Stagnation Analysis in Particle Swarm Optimisation

or

What Happens When Nothing Happens

Maurice Clerc
(Edited by Riccardo Poli)

Department of Computer Science
University of Essex
Technical Report CSM-460
ISSN: 1744-8050
August 2006

Introduction

The "classical" version of Particle Swarm Optimiser (PSO) is very simple but, before it can be executed, the user needs to define some parameters (swarm size, neighbourhoods, and some coefficients). Adaptive versions, like TRIBES [1, 2, 4], don't have this drawback. Also, they can often be more effective when compared to the classical PSO using a performance criterion which takes into account only the best fitness value achieved and the number of fitness evaluations. However, adaptive PSO may require more computation per fitness evaluation, since they perform some intermediate computations in order to take advantage of the information collected during the search process. This overhead can reduce the benefits of adaptive PSOs, and it can even rule them out for some (quasi) real time applications.

That is why it is still interesting to just improve the classical version of PSO, without complicating it too much. The aim of this paper is to mathematically analyse the behaviour of the particles when there is no improvement over several time steps, and to derive optimal parameter settings for the PSO. Five of the PSO variants suggested by this analysis are tested on five benchmark fitness functions. This allows us to identify the most interesting ones.

Classical PSOs

Principles

Particles are agents moving in a D-dimension search space. The set of the particles in the space is called a swarm. Let S be the size of the swarm. At each time step, each particle is described by four features:

- its position $x = (x_1,...,x_D)$
- its velocity $v = (v_1,...,v_D)$
- the best position found so far $p = (p_1,...,p_D)$
- its "informant group", sometimes called its neighbourhood. This is effectively a set of links to other particles, which indicate who informs who in the swarm. These links are often defined at the beginning of a run and are kept fixed thereafter. However, no topology has been proved to be the best for all possible problems, and it is arguable that PSO with dynamic neighbourhoods may present some advantages. So, here we use a robust method (see details in [1, 4]) where each particle has K links which are randomly initialised at the beginning of a run and then again every time the swarm is unable to find an improved solution during an iteration of the algorithm.

The motion equations for a particle and for each dimension d are given by

$$\begin{cases} v_d \leftarrow c_1 v_d + c_{\max} alea(0,1)(p_d - x_d) + c_{\max} alea(0,1)(g_d - x_d) \\ x_d \leftarrow x_d + v_d \end{cases} \quad (1)$$

where $g = (g_1,...,g_D)$ is the best position found by the best informant and the two *alea(0,1)* are stochastic variables uniformly distributed between 0 and 1.

PSO-0

We need a "reference" algorithm, which we will call PSO-0. It is the one described above, with the following parameters:

$$\begin{cases} S = 30 \\ K = 3 \\ c_1 = 1/(2\ln(2)) \cong 0{,}72 \\ c_{\max} = (c_1 + 1)^2 / 2 \cong 1{,}48 \end{cases}$$

The first two values have been chosen so that they can be kept in all variants described later. The coefficient $c_{\max}$ is derived from c_1 by using the constriction method explained in [3] (see the Appendix for a brief explanation).

Stagnation phenomena

In classical PSOs as defined by Equation 1 each dimension is independent. So, the analysis can be performed on just one dimension. In order to simplify our notation, we will write *c* instead of $c_{\max}$, and we will omit the index *d*. In order to explicitly represent the time step we will use the variable *t*. Finally, the realisation of the random variable uniformly distributed in $[0, c]$ will be denoted with $\tilde{c}^k$, where the superscript *k* is used to distinguish between realisations. Equation 1 is then rewritten as

$$\begin{cases} v(t+1) = c_1 v(t) + \tilde{c}^1 (p - x(t)) + \tilde{c}^2 (g - x(t)) \\ x(t+1) = x(t) + v(t+1) \end{cases} \quad \textbf{(2)}$$

We say there is stagnation if *p* and *g* (which are now just numerical values) don't change over some time steps. In order to qualitatively observe the behaviour of the particles (along just one dimension, as already noted), we choose some usual values for parameters and initialisations:

$$\begin{cases} c_1 = 0{,}7 \\ c = 1{,}4 \\ p = 0 \\ g = 0{,}5 \\ v(0) = alea(-0{,}5, 0{,}5) \\ x(0) = alea(0, 1) \end{cases} \quad \textbf{(3)}$$

If the particle is the neighbourhood best

Let us study the behaviour of the best particle in the swarm. For this particle $p = g$. In figure 1 we can see the evolution of the velocity in a typical run: the velocity rapidly goes to zero, and the particle tends to its best known position, *g*. In other words, the particle quite rapidly gives up its search. In practice, it means that some fitness evaluations are wasted since the best particle (almost) freezes after a few iterations.

However, whether or not the local-best particle behaves in this way depends on the values of c_1 and *c*. For example, with the values 0.9 and 2.1, respectively, we obtain a divergent evolution as illustrated in figure 2. This behaviour also leads to wasted fitness evaluations. (Of course, it is always possible that the velocity will tend to zero later, although it appears unlikely.) The lesson that can be drawn is that there probably exist intermediate parameter settings for which the particle neither converges to zero nor diverges. We are interested in these values

since they ensure that the local best particle actively contributes to the search of the swarm. As we will see, the examination of what happens with particles that are not local best will suggests some possible solutions.

