# Genetic Algorithm and Particle Swarm Optimization Combined with Powell Method

David Bento*,†, Diana Pinho*,†, Ana I. Pereira*,** and Rui Lima*,†

*Polytechnic Institute of Bragança, Portugal
†CEFT, FEUP, Porto University, Portugal
**Algoritmi, University of Minho, Portugal

**Abstract.** In recent years, the population algorithms are becoming increasingly robust and easy to use, based on Darwin's Theory of Evolution, perform a search for the best solution around a population that will progress according to several generations. This paper present variants of hybrid genetic algorithm - Genetic Algorithm and a bio-inspired hybrid algorithm - Particle Swarm Optimization, both combined with the local method - Powell Method. The developed methods were tested with twelve test functions from unconstrained optimization context.

**Keywords:** Genetic Algorithm. Global Optimization. Local Optimization.

## INTRODUCTION

In this paper it is analyzed the behavior of genetic algorithm and the particle swarm algorithm combined with local search. The main proposal of the present work is to obtain an efficient and robust population-based method that is able to identify the global solution with good precision under an unconstrained optimization context. For these methods non specific conditions can be imposed to the optimization problems.

This paper is organized as follows. The Section 2 describes the local and global methods that were tested and implemented. The 3rd section presents the numerical results and finally the last section presents the conclusions as well as future work.

## GLOBAL AND LOCAL OPTIMIZATION

The global optimization aims to identify the global solution of an optimization problem. In this work, two methods were used for global optimization, Genetic Algorithm (GA) and Particle Swarm Optimization (PSO). These techniques allow to find an approximate solution of the problems under study. It has also been used a local optimization, Powell Method, which allows to refine the results obtained in the global search method and as a result it is possible to obtain solutions with higher accuracy.

### Powell Method

The classical methods of non-evolutionary optimization for continuous problems can be mainly classified into direct methods, gradient and Hessian research methods. Direct methods determine the direction of research without the use of derivatives. The Powell method belongs to the direct testing methods, i.e, there is absence of derivatives of first or second order, based on the concept of conjugate directions and states that if the objective function is quadratic then can be minimized in $n$ iterations [5].

---

Powell Method

1. Starting from the initial point, calculate the first search direction: $S^i, i = 1, ..., n$.
2. Find the step $\alpha^*$ by minimizing $F(X + \alpha^* S^i)$.
3. Update $X^{i+1} = X^i + \alpha_i^* S^i$.
4. Update the $H$ and replace in the column $n$ for $\alpha_{n+1}^* S^{n+1}$: $H = [\alpha_1^* S^1, \alpha_2^* S^2, ..., \alpha_n^* S^n]$

*11th International Conference of Numerical Analysis and Applied Mathematics 2013*
AIP Conf. Proc. 1558, 578-581 (2013); doi: 10.1063/1.4825557
© 2013 AIP Publishing LLC 978-0-7354-1184-5/$30.00

5. Calculate the new conjugate direction from H: $S^{n+1} = \sum_{i=1}^{n} \alpha_i^* S^i$
6. Repeat the steps until the stop criterion is not satisfied: $\left|F(X^{k+1}) - F(X^K)\right| < 10^{-6}$.

---

## Genetic Method

Genetic algorithms are based on theory of evolution of species from Darwin [1]. The genetic algorithm starts with a set of solutions called population, where the solution is represented by an individual and the population size is preserved through each generation. The objective function is evaluated in each individual. Then individuals are selected according to their objective value. Those selected will be reproduced up randomly,by using genetic operators such as mutation and crossover. Individuals with less values have a high probability of being selected whereas the new generation of individuals may have a minor objective value than the previous generation. The evolution process is repeated until the stopping criterion is satisfied [2].

---

Hybrid Genetic Method

1. Generating population of chromosomes randomly with dimension $N$.
2. From this population are selected the best 50% - elitism.
3. From the elite population selected before, with 50% of them is made the crossing and in the other 50% is applied mutation.
4. The new population is evaluated.
5. Until the stopping criterion is satisfied ($|F(X^{k+1}) - F(X^K)| < 10^{-6}$ ), repeat the step 2 to 5.
6. Apply the Powell Method to the best solution obtained by genetic algorithm.

