A Comparative Analysis of FSS with CMA-ES and S-PSO in Ill-Conditioned Problems

Anthony J. da C.C. Lins[1], Fernando B. Lima-Neto[1], François Fages[2], and Carmelo J.A. Bastos-Filho[1]

[1] Polytechnic School of Engineering of University of Pernambuco
{ajccl,fbln,carmelofilho}@ecomp.poli.br
[2] Contraintes, INRIA Rocquencourt
francois.fages@inria.fr

Abstract. This paper presents a comparative analyzes between three search algorithms, named Fish School Search, Particle Swarm Optimization and Covariance Matrix Adaptation Evolution Strategy applied to ill-conditioned problems. We aim to demonstrate the effectiveness of the Fish School Search in the optimization processes when the objective function has ill-conditioned properties. We achieved good results for the Fish School Search and in some cases we obtained superior results when compared to the other algorithms.

Keywords: Fish School Search, Covariance matrix adaptation, Particle Swarm Optimization, Ill-conditioned problems, Invariance, Non-separable problems.

1 Introduction

Several real world problems are modeled through ill-conditioned functions due to the complexity of the relation between their components. An ill-conditioned function is a type of function that has variables which can generate great impact on the final results upon minimal adjustments. Hansen *et al.* [1] presented a comprehensive study on the performance of different search algorithms under different conditions of optimization when tackling problems with ill-conditioned characteristics. Hansen *et al.* investigated the performance of two stochastic search methods in ill-conditioned functions and non-separable problems for a set of benchmark functions with a different number of conditions. They assessed a particular version of the PSO (particle swarm optimization) [2][3] and also an implementation of the CMA-ES (covariance matrix adaptation evolution strategy) algorithm [4] [5] [6] .

Recently, Bastos-Filho and Lima-Neto [7][8] proposed a swarm intelligence technique for searching and optimization, called Fish School Search (FSS). FSS was inspired by the gregarious behavior of fish schools and it can tackle multimodal problems in high dimensional search spaces. In this paper, we assess the performance of the FSS algorithm when applied to problems with specific properties, such as ill-conditioning and non-separability. We also compare the performance of the FSS to the performance of two other approaches, S-PSO and CMA-ES.

The remainder of the paper is organized as follows: in section 2 we present an overview of the FSS algorithm. The concepts about ill-conditioned functions are described in section 3. The simulation setup, the benchmark functions and some other aspects are addressed in section 4. In section 4, we also present an analysis regarding a great number of parameter combinations for the simulated functions. In section V we give our conclusions.

2 FSS - Fish School Search

Fish School Search (FSS) is a bio-inspired algorithm for searching in high-dimensional and multimodal search spaces. The FSS functionality is based on a population of limited-memory individuals, called fish. FSS has two classes of operators [9], the feeding operator and the swimming operators. The fish school aims to achieve a collective goal that is to find food. Food is a metaphor for the fitness function, which measures the quality of the current solution. The pseudocode of the FSS algorithm can also be found in [9].

2.1 FSS Operators

Feeding Operator: The feeding operation is executed after the individual movement of the fish. Depending on the individual movement, the weight of the fish may increase or decrease, which indicates if this movement was successful or not. Bastos-Filho *et al.* [9] proposed to use an initial weight equal to 1, and it can vary up to a maximum scale value (W_{scale}). The weight is updated based on the current value of the weight summed to the normalized difference between the current fitness and the fitness at the new position considering the whole school. The weights of the fish are updated once in every FSS cycle by the feeding operator, according to equation (1):

$$W_i(t+1) = W_i(t) + \frac{\Delta f_i}{max(\Delta f)}, \tag{1}$$

where $W_i(t)$ is the weight of the fish i, Δf_i is the difference between the fitness value at the new position and the fitness value at the current position of each fish. $max(\Delta f)$ is the maximum value of Δf_i in the iteration considering the whole school.

Swimming Operator. In nature, animals react instinctively to environmental stimuli. For the fish, swimming is directly related to all significant individual and collective behaviors of the school, such as feeding, reproduction, escape from predators, move to safer positions of the aquarium or just stay grouped. In FSS, swimming is a vectorial composition of three movements: individual, collective-instinctive and collective-volitive. These movements are detailed below.

i. **Individual movement:** The individual movement operator is executed for all fish at each iteration of the algorithm. Each fish calculates randomly a new position within the search space in its neighborhood and evaluates the new position using the fitness function. If the fitness of the neighbor position is better, then the fish moves to the new position. Otherwise, the fish remains in the current position.

