SOME IDEAS ABOUT PARTICLE SWARM OPTIMISATION

Maurice.Clerc@WriteMe.com
Version 2006-11-06

1. Introduction

Here are just a few ideas about PSO. Some of them are so crazy that they might work. Some others seem reasonable but probably do not work very well.

2. PSO equivalence

2.1. **Definition.** Two objective functions f and g on the same space are PSO equivalent if for all x and all y we have $f(x) < f(y) \Leftrightarrow g(x) < g(y)$. It is almost obvious that this relation indeed is an equivalence relation. For a given PSO version, we can note it $f \sim g$.

Actually this definition is interesting only for PSO versions whose movement equations do not use values like $f(x)$ but only order relations like $f(x) < f(y)$. This is the case of all "classical versions", and the case of FIPS, the Fully Informed Particle Swarm we study below. We do suppose here we are working with such a PSO.

2.2. **Properties.**

2.2.1. *Equalities.* We immediately have $f(x) = f(y) \Leftrightarrow g(x) = g(y)$.

2.2.2. *Minima.* If $f \sim g$ then the two functions have the same local minima. This is quite straightforward. A local minimum x^* is so that for all positions "around" it, the function value is higher. More precisely, in dimension 1

$$\exists \delta, \forall \varepsilon < \delta, f(x^*) < f(x^* + \varepsilon)$$

Therefore we have also $g(x^*) < g(x^* + \varepsilon)$ under the same condition. So any local minimum of f is a local minimum of g. And vice versa.

2.2.3. *Trajectories.* An important result is that for PSO equivalent functions the trajectories of the particles are exactly the same. It is quite simple to see why. Let us consider the canonical following pseudo-code that describes how a particle is moving:

1) at time t consider some known positions $p_1, \ldots, p_N$. Optionnaly select just some of them (for example the best one(s)), thanks to relations like $f(p_j) \leq f(p_i)$.

2) combine the current velocity, the current position $x(t)$, and the chosen p_i to define the new position $x(t+1)$

3) if this new position is better than a given p_k(typically what it is called the best previous position of the particle) , i.e. if $f(x(t+1)) < f(p_k)$, then replace p_k by $x(t+1)$

As we can see none of these steps is modified for an equivalent function g. In step 2 the chosen p_i are the sames, and in step 3 the condition is the same. It means that we can replace a fitness function by an equivalent one easier to compute. Of course the fitness *values* are not the same, and if the stop criterion is something like $f(x) < \varepsilon$ the process won't stop at the same time for g. In particular if around the optimum we have $g(x) < f(x)$it stops too early. So from time to time it is still necessary to compute the "real" fitness value $f(x)$ just to see if the search has to be continued. However if the stop criterion is a maximum "search effort", like the number of fitness evaluation, we don't even have to do that.

2.3. **Examples.** All functions like

$$f(x) = \sum (x_d - \delta_d)^{2k}$$

are PSO-equivalent for any $k \in N$. Also for any function with a finite number of discontinuity points there are an infinity of linear piece-wise approximations that are PSO-equivalent to it.

Also all functions like $a + bf$ with $b > 0$ are equivalent to f.

Let H be the search space, and $H' = f(H) \subset \mathbb{R}$. Then for any function h strictly increasing on R' we have $h \circ f \sim f$. Three such equivalent functions are shown on figure 2.1.

2.4. **Geometrical interpretation.** *** TO COMPLETE ***

Two PSO equivalent functions have the same "level lines" representation.

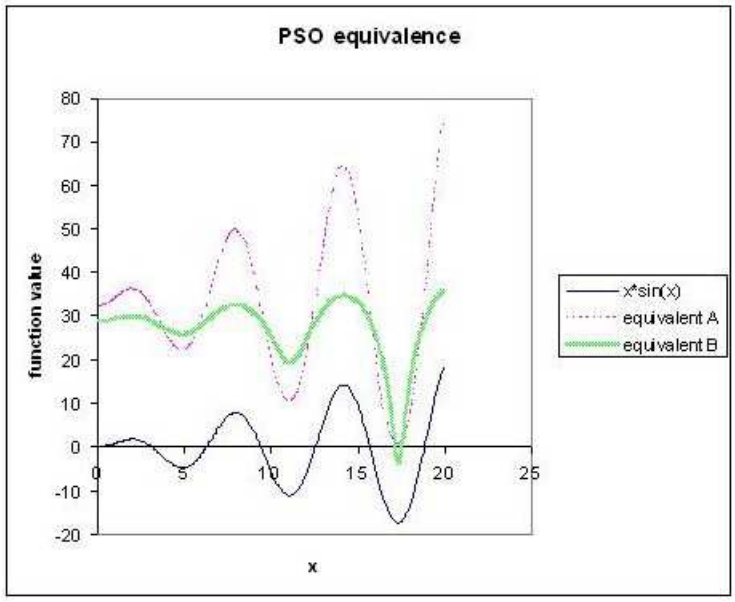


FIGURE 2.1. Three PSO equivalent functions

Pair	$\alpha_{1,1}$	$\alpha_{1,2}$	$\alpha_{2,1}$	$\alpha_{2,2}$	Equivalence
1	0.8	1	10	10	0.995
2	0.8	1	50	10	0.991
3	1	0.5	10	2	1.000

TABLE 1. Fuzzy PSO equivalence of combinations of two gaussian kernels

2.5. **Fuzzy equivalence.** PSO equivalence is useful for performance prediction (as the moves are the same for two equivalent functions, it is easy to compute the performance one one of them knowing it for the other). However it is too restrictive. So a better tool is fuzzy equivalence, which assign a truth value to the assertion "f is equivalent to g".

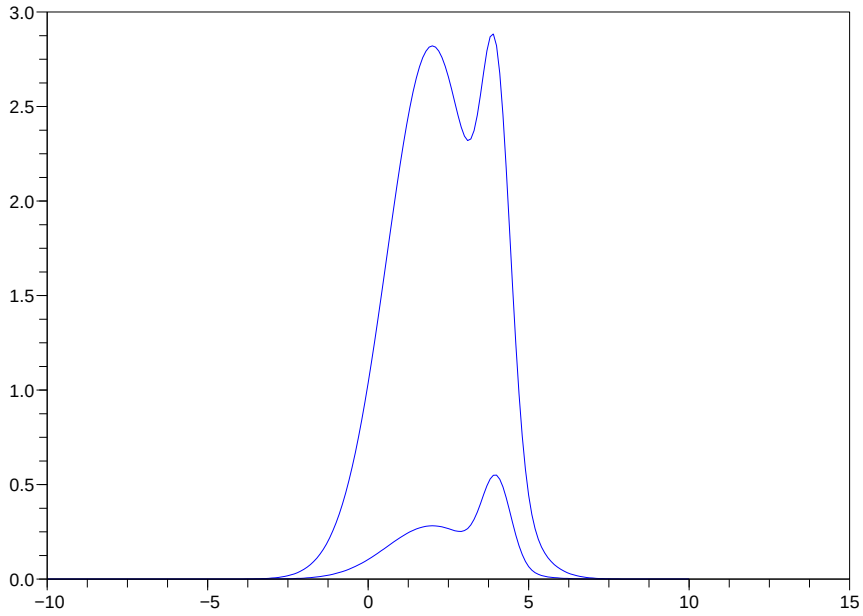
Let us consider first a finite search space $\Omega$with N elements. This can be noted $|\Omega| = N$. For each pair of elements $\{x, y\}$we define a function δ

$$\begin{cases} \delta(x,y) & = & 1 \text{ if } f(x) < f(y) \text{ and } g(x) < g(y) \\ & = & 0 \text{ else} \end{cases}$$

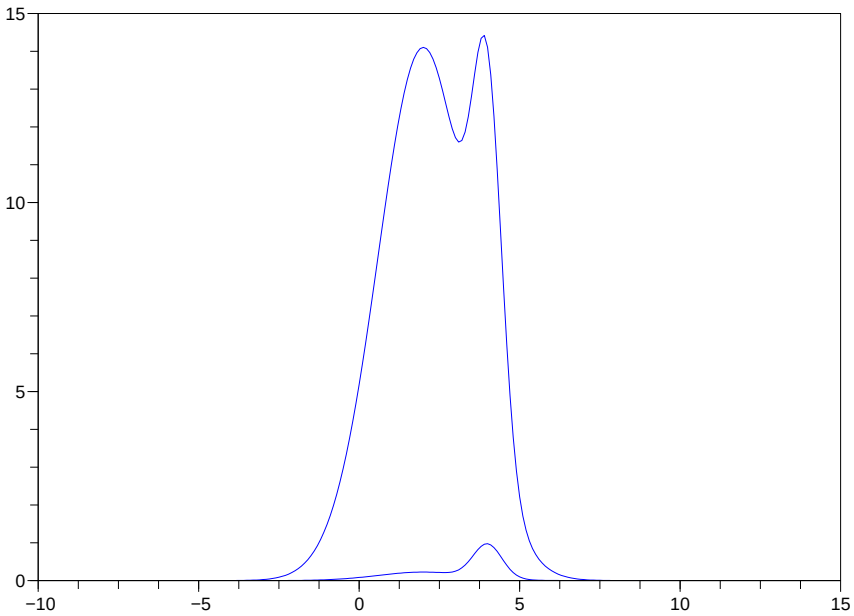
The truth value of the fuzzy PSO equivalence is then defined by

$$\frac{1}{|\Omega|^2} \sum_{\Omega^2} \delta(x,y)$$

2.5.1. *Example.* As any "reasonable" function can be approximated by a linear combination of Gaussian kernels, it is interesting to compare how much two such approximations are equivalent. Kernel 1 is defined by $k_1 = N\left(2, \sqrt{2}\right)$, and kernel 2 by $k_2 = N\left(4, \sqrt{0.2}\right)$. Amongst all functions defined by $f_i = \alpha_{i,1}k_1 + \alpha_{i,2}k_2$we consider the bimodal ones on $x_1$and x_2 with $x_1 < x_2$ and $f_i(x_1) < f(x_2)$. On what extent are they PSO equivalent? As we can see on Table 1 and on figure 2.2, even some apparently quite different functions are equivalent on a great extent.



