# The Swarm & The Queen. Towards a Deterministic and Adaptive Particle Swarm Optimization.

Maurice Clerc
Maurice.Clerc@WriteMe.com
France Télécom, 12 av. de Chevêne, 74000 Annecy, France

*Un trésor est caché dedans.*
*Je ne sais pas l'endroit; mais un peu de courage*
*Vous le fera trouver.*
Le Laboureur et ses Enfants. Jean de La Fontaine [1]

***Abstract***

We present a very simple Particle Swarm Optimization iterative algorithm, with just one equation and one social/confidence parameter. We define a "no-hope" convergence criterion and a "re-hope" method so that, from time to time, the swarm re-initializes its position, according to some gradient estimations of the objective function and to the previous re-initialization (it means it has a kind of very rudimentary memory).

We study then two different cases, a quite "easy" one (the Alpine function) and a "difficult" one (the Banana function), but both just in dimension two. The process is improved by taken into account the swarm gravity center (the "queen") and the results are good enough so that it is certainly worth to try the method on more complex problems.

## *All for one and one for all*

Suppose you and your friends are looking for a treasure in a field. Each digger has a audio detector and can communicate to his *n* nearest neighbours the level of the sound he heards and his own position. So each digger knows whether one of his neighbours is nearer to the objective than him and, if it is the case, can move more or less towards this damned lucky guy, depending on how much he trusts him. So, all together, you may find the treasure more quickly than if you were alone. Or suppose you and your friends are looking for the highest mountain in a given country. You all have altimeters and can communicate , etc...

This kind of algorithm, called Particle Swarm Optimization (PSO), has been first and is still largely experimentally studied: it is indeed extremely efficient ([1-7]). In 1998, a mathematical explanation, introducing a five dimensional space of states and some constriction coefficients for sure convergence, has been found ([8]) but as there where still no sure way to choose a priori the best parameters, one of them (the social/confidence coefficient $\varphi$) is randomly modified at each time step. Also PSO is usually using a system of two iterative equations, one for the positions and one for the velocities of the particles, with several parameters. It gives more "freedom" to the system but it is also then quite difficult to find the best parameters values.

On the contrary, we present here a pure deterministic algorithm, with just one equation, one confidence coefficient, and one "memory" parameter. The core of what we could call a No-hope/Re-hope method is an adaptive process freely inspired by the *outer iteration* procedure as described in [9]. At each time step we examine whether there is still " hope" to reach the objective and, if not, we reinitialize all positions around the best one, taken into account the local shape of the objective function, as it can be estimated by using the particles current and previous positions and

[1] A treasure is hidden inside / I don't know the place: but with a bit of courage / You will find it.

The Ploughman and his Children. Jean de La Fontaine function values. Also, we examine if the gravity center (the "queen") is itself in a solution point: it does not cost a much in processor time and it appears it sometimes greatly speeds up the convergence.

### *Equations and Theoretical Convergence Curves*

A numerical function *f* is defined on a subspace $\Omega$ of $R^D$. We are looking for a point $x_S = \left(x_{S,1},\ldots,x_{S,D}\right)$ so that, for a given objective value *S* and a given acceptable error value $\varepsilon$, we have $f\left(x_S\right) \in V = [S-\varepsilon, S+\varepsilon]$. For simplicity the space of search is a hypercube $H = \left[x_{\min}, x_{\max}\right]^D, H \subset \Omega$. It means we have at least an idea of where the objective point is. We define the objective function *g*

$$g(x) = \left|S - f(x)\right| \qquad \textbf{Equ. 1}$$

The basic iterative representation we use here is just

$$x(t+1) = x(t) + \varphi\left(p - x(t)\right) \qquad \textbf{Equ. 2}$$

where $\varphi$ is the social or confidence coefficient, and *p* the point to which the particles have to converge.
From the Equ. 2, if we consider a sequence of time steps in which *p* is a constant, we have immediately

$$p - x(t) = (1-\varphi)^t\left(p - x(0)\right) \qquad \textbf{Equ. 3}$$

The convergence condition is then

$$\left|1-\varphi\right| \in \left]0,1\right[ \text{ or } \varphi \in \left]0,1\right[ \cup \left]1,2\right[ \qquad \textbf{Equ. 4}$$