   The neighbor position is calculated by adding to each component of the current position a random value generated by an uniform distribution in the interval [-1,1] multiplied by a step as shown in equation (2).

$$\boldsymbol{n}_i(t) = \boldsymbol{x}_i(t) + \boldsymbol{rand}(-1,1)\, step_{ind}, \tag{2}$$

   in which $\boldsymbol{x}_i(t)$ is the current position of the i^{th} fish, $\boldsymbol{n}_i(t)$ is the candidate position of the fish and $\boldsymbol{rand}()$ is a vector of random number generated in each iteration. $step_{ind}$ is given as a percentage of the search space and decays linearly with the iterations. The individual step size of the fish is updated according to the equation (3) in order to allow more exploration in the beginning and more exploitation in the end of the search process.

$$step_{ind}(t+1) = step_{ind}(t) - \frac{(step_{ind\,initial} - step_{ind\,final})}{iterations}, \tag{3}$$

   in which *iterations* is the total number of iterations, $step_{ind\,initial}$ and $step_{ind\,final}$ are the initial and final steps, respectively.

ii. **Collective-Instinctive movement:** After the individual movement of all fish, the algorithm calculates the weighted movement of the whole school for fish that performed the individual movement. The resultant direction ($\boldsymbol{I}(t)$) is more influenced by the fish that have highest weights. $\boldsymbol{I}(t)$ is calculated according to the equation (4). Then, all the fish update their positions by using equation (5).

$$\boldsymbol{I}(t) = \frac{\sum_{i=1}^{N} \Delta\boldsymbol{x}_i \Delta f_i}{\sum_{i=1}^{N} \Delta f_i}, \tag{4}$$

$$\boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + \boldsymbol{I}(t). \tag{5}$$

iii. **Collective-Volitive movement:** After the two former movements, the algorithm calculates if the weight of the whole school had increased. If it is the case, we have an indication that the search process had success in the iteration and the radius of the school should contract to allow more exploitation. Otherwise, the radius should increase in order to allow the school to get out of local minima and find better regions of the search space. This operator balances the exploration/exploitation trade-off.

   The expansion or contraction of the school is applied as a small adjustment for each position of the fish with respect to the barycenter of the school. The calculation of the school barycenter is obtained by a weighted average position of all fish weighted by its respective weight, as shown in equation (6).

$$\boldsymbol{B}(t) = \frac{\sum_{i=1}^{N} \boldsymbol{x}_i W_i(t)}{\sum_{i=1}^{N} W_i(t)}. \tag{6}$$

The expansion or contraction of the radius of the school is calculated by comparing the weight of the school in the previous iteration and the current iteration. If the weight of the school had increased, the fish must update their positions according to (7). Otherwise, all fish must update their positions using equation (8).

$$\boldsymbol{x}(t+1) = \boldsymbol{x}(t) - step_{vol}\, rand(0,1)\, \frac{(\boldsymbol{x}(t) - \boldsymbol{B}(t))}{distance(\boldsymbol{x}(t), \boldsymbol{B}(t))}, \tag{7}$$

$$\boldsymbol{x}(t+1) = \boldsymbol{x}(t) + step_{vol}\, rand(0,1)\, \frac{(\boldsymbol{x}(t) - \boldsymbol{B}(t))}{distance(\boldsymbol{x}(t), \boldsymbol{B}(t))}, \tag{8}$$

in which $distance()$ is a function that calculates the Euclidean distance between the barycenter and the current position of the fish. $step_{vol}$ is the step size used to control the movement of fish to or from the barycenter. We used $step_{vol} = 2step_{ind}$.

3 Ill-Conditioned Functions

In [1], Hansen et al. investigate the behavior of PSO (particle swarm optimization) [2][3] and CMA-ES (covariance matrix adaptation evolution strategy) [4] [5] [6] on ill-conditioned and non-separable objective functions.

An objective function is separable with respect to coordinate i if the optimal value for the ith coordinate does not depend on the values for the other coordinates. A separable optimization problem in dimension n can be solved by solving n optimization problems in one dimension. Separable problems are thus easy to solve with a linear complexity in the dimension of the problem, while non-separable problems are hard to solve and may involve an exponential complexity in the number of dimensions.