(a) Pair 2, equivalence 0.991



(b) Pair 3. Equivalence 1.000

FIGURE 2.2. Two examples of fuzzy equivalent functions

2.6. **Functional PSO equivalence.** *** TO DO ***

Two functions f and g are said *functionally PSO equivalent* if there exist a function h with some special properties so that $h \circ g \sim f$. We have seen the case $h = identity$, but of course we are now studying something more general.

3. Gaussian-equivalent landscape

Let f be the fitness function defined on the search space H. We are now trying to define an approximation $\widetilde{f}$ of f as a combination of semi-gaussians kernels, so that the distance between f and $\widetilde{f}$is as small as possible. The distance may be

$$\sqrt{\int_H \left(f(x) - \widetilde{f}(x)\right)^2}$$

For each optimum $o_i = (o_{i,1}, ..., o_{i,d}, ..., o_{i,D})$ and for each dimension d we define first a monodimensional function g_d by

$$\begin{cases} g_{i,d}(x_d) &= N\left(o_{i,d}, \sigma^+_{i,d}\right)(x_d) \text{ if } x_d \geq o_{i,d} \\ &= N\left(o_{i,d}, \sigma^-_{i,d}\right)(x_d) \text{ else} \end{cases}$$

where $N(a,b)$is the classical Gaussian (Normal) function with a mean equal to a, and a standard deviation equal to b. If $x = (x_1, ..., x_D)$ we define then the near-gaussian D-approximation related to this optimum by

$$g_i(x) = f(o_i) \prod_{d=1}^{D} g_{i,d}(x_d)$$

for a maximum and by

$$g_i(x) = f(o_i)\left(1 - \prod_{d=1}^{D} g_{i,d}(x_d)\right)$$

for a minimum.

We define then an approximation g of f by

$$g(x) = g_i(x)$$

in which i is then rank of the optimum that is the nearest one of x. So g is depending on the parameters $\sigma^+_{i,d}$ and $\sigma^-_{i,d}$, and as said, we finally define the Gaussian-equivalent as the g which is the best approximation.

4. Equivalent PSOs

*** TO DO ***

This is almost the dual notion of PSO equivalence. Two PSO variants are said equivalent for a given function f , or *f-equivalent*, if they have the same behaviour when looking for the optimum. Now, of course, the point is to define what "same behaviour" means. We may interested only on the best fitness value vs number of function evaluations, or we may want something more precise, like similar trajectories.

5. Fully Informed Particle Swarm (FIPS)

5.1. **Description and equations.** The first FIPS version a probably been defined in [1]. It has been studied in [2]. Let us consider a swarm of S particles in search space of dimension D. As usually in PSO we define the following mathematical objects:

$x_i(t) = (x_{i,1}(t), \ldots, x_{i,d}(t), \ldots, x_{i,D}(t))$ the position of particle i at time t

$v_i(t) = (v_{i,1}(t), \ldots, v_{i,d}(t), \ldots, v_{i,D}(t))$ the velocity of particle i at time t

$p_i(t) = (p_{i,1}(t), \ldots, p_{i,d}(t), \ldots, p_{i,D}(t))$ the best previous position of particle i at time t

N_i the list of particles that inform particle i. In practice this is a list of integer numbers of $[1, S]$. Note that we suppose here it is not depending on t. The size of this set (the number of elements is noted by $|N_i|$. The jth element of this list is noted $N_{i,j}$.

w and c, two real numbers. Let us call them here first and second confidence coefficients.

At last $\widetilde{u}$ is a realisation of a given random distribution R. We do suppose here that this distribution is the uniform one on $[0, u]$.

Now the iterative movement equations are, for each particle i and for each dimension d:

$$\begin{cases} v_{i,d}(t+1) &= wv_{i,d}(t) + \dfrac{1}{|N_i|}\displaystyle\sum_{j=1}^{|N_i|} \widetilde{c}\left(p_{N_{i,j},d}(t) - x_{i,d}(t)\right) \\ x_{i,d}(t+1) &= v_{i,d}(t+1) + x_{i,d}(t) \end{cases} \tag{5.1}$$

Compared to classical PSO here the particle is not looking for its best informant (neighbour), but take them all into account. Note that the "weight" of each contribution is the same (i.e. $1/|N_i|$). We may think at something like

"the better the neighbour the bigger the weight", but if the weight were depending on, say, $f\left(p_{N_{i,j}}\right)$, it would be not consistent with the concept of PSO equivalence. Moreover it seems such a weighted variant is not really better [2].

5.2. **Deterministic form analysis.** For the mathematical analysis we need first to consider a simpler version:

- deterministic, i.e. $\widetilde{c} = c$
- global, i.e. $N_i = (1, \ldots, S)$. Each particle "sees" all the others.

Then, if we define

$$P_d(t) = \frac{c}{S} \sum_{j=1}^{S} p_{j,d}(t) \tag{5.2}$$

the system 5.1 becomes

$$\left\{ \begin{array}{rcl} v_{i,d}(t+1) & = & w v_{i,d}(t) + P_d(t) - c x_{i,d}(t) \\ x_{i,d}(t+1) & = & w v_{i,d}(t) + P_d(t) + (1-c) x_{i,d}(t) \end{array} \right. \tag{5.3}$$

Note: in what follows we do consider that all particles are moving on the same time (parallel mode), and not each at a time in a loop, as in most of PSO versions. It is worth to note, though, that this mode is a bit less effective that the sequential one [3].

5.2.1. *"Differential" explicit representation.* At a given time t we consider two particles, i and k, the difference of their velocities, and the difference of their positions. With quite understandable notations, and by using system 5.3 at times t and $t-1$, we have then

$$\left[\begin{array}{c} \Delta_{i,k} v_d \\ \Delta_{i,k} x_d \end{array} \right]_t = \left[\begin{array}{cc} w & -c \\ w & 1-c \end{array} \right] \left[\begin{array}{c} \Delta_{i,k} v_d \\ \Delta_{i,k} x_d \end{array} \right]_{t-1} \tag{5.4}$$

So we finally have

$$\left[\begin{array}{c} \Delta_{i,k} v_d \\ \Delta_{i,k} x_d \end{array} \right]_t = \left[\begin{array}{cc} w & -c \\ w & 1-c \end{array} \right]^t \left[\begin{array}{c} \Delta_{i,k} v_d \\ \Delta_{i,k} x_d \end{array} \right]_0 \tag{5.5}$$

Let us define

$$M = \left[\begin{array}{cc} w & -c \\ w & 1-c \end{array} \right] \tag{5.6}$$

It is not that difficult to exactly compute M^t and some convergence conditions, but here we just need to say that its general form is

$$\left[\begin{array}{cc} m_{t,1,1} & m_{t,1,2} \\ m_{t,2,1} & m_{t,2,2} \end{array} \right]$$

We can simply initialise the velocities to zero. After all, if the positions are initialised at random, after the first time step we do have anyway random velocities. As a result, we can derive a very interesting relation

$$\Delta_{i,k} x_d(t) = m_{t,2,2} \Delta_{i,k} x_d(0) \tag{5.7}$$

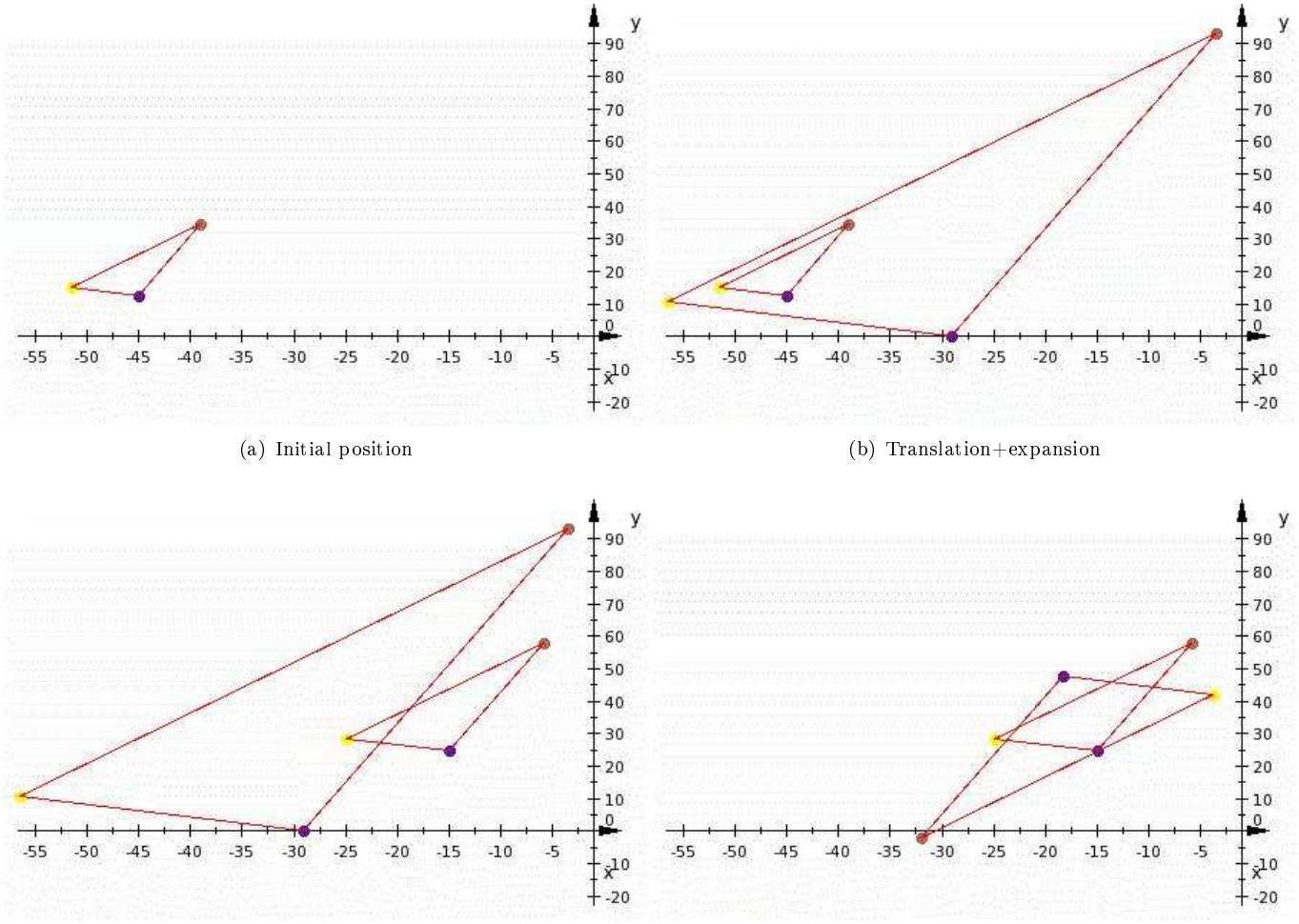
Let us see now what it means.