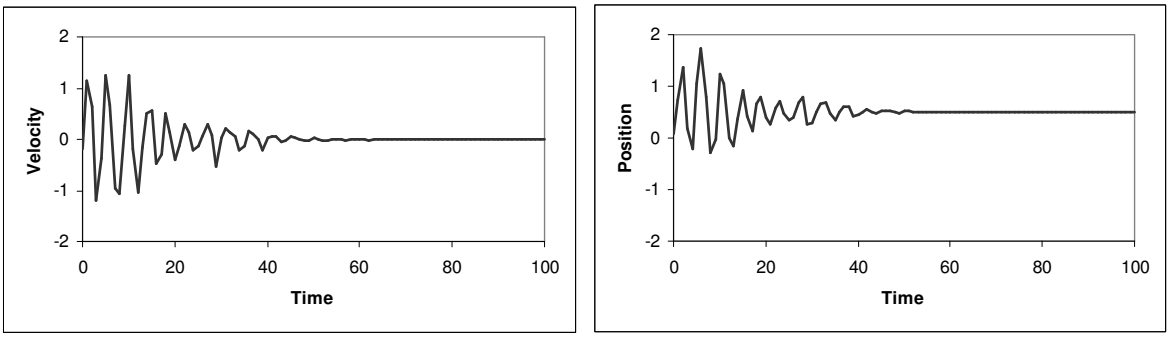

Figure 1. PSO-0. Behaviour of the best particle in case of stagnation (for typical parameter values)

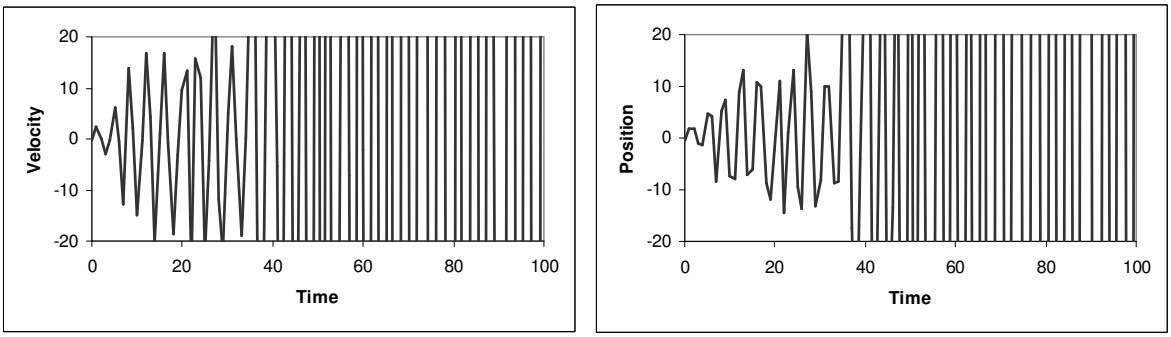

Figure 2. PSO-0. Behaviour of the best particle in case of stagnation (for unusual parameter values)

If the particle is not the best one

For non-best particles $p \neq g$ .The behaviour of these particles is indeed very different from that of the local best particle, even with typical parameter values. As we can see on figure 3, the velocity of such particles does not tend to zero, and, so, their position is permanently oscillates around a particular point, halfway between p and g.

A deeper examination shows that the module of the velocity follows a statistical law that is exponentially decreasing (cf. figure 4): small oscillations are frequent, more important ones are more rare, big ones are even more rare, etc. In other words, the longer the stagnation, the more probably big moves will occur. This is a quite desirable behaviour for then, even during stagnation, it performs a kind of local search that is usually fruitful: the particle will probably find a better position. So a quite obvious (small) improvement could be to do something so that the best particle has the same behaviour.

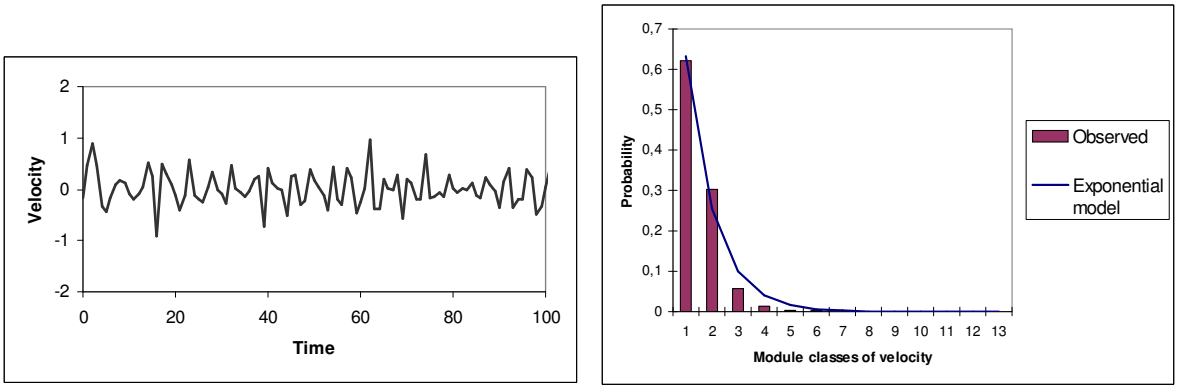

Figure 3. PSO-0. Behaviour of a non-best particle in case of stagnation (for typical parameter values). In case of long stagnation (100000 time steps in the plot on the right) the particles oscillates according to an approximately exponential probability distribution where small moves are quite frequent, while big ones are rarer (but do exist)

Stagnation analysis

.