In particular, if we want a convergence to *p* with an admissible error $\varepsilon$, and if the objective function is not too "sharp" and, more important, not too "flat" around *p*, that is to say if we have in practice

$$\left|g(p) - g(x)\right| \approx \varepsilon \text{ for } |\text{x - p}| = \varepsilon \qquad \textbf{Equ. 5}$$

we obtain the convergence time *T*

$$T = \frac{\ln\left(\dfrac{\varepsilon}{\left|p - x(0)\right|}\right)}{\ln\left(\left|1-\varphi\right|\right)} \qquad \textbf{Equ. 6}$$

If we know nothing about *p* nor $x(0)$, a theoretical estimation of the convergence time $T_{theor}$ is then (defining $\Delta x = x_{\max} - x_{\min}$)

$$T_{theor} = \frac{\ln\left(\dfrac{4\varepsilon}{\Delta x}\right)}{\ln\left(\left|1-\varphi\right|\right)}, \varphi \in \left]0,1\right[ \cup \left]1,2\right[ \qquad \textbf{Equ. 7}$$

By plotting $T_{theor}$ versus $\varphi$, we obtain a performance curve. Figure 1 shows a typical one. Note that due to the hypothesis in Equ. 5, we may in practice obtain a better performance curve with some particular objective functions: the theoretical one is just a good guideline.

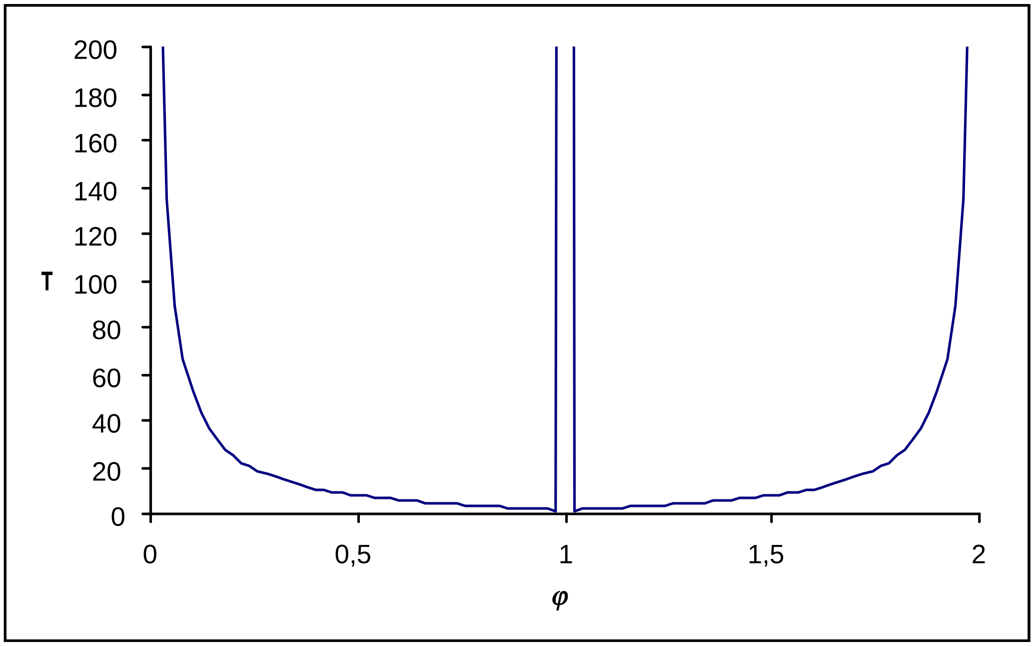


**Figure 1. Typical theoretical performance curve ($\varepsilon$=0.01, $\Delta$x=10).**

### ***No-hope criterion***

We can define the "velocity" of a particle by

$$v(t) = x(t+1) - x(t) \qquad \textbf{Equ. 8}$$

It means we have here

$$v(t) = \varphi(p - x(t-1)) = \varphi(1-\varphi)^t (p - x(0)) \qquad \textbf{Equ. 9}$$

