Study of Flower Pollination Algorithm for Continuous Optimization

Szymon Łukasik[1,2] and Piotr A. Kowalski[1,2]

[1] Systems Research Institute, Polish Academy of Sciences,
ul. Newelska 6, 01-447 Warsaw, Poland
[2] Department of Automatic Control and Information Technology, Cracow University of Technology
ul. Warszawska 24, 31-155 Cracow, Poland
`slukasik@ibspan.waw.pl`
`pakowal@ibspan.waw.pl`

Abstract. Modern optimization has in its disposal an immense variety of heuristic algorithms which can effectively deal with both continuous and combinatorial optimization problems. Recent years brought in this area fast development of unconventional methods inspired by phenomena found in nature. Flower Pollination Algorithm based on pollination mechanisms of flowering plants constitutes an example of such technique. The paper presents first a detailed description of this algorithm. Then results of experimental study of its properties for selected benchmark continuous optimization problems are given. Finally, the performance the algorithm is discussed, predominantly in comparison with the well-known Particle Swarm Optimization Algorithm.

Key words: flower pollination algorithm, metaheuristics, optimization, nature-inspired algorithms

1 Introduction

Inspiration from nature is a feature found in many traditional [1, 2] or newly developed metaheuristics. Among the examples of mechanisms mimicked by recent unconventional heuristic algorithms one can name, e.g. bioluminescent communication of fireflies [3], social spider's cooperative schemes [4] or swarm behavior of krill herds [5]. This paper studies an approach based on flower pollination, namely Flower Polination Algorithm (FPA) proposed in 2012 by Xin-She Yang [6].

The goal of the paper is to evaluate FPA performance and study its properties in continuous optimization. Up to date the only contributions in this area have been made by creator of the algorithm. They establish the basic form of FPA [6] and describe its multi-objective variant [11]. Here we try to supplement those studies, e.g. by more detailed analysis of parameters and evaluation of algorithm's performance with regards to standard variant of Particle Swarm Optimization (PSO) [12].

The paper is organized as follows. First, it provides a foundation upon the algorithm was constructed along with its formal description. Then, it describes the results of numerical experiments, devoted to exploring possible parameter values. Algorithm's performance in relation to PSO method is also studied therein. Finally, the last part of the contribution contains some concluding remarks on FPA and proposals for its further development.

2 FPA Formulation

Pollination itself as a natural phenomenon constitutes a process of transferring pollen grains from the stamens, the flower parts that produce them, to the ovule-bearing organs or to the ovules (seed precursors) themselves [7]. It is crucial for fertilization and reproduction of flowering plants also known as the angiosperms – the largest and most conspicuous group of modern plants [8]. Two types of pollination can be distinguished with regards to the methods of pollen transfer. First, abiotic pollination that does not involve using other organisms and employs wind, water or gravity as pollination mediators. Second, biotic pollination, which is the dominant one, requires pollinators, i.e. organisms, predominantly insects, that carry or move the pollen grains. Evolutionary, pollination by insects probably occurred in primitive seed plants, with reliance on other means being a relatively recent development [7]. Pollination can be accomplished by cross-pollination or by self-pollination. Self-pollination occurs when pollen from one flower pollinates the same flower or other flowers of the same individual [9]. In contrast to that, cross-pollination happens when pollen is delivered to a flower from a different plant. About 70% of the present day angiosperms are cross-pollinated while the rest are self-pollinated [10]. To sum up, it is worth to mention that nowadays in nature all kind of pollination modes can be found, with some plants being able to effectively employ each and every one of them. Consequently both biotic and cross-pollination as well as abiotic and self-polination strategies were translated into the optimization domain and included in novel unconventional metaheuristic of FPA.

Pollination process constitutes a set of complex mechanisms crucial to the success of plants reproductive strategies. To build an effective algorithm mimicking those mechanisms several simplifications in understanding their fundamental aspects have to be made. Firstly, global, i.e. long-distance pollination should occur for biotic and cross-pollination pollen transfers. At the same time abiotic and self-pollination are to be associated with local reproductive strategies. Plants can employ both pollination schemes, however attractiveness of selected plant should influence the tendency of individual pollinators to exclusively visit it. The strength of this attraction is referred to as flower constancy.