---

A second version of the algorithm described above was implemented where the difference lies in the fact that GA2 has implemented a different elitism step. This means that the generation of the population by crossing the GA2 chooses only the best "son" of each crossover, while the GA1 for each intersection have generated two "children." The remaining steps are all equivalent between these two algorithms.

## Particle Swarm Optimization

The Particle Swarm Optimization (PSO) belongs to a class of algorithms inspired by natural social intelligent behaviors, called Swarm Intelligence (SI). Developed by Kennedy and Eberhart (1995) [3], this method consists in optimizing an objective function through the exchange of information between individuals (particles) of a population (swarm). The PSO idea is to perform a set of operations and move each particle to promising regions in the search space. At each iteration the particles are moved as if they were changed in $n$-dimensional space, this occurs by the application of respective velocities. At each iteration the velocity of each particle is adjusted. The velocity calculation is based on the best position found by the neighborhood of the particle, the best position found by the particle itself - *pbest* and the best position found by the whole population, taking into account all particles - *gbest* or the best position overall [4].

---

Particle Swarm Optimization Method

1. Generate a population of particles with random positions and velocities.
2. Compare the value of *pbest* obtained with the particle $i$. If the value is better, update the *pbest* with the new value.
3. Compare the value obtained with the best overall value. If it is better, update *gbest* with the new value.
4. Update the particle velocity according to: $v_i^{k+1} = wv_i^k + c_1 r_1 (pbest_i - x_i^k) + c_2 r_2 (gbest_i - x_i^k)$, where $v_i^{k+1}$ is the velocity of the particle $i$, $k$ is the number of iterations and $w$ a parameter that representing the inertial particle and controls the operating capacity of the space of solutions; $c_1$ and $c_2$ are the parameters of confidence and $r_1$ and $r_2$ are random numbers in $[0,1]$ [4].
5. Update the particle position according to: $x_i^{k+1} = x_i^k + v_i^{k+1}$.
6. Repeat the step 2 to 6, until the stopping criterion is satisfied ($|F(X^{k+1}) - F(X^K)| < 10^{-6}$).
7. When the criterion is satisfied the Powell Method is applied.

---

## NUMERICAL RESULTS

The numerical results were obtained using a Intel(R) Core (TM) i3 CPU M330@2.13GHz with 8.00 GB of RAM.

The algorithms AG1, AG2 and PSO were implemented in Matlab to perform a global search of the unconstrained problem, hybridized with local search method, Powell Method, in order to refine the global solution found and obtain solutions with higher accuracy. Each problem was tested in 12 unconstrained test functions, obtained from Global Optimization Test Problems[6]. All methods are compared with the genetic algorithm present in the Matlab, hybridized with the local search performed by the Nelder Mead method.