General iterative equation and interpretation

In the particular case of $c = 0$, we obviously have $v(t+1) = c_1 v(t)$, and the convergence criterion is simply $c_1 < 1$ (we assume it is non negative). In the general case, from equation 2 we can successively write

$$v(t+2) = c_1 v(t) + \tilde{c}^3 (p - x(t+1)) + \tilde{c}^4 (g - x(t+1))$$

$$-x(t+1) = -v(t+1) + x(t)$$

$$-x(t) = \frac{1}{\tilde{c}^1 + \tilde{c}^2} \left(v(t+1) - c_1 v(t) - \tilde{c}^1 p - \tilde{c}^2 g \right)$$

$$v(t+2) = (c_1 - \tilde{s} + \tilde{q}) v(t+1) - c_1 \tilde{q} v(t) - (p - g) \tilde{w} \qquad \textbf{(4)}$$

with

$$\begin{cases} \tilde{s} = \tilde{c}^3 + \tilde{c}^4 \\ \tilde{q} = \dfrac{\tilde{c}^3 + \tilde{c}^4}{\tilde{c}^1 + \tilde{c}^2} \\ \tilde{w} = \tilde{c}^3 - \tilde{c}^1 \tilde{q} \end{cases} \qquad \textbf{(5)}$$

The quantities $\tilde{s}$, $\tilde{q}$, et $\tilde{w}$ are realisations of corresponding random variables *S*, *Q* and *W*. Let's also define $Z = c_1 - S + Q$. The general iterative equation (equation 4) is therefore based on the three random variables *Z*, *Q* and *W*. It can be rewritten as:

$$v(t+1) = Z v(t) - c_1 Q v(t-1) + (p - g) W \qquad \textbf{(6)}$$

The probability densities of *S* and *Q* can quite easily be formulated (cf. Appendix A). The ones of *Z* and *W* are extremely complicated, but we can get an idea of their "shape" by performing simulations. Also, their main features (minimum, maximum, and mean) can be computed (see Table 1). (Further details and figures are in Appendix A). Note that *Z* and *Q* have an infinite support, with a density tending towards zero for high values of the variables. This explains the negative exponential behaviour we have seen in figure 3.

Table 1. Main features of the three random variables in equation 6

	Minimun	Maximum	Mean
Z	$c_1 + 1 - 2c$	∞	$c_1 - c + 2\ln(2)$
Q	0	∞	$2\ln(2)$
W	$-c$	c	0

Let's try now to interpret the three stochastic variables in equation 6. The last one (*W*) can be seen as "noise". The bigger $p - g$, the bigger the effect of noise. Intuitively this makes sense: the more the particle is far from the best position it knows, the more it tends to try big moves. Note, however, that the term (*p*-*g*) *W* has finite maximum and minimum values. Therefore, the search never explores beyond certain limits. This suggests that there might be advantages in replacing this source of noise with a traditional Gaussian noise source.

As $-c_1 Q$ is always negative and as *Q* is parameter free, this term can be seen as a "back force", whose intensity only depends on c_1.

Finally, let us consider *Z*. This variable will often be positive (forth force) but can also take negative values (back force) if $c_1 + 1 - 2c$ is negative. If we set

$$c = \frac{c_1 + 1}{2} \qquad (7)$$

this can never happen and, so, also *Z* has an unambiguous interpretation.

This analysis clarifies the role of each random variable: noise for *W*, back force for *Q*, and forth force for *Z*. However, in order to find some other constraints on the parameters we have to go further in our analysis.

If a particle is a local best

For the best particle we have $p = g$ and we clearly see from equation 6 that there is no noise component. This explains why the particle may converge for some parameter values. If we consider two consecutive time steps (don't forget we are in stagnation, so *p* is constant), the equation can be rewritten in a matrix form as

$$\begin{bmatrix} v(t+1) \\ v(t) \end{bmatrix} = \begin{bmatrix} \tilde{z}^t & -c_1\tilde{q}^t \\ 1 & 0 \end{bmatrix} \begin{bmatrix} v(t) \\ v(t-1) \end{bmatrix} = \tilde{M}^t \begin{bmatrix} v(t) \\ v(t-1) \end{bmatrix} \qquad (8)$$

Then the behaviour of the particle over multiple time steps is determined by $\prod_{k=0}^{t} \tilde{M}^k$. Let us define

$$M = \begin{bmatrix} \hat{Z} & -c_1\hat{Q} \\ 1 & 0 \end{bmatrix} \qquad (9)$$

where $\hat{Z}$ and $\hat{Q}$ are the means of the random variables *Z* and *Q*. The "mean" behaviour of local best particles is given by the powers M , M^2 ,... etc. of this matrix. If λ_1 and λ_2 are the eigenvalues of *M*, we should expect to see convergence if and only if the following condition holds:

$$\max(|\lambda_1|, |\lambda_2|) < 1 \qquad (10)$$

(this is a classical result for second order dynamical systems).