5.2.2. *The swarm as a polyhedron.* According to equation 5.7, if we compute the Euclidean distance between particles i and k, we find

$$\|x_i - x_k\|_t = m_{t,2,2} \|x_i - x_k\|_0$$

Note that the coefficient $m_{t,2,2}$ is not depending on i nor on k. In other words, at each time step all distances between particles are multiplied by the same coefficients. If we see the particles positions as vertices of a polyhedron only very few geometrical transformations and their combinations are therefore possible:

- scaling (expansion, contraction)
- translation
- reflection



(a) Initial position

(b) Translation+expansion

(c) Translation+contraction

(d) Translation+contraction+reflection

FIGURE 5.1. Deterministic global FIPS on Sphere/Parabola 2D. First four steps with a swarm of three particles

Note that as equation 5.7 is valid for each dimension independently, even rotation is not admissible: it is easy to see that for any pair (i,k)all vectors $(x_i - x_k)_t$ are parallel to $(x_i - x_k)_0$, for any t value. In figure 5.1 we can see the four first steps of a three particle swarm looking for the minimum of the two-dimensional Sphere/Parabola function.

5.2.3. *Some maths for fun.* The eigen values of the matrix M (equation 5.6) are given by

$$\left\{ \begin{array}{rcl} \Delta & = & \frac{(w-c+1)^2}{4} - 4w \\ \lambda & = & \frac{w-c+1}{2} \pm \frac{\sqrt{\Delta}}{2} \end{array} \right.$$

For usual w and c values the condition $\Delta < 0$ holds, and we then have $|\lambda| = \sqrt{w}$. So the two eigenvalues are

$$\left\{ \begin{array}{rcl} \lambda_1 & = & \sqrt{w}\left(\cos(\theta) - i\sin(\theta)\right) \\ \lambda_2 & = & \sqrt{w}\left(\cos(\theta) + i\sin(\theta)\right) \end{array} \right.$$

with $\cos(\theta) = \frac{w-c+1}{2\sqrt{w}}$, and $\sin(\theta) = \frac{\sqrt{-\Delta}}{2\sqrt{w}}$. On the other hand there exists a matrix A so that $M = A\Lambda A^{-1}$, with $\Lambda = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$. So we have $M^t = A\Lambda^t A^{-1}$. After some boring calculuses, it means that if the initial velocity is null, the value of the term $m_{t,2,2}$ is given by

$$m_{t,2,2} = \sqrt{w}^t \left(\frac{1-c-w}{\sqrt{-\Delta}} \sin(t\theta) + 2\cos(t\theta) \right)$$

Its decreasing oscillation is shown on figure .

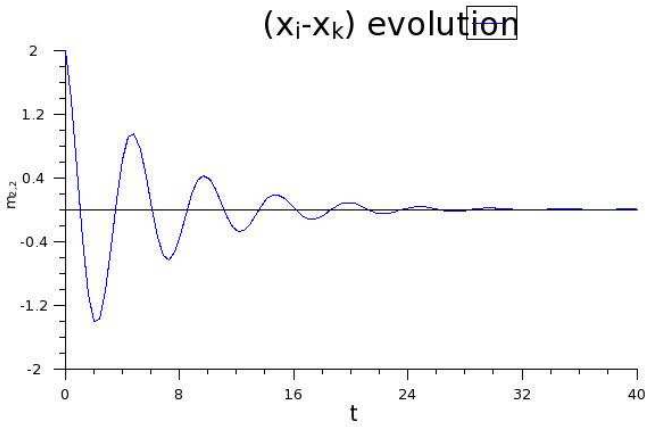


FIGURE 5.2. FIPS (deterministic). A typical evolution of the difference between two coordinates of two particles on a given dimension

5.2.4. *FIPS and Simplex Method.* If you know a bit about non-linear optimisation by Simplex Method, what has been said above should sounds familiar: the admissible operations (expansion, contraction, reflection) are almost the same. However FIPS is also using translation, and there is no need of explicit reevaluations rules. Moreover, in the complete FIPS form some parameters are random, as the Simplex Method is deterministic (except the initialisation).

5.3. **Random form analysis.** ** TO DO ** Replace "sum of random variables" by "Gaussian random variable" (see TCL)

6. Central force PSO

6.1. **Description and equations.** A more general PSO version can be defined by using a gravitational analogy. The idea is not completely new (see for example [4]) , but can be generalised. The space is supposed to be "viscous". It means during each time step Δt the velocity is multiplied by a positive coefficient w smaller than 1. Let us suppose all $p_i(t)$ are "bodies" of mass 1 inside the search space. The particle is attracted by these bodies, according to a central force law. Let us say its mass is also 1. So the equations of movement are something like

$$\left\{ \begin{array}{lcl} v_{i,d}(t+1) & = & w v_{i,d}(t) + \gamma \Delta t \sum_{j=1}^{|N_i|} \frac{p_{N_{i,j},d}(t) - x_{i,d}(t)}{\left\| p_{N_{i,j}}(t) - x_i(t) \right\|^{\alpha}} \\ x_{i,d}(t+1) & = & v_{i,d}(t+1)\,\Delta t + x_{i,d}(t) \end{array} \right. \tag{6.1}$$

When $\Delta t = 1$, $\gamma = c/|N_i|$and $\alpha = 0$, we have exactly the deterministic form of FIPS. However this form is more interesting. For example, even if not any p_i is modified (no improvement) it is well known that as soon as $\alpha > 0$, and $|N_i| \geq 3$, the trajectory of the particle is chaotic: there is no need of randomness. Strictly speaking it depends on the friction coefficient. High friction (small coefficient) implies less chaos, but if we set w to about 0.7, and c to about 1.4, as in classical PSO, we do easily obtain chaos.

Also, contrarily to FIPS (and to any classical PSO version), the particle speeds up when its distance to a p_i is decreasing: it is not easily trapped into a local minimum.

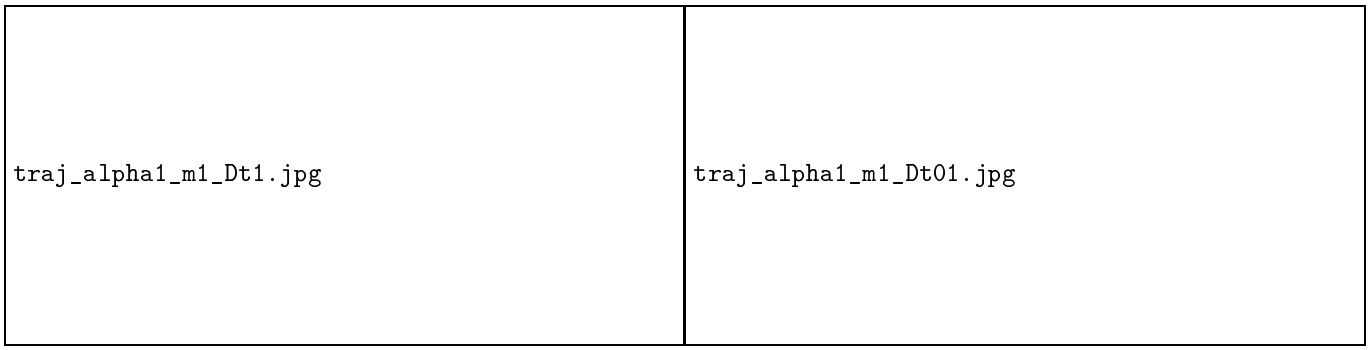
There is a drawback, though: we have to define Δt. And, in particular, we usually can not just set it to 1, for it is often a too high value that does not preserve the dynamic of the underlying continuous system, as we can see on figure 6.1. Actually a good way may be an adaptive Δt, decreasing function of the velocity module.

Another drawback is that the term $\left\| p_{N_{i,j}}(t) - x_i(t) \right\|$ of equation 6.1 can perfectly be equal to zero, or at least very small. There are many ways to cope with this problem (the simplest one is to just not take into account "bodies" p_i that are too near of the particle), but none of them is completely satisfying.