**TABLE 1.** Test functions used to test the robustness of the implemented methods.

| Test Function | Methods | Rate of convergence (%) | Average objective function evaluations | Average time (s) |
|---|---|---|---|---|
| Ackley | GA1 | 100 | 25643 | 3.7 |
| | GA2 | 100 | 45423 | 4.3 |
| | GA | 100 | 1082 | 0.2 |
| | PSO | 100 | 3000038 | 33.8 |
| Beale | GA1 | 100 | 44411 | 5.9 |
| | GA2 | 100 | 83883 | 6.9 |
| | GA | 100 | 1285 | 0.2 |
| | PSO | 100 | 1065183 | 5.6 |
| Bohachevky | GA1 | 100 | 23925 | 3.1 |
| | GA2 | 100 | 48696 | 4.0 |
| | GA | 100 | 1046 | 0.2 |
| | PSO | 100 | 2504937 | 13.1 |
| Branin | GA1 | 100 | 19925 | 2.6 |
| | GA2 | 100 | 40596 | 3.4 |
| | GA | 100 | 1046 | 0.2 |
| | PSO | 100 | 3000043 | 15.4 |
| Griewank | GA1 | 100 | 187864 | 123.8 |
| | GA2 | 100 | 184984 | 123.7 |
| | GA | 100 | 1043 | 0.2 |
| | PSO | 100 | 2713671 | 29.4 |
| Hartmann | GA1 | 100 | 3021 | 0.5 |
| | GA2 | 100 | 10122 | 1.1 |
| | GA | 100 | 1064 | 0.2 |
| | PSO | 100 | 43 | 0.0 |
| Hump | GA1 | 100 | 19710 | 2.6 |
| | GA2 | 100 | 34754 | 2.9 |
| | GA | 100 | 1063 | 0.2 |
| | PSO | 100 | 844752 | 5.0 |
| Levy | GA1 | 100 | 61032 | 11.2 |
| | GA2 | 100 | 127059 | 16.9 |
| | GA | 100 | 2229 | 0.4 |
| | PSO | 90 | 3000139 | 91.4 |
| Powell | GA1 | 100 | 90459 | 15.9 |
| | GA2 | 100 | 151840 | 19.7 |
| | GA | 100 | 2941 | 0.5 |
| | PSO | 90 | 3000180 | 82.1 |
| Rastrigin | GA1 | 100 | 23674 | 3.3 |
| | GA2 | 100 | 49516 | 4.4 |
| | GA | 100 | 1052 | 0.2 |
| | PSO | 90 | 1332968 | 7.1 |
| Schwefel | GA1 | 100 | 22895 | 3.3 |
| | GA2 | 100 | 35295 | 3.6 |
| | GA | - | - | - |
| | PSO | - | - | - |
| Shekel | GA1 | 60 | 34977 | 6.0 |
| | GA2 | 60 | 58487 | 7.3 |
| | GA | - | - | - |
| | PSO | 10 | 60 | 0.0 |

Table 1 includes the results of all implemented methods, in which is shown the rate of convergence of each, the average ratings of the objective functions, and the average time taken to converge for a total of 10 runs of the problem. From the results it can be seen that the methods GA (1 and 2) have much higher convergence rates for most of the tested functions. The PSO in some cases has very low convergence rate and also appears that the PSO is a method that need more evaluations of the objective function, which may explain the fact that the method provides the time average processing, in general, higher than the other methods.

In comparison with the GA present in the Matlab, the two GA implemented in this work present better results for some problems, but the time of convergence and the number of evaluations of the objective function are less than for the others.

## CONCLUSIONS AND FUTURE WORK

In this study, it was compared some variants of the genetic algorithm and one variant of the particle swarm optimization algorithm. From the numerical results, it is possible to conclude that the methods GA (1,2) presented a greater robustness since managed to find an approximate solution for almost every problem.


## ACKNOWLEDGMENTS

The authors acknowledge the financial support provided by: PTDC/SAU-BEB/108728/2008, PTDC/SAU-BEB/105650/2008, PTDC/EME-MFE/099109/2008 and PTDC/SAU-ENB/116929/2010 from FCT (Science and Technology Foundation), COMPETE, QREN and European Union (FEDER), and by portuguese funds through FCT (Foundation for Science and Technology) within the project FCOMP-01-0124-FEDER-022674.


## REFERENCES


1. J. H. Holland,*Adaptation in Natural and Artificial Systems*, Ann Arbor, The University of MichiAGn Press, 1975.
2. M. Kumar, M.Husian, N. Upreti, D. Gupta, *Genetic Algorithm: Review and Application*, International Journal of Information Technology and Knowledge Management, 2(2), pp. 451-454, 2010.
3. J. Kennedy, R. Eberhart,*Particle Swarm Optimization*, IEEE International Conference on Neural Network, Perth, , pp. 1942-1948, 1995.
4. D. Bratton, J. Kennedy, *Defining a Standard for Particle Swarm Optimization*, IEEE Swarm Intelligence Symposium, 2007.
5. O. Kramer, *Iterated local search with Powell's method: a memetic algorithm for continuous global optimization*, Memetic Comp., 2, pp. 69 - 83, 2010.
6. Global Optimization Test Problems, `http://www-optima.amp.i.kyoto-u.ac.jp/member/student/hedar/Hedar_files/TestGO_files/Page364.htm`