$-3.815e+03$	$-3.062e+03$	$-5.863e+03$	$-3.902e+03$	$-1.834e+117$	$-2.664e+03$	$-7.372e+09$	$-3.030e+03$
	std	$1.868e+02$	$3.374e+02$	$3.184e+03$	$1.402e+02$	$5.933e+117$	$3.087e+02$	$8.628e+08$	$5.923e+02$
$F9$	mean	$3.551e+00$	$1.304e+01$	$3.616e+02$	$9.013e+00$	$2.688e+01$	$1.436e+01$	$5.433e+07$	$1.726e+01$
	std	$2.335e+00$	$7.248e+00$	$3.566e+01$	$2.943e+00$	$4.782e+00$	$8.774e+00$	$1.665e+07$	$1.527e+01$
$F10$	mean	$1.109e-12$	$9.126e-02$	$1.968e+01$	$5.904e+00$	$5.893e-03$	$2.161e+00$	$2.682e+06$	$7.305e+00$
	std	$1.082e-12$	$3.636e-01$	$7.209e-01$	$1.419e+00$	$3.875e-03$	$1.242e+00$	$7.733e+05$	$9.765e+00$
$F11$	mean	$7.983e-02$	$2.827e-01$	$3.478e+02$	$2.889e+00$	$3.571e-01$	$3.064e-01$	$2.039e+04$	$1.582e-01$
	std	$2.837e-02$	$1.064e-01$	$1.091e+02$	$1.098e+00$	$7.604e-02$	$2.043e-01$	$1.495e+04$	$9.304e-02$
$F12$	mean	$2.922e-24$	$2.096e-02$	$1.343e+08$	$9.438e+00$	$7.148e-06$	$2.379e+00$	$7.396e+06$	$2.972e-12$
	std	$1.104e-23$	$7.903e-02$	$8.176e+07$	$5.676e+00$	$6.177e-06$	$1.687e+00$	$4.419e+06$	$1.529e-12$
$F13$	mean	$3.767e-23$	$2.183e-03$	$3.591e+08$	$1.265e+03$	$1.063e-04$	$3.823e-01$	$4.546e+07$	$1.205e-11$
	std	$1.712e-22$	$4.056e-03$	$2.946e+08$	$3.400e+03$	$1.723e-04$	$7.603e-01$	$4.820e+07$	$6.464e-12$
Friedman value		23	47	92	73	46	62	89	36
Friedman rank		1	4	8	6	3	5	7	2