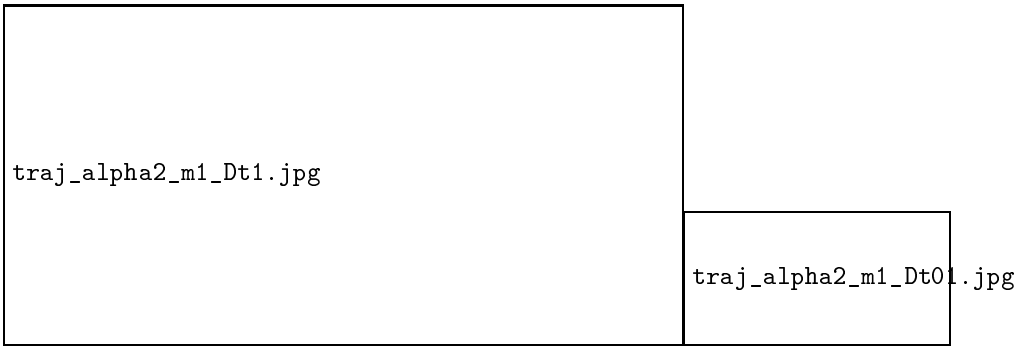
6.2. **Tests and results.** ** TO DO ** Preliminary tests are not very promising. The dissipation function has probably to be modified (see section 7 Dissipative system approach).

7. Dissipative system approach

*** TO COMPLETE *** Inspired by [5].



(a) alpha=1, Dt=1 (b) alpha=1, Dt=0.1



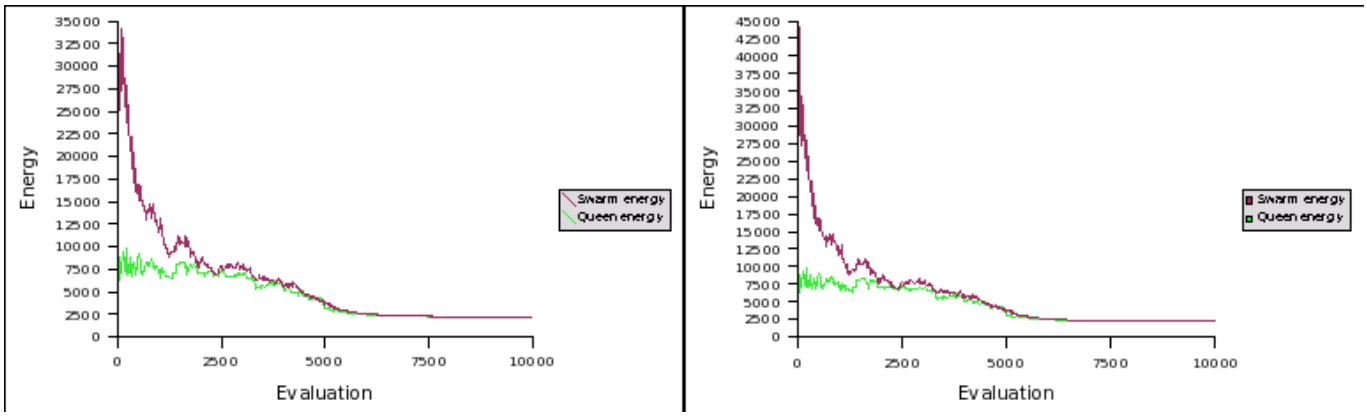
(c) alpha=2, Dt=1 (d) alpha=2, Dt=0.1

FIGURE 6.1. Trajectories with central force laws in viscous space. The smaller Δt, and the higher α, the easier chaos occurs

7.1. **Energy.** When plotting the total energy of the swarm (kinetic+potential) it appears that it globally decays with just some small oscillations (see figure 7.1). Note that the velocity of a particle is given by $x(t) - x(t-1)$, so it is defined even for PSO versions without any explicit velocity, like Bare-Bones.

Such a decay means that the dissipation function g in the Hamiltonian H of the system is quite similar to a friction. If the value of the global minimum is x^*then its root is $|f(x^*)|$, for on the long term all velocities tend to zero. We can write

$$g(|f(x^*)|) = 0$$



(a) Standard PSO (b) Gaussian Bare-Bones PSO

FIGURE 7.1. Typical energy evolutions. Rastrigin 30D, 100000 fitness evaluations

7.2. **Correlations between particles.**

7.2.1. *Coupling to the queen.* How is the swarm coupled to its centre of mass (the "queen" as defined in [6])? As we can se on figure 7.1, if we plot the energy of the queen and compare it to the one of the swarm, there is a high correlation after a while, but not on the beginning. It is not visible on the figure, but here the kinetic energy is quite small compared to the potential one. It means the potential energy of the queen (defined as $f(queen)\,S$ is almost equal to the sum of all potential energies of the particles.

So it might be possible to dramatically reduce the number of fitness evaluations: after a while most of the time just one for the queen (and the fitness value is also assigned to all particle), and from time to time for the whole swarm. Of course the main difficulty is to precisely define "after a while" and "from time to time".

7.2.2. *Relative trajectories.* It may be interesting to see how particles are flying around the queen, i.e. their relative trajectories.

8. Stagnation and progression phases

In all iterative algorithms an iteration can be said to be "successful" (progress) if the best known fitness value has decreased (when looking for a minimum), or "stagnant" if not. We are studying here the stagnation/progress sequence along a run. Let us define a coding: a sequence like $(1, -2, 3...)$means "progress during one iteration, stagnation during 2, progress during 3, ...".

Such a study may give us some insights on the dynamics of the algorithm, and even on its quality. For example, if there are only progresses (like say deterministic Gradient Descent), we are sure the algorithm is bad as soon as the function is multimodal. More generally, and intuitively, a good algorithm (meaning good on a large set of problems) should probably have a"balanced" stag/prog sequence.

On figure 8.1 we can see a typical sequence given by Standard PSO 2006 [7] on a classical problem. It seems it is indeed balanced, but plotting the histogram is more informative. In figure 8.2 we have some such histograms, for Standard PSO, FIPS (equation 5.1), and Gaussian Bare-Bones PSO (see below equation 10.1) . Although there are on the whole and in all cases as many stagnations as progresses, the behaviours are clearly very different, so it might possible that such histograms are "signatures" of the algorithms.

For example, for a given problem, is there a relation between the performance of a PSO variant, and the skewness of its stag/prog sequences? Table 2 gives an idea of the performance over 50 runs (mean of the best values after 10000 fitness evaluations for Parabola/Sphere, and after 40000 for Rastrigin), and of the mean skewness of the corresponding stag/prog sequences.

	Parabola on$[-100,100]^{30}$	Rastrigin on $[-10,10]^{30}$
Standard PSO	0.00011 (0.03)	67.08 (-23.09)
FIPS	0.000092 (0.07)	178.92 (-3.58)
Bare-Bones	0.00013 (-1.6)	61.31 (-20.43)

Table 2. Performance of three PSO variants, and mean skewness of the stagnation/progress sequences

First of all we can note that figures 8.2 do not always give a good idea of the real skewness. For example on b), the big skweness is in fact due to a few very long stagnation sequences (typically 50 iterations) that do not appear on the figure. Second, on these small examples performance there is no clear relation ship between skweness and performance.

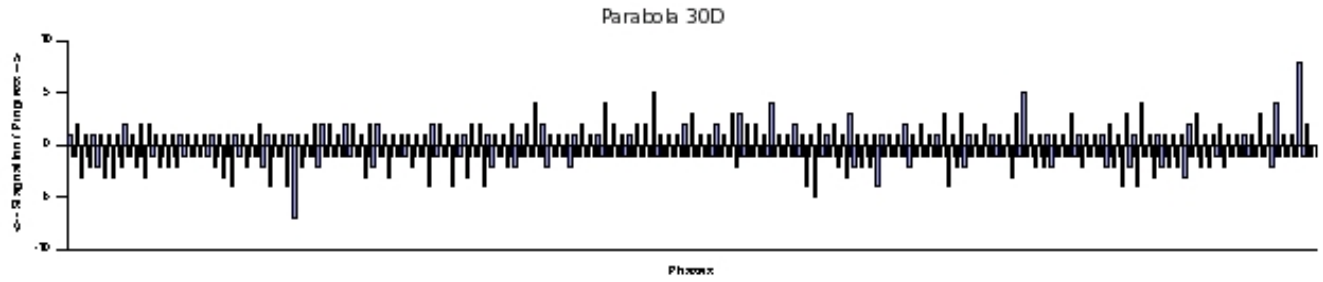
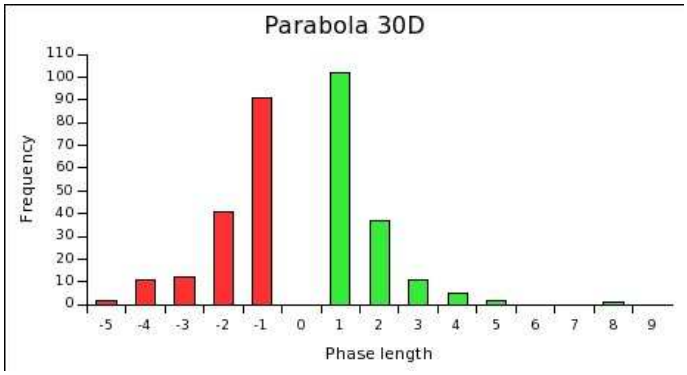
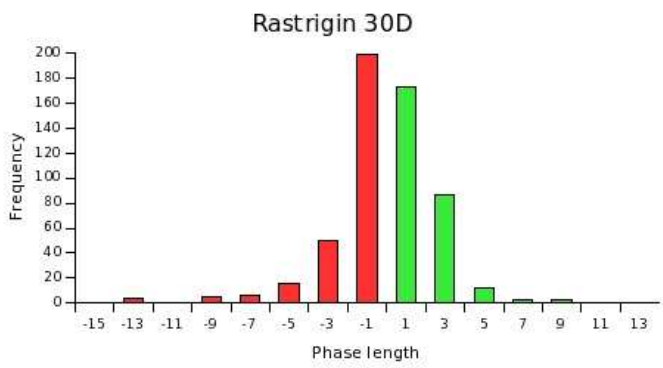