When moving to the optimization domain pollination concept can be understood as follows. A single flower or pollen gamete will constitute a solution of the optimization problem, with the whole flower population being actually used. Their constancy will be understood as solution fitness. Pollen will be transferred in the course of two operations used interchangeably, that is: global and local pollination. The first one employs pollinators to carry pollen to long distances towards individual characterized by higher fitness. Local pollination on the other hand occurs within limited range of individual flower thanks to pollination mediators like wind or water. FPA for continuous optimization built on assumptions presented above will be more strictly formulated in the subsequent part of this Section.

The general goal of continuous optimization is to find x^* which satisfies:

$$f(x^*) = \min_{x \in S} f(x), \tag{1}$$

where $S \subset R^N$, and $f(x)$ constitutes solution's x cost function value. Therefore actual task of the optimizer is to find argument minimizing f.

To solve the optimization problem (1) the population of M individuals will be used. It is represented by a set of N-dimensional vectors – equivalent to individuals' positions – within the iteration k denoted by:

$$x_1(k), x_2(k), ..., x_M(k). \tag{2}$$

The best position found by given individual m prior to iteration k is given by $x_m^*(k)$ with cost/fitness function value $f(x_m^*(k))$. At the same time:

$$x^*(k) = \arg \min_{m=1,...,M} f(x_m(k)), \tag{3}$$

or

$$x^*(k) = \arg \max_{m=1,...,M} f(x_m(k)), \tag{4}$$

corresponds to the best solution found by the algorithm in its k iterations, with $f(x^*(k))$ representing its related cost (3) or fitness (4) function value. The formula used depends on type of optimization task (minimization or maximization of function f).

Flower Pollination Algorithm's formal description using aforementioned notation will be given below (as Algorithm 1). It can be seen that global pollination occurs with probability p defined by so called switch probability. If this phase is omitted local pollination takes place instead. The first one constitutes of pollinator's movement towards best solution $x^*(k)$ found by the algorithm, with s representing N-dimensional step vector following a Lévy distribution :

$$L(s) \sim \frac{\lambda \Gamma(\lambda) \sin(\pi\lambda/2)}{\pi s^{1+\lambda}}, \; (s \gg s_0 > 0), \tag{5}$$

with Γ being the standard gamma function and parameters $\lambda = 1.5, s_0 = 0.1$ as suggested by Yang [11]. Practical method for drawing step sizes s following this distribution by means of Mantegna algorithm are given in [13]. Local pollination includes two randomly selected members of the population and is performed via movement towards them, with randomly selected step size ϵ. Finally, the algorithm is terminated when number of iteration k reaches predetermined limit defined by K.

Algorithm 1 Flower Pollination Algorithm

```
1: $k \leftarrow 1$ {initialization}
2: $f(x^*(0)) \leftarrow \infty$
3: **for** $m = 1$ to $M$ **do**
4:    Generate_Solution($x_m(k)$)
5: **end for**
6: {find best}
7: **for** $m = 1$ to $M$ **do**
8:    $f(x_m(k)) \leftarrow$ Evaluate_quality($x_m(k)$)
9:    **if** $f(x_m(k)) < f(x^*(k-1))$ **then**
10:      $x^*(k) \leftarrow x_m(k)$
11:   **else**
12:      $x^*(k) \leftarrow x^*(k-1)$
13:   **end if**
14: **end for**
15: {main loop}
16: **repeat**
17:   **for** $m = 1$ to $M$ **do**
18:      **if** $Real\_Rand\_in\_(0,1) < p$ **then**
19:         {Global pollination}
20:         $s \leftarrow Levy(s_0, \gamma)$
21:         $x_{trial} \leftarrow x_m(k) + s(x^*(k) - x_m(k))$
22:      **else**
23:         {Local pollination}
24:         $\epsilon \leftarrow Real\_Rand\_in\_(0,1)$
25:         $r, q \leftarrow Integer\_Rand\_in(1, M)$
26:         $x_{trial} \leftarrow x_m(k) + \epsilon(x_q(k) - x_r(k))$
27:      **end if**
28:      {Check if new solution better}
29:      $f(x_{trial}) \leftarrow$ Evaluate_quality($x_{trial}$)
30:      **if** $f(x_{trial}) < f(x_m(k))$ **then**
31:         $x_m(k) \leftarrow x_{trial}$
32:         $f(x_m(k)) \leftarrow f(x_{trial})$
33:      **end if**
34:   **end for**
35:   {find best and copy population}
36:   **for** $m = 1$ to $M$ **do**
37:      **if** $f(x_m(k)) < f(x^*(k-1))$ **then**
38:         $x^*(k) \leftarrow x_m(k)$
39:      **else**
40:         $x^*(k) \leftarrow x^*(k-1)$
41:      **end if**
42:      $f(x(k+1)) \leftarrow f(x_k)$
43:      $x(k+1) \leftarrow x(k)$
44:   **end for**
45:   $f(x^*(k+1)) \leftarrow f(x^*k)$
46:   $x^*(k+1) \leftarrow x^*(k)$
47:   $stop\_condition \leftarrow Check\_stop\_condition()$
48:   $k \leftarrow k+1$
49: **until** $stop\_condition$ = **false**
50: **return** $f(x^*(k))$, $x^*(k)$, $k$
```