The eigenvalues of *M* are given by

$$\lambda = \frac{\hat{Z}}{2} \pm \sqrt{\Delta}$$

with

$$\Delta = -c_1\hat{Q} + \left(\frac{\hat{Z}}{2}\right)^2$$

Let us consider all possible cases.

Case $\Delta<0$

The discriminant is negative if the following condition holds

$$c \in \left]c_1 + \hat{Q} \pm 2\sqrt{c_1\hat{Q}}\right[\qquad (11)$$

Since we assume that c_1 is not negative, by replacing $\hat{Q}$ by its value, the convergence condition is simply

$$c_1\hat{Q} = c_1 2\ln(2) < 1 \qquad \textbf{(12)}$$

In the field of iterative optimisation, it is often a good idea to work "on the edge of chaos". We can then try the condition $c_1\hat{Q} = 1$. By combining with equation 7 this gives us the following parameter set

$$\begin{cases} c_1 = \dfrac{1}{2\ln(2)} \cong 0{,}72 \\ c = \dfrac{c_1 + 1}{2} \cong 0{,}86 \end{cases} \qquad \textbf{(13)}$$

It is easy to verify that $(c_1 + 1)/2$ always satisfies the condition of equation 11, and that we are therefore indeed in the case $\Delta < 0$.

Case Δ>0

When the discriminant is positive the convergence condition is

$$\left| \frac{c_1 - c + \hat{Q}}{2} \pm \sqrt{\Delta} \right| < 1 \qquad \textbf{(14)}$$

If we combine with the condition on *c* that ensures that the discriminant is positive, we see that *c* must be in the interval $\left[\hat{Q} + c_1 + c_1\hat{Q} - 1, \hat{Q} + c_1 - 2\sqrt{c_1\hat{Q}}\right]$. For typical values of c_1 this interval is very small (it has size 0.028 for $c_1 = 0.5$, and 0.0028 pour $c_1 = 0.8$). So, in what follows we will simply ignore this case.

If a particle is not a local best

During stagnation neither *p* nor *g* are modified. Then, when the particle is not the best one, that is to say when $p \neq g$, it can't converge, because of the "noise" component *W*. This is indeed what one empirically observes, no matter how small c_1 is. Even for $c_1 = 0$ the particle keeps oscillating. When this coefficient is too big, however, the particle can diverge.

More precisely, because the mean of *W* is zero, the conditions that we have found for the mean behaviour of local best particles are still valid. However, in this case satisfying equation 10 does not imply convergence but non-divergent oscillations. So, the conditions in equation 12 (or 13) should lead the PSO to performing a good form of local search.

Generalisation and variants

The three distributions

We can now rewrite equation 6 in a more general form

$$v(t+1) = R_1 v(t) - R_2 v(t-1) + (p - g)R_3 \qquad \textbf{(15)}$$

where the three probability distributions have the following features

R_1 is positive unimodal with infinite support

R_2 is positive unimodal with infinite support

R_3 is symmetrical and unimodal with zero mean

We have seen that we can choose $R_3 = W$, $R_2 = c_1 Q$, and $R_1 = Z$, with the condition $c \leq (c_1 + 1)/2$, but some other choices are clearly interesting. In particular, as R_1 is used as a "forth force" and $-R_2$ as a "back force", we can manipulate *c* and c_1 so that the two forces are of similar intensity. We will see a simple example below.

Starting from PSO-0, we will now progressively add some small modifications which have the objective of providing improvements while keeping the algorithm as fast as possible.

There will be two kinds of modifications suggested by our stagnation analysis:

- we define some new parameter settings and use them irrespective of whether there is no stagnation or not
- we modify the basic PSO so as to detect stagnation and to then use special strategies to deal with it

Theoretically, stagnation occurs as soon as *p* and *g* don't change for two time steps. However, it does not seem appropriate to do something special if there is no improvement for a few time steps as long as the swarm finds improvements often enough. So, first we must determine how long the stagnation should be before one should modify the PSO search strategy.

Defining a stagnation threshold

It seems reasonable for the PSO to try something different if:

- the local best particles have had enough time to be informed (either directly or indirectly) by a possible better particle
- nevertheless, they have not improved their best known position

Note that this criterion is not applicable to the global best, but in order to keep the algorithm simple, we neglect this particular case.

The probability that a given particle *A* is still not informed that there exists a better particle *G* in the swarm after *t* time steps is given by

$$p_t(G\neg\rightarrow A)=\left(1-\frac{1}{S}\right)^{Kt} \qquad \textbf{(16)}$$

where *S* is the swarm size and *K* the number of information links. These are drawn at random at each time step, since during stagnation there is no improvement. (For detailed formulas characterising the connectivity in the PSO over time, see [1, 4]) . If we want this probability to be smaller than a threshold ε, the number of time steps to wait before we react to stagnation should be

$$t_{stag}=\frac{\ln(\varepsilon)}{K\ln\left(1-\frac{1}{S}\right)} \qquad \textbf{(17)}$$

For example, for $S=30$, $K=3$ and $\varepsilon=0.0001$, we should react to stagnation if the local best particles didn't improve their position for at least 91 time steps. The additional parameter ε must, unfortunately, be set empirically: this is the price to pay to be sure the algorithm remains fast.