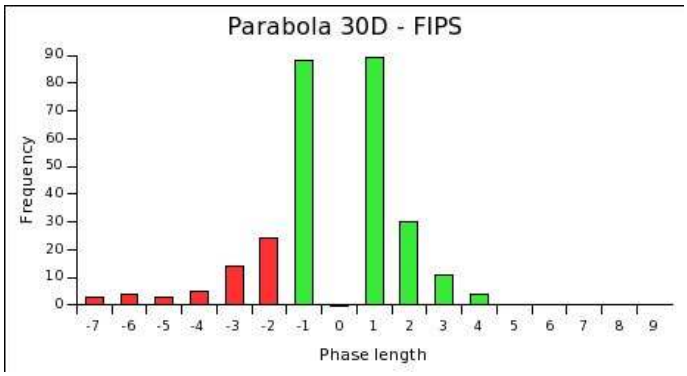
Figure 8.1. Stagnation-Progression sequence. Parabola (Sphere) 30. Standard PSO



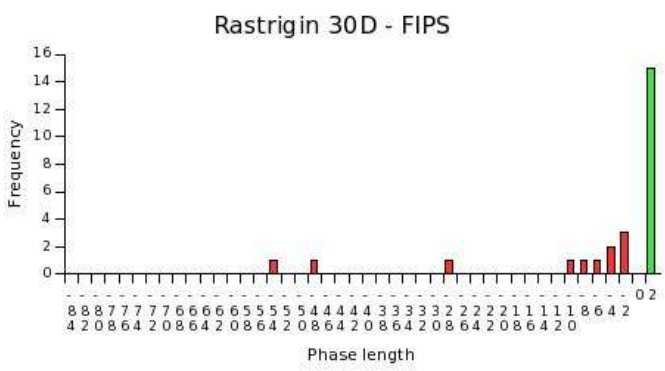
(a) Standard PSO. Skewness = 0.03



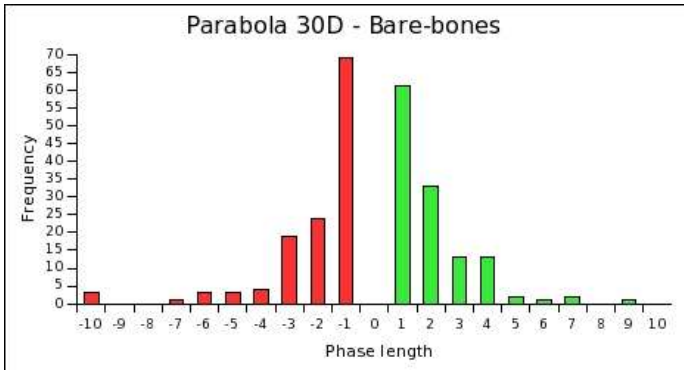
(b) Standard PSO. Skewness = -23.09



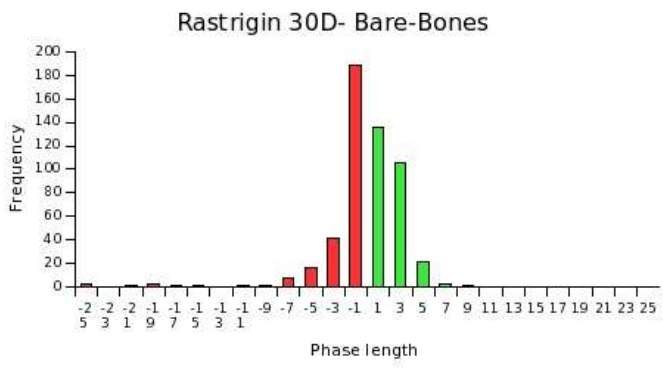
(c) FIPS. Skewness = 3.67



(d) FIPS. Skewness = 2.02



(e) Gaussian Bare-Bones. Skewness = -1.6



(f) Gaussian Bare-Bones. -20.43

FIGURE 8.2. Stagnation-Progression histograms. Standard PSO, FIPS, Gaussian Bare-Bones

9. Distorted Search Space

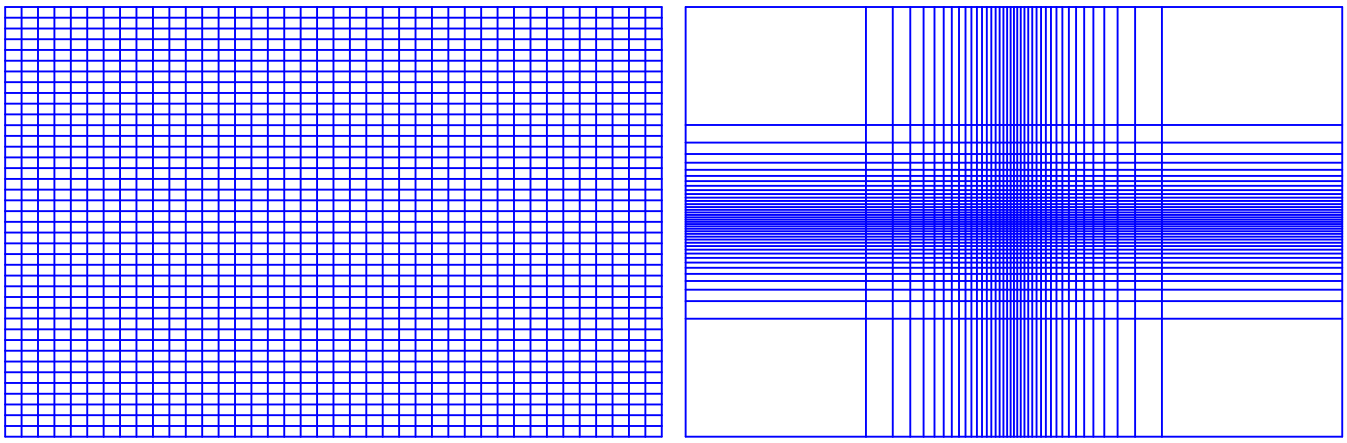
9.1. **Centre bias.** The starting point is a PSO variant with a bias in favour of the centre O of the search space $\mathcal{H}$, for example Standard PSO without any clamping (i.e. when a particle leaves the search space, its position is not modified and not evaluated). Let us suppose there is a metric defined on $\mathcal{H}$, so that the distance between two points can be calculated. We also define a mapping Φ of $\mathcal{H}$ on itself so that

- Φ is a bijection. So Φ^{-1} does exist
- $\Phi(O) = O$
- if x is on a bound then $\Phi(x) = x$
- for any other point we have $distance(\Phi(x), O) < distance(x, O)$

It means the search space is contracted towards its centre. Now, for a given fitness function f we define g by $g(y) = f\left(\Phi^{-1}(y)\right)$. Then if y^* is the position of a global minimum of g, the position of a global minimum of f is $x^* = \Phi^{-1}(y^*)$. The idea is to define Φ so that finding $y^*$for g is easier than directly finding x^* for f. Let us try with a simple continuous mapping (homeomorphism). If $x = (x_1, \ldots, x_d, \ldots, x_D)$we define $\Phi(x) = (\phi(x_1), \ldots, \phi(x_d), \ldots, \phi(x_D))$ with

$$\phi(x_d) = \Delta_d sign(x_d)\left(1 - \left(1 - \frac{x_d}{\Delta_d}\right)^{\alpha}\right)$$

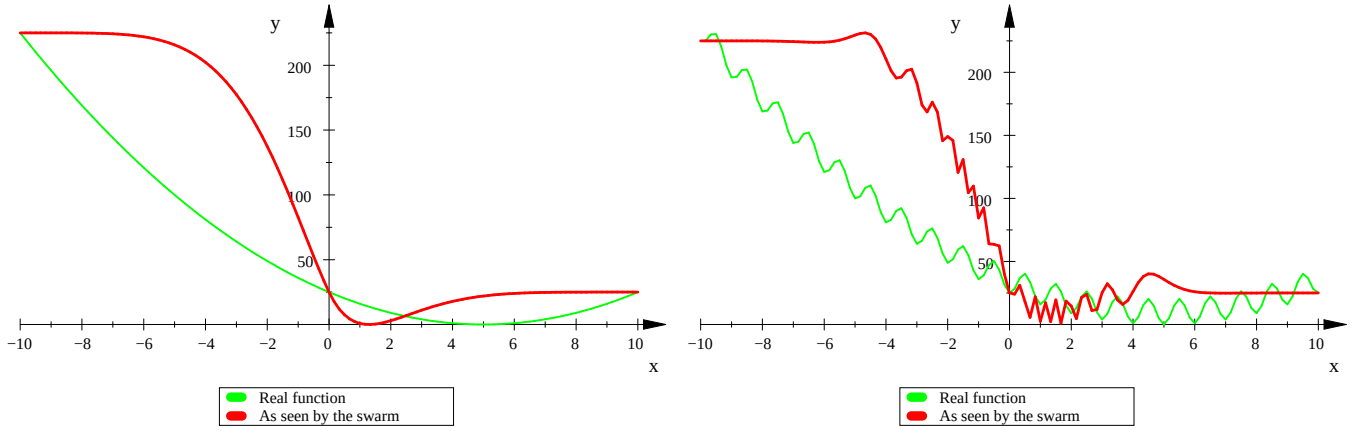
where $\Delta_d = \frac{x_{max,d} - x_{min,d}}{2}$. We can have an idea of the result on figure 9.1.