So the maximum fly of a particle between time steps $t_1$ and $t_2$, i.e. the distance to the farthest point it can reach, is

$$\begin{cases} F_{t_1,t_2} = \sum_{t=t_1}^{t_2} |v(t)| = |p - x(0)| \left( (1-\varphi)^{t_1} - (1-\varphi)^{t_2} \right) \text{if } \varphi \in ]0,1[ \\ \quad = |v(t_1)| \text{ if } \varphi \in ]1,2[ \end{cases} \qquad \textbf{Equ. 10}$$

In particular, the maximum remaining possible fly at time $t$ is

$$\begin{cases} F_{t,\infty} = |p - x(0)|(1-\varphi)^t \text{ if } \varphi \in ]0,1[ \\ F_{t,\infty} = |p - x(0)|\varphi|1-\varphi|^t \text{ if } \varphi \in ]1,2[ \end{cases} \qquad \textbf{Equ. 11}$$

If we define the diameter of the swarm by

$$\Theta(t) = \max_{(i,j)\in[1,N]^D} |x_i - x_j| \qquad \textbf{Equ. 12}$$

and by noting that we have

$$\Theta(t) = |p - x(0)||1-\varphi|^t \qquad \textbf{Equ. 13}$$

an estimation of the space of search diameter at each time step is then given by the formula

$$\begin{cases} \tilde{H}_t = \Theta(t) \text{ if } \varphi \in ]0,1[ \\ \tilde{H}_t = (2\varphi - 1)\Theta(t) \text{ if } \varphi \in ]1,2[ \end{cases} \qquad \textbf{Equ. 14}$$

As we can see, it is decreasing.

Now let $x_{grav}$ be the gravity center of the swarm (see the pseudo-code below for two possible definitions). We can have an idea of the local shape of the objective function by examining the values

$$\frac{\Delta g_i}{\Delta x_i} = \frac{g(x_i) - g(x_{grav})}{x_i - x_{grav}} \qquad \textbf{Equ. 15}$$

Finally, we define our "no-hope" criterion by writing the space of search is too small (in the case, of course, we have still not found a solution, even in the gravity center)

$$\begin{cases} \tilde{H}_t < 2\varepsilon', \text{ with } \varepsilon' = \varepsilon \dfrac{N}{\sum_{i=1}^{N} \dfrac{\Delta g_i}{\Delta x_i}} \\ g(x_{grav}) \notin V \end{cases} \qquad \textbf{Equ. 16}$$

### *Re-hope method*

In practice, $p$ may be modified during the search. The simplest way is probably to consider, for a given particle, $p$ is the best position found in the neighbourhood of this particle at each time step, and even to consider this neighbourhood itself is the whole swarm. So let us call $x_{best}$ the best position at a given time step.

Let $n_{re\_init}$ be the number or reinitializations which have already happened, and $\varphi_{re\_hope}(n_{re\_init})$ be the memory parameter. In this Re-hope method, we define a new swarm position "around" the previous best particle $x_{best}$ so that its diameter along the dimension $d$ is defined by

$$\Theta'_d(t) = \varphi_{re\_hope}(n_{re\_init})\left(\Theta_d(t) + 2\frac{g(x_{best})}{\Delta_d g}\right) \qquad \textbf{Equ. 17}$$

with

$$\varphi_{re\_hope}(n_{re\_init}) = \varphi_{re\_hope}(n_{re\_init} - 1)\varphi_{re\_hope}(0) \qquad \textbf{Equ. 18}$$

$$\Theta_d(t) = \max_{(i,j)\in[1,N]^D} \left| x_{i,d} - x_{j,d} \right| \qquad \textbf{Equ. 19}$$

and $\Delta_d g$ an estimation of the gradient of $g$ along the dimension $d$. In practice, it is calculated as follow:

$$\begin{cases} x'_{i,d'} = x_{i,d'}(t) \text{ if } d' \neq d \\ \quad\;\;\; = x_{i,d'}(t-1) \text{ if } d' = d \end{cases} \qquad \textbf{Equ. 20}$$

$$\Delta_d g = \frac{1}{N}\sum_{i=1}^{N} \frac{g(x_i) - g(x'_i)}{x_{i,d}(t) - x'_{i,d}}$$

Of course, we suppose we have $\varphi_{re\_hope}(0) \geq 1$ (typically 1.1). In practice, in the examples below, each particle has its coordinates redefined by

$$x_{i,d}(t) = x_{best,d}(t) + \Theta'_d(t)\left(\frac{x_{i,d}(0)}{\Delta x} - \frac{1}{2}\right) \qquad \textbf{Equ. 21}$$

It means the new swarm position is quite similar to the initial position, but usually "distorted" along some dimensions.