Test bed

Based on the previous analysis we can now define some new variants of PSO and we can test them on some functions in order to compare them. The five functions used here are very classical (see table 2). Their minimum is always zero. The dimensions (30), search spaces and required accuracies are chosen so that our "reference" algorithm, PSO-0, has a success rate which is neither zero nor 100% over 100 runs with at most 40000 fitness evaluations. This success rate, for each function, can therefore be chosen as a comparison criterion for algorithms. For a quick comparison the mean success rate is enough. In case of a tie (or almost) between two algorithms, we will prefer the variant that gives the smaller standard deviation, since this is an indication of a superior robustness.

Table 2. Success rates of PSO-0 over five test functions

Function	Search space	Accuracy	Success rate with PSO-0
Parabola (Sphere)	$[-20,20]^{30}$	10^{-9}	32%
Griewank	$[-300,300]^{30}$	0.0001	44%
Rosenbrock	$[-10,10]^{30}$	25	21%
Rastrigin	$[-5{,}12,5{,}12]^{30}$	35	12%
Ackley	$[-32,32]^{30}$	0.0002	5%

Five variants

We will call our new variants of PSO 3PD-PSO, which stands for "Three Probability Distributions Particle Swarm Optimisation". The following variants are just the most obvious ones suggested by stagnation analysis, others could be defined. In particular we don't modify at all the distribution of R_2, which is still equal to $c_1 Q$, nor the distribution of R_1, which is equal to *Z*, except possibly for translations. We describe the five variants in the next subsections while Table 3 summarizes their performance on our five test problems.

Table 3. Results with five new variants of PSO (when the success rate is 100%, the mean number of fitness evaluations is about 18000).

Function	3PD-PSO-0	3PD-PSO-0'	3PD-PSO-1	**3PD-PSO-2**	3PD-PSO-3
Parabola (Sphere)	100%	100%	100%	100%	100%
Griewank	35%	49%	52%	56%	53%
Rosenbrock	74%	74%	65%	73%	72%
Rastrigin	58%	35%	24%	37%	33%
Ackley	5%	66%	86%	61%	64%
Mean	*54%*	*65%*	*65%*	*65%*	*64%*
Std. dev.	0.33	0.22	0.27	0.21	0.22

3PD-PSO-0

This variant is simply PSO-0 but with the parameters values defined in equation 13. The motion equation is then directly derived from equation 1 and can be written for dimension *d* as

$$v_d \leftarrow 0.72 v_d + 0.86 alea(0,1)(p_d - x_d) + 0.86 alea(0,1)(g_d - x_d) \qquad \textbf{(18)}$$

Other parameters are the same (30 particles, 3 neighbours drawn at random for each particle). As one can see from table 3, the effectiveness w.r.t. PSO-0 is significantly improved.

3PD-PSO-0'

Here we keep $c_1 = 1/2\ln(2) \cong 0{,}72$, but we choose *c* so that the mean of *Z* (for the "forth force") is equal to the mean of $c_1 Q$ (for the "back force"). This is achieved for $c = 1/2\ln(2) + 2\ln(2) - 1 \cong 1.108$. The velocity is then updated according to the following equation

$$v_d \leftarrow 0.72 v_d + 1.108 alea(0,1)(p_d - x_d) + 01.108 alea(0,1)(g_d - x_d) \qquad \textbf{(19)}$$

This further improves the average effectiveness of the algorithm and reduces the standard deviation over its performance.

3PD-PSO-1

For $c_1 = 1/2\ln(2) \cong 0{,}72$, stagnation analysis suggested a possible c ($c_{\max}$ in equation 1) equal to 0.86. An idea would be to define another plausible value of c and to choose at random between the two values at each move. For example, the value 1.48 used in PSO-0, which comes from a deterministic convergence analysis (cf. Appendix A), seems an appropriate alternative. An even more interesting possibility, which we test in 3PD-PSO-1, is to try the variation interval

$$[a,c] = \left[\frac{c_1+1}{2}, \frac{(c_1+1)^2}{2}\right] \qquad \textbf{(20)}$$

So, in practice, in 3PD-PSO-1, to compute the new velocity of a particle, $c_{\max}$ is first chosen at random

$$c_{\max} = alea(a,c)$$

and then for each dimension d equation 1 is applied. Formally, it is equivalent to the following

$$v_d \leftarrow c_1 v_d + \tilde{b}^1(p_d - x_d) + \tilde{b}^2(g_d - x_d) \qquad \textbf{(21)}$$

where $\tilde{b}^1$ et $\tilde{b}^2$ are two realisations of a random variable B, defined by

$$B = (a + (c-a)U_1)U_2 \qquad \textbf{(22)}$$

U_1 and U_2 being two uniform random variables on $[0,1]$. The distribution of B is obviously non-uniform, as we can see in figure 8. Intuitively this seems a more appropriate distribution, since big moves are now less probable. The performance improvement obtained by 3PD-PSO-1 is comparable to the one provided by 3PD-PSO-0, although the standard deviation for 3PD-PSO-1 is slightly higher.

3PD-PSO-2

We have seen that in case of stagnation the local best particle has a special behaviour that rapidly leads to wasting fitness evaluations. This can easily be corrected.

Let G be the local best particle. In order to compute its new velocity, we consider all best known positions in the swarm (i.e., all p's), and keep the ones that are better than the best position found by G (i.e., g). Then we choose one of the remaining p's at random, let us call it g_2, and in the movement equation for G we replace g by g_2. So, except for the swarm best, the motion equations of local best particles are now similar to the ones of non-best particles.