(a) Original Euclidean space (b) Contracted space

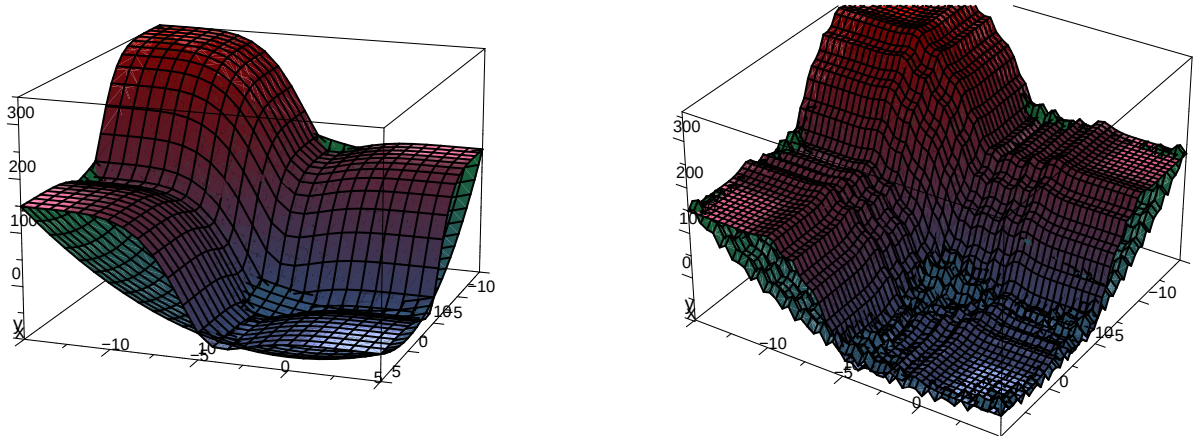
FIGURE 9.1. Simple space contraction, $\alpha = 0.2$

Now the objective function, as "seen" by the swarm is of course also different. By construction the minimum is nearer of the centre of the search space (except if it was on the bounds), but the global shape is not the same, as we can see on figure 9.2.



(a) Parabola 1D

(b) Rastrigin 1D



(c) Parabola 2D

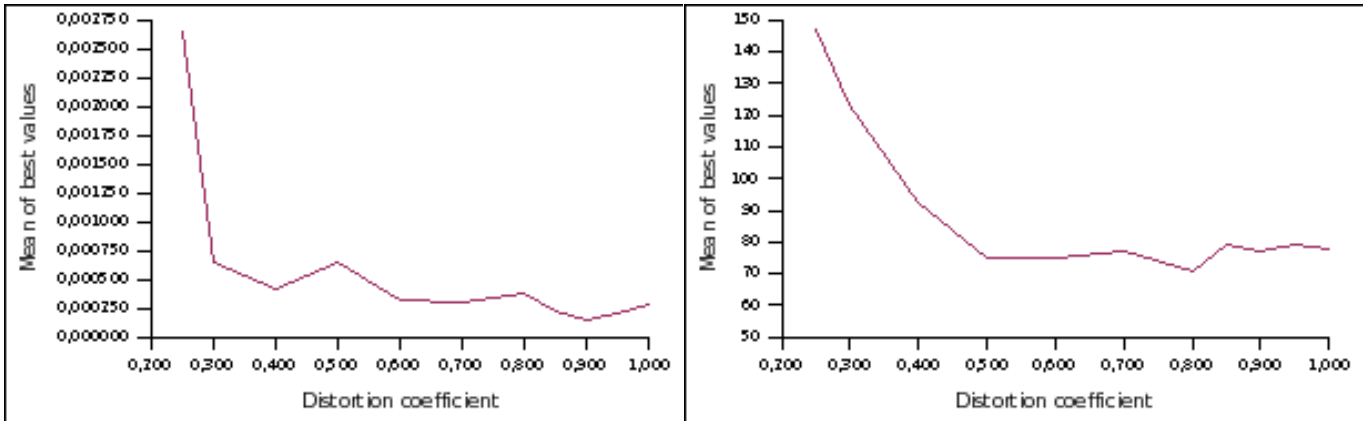
(d) Rastrigin 2D

FIGURE 9.2. Fitness function deformation. $\alpha = 0.2$

Of course the question now is to see if the performance is better. Let us compare results on two functions:

- Parabola on$[-100,100]^{30}$, offset=50
- Rastrigin on $[-10,10]^{30}$, offset=5

and over 50 runs (mean of the best values after 10000 fitness evaluations for Parabola/Sphere, and after 40000 for Rastrigin).

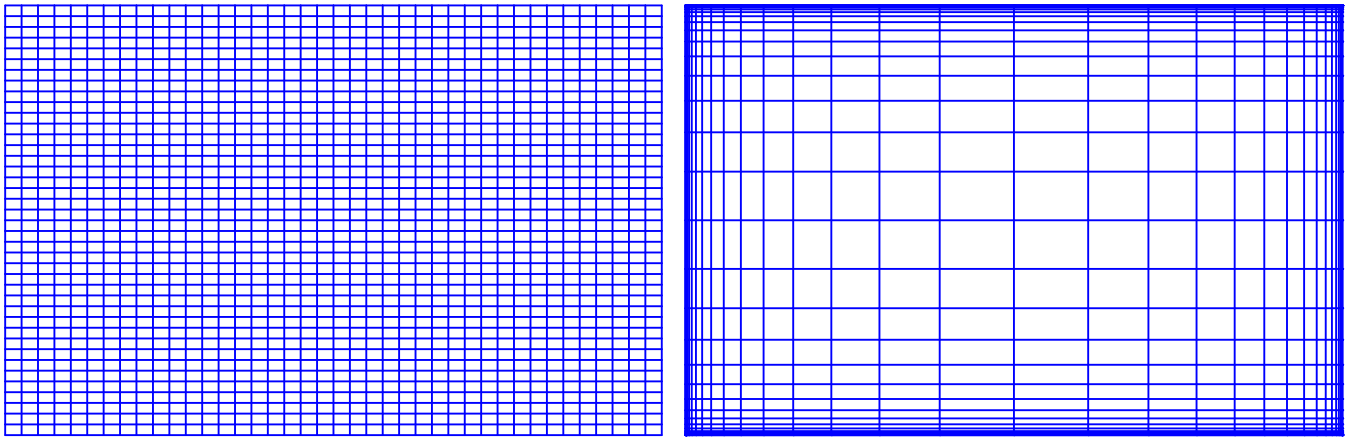


(a) Parabola

(b) Rastrigin

FIGURE 9.3. Performance by using a contracted search space and Standard PSO

9.2. **Bounds bias.** Now we try to take advantage of a bias in favour of the bounds, for example induced by using a classical clamping method: when a particle tends to leave the search space along dimension d, the coordinate is set to the bound value, and the corresponding velocity component is set to zero.



(a) Original Euclidean space (b) Expanded search space

FIGURE 9.4. Simple space expansion, $\alpha = 1.5$

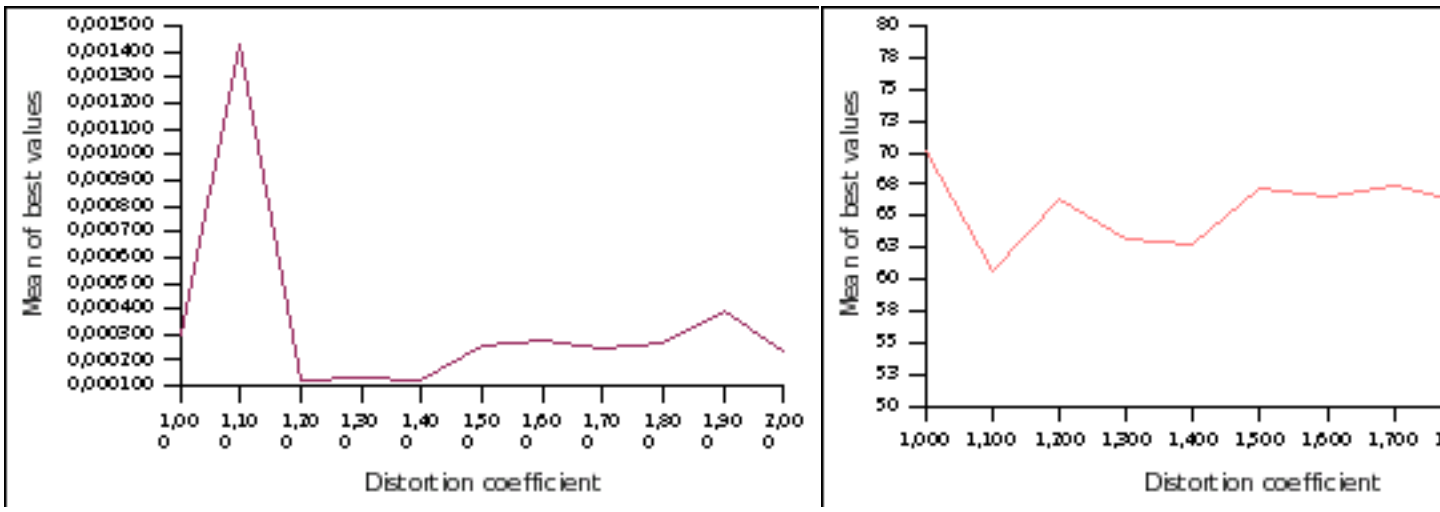


FIGURE 9.5. Performance by using an expanded search space, and Standard PSO without clamping

9.3. **Comments on results.** In both cases there is an effect, but not very clear. It seems that taking advantage of a bias in favour of the bounds is a bit more effective. However as there is always also an intrinsic bias in favour of the centre it may be better to use another mapping that simultaneously contract what is near of the center, and expand what is near of the bounds.