Furthermore, ill-conditioned functions correspond to situations where small differences in variables, *i.e.* in different directions in the search space, can generate very different results for the evaluation function, by several orders of magnitude. Ill-conditoned objective functions can hardly guide the search of optimal solutions and thus provide challenges for any optimization methods.

4 Comparing FSS to CMA-ES and S-PSO

4.1 Simulation Setup

In [1], a benchmark of hard non-separable ill-conditioned functions was tried with two well-established optimization methods, each of them with a specific bio-inspired stochastic search: swarm intelligence (PSO) and evolutionary strategy (CMA-ES). In this section, we present the results obtained with FSS on this benchmark.

The objective functions are given in Table 1. The search space domain is $[-20, 80]^n$ in both cases. According to the original paper, in the Rosenbrock function the parameter α tunes the width of the bent ridge that guides to the global optimum. In the classical Rosenbrock function α equals 100. For smaller α the ridge becomes wider and the function becomes less difficult to solve. The α value will be vary between one and 10^8. Rastringin function was implemented and tested without any kind of modifications.

Table 1. Test functions rosenbrock and rastrigin and target function values

Function	f_{target}
$F_{Rosenbrock}(\boldsymbol{x}) = \sum_{i=1}^{n}[\alpha(x_{i+1} - x_i^2)^2 + (1 - x_i)^2]$	10^{-9}
$F_{Rastrigin}(\boldsymbol{x}) = 10n + \sum_{i=1}^{n}[x_i^2 - 10cos(2\pi x_i)]$	10^{-9}

For the configuration setup and tests, we used all parameters according to the S-PSO presented in the original paper. We used 10 dimensions for both benchmark functions. For the Rosenbrock function, we used 16 fish. For the Rastrigin function, we used the following swarm sizes 10, 16, 30, 100, 300 and1000. The simulations were performed using 21 runs of the algorithm, for each configuration in both functions. In each of the executions of the algorithm, the evaluation function call was 10^7 times. The stopping criterion for each run is when the maximum number of function evaluations was reached. The FSS implementation was based on the *FSS-Vanilla* Java$^{\text{TM}}$version[1].

4.2 Simulation Results

Table 2 is a summary of the results obtained from the simulations using FSS for both benchmark functions. The table depicts the average and the (standard deviation) values for the fitness and the execution time for all simulations. We just achieved the target value for the optimization process in the Rastrigin function using 16 and 30 fish. One can observe that the target value (10^{-9}) was not reached for the Rastrigin function for the other configuration settings. We believe that 10 fish are not enough to provide the proper exploration of the entire search space, whereas too many fish can mitigate the exploitation capacity. We did not achieved the target value for the Rosenbrock function in any case.

[1] Published: `http://www.fbln.pro.br/fss/versions.htm`

Table 2. FSS simulations summary, considering fitness value (average and std deviation) and time, for each configuration

FSS				Fitness		Time(ms)	
Function	Dimensions	Size	α	Average	Std Dev	Average	Std Dev
Rosenbrock	10	16	1	2,76E-03	4,76E-04	3,11E+05	7,04E+02
			10	2,44E-02	4,88E-03	3,21E+05	4,64E+03
			100	2,67E-04	3,74765E-05	2,67E-04	3,74765E-05
			300	5,54E+00	3,94E-01	3,23E+05	1,21E+03
			1000	2,28E-04	3,51995E-05	2,28E-04	3,51995E-05
Rastrigin	10	10	-	3,05E-06	6,65E-07	2,35E+05	4,93E+02
		16	-	1,75514E-10	2,49508E-11	1,75514E-10	2,49508E-11
		30	-	8,38232E-11	1,80445E-11	8,38232E-11	1,80445E-11
		100	-	1,82707E-07	3,64708E-08	2,34E+06	2,06E+04
		300	-	4,56812E-08	7,84523E-09	7,09E+06	2,67E+05
		1000	-	9,56496E-09	1,37591E-09	2,38E+07	8,98E+04

Table 3 shows the results for the analysis of the effectiveness and (number of times the algorithm obtained the fitness target - maximum is 21 trials) considering the FSS, S-PSO and CMA-ES. Although the standard version of the FSS is not able to solve configurations tests, one can observe that the FSS can solve the problem with low number of fish, while the S-PSO and the CMA-ES need 300 particles and 1000 individuals, respectively, to solve the same problem.