### *Swarm & Queen algorithm*

The high level pseudocode of the algorithm for a given swarm size is given below. Note that as the space of search is globally decreasing during the process, the best initial position seems to be a regular disposition on the "frontier" of the hypercube $H$, but this particular point would need more investigation, particularly for $\varphi$ values greater than 1. So we will just examine the case $\varphi \in ]0,1[$ Note also that the algorithm could be easily modified to search an extremum, but we would not have then a rigorous success criterion.

THE SWARM&QUEEN ALGORITHM

<choose the acceptable error $\varepsilon$>
<initialize the particle positions>
(comment: in our particular case, put the particles regularly on the frontier of $H$>
<choose the $\varphi$ value>
<choose the $\varphi_{re\_hope}(0)$ value>
<choose the maximum acceptable number of time steps $t_{\max}$ >
(comment: the theoretical convergence time (Equ. 7) gives an idea of what it should be)
$t = 0$
while $t \le t_{\max}$ do
    <compute the gravity center $x_{grav}$> (comment: either weighted or unweighted, see below)
    if $f\left(x_{grav}\right) \in V$ then <SUCCESS; STOP> else
        <find the *best* particle in the swarm>
        (comment: the one for which $g(x_i)$ is the smallest)
        for each particle $i$ do
            $x_i(t+1) = x_i(t) + \varphi_t\left(x_{best}(t) - x_i(t)\right)$
            if $x_i(t+1) \notin \Omega$ then <keep it on the frontier of $\Omega$>
            (comment: mathematically necessary to be always able to compute $g(x)$)
            if $f\left(x_i(t+1)\right) \in V$ then <SUCCESS; STOP>
        if $\tilde{H}_t < 2\varepsilon'$ then <use the Re-hope method>
    $t = t + 1$
end while
FAILURE
end

UNWEIGHTED GRAVITY CENTER (UGC)

$$x_{grav} = \frac{\sum_{i=1}^{N} x_i}{N} \qquad \textbf{Equ. 22}$$

WEIGHTED GRAVITY CENTER (WGC)

$$x_{grav} = \frac{\sum_{i=1}^{N} x_i / g(x_i)}{\sum_{i=1}^{N} 1/g(x_i)} \qquad \textbf{Equ. 23}$$

(comment: the better the particle the bigger its weight)

We now examine some results in two cases: the Alpine function, which is quite easy, and the Banana function, a bit more difficult, both in dimension 2, for we are just trying for the moment to *understand* what happens.

## *Two examples*

### The Alpine function

This function is defined by

$$f(x_1, \ldots, x_D) = \sin(x_1) \ldots \sin(x_D) \sqrt{x_1 \ldots x_D}, (x_1, \ldots, x_D) \in [0, x_{\max}]^D \qquad \textbf{Equ. 24}$$

In dimension two and on $[0,10]^2$ it gives Figure 2. With a lot of imagination, you can almost recognize the French Côte d'Azur on the south and the Mont Blanc as the highest summit. This function is interesting for testing the search of an extremum for the following reasons:

- there are as many local extrema as we want, just by increasing $x_{max}$,
- there is just one global extremum,
- the solution can easily be directly computed.