Another point is that when contraction is too important the algorithm is not stable. It seems quite normal, for a small move in the contracted space may be in fact a very big move in the real space. Actually, for this precise reason, the method may not work at all. We have to define a biased PSO *and* a mapping so that the final algorithm is indeed an improvement.

10. PSO and Information Theory

We are considering here "classical" PSO, i.e. with a constant swarm size S and with velocities. The swarm has S particles positions x_i, S memories p_i, S velocities v_i. At each time it then stores $3S$ positions ($2S$ directly, and S thanks to the velocities). More precisely, for each particle i the known positions are $x_i(t)$, $p_i(t)$, and $x_i(t-1)$ (thanks to $x_i(t-1) = x_i(t) - v_i(t)$).

At each time the swarm "learns" S new positions. How many does it forget? If there is no improvement, exactly S (the previous positions): the quantity of information is constant. If there are m improvements (i.e. for m particles i we have $f(x_i(t+1)) < f(p_i(t))$) then for each one the data are $x_i(t+1)$, $p_i(t+1) = x_i(t+1)$, $x_i(t)$. It means we have temporarily lost m informations (positions).

After that, if the particle does not improve its p_i, we have again three different informations $x(t+2)$, $p(t+2) = p(t+1) \neq x(t+2)$, $x(t+1)$. So, on the whole, the number of known different positions is oscillating between $3S$ and $2S$: $3S$ when there is no improvement at all, $2S$ when all particles improve their p_i.

In passing, it means it would probably be better to define a "Constant Informed PSO", in which the number of different known positions is constant.

Anyway, as soon as the dimension is high, the volume of the successful possible positions is very small compared to the volume of the search space. There is no chance to find one just by sampling at random $2S$ or even $3S$ positions at each time, according to an uniform distribution. But precisely, this sampling is not uniform. So it can work (when it works) only for two reasons:

(1) the fitness landscape is so that the assertion "nearer (of the optimum x^*) is better" has a truth value not too small. Or, in other words, when sampling a point at random, the probability it is in the attraction basin of x^* is not too small
(2) at least after a given time t, and at least for a given $p_i(t)$ inside the attraction basin, there exists x_j so that the probability that is on average greater than 0.5.

Of course we don't know when a p_i is in the attraction basin. So to be sure the condition 2 should be in fact: for any p_i, there is at least one x_j so that the probability $distance(x_j(t+1), p_i(t)) < distance(x_j(t), p_i(t))$ is greater than 0.5. Or, roughly speaking "for any memory there is at least one particle that tends more or less towards it".

For example let's consider the Gaussian (Normal) bare-bones PSO defined by

$$x(t+1) = \mathcal{N}\left(\frac{p_i + g_i}{2}, w|p_i - g_i|\right) \tag{10.1}$$

in which g_i is, as usually, the best known position in the neighbourhood of the particle. It means a good w value should be about 0.7. Actually it seems the best results are obtained with the "magic" value $1/(2\ln(2)) \simeq 0.72$ found in the Stagnation Analysis [8]. There might be something "deep" here ...

11. Constant Informed PSO

Keep constant the number of *different* known positions, for example by memorizing two "best" positions instead of one: $p(t)$and $p'(t)$and by applying the following rules:

$p(0) = p'(0) = x(0)$
$p(t-1) \to p'(t)$
if $f(x(t)) < f(x(t-1))$
then $x(t) \to p(t)$ else $p(t-1) \to p(t)$

So at each time after 0 we usually know two different positions, either x and $p = p'$ or $x = p$ and p'. Of course the movement equations should now taking into account not only the classical $\{x, p, g\}$where g is the neighbourhood best, but $\{x, p, p', g\}$. Note that we always have

$$f(x) \geq f(p) \geq f(p') \geq f(g)$$

Option 1: if the particle has improved its position then use $\{x, p', g\}$ else use $\{x, p, g\}$.
Option 2: use $\{x, p, p', g\}$

$$v_{i,d}(t+1) = wv_{i,d}(t) + \lambda\widetilde{c}(p_d(t) - x_{i,d}(t)) + \lambda^2\widetilde{c}(p'_d(t) - x_{i,d}(t)) + \widetilde{c}(g_d(t) - x_{i,d}(t))$$

with
$1 + \lambda + \lambda^2 = 2$ i.e. $\lambda = \left(\sqrt{5} - 1\right)/2$

The idea is "tend towards g", a bit less "towards p", and even less "towards p'". According to table 3 Option 1 is not interesting, but Option 2 is. Note that the idea to tends more towards g than towards p (but not by taking into account their fitness values), might be applied to classical PSO or, more interesting, to FIPS (by sorting the $p_i$according to their fitness values).

12. Quantisation

12.1. **Description and equations.** Working on a quantised search space has at least two advantages ... and one drawback. On the one hand

(1) if the fitness landscape has a global curvature, by using quantisation you should quite quickly reduce the search area to a smaller one around the optimum.

	Parabola on[-100,100]^{30}	Rastrigin on [-10,10]^{30}
Standard PSO	0.00011 (0.03)	67.08
CI option 1	0.028	134.4
CI option 2	0.00061	49.54

TABLE 3. Constand Informed PSO

(2) the particle may also easily escape a local optimum, for it simply does not "see" it. This is a kind of "tunnel effect".

Now, on the other hand, you may *never* find an acceptable position if the optimum is too far from a quantised position. So we not only have to define a quantisation step, but also to adapt it according to what the swarm learns about the search space during the process. Roughly speaking the rule should be something like "if no improvement, decrease it, and if improvement, increase it". Now the main difficulty is to mathematically define each term of such a rule.

In the context of Tribes [3], a parameter free PSO which is by nature completely adaptive, "improvement" is already perfectly defined, and is usually locally different for each tribe. However we try here to do something simpler, in the context of Standard PSO. So we use a more classical definition: "improvement" just means either "improvement of the global best" (global adaptation), or "improvement of the personal best" (individual adaptation). Note that in this last case each particle has its own quantisation.

Now, we need a formula to define how to increase/decrease the quantisation step. As a first approach, we just try here a simple deterministic one. Let $q(t)$ be the relative step at time t. It means that on each dimension d the real step is $Q_d(t) = q(t)(x_{max,d} - x_{min,d})/2$. The adaptation rule is defined by

$$\left\{ \begin{array}{rcl} q(0) & = & \text{any value in }]0,1[\\ q(t+1) & = & MIN(s_1 q(t), q(0)), s_1 > 1 \text{ if improvement} \\ & = & s_2 q(t), s_2 \in]0,1[\text{ if no improvement} \end{array} \right.$$

At last when a new quantisation is used it is also useful to partly re-initialise the velocities, particularly when the step has been increased: if the velocity is too small the particle may not move at all. In practice we use here the following formula on each dimension

$$\left\{ \begin{array}{rcl} u & = & U(0,Q_d) - U(0,Q_d) \\ |u| > v_d & \Rightarrow & v_d \leftarrow u \end{array} \right.$$

12.2. **Tests and results.** Note that, of course, results may easily be extraordinary good when the solution is on the centre of the search space, or even right on a quantised point, as for most of test functions. So we use an offset, and one whose relative value is irrational, for example $1/\sqrt{3} \simeq 0.577$. For the same reason, the worst case is to choose a transcendant number for $q(0)$, for example $e/10 \simeq 0.27$. On the contrary it may be a good idea to choose a transcendant number for one of the scaling coefficients. On table 4 we have set $s_2 = \pi/4 \simeq 0.79$. As we can see it seems quantisation may slightly improve the algorithm.

s_1	Parabola on[-100,100]^{30}	Rastrigin on [-10,10]^{30}
0	0.000230	86.96
1	0.0000134	89.35
1.5	0.000195	86.20
2	0.000079	84.49

TABLE 4. Performance of quantisation. Mean of the best values over 50 runs (10000 evaluations/run for Parabola, 40000 for Rastrigin). In the case $s_1 = 0$ there is in fact no quantisation at all . When $s_1 = 1$ it means the step size can not increase

13. TIME ASYMMETRY

** TO DO **

given x(t)=x*, find the probability distribution of all possible x(t-1), then of all x(t-2), ..., and, finally, the distribution of the successful starting points. May be easier to do with FIPS, and even easier with Bare-Bones. Expected result: a non uniform distribution that should be used for initialisation. As PSO has a "zero" bias, a good initial distribution should probably be less dense around the centre.

Evaluation probability	Parabola on[-100,100]^{30}	Rastrigin on [-10,10]^{30}
1	0.000230	86.96
0.75	0.000056	87.05
0.5	0.00071	84.36
0.25	0.000240	90.77

TABLE 5. Performance with probabilistic evaluation. Relative offset$1/\sqrt{3} \simeq 0.577$. Mean of best values over 50 runs. 10000 evaluations for Parabola, 40000 for Rastrigin

14. Why is Sphere linear?

Experimentally, when we plot $\ln(f(p(t)))$vs FE (number of fitness evaluations), we find a curve whose mean slpoe is clearly (decreasing) linear when $f(x)$ is defined by something like x^α with $\alpha > 0$. It has probably something to do with the PSO equivalence between $x^\alpha$and $|x|$. Let us examine what happens when the dimension is simply 1.

14.1. **Exactly true for Gradient Descent.** With classical gradient descent (GD), and $f(x) = x^\alpha$ we have

$$x(t) = \frac{\alpha - 1}{\alpha} x(t-1)$$

So $f(x(t)) = \frac{\alpha-1}{\alpha}^{t\alpha} x(0)^\alpha$. When taking the logarithm we indeed obtain

$$\ln(f(x)) = \alpha \ln\left(\frac{\alpha - 1}{\alpha}\right) t + \alpha \ln(x(0))$$

i.e. a line whose slope is $\alpha \ln\left(\frac{\alpha-1}{\alpha}\right)$.