We add this strategy to 3PD-PSO-0', obtaining a new PSO, thhat we call 3PD-PSO-2, the effectiveness of which is more or less the same as that of 3PD-PSO-0' (the standard deviation is a bit smaller, but this difference may not be statistically significant).

3PD-PSO-3

In Appendix A, we show that the "noise" distribution, the pdf of W, is vaguely similar to a truncated Gaussian curve. Therefore, in case of stagnation, it is reasonable to ask what would happen if we replaced it with a Gaussian distribution. To avoid the introduction of a new parameter in the system, we choose a Gaussian distribution with the same standard deviation as W, i.e. $\sigma = c\sqrt{(11-14\ln(2))/12}$ (which is approximately $c/3$).

The resulting algorithm is a bit more complicated than the other versions. We need to memorize velocities at time t and t-1, and also the random values that are used. Then, when stagnation is detected stagnation, a particle's velocity is directly computed by the following rewrite of equation 6:

$$v_d(t+1) = (c_1 - \tilde{s} + \tilde{q})v_d(t) - c_1\tilde{q}v(t-1) - (p-g)alea\_normal(0,\sigma) \qquad \textbf{(23)}$$

So, clearly the PSO's search process is not really slowed down. We applied this method to 3PD-PSO-0', obtaining an algorithm that we call 3PD-PSO-3. Performance, however, were not significantly improved (the result is the same when equation 23 is used within 3PD-PSO-2).

Conclusions

Variants 3PD-PSO-0', 3PD-PSO-1 and 3PD-PSO-2 are more or less equivalent, with, perhaps, a slight superiority of 3PD-PSO-2 which seems a bit more robust. Let's summarize their differences w.r.t. PSO-0:

- the $c_{\max}$ coefficient is chosen so that in case of stagnation the "forth force" has the same mean as the "back force"
- for the local best the set of informants is the whole swarm.

Other variants based on stagnation analysis are of course possible. In particular we could manipulate the "forth force" and "back force" components more radically, for example by using Beta distributions. Also, more tests are needed to choose discriminate between the variants we have proposed in this paper.

Appendix A

Mathematical developments below have been simplified. In particular the details of some complex probability densities are not given, for they are not really more helpful than their graphical representations for the purpose of understanding this study.

Relationship between coefficients resulting from a deterministic analysis

In [3] several constriction methods are defined, that guarantee the non-divergence of the swarm. For historical reasons, the most commonly used gives the two coefficient c_1 and $c_{\max}$ as a function of a single parameter, φ:

$$\begin{cases} c_1 = \dfrac{2}{\varphi - 2 + \sqrt{\varphi^2 - 4\varphi}} \\ c_{\max} = c_1 \dfrac{\varphi}{2} \end{cases}$$

However this equation can easily be rewritten as a relationship between them

$$c_{\max} = \frac{(c_1 + 1)^2}{2}$$

S study

Let X_1 and X_2 be two uniform random variables in $[0, c]$. The density of $S = X_1 + X_2$ is given by

$$s(u) = \int_{-\infty}^{+\infty} g(x, u - x)dx$$

with $g(x, u - x) = (1/c)(1/c)$ if *x* and $u - x$ are both in $[0, c]$, and 0 otherwise. One can distinguish two cases (we assume *c* is not 0):

$$\begin{cases} 0 < u < c \Rightarrow s(u) = s_1(u) = \dfrac{1}{c^2} \displaystyle\int_0^u dx = \frac{u}{c^2} \\ c \le u \le 2c \Rightarrow s(u) = s_2(u) = \dfrac{1}{c^2} \displaystyle\int_{u-c}^{c} dx = \frac{2c - u}{c^2} \end{cases}$$

The support of *S* is $[0,2c]$ and the mean is the sum of the means of X_1 and X_2, i.e. $\hat{S} = c$. Actually *S* can be rewritten $S = cU$ where *U* is a uniform random variable in $[0,1]$.

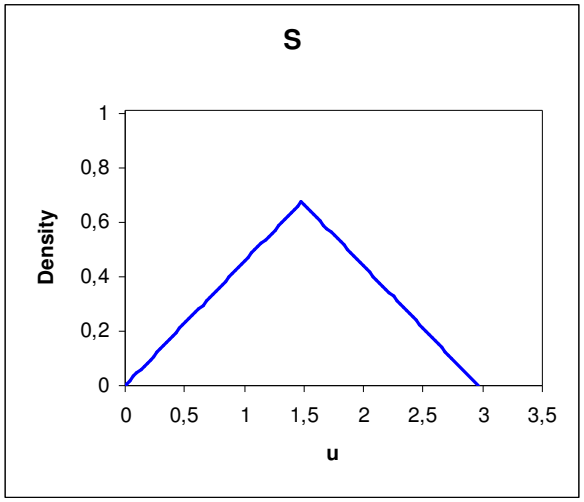

Figure 4. S density for c=1.48

Q study

Let S_1 and S_2 be two random variables like *S*. Their quotient is $Q = S_1/S_2$. The corresponding probability density is given by

$$q(u) = \int_{-\infty}^{+\infty} |x| s(ux) s(x) dx$$

To more easily compute this definite integral one can consider four cases.