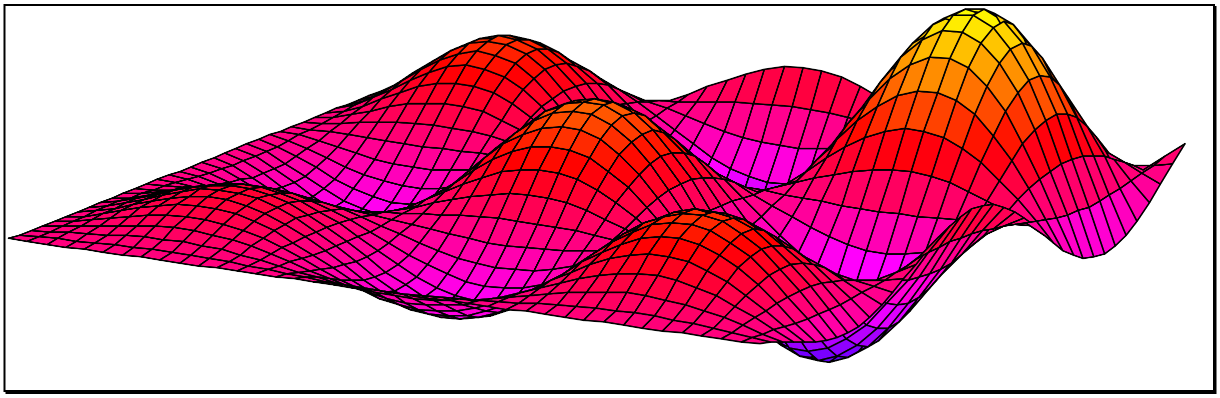

**Figure 2. The 2D Alpine function.**

In any dimension $D$, in the hypercube $[0,10]^D$, the maximum is at the point $(x_s, x_s, \ldots, x_s) \in [0,1]^D$ where $x_s$ is the solution of

$$\tan(x) + 2x = 0, x \in \left]\frac{5\pi}{2}, \frac{6\pi}{2}\right[ \quad \textbf{Equ. 25}$$

that is to say the point $(7.917, \ldots, 7.917)$. The maximal value is then about $2.808^D$.

## The Banana function

Rosenbrock's valley (De Jong's function 2, Banana function) is a classic optimization problem. The global minimum ($S$=0) is inside a long, narrow, parabolic shaped flat valley, and convergence to the solution point (1,1) is well known to be difficult. In 2D the equation of the surface is

$$f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2 \quad \textbf{Equ. 26}$$

Figure 3 shows how looks like the function for $(x_1, x_2) \in [-2,2]^2$ (as the values of the function are quite big, the scale has been reduced along the third dimension).