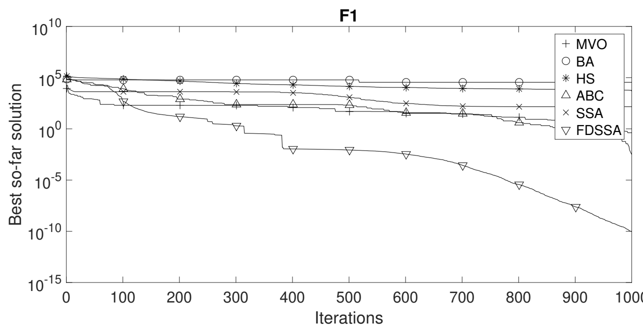

Fig. 1. Convergence curves on $F1$

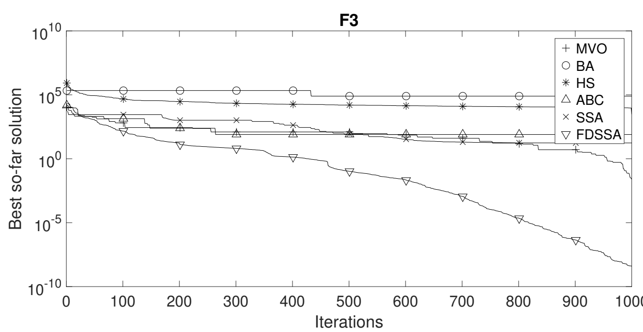

Fig. 2. Convergence curves on $F3$

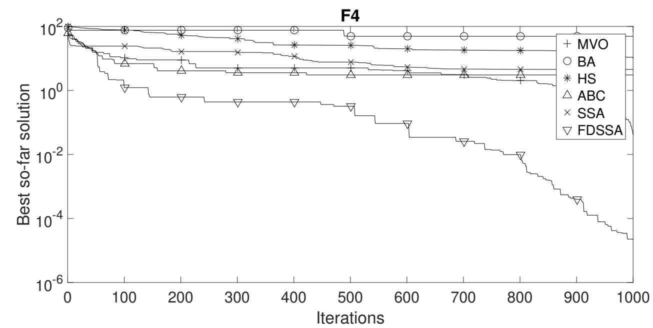

Fig. 3. Convergence curves on $F4$

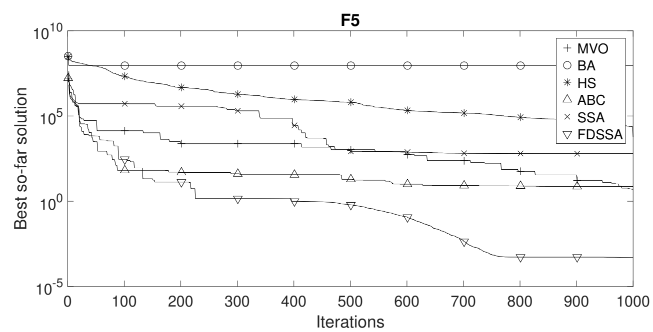

Fig. 4. Convergence curves on $F5$

[2] R. Eberhart , J. Kennedy, "A new optimizer using particle swarm theory.", In: MHS95. Proceedings of the 6th international symposium on micro machine and human science. IEEE, New York, pp 39-43, 1995

[3] X-S Yang , "A new metaheuristic bat-inspired algorithm". In: Pelta DA, Krasnogor N, Dumitrescu D, Chira C, Lung R (eds) Nature inspired cooperative strategies for optimization (NICSO 2010). Springer, Berlin, pp 6574, 2010

[4] D . Karaboga , "An idea based on honey bee swarm for numerical optimization." Technical report-tr06, Erciyes University, Engineering Faculty, Computer, 2005

[5] S. Mirjalili, "Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm". Knowl Based Syst 89:228249, 2015

[6] S. Mirjalili, SM Mirjalili, A Lewis, "Grey wolf optimizer." Adv Eng Softw 69:4661, 2014.

[7] S. Mirjalili, S. M. Mirjalili, A. Hatamlou, "Multi-Verse Optimizer: a nature-inspired algorithm for global optimization", Neural Computing and Applications, 27, 495-513, 2016.

[8] Z.W.Geem, J.H. Kim , G.V. Loganathan, "A new heuristic optimization algorithm: harmony search", Simulation 76 (2) 60-68, 2001.

[9] D. Pelusi, S. Elmougy, L.G. Tallini, B. Bose, "m-ary balanced codes with parallel decoding", IEEE Transactions on Information Theory, 61(6), 3251-3264, 2015.

[10] L.G. Tallini, D. Pelusi, R. Mascella, L. Pezza, S. Elmougy, B. Bose, "Efficient Non-Recursive Design of Second-Order Spectral-Null Codes",

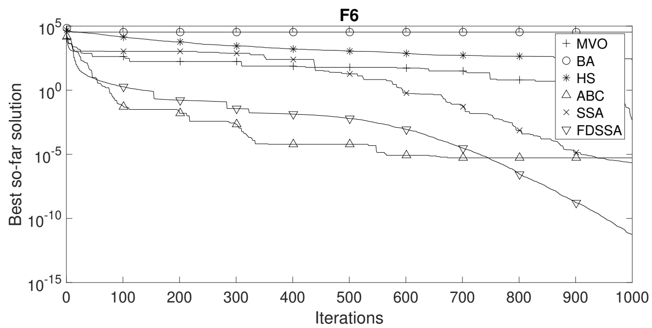

Fig. 5. Convergence curves on $F6$

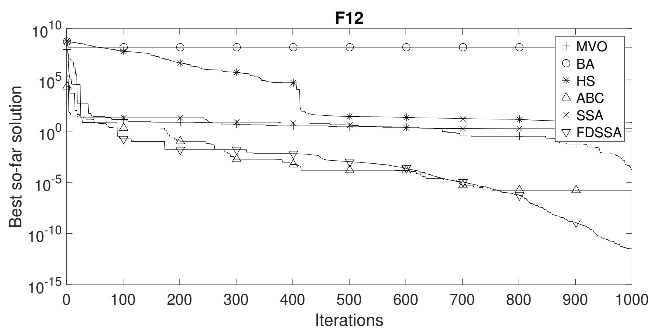

Fig. 6. Convergence curves on $F12$

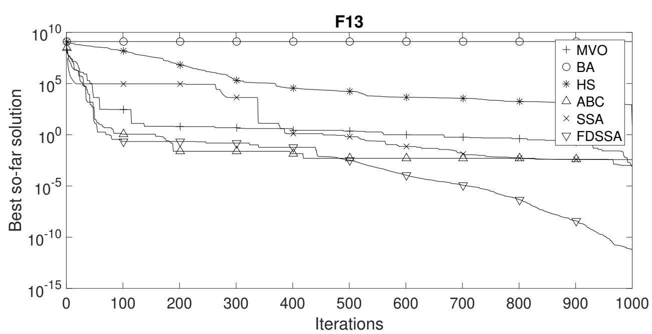

Fig. 7. Convergence curves on $F13$

IEEE Transactions on Information Theory, 62(6), 3084-3102, 2016.