Table 3. Analysis of the effectiveness (number of times the algorithm obtained the fitness target) considering the FSS, S-PSO and CMA-ES

Entities	10	16	30	100	300	1000
S-PSO	-	-	5%(1)	71%(15)	100%(21)	100%(21)
CMA-ES	-	-	-	24%(5)	76%(16)	100%(21)
FSS	-	100%(21)	100%(21)	-	-	-

5 Conclusions

In this paper we assessed the performance of the FSS for two well known ill-conditioned functions, Rastrigin and Rosenbrock. We compared the results with the results obtained by CMA-ES and S-PSO. The overall results were somehow counterintuitive, since FSS excelled for the apparently more ill-conditioned function (Rastrigin), producing excellent results for situations in which the two competing techniques not even could produce anything. And this could be due to the built-in set of mechanisms that endow FSS to perceive 'ill-conditions' of problems.

On the other hand, the topology of Rosenbrock function, with the global minimum inside a long parabolic shaped flat valley, makes it hard to spot the global minimum during the optimization process. The poor results obtained with FSS (for that particular function) are likely due to a premature convergence of the search process convergence in local minima. Though FSS has a mechanism to self-adjust exploration and exploitation modes, the topology of this apparently simple function does not allow FSS to escape from local minima. We hypothesize that the premature convergence of the FSS can be related to the linear decreasing of search steps for Individual and Volitive movement operators. In a future work, this kind of problem will be addressed.

Another point quite interesting is that FSS method is rather easy to adjust, as many parameters can be left without change. One reason for example is due to its self-control of exploration and exploitation modes.

Although counterintuitive, the increase of the number of fish may compromise the performance of the FSS method because not-directly it adds on the granularity of the search. This eventual artifact of acceleration of convergence is subject to change in the next release of FSS.

References

1. Hansen, N., Ros, R., Mauny, N., Schoenauer, M., Auger, A.: Impacts of invariance in search: When cma-es and pso face ill-conditioned and non-separable problems. Applied Soft Computing 11(8), 5755–5769 (2011)
2. Kennedy, J., Eberhart, R.: Particle swarm optimization, vol. 4, pp. 1942–1948 (1995)
3. Eberhart, R., Shi, Y.: Comparing inertia weights and constriction factors in particle swarm optimization. In: Proceedings of the 2000 Congress on Evolutionary Computation, vol. 1, pp. 84–88 (2000)
4. Hansen, N., Ostermeier, A.: Completely derandomized self-adaptation in evolution strategies. Evolutionary Computation 9, 159–195 (2001)
5. Hansen, N.: The CMA evolution strategy: a comparing review. In: Lozano, J., Larranaga, P., Inza, I., Bengoetxea, E. (eds.) Towards a New Evolutionary Computation. Advances on Estimation of Distribution Algorithms, pp. 75–102. Springer (2006)
6. Hansen, N., Ostermeier, A.: Adapting arbitrary normal mutation distributions in evolution strategies: the covariance matrix adaptation. In: Proceedings of IEEE International Conference on Evolutionary Computation, pp. 312–317 (May 1996)
7. Bastos-Filho, C.J.A., de Lima Neto, F.B., Lins, A.J.C.C., Nascimento, A.I.S., Lima, M.P.: A novel search algorithm based on fish school behavior. In: IEEE International Conference on Systems, Man and Cybernetics, SMC 2008, pp. 2646–2651 (2008)
8. Bastos Filho, C.J.A., de Lima Neto, F.B., Lins, A.J.C.C., Nascimento, A.I.S., Lima, M.P.: Fish School Search. In: Chiong, R. (ed.) Nature-Inspired Algorithms for Optimisation. SCI, vol. 193, pp. 261–277. Springer, Heidelberg (2009)
9. Bastos-Filho, C.J.A., de Lima-Neto, F.B., Sousa, M.F.C., Pontes, M.R., Madeiro, S.S.: On the influence of the swimming operators in the fish school search algorithm. In: IEEE International Conference on Systems, Man and Cybernetics, SMC 2009, pp. 5012–5017 (2009)