Next Section of the paper will present results of conducted tests which were aimed at exploring the performance of the Flower Pollination Algorithm in the form being presented here extensively.

3 Experimental results

For computational experiments examining properties of the FPA set of benchmark problems already considered in CEC'13 competition was used [14]. Table 1 presents those functions along with their mathematical expressions, dimensionality and optimum values f^*.

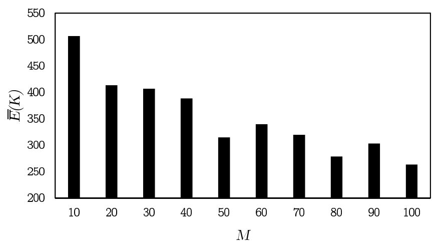
Table 1: Benchmark functions used for experimental studies

f	Name	Expression	Feasible bounds	N	f^*
f_1	Sphere	$f_1(x) = \sum_{i=1}^{N} z_i^2 + f_1^*$ $z = x - o$	$[-100, 100]^N$	10	-1400
f_2	Rotated Bent Cigar	$f_2(x) = z_1^2 + 10^6 \sum_{i=2}^{N} z_i^2 + f_2^*$ $z = M_2 T_{asy}^{0.5}(M_1(x - o))$	$[-100, 100]^N$	10	-1000
f_3	Different Powers	$f_3(x) = \sqrt{\sum_{i=1}^{N} \lvert z_i\rvert^{2+4\frac{i-1}{N-1}}} + f_3^*$ $z = x - o$	$[-100, 100]^N$	10	-1000
f_4	Rotated Rastrigin	$f_4(x) = \sum_{i=1}^{N} (z_i^2 - 10\cos(2\pi z_i) + 10) + f_4^*$ $z = M_1 \Lambda^{10} M_2 T_{asy}^{0.2}(T_{osz}(M_1 \frac{5.12(x-o)}{100}))$	$[-100, 100]^N$	10	-300
f_5	Schwefel	$f_5(x) = 418.9829N \sum_{i=1}^{N} g(z_i) + f_5^*$ $z = \Lambda^{10}(\frac{1000(x-o)}{100}) + 4.209687462275036 * 10^2$ $g(z_i) = z_i \sin(\lvert z_i\rvert^{1/2})$	$[-100, 100]^N$	10	-100
f_6	Rotated Katsuura	$f_6(x) = \frac{10}{N^2} \prod_{i=1}^{N}(1 + i \sum_{j=1}^{32} \frac{\lvert 2^j z_i - round(2^j z_i)\rvert}{2^j})^{\frac{10}{N^{1.2}}} - \frac{10}{N^2} + f_6^*$ $z = M_2 \Lambda^{100}(M_1 \frac{5(x-o)}{100})$	$[-100, 100]^N$	10	200

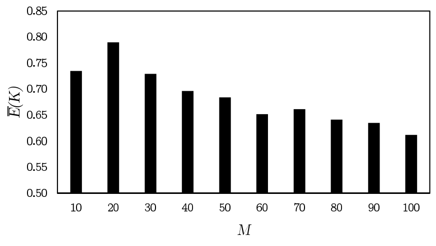
Symbols:
$o = x^*$ the shifted global optimum , which is randomly distributed in $[-80, 80]^N$,
M_1, M_2 - orthogonal (rotation) matrix generated from standard normally distributed entries by Gram-Schmidt orthonormalization.
Λ^α - a diagonal matrix in N dimensions with the i^{th} diagonal element as $\lambda_{ii} = \alpha^{\frac{i-1}{2(N-1)}}$ for $i = 1, 2, ..., N$.
T_{asy}^{β} - if $x_i > 0, x_i = x_i^{1+\beta\frac{i-1}{N-1}\sqrt{x_i}}$ for $i = 1, 2, ..., N$.
T_{osz} - for $x_i = sign(x_i)\exp(\hat{x}_i + 0.049(sin(c_1\hat{x}_i) + sin(c_2\hat{x}_i)))$ for $i = 1, 2, ..., N$.
where:
$\hat{x}_i = log(\lvert x_i\rvert)$ for $x_i \neq 0$, otherwise $\hat{x}_i = 0$,
$c_1 = 10$ if $x_i > 0$, otherwise $c_1 = 5.5$,
$c_2 = 7.9$ if $x_i > 0$, otherwise $c_2 = 3.1$.

