

Particle Swarm Optimization and Information

Part 1 : Experimental results and comments

Maurice.Clerc@WriteMe.com

Introduction

There are different versions of Particle Swarm Optimization algorithms, but they informational point of view: what kind of information each particle has access to, a this, I study here two versions, a constricted one and an adaptive one, on three p easily completely analyzed in terms of probability to find a solution. They are compa

The three problems

Tableau1. The tree problems analyzed

Function name	Search space	Formula	Target value
Sphere 2D	$[-20, 20]^2$	$\sum_{d=1}^D x_d^2$	0
Rosenbrock 2D	$[-50, 50]^2$	$\sum_{d=1}^{D-1} 100(x_d^2 - x_{d+1})^2 + (1 - x_d)^2$	0
Rastrigin 2D	$[-5, 5]^2$	$\sum_{d=1}^D (x_d^2 - 10 \cos(2\pi x_d) + 10)$	0

The three methods

Random search rPSO

The dimensionality of the search space is finite, so that for each dimension k , there is a minimum value $x_{\min, k}$ and a maximum one $x_{\max, k}$. So, for each particle, the move is defined by:

$$\begin{cases} x(t+1) = (x_k), & k = 1, 2, \dots, D \\ x_k = \text{alea}(x_{\min, k} \dots x_{\max, k}) \end{cases}$$

At time step, the particle uses no (variable) information at all.

Constricted version cPSO

The version used here is PSO Type 1", as defined in [Kennedy 01] and tested in

$$\begin{cases} \varphi > 4 \\ \kappa \in]0, 1[\\ \chi = \frac{2\kappa}{\varphi - 2 + \sqrt{\varphi^2 - 4\varphi}} \\ v(t+1) = \chi(v(t) + \text{alea}(0 \dots \varphi)(p_i(t) - x(t)) + \text{alea}(0 \dots \varphi)(p_g(t) - x(t))) \\ x(t+1) = x(t) + v(t+1) \end{cases}$$

with $\varphi = 4.1$, $\kappa = 0.8$, a swarm size equal to 20, and a neighbourhood size equal to 3.

At each time step, the pieces of variable information a given particle knows and can

- its current position and the corresponding objective function value,
- its best position found so far, and the corresponding objective function value.

Adaptive version aPSO

The version is a simplified PSO (Adaptive Autonomous PSO):

$$\begin{cases} v(t+1) = \alpha v(t) + \text{beta} (p_g(t) - x(t)) \\ x(t+1) = x(t) + v(t+1) \end{cases}$$

with $\alpha = 0.5$, adaptive swarm size N in $[3, \infty]$, adaptive neighbourhood size in $[3, \infty]$.

At each time step, the particles of information a given particle can transmit are

- its current position and the corresponding objective function value,
- its best position found so far, and the corresponding objective function value
- its previous position (to estimate its improvement)
- its neighbourhood size,
- the swarm size (global information)

As we can see, and to summarize, depending on the algorithm, each particle knows:

- no (variable) information at all (rPSO),
- only local information (cPSO),
- a bit more local information and a global one (swarm size) (aPSO).

However, know is not enough, if you don't know edge. So, let us see now how "difficult" three problems and how the three algorithms cope with them.

Objective function complexity

Method

For a given objective function f , on a given search space S , is to find a solution? that, this question has no meaning. We have to precisely define what target value and an admissible error ϵ so that $|f(x) - T| < \epsilon$.

The solution set is then the set of all solution points, and is not depending on ϵ . It is one connected part, particularly if the function has several local optima. However, if except for some « strange » functions, it is always possible to define a measure (the cardinality $|S|$). For example, for a discrete search space it is just the number of solution points in S . It may be the total length of an union of intervals.

So we can study the relation between the admissible error and the solution area size so that $|S| = \xi(\epsilon)$. We do suppose here there is at least one solution point in the search space. If the objective function is finite, that is to say $f_{\min} < f_{\max}$, for simplicity, we do suppose target is equal to f_{\min} . So we have immediately an obvious point $(f_{\min}, 0)$ on the curve you admit any possible function value, so any point of the search space is a solution! Also minimum, we have the point $(f_{\min}, 0)$. But what happens between these values?