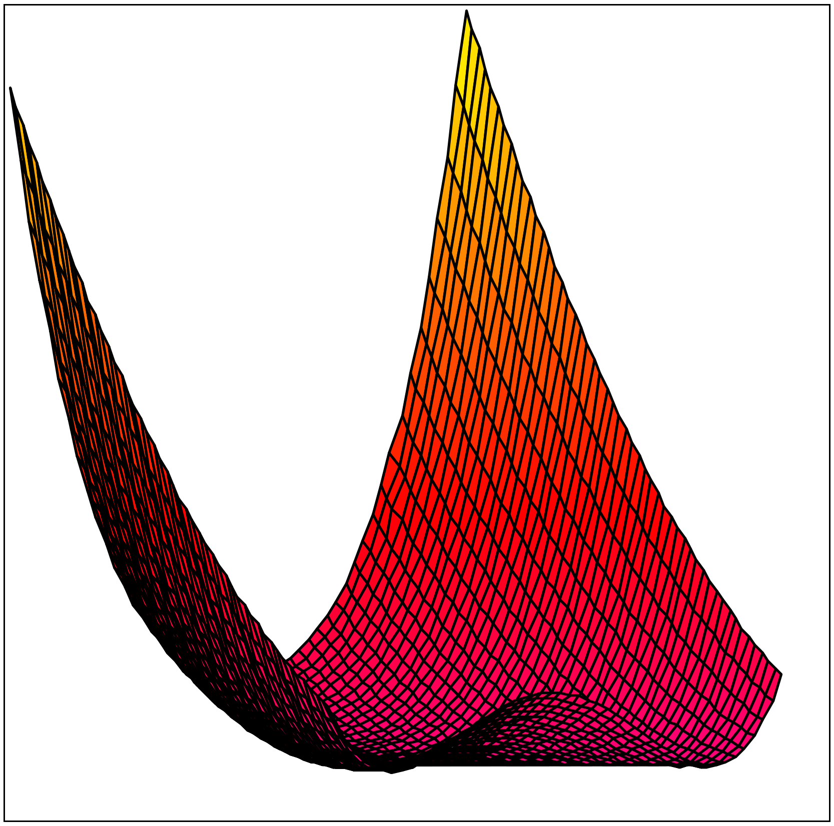

**Figure 3. The Banana function (reduced scale for function values).**

## Results

We study here what happens when we are looking for the (known) maximal value of the 2D-Alpine function on $[0,10]^2$, and the (known) minimal value of the 2D-Banana function. In both cases the admissible error is $\varepsilon$=0.01. We try the 49 $\varphi$ values (0.02, 0.06,...,0.98), with $t_{\max} = 400$ and $\varphi_{re\_hope}(0) = 1.05$. Figure 4 shows a typical convergence sequence. It is interesting to note how the swarm *almost* find a solution in just two time steps, but the solution area is so thin that it doesn't "see" it, and so it has to slowly come back.