The experiments were conducted for fixed number of iterations $K = 10000$ $(1000 * N)$, and 30 trials for each function. As a performance measure mean optimization error $\overline{E}(K)$ was used (with $E(K) = |f(x^*(K)) - f^*|$) along with its standard deviation $\sigma_{E(K)}$. We have also studied mean execution time $\overline{t}$ in seconds needed for one algorithm's run.

First the influence of population size M was under investigation. We tested FPA with $M = \{10, 20, 30, ..., 100\}$ and $p = 0.8$ as suggested in [6]. Obviously, as the algorithm is characterized by $O(M)$ complexity, it is recommended to limit population size if satisfactory outcomes are achieved for smaller M. It is the case for unimodal functions f_1, f_2 and f_3 where acceptable results were obtained for 20 individuals. For multimodal functions it is recommended to further increase population size, e.g. to 80 as significant improvement in optimization performance is observed. It is illustrated by mean optimization error shown for f_5 and f_6 on Fig. 1.



(a) Function f_5



(b) Function f_6

Fig. 1: Mean error values for varying population size

In the subsequent phase of numerical experiments we used $M = 20$ for f_1, f_2, f_3 and f_4 and $M = 80$ for f_5 and f_6. During these tests suggested value for switch probability p were under investigation. Most significant findings were summarized in Table 2. It presents values of p which ensure minimum error value in single run. It was established that for unimodal functions FPA provided optimum values regardless of p being used. It was noted however that higher value of switch probability is recommended, as computational cost of the algorithm is lower in this case. For multimodal problems best results can be achieved for $p \in [0.5, 0.8]$.

Table 2: Switch probability values ensuring $min(E(K))$

Function	Minimum error for
f_1	p={0.1,...,0.9}
f_2	p={0.3,...,0.8}
f_3	p={0.1,...,0.9}
f_4	p=0.8
f_5	p=0.5
f_6	p=0.7

Finally, experimental evaluation of algorithm's performance in reference to the standard Particle Swarm Optimization was performed. PSO variant with constriction factor χ was used [2]. For parameter values $\chi = 0.72984$ and $c_1 = c_2 = 2.05$ were chosen [15]. Experiments were executed for the same population sizes and switch probabilities established in previous tests. Mean error, its deviation and minimum value along with execution time were under investigation. Summary of obtained results is provided in Table 3.

It can be seen that FPA and PSO exhibit similar performance for selected benchmark functions. The algorithm studied in this paper performs better for f_2 and f_4. Choice between PSO and FPA is not so obvious for f_1, f_3 and f_6 where indistinguishable performance differences were observed. PSO in turn would be an apparent choice in case of f_5 function.

Table 3: Performance comparison of Flower Pollination Algorithm and Particle Swarm Optimization for selected benchmark functions

Function	Algorithm	$\overline{E}(K)$	$\sigma_{E(K)}$	$\min E(K)$	$t[s]$
f_1	FPA	0.00	0.00	0.00	3.69
	PSO	0.00	0.00	0.00	2.88
f_2	FPA	1.51	2.36	0.00	3.78
	PSO	3288692	7359309	0.16	3.03
f_3	FPA	0.00	0.00	0.00	3.72
	PSO	0.00	0.00	0.00	2.91
f_4	FPA	12.55	6.58	2.98	3.84
	PSO	13.69	5.27	3.98	3.19
f_5	FPA	227.78	70.78	39.02	18.78
	PSO	130.30	80.71	0.31	7.96
f_6	FPA	0.59	0.16	0.16	18.02
	PSO	0.42	0.13	0.19	11.02

To illustrate the dynamics of both algorithms average error values obtained for f_4 function during 10000 iterations were enclosed on Figure 2. It can be seen that in this case PSO converges faster in the first phase of the optimization process. It stucks in the local minimum however and is outperformed by FPA in the latter part of the process.

4 Conclusion

The paper examined novel nature-inspired metaheuristics of Flower Pollination Algorithm. Besides its detailed description, experimental evaluation of its sensitivity to parameter values and performance with regards to the Particle Swarm Optimization were under consideration.