To study that "experimentally", we define some objective function value classes, t f_{\min} into n intervals $[f_i, f_{i+1}]$, with $f_0 = f_{\min}$ and $f_n = f_{\max}$. We also divide the search space into a lot of "cells" of size Δx and compute f for each cell. For each cell i , we compute f_i and f_{i+1} if $f_{i+1} \leq f_{\max}$.

So we finally have numbers. We cumulate them by $\sum_{j=1}^i c_j$, so that each finally an estimation $\xi(f)$.

In order to compare the three functions, we normalize the results by dividing them by

Results and comments

As we can see on Figure 1, we have three typical shapes. If we define each curve gives us in fact an estimation of how difficult it is to find a solution by chance, for a given course very easy (normalized solution area size almost equal to one) but when ε decreasing, we clearly see some differences.

What do say us the curves ? For all functions, they of course say the difficulty is increasing in a quite « normal » way (i.e. function, it says it is first quite easy (at the very beginning a bit more difficult far easier) and then suddenly far more difficult. For the Rastrigin function, it is a bit more difficult than for the two others (except at the very beginning), but it becomes a bit easier at the end.

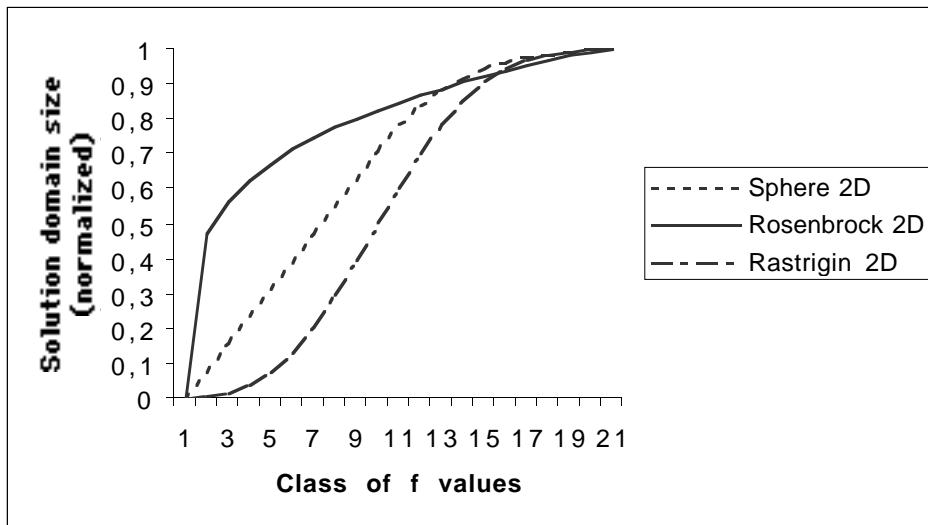


Figure.1 Difficulty to find a solution versus admissible error

Now, of course, we usually are not looking for a solution purely at random: we use an « intelligent » algorithm which find « interesting » positions and try to improve them. Results suggest, in particular, is that for some functions, it is more efficient to start at random. Below, we will try to see more precisely why.

Efficiency and success rate

Method

Now we use the three algorithms : Random, Constricted PSO, and Adaptive PSO. A move is considered "successful" if the new position is strictly better than the best one ever found so far.

At each time step, we count the total number of objective function evaluations (number of moves for there is no additional local search) and the total number of successful moves. The success rate (said « cumulated », for it takes into account all moves from the beginning only the moves during the current time step).

Results and comments

For each function, and each algorithm, the stop criterion is an admissible error. The Adaptive method is always better, for it finds an admissible solution is here not the point. The point is that for Rosenbrock 2D and Rastrign 2D, the random search, to find some interesting positions. It is particularly clear for the compare with Constricted PSO. Nevertheless, the good news is that even for this case beginning, Adaptive PSO is almost as good as the random search. This is in fact better times to times new particle almost at random.

So, in some cases, it would be better to begin with pure random moves. However, in some cases, and how long you should do that, and, as we can clearly see, using the random search gives very bad results. So, finally, it appears Adaptive PSO is a good compromise. The intuitive idea seems to be right: the more information you use, the more efficient you are.

However, it is not so obvious, for using information is not at all costless, so, if you look on the results but also on the cost to obtain them, the function "efficiency vs cost" is always be increasing, and, more precisely, may have a maximum. Among other things, see in Part 2.