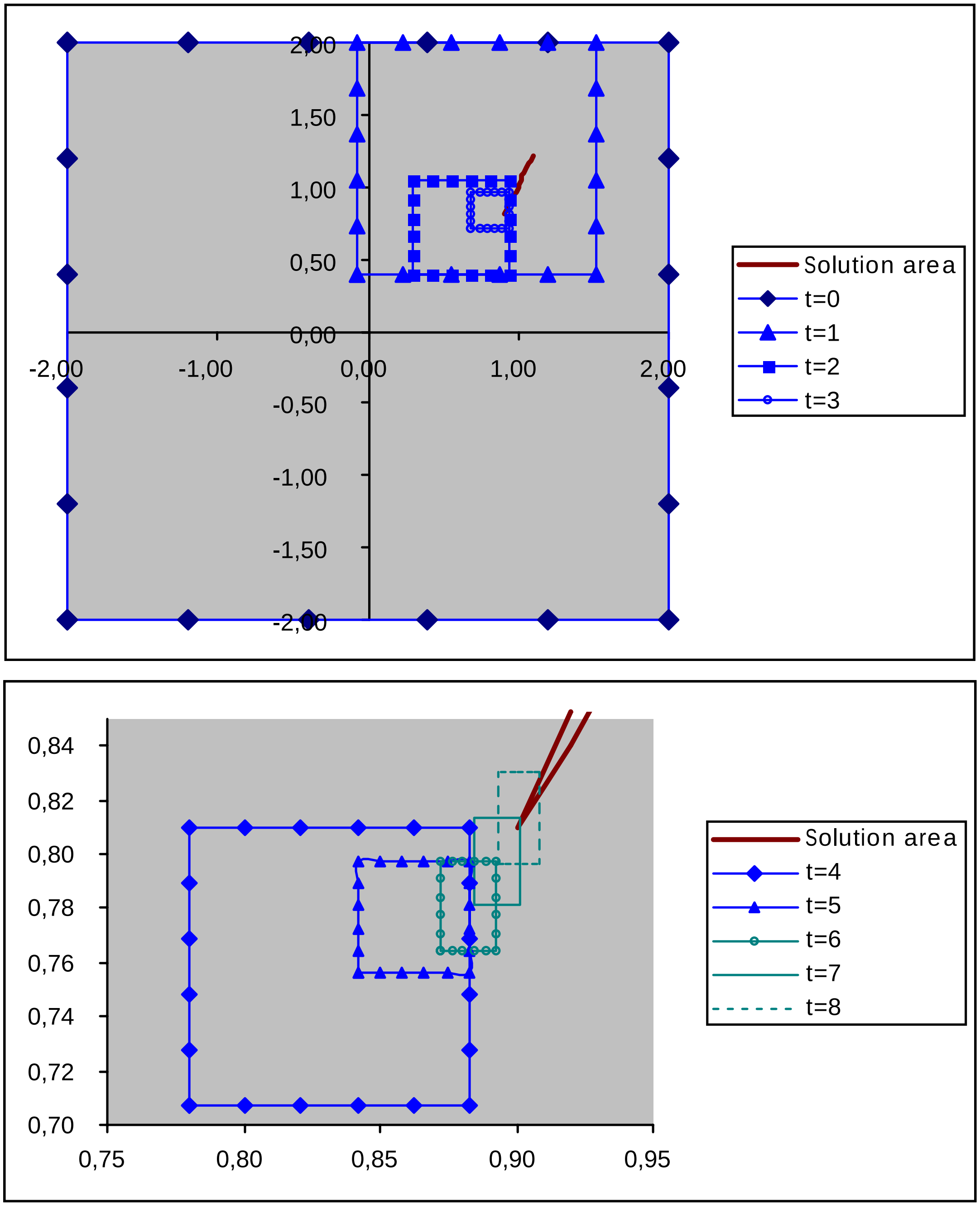


**Figure 4. A convergence sequence for the Banana function ($\varphi$=0.6, weighted gravity center).**

Some global performance curves are in Figure 5 (with each Re-hope process counted as a time step). We clearly see the swarm has indeed often some difficulties to find a solution for the Banana function.

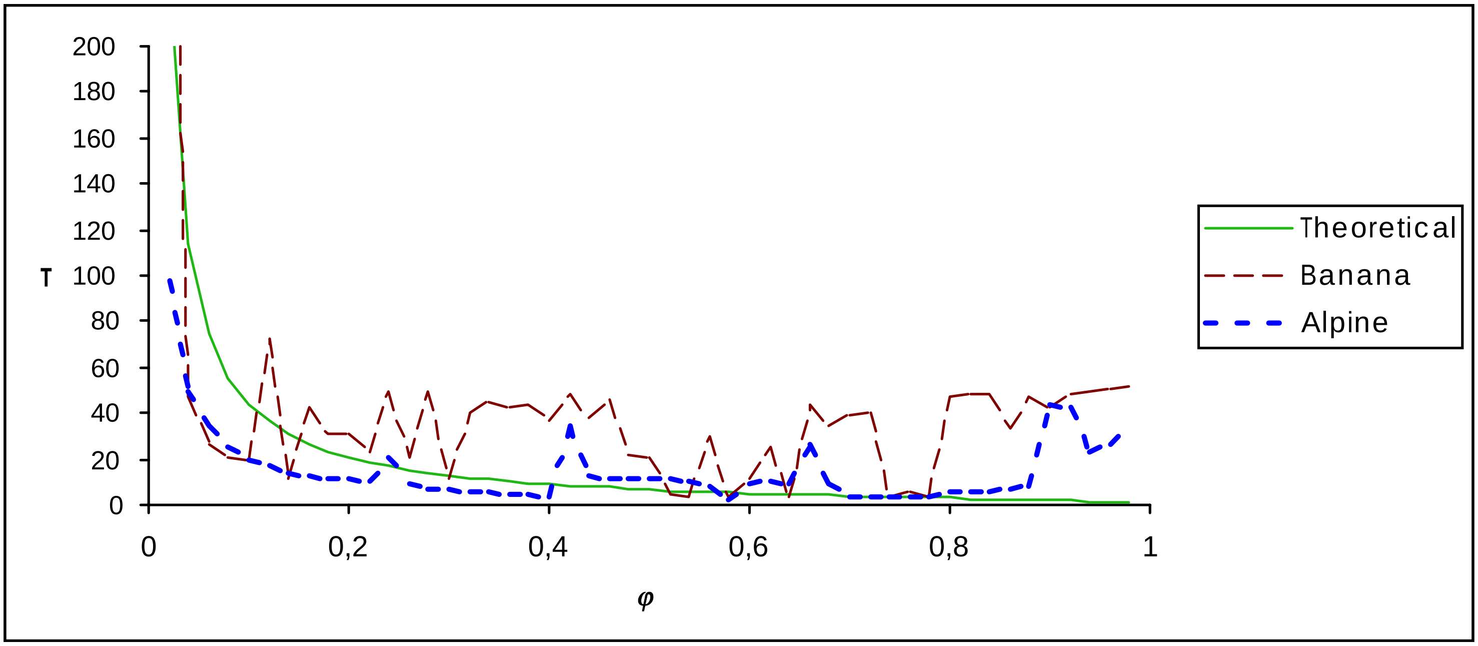


**Figure 5. Performance curves for the 2D Alpine and Banana functions (weighted grav. cent.).**

As the success ratio is simply 100% (which is *not* the case, for example, by using a "classical" PSO with a random $\varphi$), it makes sense to globally compare the curves to the theoretical one