14.2. **More generally.** Let $\widetilde{u}$ be the expectation of the random variable u (we do suppose here it exists). By analogy with the previous analysis, if we can prove that we have

$$\widetilde{p}(t) = \widetilde{\beta}\widetilde{p}(t-1)$$

then we would have

$$\widetilde{\ln(f(p(t)))} = \alpha\widetilde{\ln(\beta)}t + \alpha \ln(p(0))$$

*** TO COMPLETE ***

15. Better to not evaluate?

When a particle tends to leave the search space it is quite usual to clamp the position, and sometimes the velocity, to an acceptable value. However a pretty good and simpler method is to do nothing special and to not evaluate the position. Of course it means the fitness value is wrong but sooner or later the particle should be attracted inside the search space anyway.

Now there is no theoretical reason to use this method only when the particle leaves the search space. After all you can define it quite arbitrarily. Sometimes fitness evaluation is very costly so the obvious question is "What happens if I not always evaluate the new position?" The idea is then to evaluate only with a probability of say 50%. We can then perform two times more moves for almost the same cost. If the convergence of the algorithm is equivalent or even slightly deteriorated, it is interesting. As we can see on table 5 it seems it is indeed the case: for the same number of fitness evaluations we easily find a significantly better result for Parabola, and a slightly better one for Rastrigin.

16. Weighted Bare-Bones

In (Gaussian) Bare-Bones PSO, the next position is simply given by

$$x(t+1) = N\left(\frac{p(t) + g(t)}{2}, |p(t) - g(t)|\right)$$

where $N(\mu, \sigma)$is the Gaussian distribution with a mean equal to μ and a standard deviation equal to σ. However it can be easily found experimentally that replacing $|p(t) - g(t)|$ by $w|p(t) - g(t)|$ with $0 < w < 1$ gives better results. Actually a short theoretical analysis is even possible. The idea is that the next position should satisfy two conditions, in probability:

- not too far from $p(t)$
- but nevertheless nearer of $g(t)$

t	x1	v1	x2	v2
0	0.1	-0.9	0.9	0.9
1	-0.8	1.15	1.8	-1.05
2	0.35	0.07	0.75	-1.41
3	0.43	-0.6	-0.66	1.15
4	-0.17	-0.16	0.49	0.55
5	-0.33	1.18	1.04	-2.04
6	0.85	0.1	-1	2.61
7	0.95	-1.62	1.61	0.76
8	-0.68	-1.17	2.37	-7.76
9	-1.84	0.84	-5.39	-5.59
10	-2.69	4.97	-10.97	29.99
11	2.28	1.4	19.01	-34.36
12	3.68	-9.72	-15.35	6.55
13	-6.05	-7	-8.8	4.72
14	-13.05	34.4	-4.08	7.97

TABLE 6. When "no clamping and no evaluation" implies "no stop".

It means that $x(t+1)$ should be in $[p(t), g(t)]$with a probability just a bit bigger than 0.5. If we define $\sigma = w|p(t) - g(t)|$we then just have to find w so that

$$\frac{1}{\sqrt{2\pi}} \int_{-1/2w}^{1/2w} e^{-v^2} dv > 0.5$$

By using the normalizes inverted cumulative normal distribution, we find $w < 0.74$. If we assume that w should just balance the behaviour between exploration and exploitation, we may set it precisely to this value.

17. No clamping and no evaluation: a false good idea

When a particle tends to leave the search space an usual way is to clamp it, and sometimes the velocity too. Something like

$$\text{if} x_d > x_{max,d} \text{then} \left\{ \begin{array}{lcl} x_d & = & x_{max,d} \\ v_d & = & -\alpha v_d, \, \alpha \in [0,1] \end{array} \right.$$

Some people claim that there are two drawbacks:

- it increases the bias towards the center of the search space
- it costs one fitness evaluation

Note that, though, this method is interesting when the solution is either near the center of the search space or near the bounds.

Another way is to simply do nothing: no clamping and no evaluation. Or, in other words, outside the search space, the particle see the landscape as flat, with a fitness value precisely equal to its current one. The underlying idea is that it will come back soon, for it is "attracted" by some positions that do are inside the search space (for example the previous best and the local best, for classical PSO).

Usually it works quite well. But not always. It may happen that all particle are outside the search space and fail to come back in. In such a case, if the stop criterion is precisely a maximum number of evaluations, the program never stops. Let us consider a very small example. Dimension is 1, search space is $[0, 1]$, fitness function is $f(x) = x$. The motion equation is $v(t+1) = wv(t) + \widetilde{c}(p(t) - x(t)) + \widetilde{c}(g(t) - x(t))$ with $w = 0.72$, and $c = 1.48$. As we can see on Table 6 the particles tend to oscillate, which is quite normal, but as they are both outside the search space after time 8, the oscillations seem to increase, again and again. It might be possible that, after a very long time, the particles indeed come back, but even in such a case, the performance would be extremely bad.

So it seems necessary either to give up this method, or to add another stop criterion, like a maximum number of iterations.

Actually there is another and more important reason for which "no clamping" is a false good idea: it is less effective. Using any clamping method indeed induces more bias, but this is not a problem. A good method gives better result in two cases: when the solution point is near of the bounds, or near of the centre of the search space, and similar result else (see my paper Confinements and Bias for details).

18. Rand(Rand(Rand()))

By discussing with some people I found there is a quite common ununderstanding of some simple probability problems. Here is one the them. It has some connections with optimisation, but basically it can be described as follows.

Let us consider the two random variables:

$x_1$uniform on $[0,1]$

$x_2$uniform on $]x_1,1]$

What is the distribution law of x_2? Let us denote $p(x_2<u)$the probability that $x_2$is smaller than u. We obviously have

$$\left\{\begin{array}{lcl} p(x_2<u) & = & 0 \text{ for} u\in[0,x_1] \\ & = & \frac{u-x_1}{1-x_1} \text{ for } u\in]x_1,1] \end{array}\right.$$

This is of course for a given x_1. The probability density of $x_1$is simply 1, so to find a formula not depending on $x_1$we just have to compute the following integral

$$\left\{\begin{array}{lcl} p(x_2<u) & = & \int_0^u \frac{u-x_1}{1-x_1}dx_1 \\ & = & u-(u-1)\ln(1-u) \end{array}\right.$$

Actually, to estimate the probability of success of some algorithms, it may be interesting to go further. Let x_3 be a uniform random variable in $[0,x_2[$. By definition, if $x_2<u$ then $x_3<u$ for sure. Else we have $p(x_3<u)=u/x_2$. Note also that the probability density of $x_2$is $-\ln(1-v)$ (i.e. $\lim_{dv\to 0}p(v<x_2<v+dv)$. In passing, we can then compute its expectation

$$\left\{\begin{array}{lcl} E(x_2) & = & \int_0^1 -v\ln(1-v)\,dv \\ & = & -\frac{1}{4}\left[2\ln(1-v)\left(v^2-1\right)-v(v+2)\right]_0^1 \\ & = & \frac{3}{4} \end{array}\right.$$

For x_3 we have immediately

$$\left\{\begin{array}{lcl} D_3(u)=p(x_3<u) & = & u-(u-1)\ln(1-u)-u\int_u^1 \frac{\ln(1-v)}{v}dv \\ & = & u-(u-1)\ln(1-u)+u\left(Li_2(1)-Li_2(u)\right) \\ & = & u-(u-1)\ln(1-u)+u\left(\frac{\pi^2}{6}-Li_2(u)\right) \end{array}\right.$$

where $Li_2$is the dilogarithm. In particular for $u=0.5$ we have $Li_2(0.5)=\frac{\pi^2}{12}-\frac{\ln^2(2)}{2}$, and $D_3(0.5)\simeq 0.68$.

The probability density for x_3 is given by $d_3(u)=\frac{\pi^2}{6}-Li_2(u)$. So we can compute the expectation by

$$\left\{\begin{array}{lcl} E(x_3) & = & \int_0^1 u\left(\frac{\pi^2}{6}-Li_2(u)\right)du \\ & = & \frac{3}{8}=0.375 \end{array}\right.$$

Note - This result has been obtained thanks to a symbolic calculus software

Links

The basic simplex method, http://www.multisimplex.com/simplex_b.htm

Nelder-Mead method, http://en.wikipedia.org/wiki/Downhill_simplex

References

[1] J. Kennedy, "Bare Bones Particle Swarms," in *IEEE Swarm Intelligence Symposium*, pp. 80–87, 2003. Fully informed PSO.

[2] R. Mendes, *Population Topologies and Their Influence in Particle Swarm Performance*. PhD thesis, Universidade do Minho, 2004.

[3] M. Clerc, *Particle Swarm Optimization*. ISTE (International Scientific and Technical Encyclopedia), 2006.

[4] T. M. Blackwell and P. J. Bentley, "Dynamic Search with Charged Swarms," in *Genetic and Evolutionary Computation Conference*, (San Francisco), pp. 19–26, Morgan Kaufmann, 2002.

[5] F. Schweitzer, W. Ebeling, and B. Tilch, "Statistical mechanics of canonical-dissipative systems and applications to swarm dynamics.," *Phys Rev E Stat Nonlin Soft Matter Phys*, vol. 64, p. 021110, Aug 2001.

[6] M. Clerc, "The Swarm and the Queen: Towards a Deterministic and Adaptive Particle Swarm Optimization," in *Congress on Evolutionary Computation*, vol. 3, (Washington DC), pp. 1951–1955, IEEE, 1999.

[7] PSC, "Particle Swarm Central, http://www.particleswarm.info."

[8] M. Clerc, "Stagnation analysis in particle swarm optimization or what happens when nothing happens.," tech. rep., 2005.