References

- [CLE 01a] Clerc M., Guided Random Generation for Adaptive Particle Swarm Optimization, France '01.
- [CLE 01c] Clerc M., Kennedy J., "The Particle Swarm: Explosion, Stability, and Convergence in Complex Space", IEEE Journal of Evolutionary Computation, vol. in press, 2001,
- [KEN 01a] Kennedy J., Eberhart R., Shi Y., Swarm Intelligence, Academic Press, 2001.

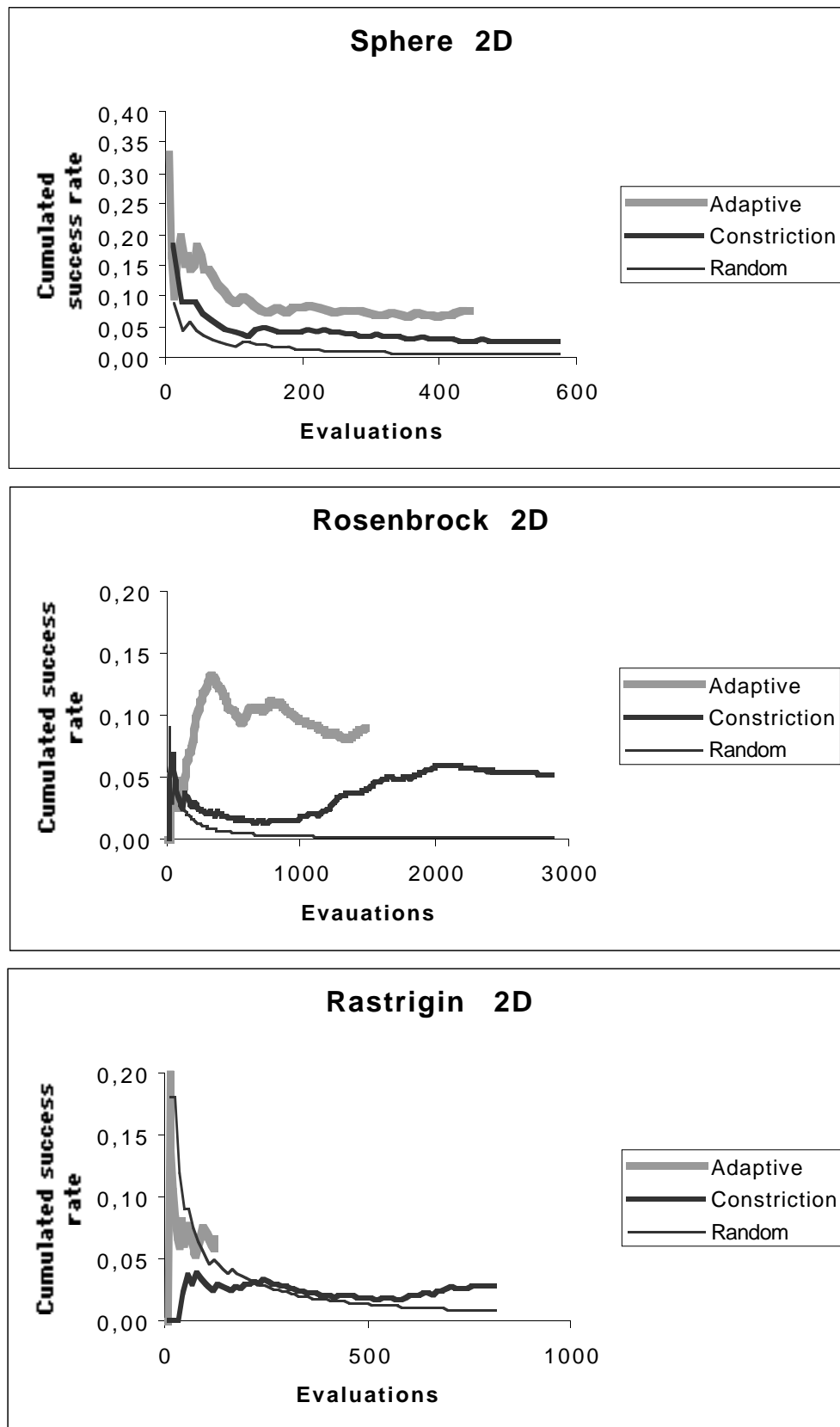


Figure.2 Cumulated success rates for three functions and three methods