[11] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, and S. M. Mirjalili, “Salp swarm algorithm: A bio-inspired optimizer for engineering design problems,” Adv. Eng. Softw., vol. 114, pp. 163-191, 2017.

[12] K. Gholami, M.H. Parvaneh, “A mtated salp swarm algorithm for optimum allocation of active and reactive power sources in radial distribution systems”, Applied Soft Computing Journal, 85, 105833, 2019.

[13] M. Tubishat, N. Idris, L. Shuib, M. A.M. Abushariah, S. Mirjalili, “Improved Salp Swarm Algorithm based on opposition based learning and novel local search algorithm for feature selection”, Expert Systems With Applications 145, 113122, 2020.

[14] M. H. Qais, H. M. Hasanien, S. Alghuwainem, “Enhanced salp swarm algorithm: Application to variable speed wind generators”, Engineering Applications of Artificial Intelligence 80, 82-96, 2019.

[15] H. Faris, M. M. Mafarja, A. A. Heidari, I. Aljarah, A. M. Al-Zoubi, S. Mirjalili, H. Fujita ,“ An efficient binary Salp Swarm Algorithm with crossover scheme for feature selection problems”, Knowledge-Based Systems 154, 43-67, 2018.

[16] A. A. Ateya, A. Muthanna, A. Vybornova, A. D. Algarni, A. Abuarqoub, Y. K., A. Koucheryavy, “Chaotic salp swarm algorithm for SDN multi-controller networks”, Engineering Science and Technology, an International Journal 22, 1001-1012, 2019.

[17] P. C. Sahu, S. Mishra, R. Chandra Prusty, S. Panda, “Improved-salp swarm optimized type-II fuzzy controller in load frequency control of multi area islanded AC microgrid”, Sustainable Energy, Grids and Networks 16, 380-392, 2018.

[18] R. Chen, C. Dong, Y. Ye, Z. Chen, Y Liu, “QSSA: Quantum Evolutionary Salp Swarm Algorithm for Mechanical Design”, IEEE Access, 7, 145582-145595, 2019.

[19] B. Ma, H. Ni, X. Zhu, R. Zhao, “A Comprehensive Improved Salp Swarm Algorithm on Redundant Container Deployment Problem”, IEEE Access, 7, 136452-136470, 2019.

[20] Z Xing, H. Jia , “Multilevel Color Image Segmentation Based on GLCM and Improved Salp Swarm Algorithm”, IEEE Access, 7, 37672-37690, 2019.

[21] B. Xiao, R. Wang, Y. Xu, J. Wang, W. Song and Y. Deng, ”Simplified Salp Swarm Algorithm,” 2019 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA), Dalian, China, 2019, pp. 226-230.

[22] RA Ibrahim, AA Ewees, D. Oliva, MA Elaziz, S. Lu, “Improved salp swarm algorithm based on particle swarm optimization for feature selection”. J Ambient Intell Humaniz Comput 10:1-15, (2018)

[23] A. E. Hegazy, M. Makhlouf, and G. S. El-Tawel, “Improved salp swarm algorithm for feature selection,” J. King Saud Univ.-Comput. Inf. Sci., doi: 10.1016/j.jksuci.2018.06.003, 2018

[24] G. I. Sayed, G. Khoriba, and M. H. Haggag , “A novel chaotic salp swarm algorithm for global optimization and feature selection,” Appl. Intell., vol. 48, no. 10, pp. 3462-3481, 2018.

[25] R. M. Rizk-Allah, A. E. Hassanien, M. Elhoseny, and M. Gunasekaran, “A new binary salp swarm algorithm: Development and application for optimization tasks,” Neural Comput. Appl., 2018. doi: 10.1007/s00521-018-3613-z.

[26] PA Anderson, Q. Bone, “Communication between individuals in salp chains II. physiology”. Proc R Soc Lond B, 210:559-74, 1980.

[27] I. Aljarah, M. Mafarja, A. A. Heidari, H. Faris, Y. Zhang, S. Mirjalili , “Asynchronous accelerating multi-leader salp chains for feature selection”, Applied Soft Computing 71, 964-979, 2018.

[28] G Zhang, H. Chen, A Asghar Heidari, X. Zhao, Y. Xu, P. Wang, Y. Li, C. Li, “Chaos-Induced and Mutation-Driven Schemes Boosting Salp Chains-Inspired Optimizers”, IEEE Access 7, 31243-31261, 2019.

[29] M. Varadarajan, K.S. Swarup, “Differential evolutionary algorithm for optimal reactive power dispatch”, Int. J. Electr. Power Energy Syst. 30 435-441, 2008.

[30] S. Das , P. Suganthan , Differential evolution: a survey of the state-of-the-art, IEEE Trans. Evolut. Comput. 15 (1) (2011) 4-31 .

[31] W. Wayne, “Friedman two-way analysis of variance by ranks”, Applied Non-parametric Statistics, Boston: PWS-Kent, 1990.