$0 < u \le 1/2$

$$\begin{cases} q(u) = \int_0^c x s_1(ux) s_1(x) dx + \int_c^{2c} x s_1(ux) s_2(x) dx \\ \quad = \frac{7}{6} u \end{cases}$$

$1/2 < u \le 1$

$$\begin{cases} q(u) = \int_0^c x s_1(ux) s_1(x) dx + \int_c^{c/u} x s_1(ux) s_2(x) dx + \int_{c/u}^{2c} x s_2(ux) s_2(x) dx \\ \quad = \frac{1}{6u^3} - \frac{2}{3u^2} + \frac{8}{3} - \frac{3}{2} u \end{cases}$$

$1 < u \le 2$

$$\begin{cases} q(u) = \int_0^{c/u} x s_1(ux) s_1(x) dx + \int_{c/u}^{c} x s_2(ux) s_1(x) dx + \int_c^{2c/u} x s_2(ux) s_2(x) dx \\ \quad = -\frac{3}{2u^3} + \frac{9}{3u^2} - \frac{2}{3} - \frac{u}{6} \end{cases}$$

$2 < u$

$$\begin{cases} q(u) = \int_0^{c/u} x s_1(ux) s_1(x) dx + \int_{c/u}^{2c/u} x s_2(ux) s_1(x) dx \\ \quad = \dfrac{7}{6u^3} \end{cases}$$

Note that Q is parameter free. The mean can be computed by noting that $Q = S_1(1/S_2)$, as a product of two independent random variables. This mean is then the one of $1/(U_1 + U_2)$. The density of this last random variable is

$$\begin{cases} \dfrac{2}{u^2} - \dfrac{1}{u^3} \text{ if } u \in [1/2,1] \\ \dfrac{1}{u^3} \text{ if } u > 1 \end{cases}$$

Its mean is then

$$\int_{1/2}^{1} \left(\frac{2}{u} - \frac{1}{u^2} \right) du + \int_{1}^{+\infty} \frac{1}{u^2} du = 2\ln(2)$$

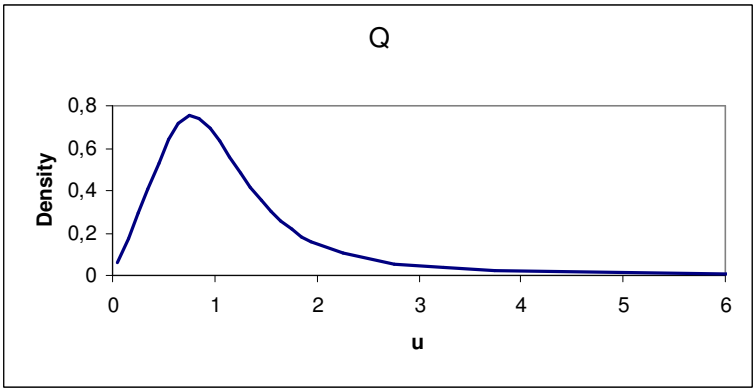

Figure 5. Q density

W study

The random variable W can be rewritten as

$$\begin{cases} W = X_3 - X_1 \dfrac{X_3 + X_4}{X_1 + X_2} \\ \quad = \dfrac{X_3 X_2 - X_1 X_4}{X_1 + X_2} \end{cases}$$

So we immediately see that it is symmetrical and that its mean is zero. In order to compute the maximum value, one can note that it is reached when X_1 is null and that in this case W is equal to X_3. This maximum is then equal to c. The formulas that give the density of W are quite complicated, since they make use of the order 2

Jonquières function (dilogarithm) $\mathrm{Li}_2(y) = \sum_{k=1}^{\infty} \frac{y^k}{k^2}$. It is enough here the visualize it and to compute the standard deviation σ. By using the fundamental theorem about the variance of a combination of random variables, and noting that each X_i is equal to cU_i, where density of U_i is 1 on $[0,1]$, we can write

$$\sigma^2 = \int_{-\infty}^{+\infty} t.density(W^2)dt =$$

$$= \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} W^2 density(X_1X_2X_3X_4)dx_1dx_2dx_3dx_4$$

$$= c^2 \int_0^1\int_0^1\int_0^1\int_0^1 \left(\frac{u_3u_2 - u_1u_4}{u_1+u_2}\right)^2 du_1du_2du_3du_4$$

$$= c^2 \frac{11-14\ln(2)}{12}$$

that is to say $\sigma = c\sqrt{(11-14\ln(2))/12}$, which almost equal to $c/3$. Thanks to this information, we can define the best Gaussian approximation of *W*.