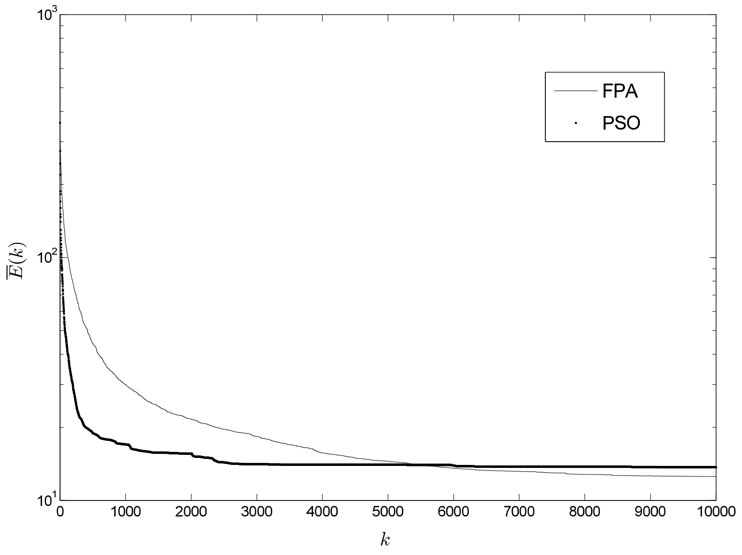


Fig. 2: Average error values during 10000 iterations for PSO and FPA (function f_4, logarithmic scale)

We found that the initial variant of the algorithm – as introduced by Yang – proved to be quite competitive even in comparison to the well-studied PSO. Mechanisms of local and global exploration embedded within the algorithm are complementary and allow the algorithm to effectively search the domain of optimization problem at hand. What is more FPA is characterized by small number of parameters which make it an attractive tool for real-life optimization problems, also multiobjective ones [11].

Further studies in the area of this contribution will concern, both studying algorithm's properties for more diverse set of benchmark instances as well as assessing more sophisticated variant of FPA which will employ variable switch probability. Those aspects are to be covered by the forthcoming follow-up paper [16].

References

1. Simon, D.: Evolutionary Optimization Algorithms. Wiley, New York (2013)
2. Clerc, M.: Particle Swarm Optimization. Wiley, New York (2006)
3. Łukasik, S., Żak, S.: Firefly algorithm for continuous constrained optimization tasks. Lecture Notes in Computer Science **5796** (2009) 97–106
4. Cuevas, E., Cienfuegos, M.: A new algorithm inspired in the behavior of the social-spider for constrained optimization. Expert Systems with Applications **41**(2) (2014) 412 – 425

5. Gandomi, A., Alavi, A.: Krill herd: A new bio-inspired optimization algorithm. Communications in Nonlinear Science and Numerical Simulation **17**(12) (2012) 4831 – 4845
6. Yang, X.S.: Flower pollination algorithm for global optimization. Lecture Notes in Computer Science **7445** (2012) 240–249
7. Encyclopædia Britannica Online: Pollination (2014) [Online; accessed 27-June-2014].
8. Takhtajan, A.: Flowering Plants. Springer, New York (2009)
9. Cronk, J., Fennessy, M., Siobhan, M.: Wetland plants: biology and ecology. CRC Press/Lewis Publishers, Boca Raton (2001)
10. Maiti, R., Satya, P., Rajkumar, D., Ramaswam, A.: Crop plant anatomy. CABI, Wallingford (2012)
11. Yang, X.S., Karamanoglu, M., He, X.: Multi-objective flower algorithm for optimization. Procedia Computer Science **18** (2013) 861–868
12. Kennedy, J., Eberhart, R.: Particle Swarm Optimization. In: Proceedings of IEEE International Conference on Neural Networks, vol. IV. (1995) 1942–1948
13. Yang, X.: Nature-Inspired Optimization Algorithms. Elsevier, London (2014)
14. Liang, J., Qu, B., Suganthan, P., Hernandez-Diaz, A.: Problem Definitions and Evaluation Criteria for the CEC 2013 Special Session and Competition on Real-Parameter Optimization. Technical Report 201212, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore (2013)
15. Bratton, D., Kennedy, J.: Defining a standard for particle swarm optimization. In: Swarm Intelligence Symposium, 2007. SIS 2007. IEEE. (2007) 120–127
16. Łukasik, S., Kowalski, P.: Flower Pollination Algorithm – facts, conjectures and improvements. In preparation (2014)