$$quality_{\text{real curve vs theoretical}} = 1 - \frac{\sum_{\text{real curve}} \text{convergence times} - \sum_{\text{theoretical curve}} \text{convergence times}}{\sum_{\text{theoretical curve}} \text{convergence times}} \quad \textbf{Equ. 27}$$

Also it is interesting to note how many successes are obtained thanks to the queen, to the Re-hope method or both.. Finally, we obtain the Table 1.

| | **Alpine function** | | **Banana function** | |
|---|---|---|---|---|
| | UGC | WGC | UGC | WGC |
| **Success ratio** | 100 % | 100 % | 100 % | 100 % |
| **a) last step thanks to the queen** | 55 % | 96 % | 2 % | 61 % |
| **b) the Re-hope procedure has been used** | 59 % | 49 % | 73 % | 75 % |
| **a) and b)** | 14 % | 47 % | 2 % | 39 % |
| **Quality vs theoretical estimation** | 1.19 | 1.40 | 0.44 | 0.47 |

**Table 1. Some quality ratios.**

### *Discussion*

In this four cases, the Swarm&Queen method works extremely well, although the WGC option clearly improves the performance. Interestingly, the convergence is not always obtained for the same reasons. Sometimes the Re-hope procedure is used, sometimes not, sometimes the successful time step is due to the queen, sometimes not. This suggests that the three mechanisms

- normal iteration
- gravity center
- Re-hope procedure

are indeed able to cooperate. Nevertheless it is still unclear how the best $\varphi_{re\_hope}(0)$ has to be chosen. It has been done here experimentally (for example results are a bit less good with the value 1.5) and this is of course not very satisfying. Some preliminary results show the optimal value is in fact slightly depending on $\varphi$. So an obvious research direction is now to clear up this theoretical point.

### Acknowledgments

To Jim Kennedy, for his kind proofreading and for the "weighted gravity center" idea.
To Neil Hickox for his nice MacOS 3D Surfaces software.

### References

[1] P. J. Angeline, “Using Selection to Improve Particle Swarm Optimization,” presented at IEEE International Conference on Evolutionary Computation, Anchorage, Alaska, May 4-9, 1998.
[2] R. C. Eberhart and J. Kennedy, “A New Optimizer Using Particles Swarm Theory,” presented at Proc. Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, 1995.
[3] J. Kennedy and R. C. Eberhart, “Particle Swarm Optimization,” presented at IEEE International Conference on Neural Networks, Perth, Australia, 1995.
[4] J. Kennedy and W. M. Spears, “Matching Algorithms to Problems: An Experimental Test of the Particle Swarm and Some Genetic Algorithms on the Multimodal Problem Generator,” presented at IEEE International Conference on Evolutionary Computation, 1998.
[5] Y. H. Shi and R. C. Eberhart, “A Modified Particle Swarm Optimizer,” presented at IEEE International Conference on Evolutionary Computation, Anchorage, Alaska, May 4-9, 1998.
[6] Y. Shi and R. C. Eberhart, “Parameter Selection in Particle Swarm Optimization,” : Indiana University Purdue University Indianapolis, 1998.
[7] J. Kennedy and R. C. Eberhart, “A discrete binary version of the particle swarm algorithm,” presented at International Conference on Systems, Man, and Cybernetics, 1997.
[8] M. Clerc and J. Kennedy, “The Particle Swarm: Explosion, Stability, and Convergence in a Multi-Dimensional Complex Space,” *IEEE Journal of Evolutionary Computation*, vol. in press, No. , 2001.
[9] H. Niederreiter and P. Peart, “Localisation of search in quasi-Monte Carlo methods for global optimization,” *SIAM J. Sci. Statist. Comput.*, vol. 7, No. , pp. 660-664, 1986.