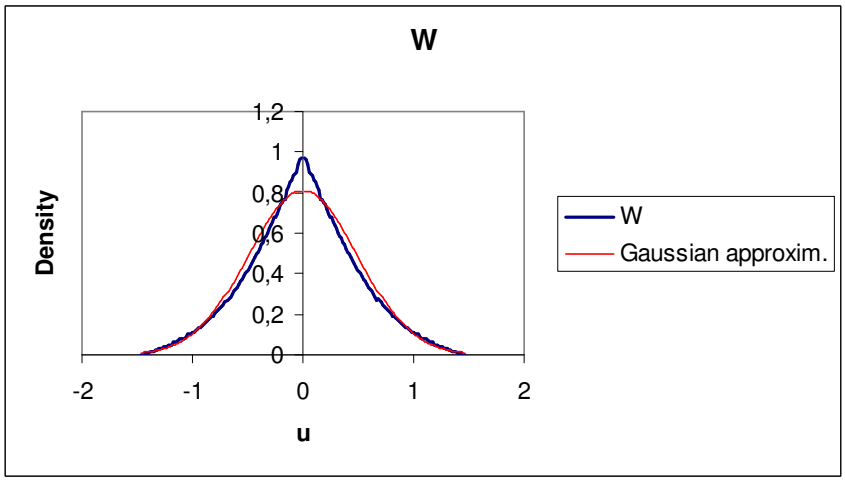

Figure 6. W density for c=1.48, and best Gaussian approximation

Z study

The random variable $Z = c_1 - S + Q$ can be rewritten

$$Z = c_1 - (U_3 + U_4)\left(c - \frac{1}{U_1+U_2}\right)$$

So we can see that the support is $[c_1 - c + 1, +\infty[$. We have seen that the mean of $1/(U_1+U_2)$ is $2\ln(2)$. Therefore, the mean of *Z* is $\hat{Z} = c_1 - c + 2\ln(2)$. Note that $\hat{Z} = c_1 - \hat{S} + \hat{Q}$, but this formula couldn't have been directly written, for random variables *S* and *Q* are not independent. Here, again, we just visualize the density curve, without giving complicated formulas.

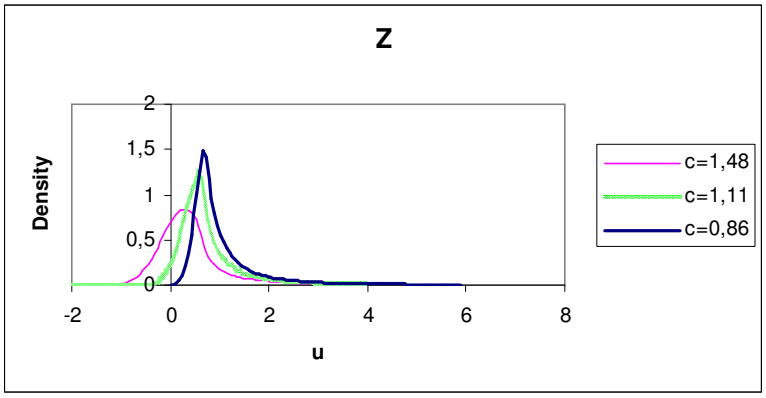

Figure 7. Some Z densities for c_1=0.72

B study

The random variable B can be rewritten $B=(a+(c-a)U_1)U_2$. Its support is obviously $[a,c]$. B can be seen as the product of the two independent random variables $X_1=(a+(c-a)U_1)$ and U_2. Its mean is the product of the means of X_1 and U_2, respectively $(a+c)/2$ and $1/2$. The mean value is then $\hat{B}=(a+c)/4$.

The probability density is given by

$$\begin{cases} b(u)=\dfrac{1}{c-a}\displaystyle\int_a^c \frac{1}{x}dx=\frac{1}{c-a}\ln\left(\frac{c}{a}\right) \text{ if } u\in[0,a] \\ \quad =\dfrac{1}{c-a}\displaystyle\int_c^u \frac{1}{x}dx=\frac{1}{c-a}\ln\left(\frac{c}{u}\right) \text{ if } u\in[a,c] \end{cases}$$

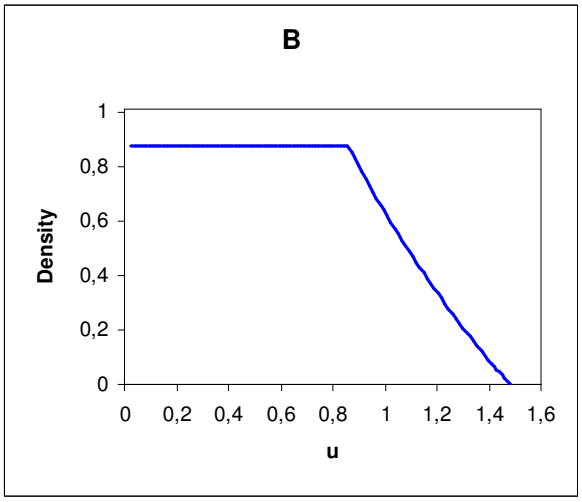

Figure 8. B density for c=1.48

Acknowledgements

The author would like to thank Jean-Pierre Levrel (newsgroup fr.sci.maths) and Ray Koopman (newsgroup alt.sci.math.probability) for useful comments.

References

[1] M. Clerc, *L'optimisation par essaims particulaires. Versions paramétriques et adaptatives*: Hermès Science, 2005.

[2] G. C. Onwubolu, "TRIBES application to the flow shop scheduling problem," in *New Optimization Techniques in Engineering*. Heidelberg, Germany: Springer, 2004, pp. 517-536.

[3] M. Clerc and J. Kennedy, "The Particle Swarm-Explosion, Stability, and Convergence in a Multidimensional Complex Space," *IEEE Transactions on Evolutionary Computation*, vol. 6, pp. 58-73, 2002.

[4] M. Clerc, *Particle Swarm Optimization,* ISTE, 2006.