Chapter 3
Theoretical Derivations and Application Issues

This chapter deals with fundamental theoretical investigations and application issues of PSO. We are mostly interested in developments that offer new insight in configuring and tuning the parameters of the method. For this purpose, the chapter opens with a discussion on initialization techniques, followed by brief presentations of investigations on particle trajectories and the stability analysis of PSO. A useful technique based on computational statistics is also presented for the optimal tuning of the algorithm on specific problems. The chapter closes with a short discussion on termination conditions.

INITIALIZATION TECHNIQUES

Initialization is perhaps the less studied phase of PSO and other evolutionary algorithms. This may be due to the general demand for developing algorithms that are not very sensitive in the initial conditions. However, it can be experimentally verified that, in various problems, initialization can have a significant impact on performance.

As already mentioned in the previous chapter, uniform random initialization is the most popular scheme in evolutionary computation due to the necessity for equally treating each part of a search space with unrevealed characteristics. However, alternative initialization methodologies that use different probability distributions or employ direct search methods to provide the first steps of the algorithm have proved very useful.

In the following sections, we discuss the most common probabilistic initialization techniques. In addition, we present a scheme based on the nonlinear simplex method of Nelder and Mead, which has been shown to work beneficially for the initialization of PSO (Parsopoulos & Vrahatis, 2002).

DOI: 10.4018/978-1-61520-666-7.ch003
Random Probabilistic Initialization

In the framework of PSO, the quantities that need to be initialized prior to application are the particles as well as their velocities and best positions. The best positions consist of the best solutions already detected by each particle, while current particle positions represent candidate new solutions. Since no information on the promising regions of the search space is expected to be available prior to initialization, the initial particles and the corresponding best positions are considered to coincide. Also, in constrained optimization, we are interested in detecting feasible solutions, i.e., solutions that do not violate problem constraints. For this purpose, the initialization of swarm and best positions within the feasible search space, $A \subset \mathbb{R}^n$, is desirable.

The most common technique in evolutionary computation is *random uniform initialization*. According to this, each particle of the initial swarm and, consequently, of the initial best positions is drawn by sampling a uniform distribution over the search space $A$. The applicability of this approach depends on the form of the search space. If $A$ is given as an $n$-dimensional bounded box:

$$A = [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_n, b_n],$$

then any of the available pseudo-random number generators, such as the number-theoretically generated Sobol sequences (Press *et al.*, 1992, Chapter 7), can be directly used to produce uniformly distributed numbers within it.

In practice, it is very common to exploit the one-dimensional pseudo-random generators that accompany all modern computer systems. Thus, each component of a particle is generated as a uniformly distributed pseudo-random value within the interval $[0,1]$ and then scaled in the magnitude of the corresponding direction of $A$. This procedure is described in the pseudocode of Table 1, where we use the function `drand48()` provided by the C++ programming language as the pseudo-random generator in $[0,1]$. The produced value is then scaled in the corresponding direction of the search space, so that the produced particles lie strictly within $A$. In addition, we scale the produced pseudo-random values of the velocity components, in order to clamp it within its limits, $[-v_{\text{max}}, v_{\text{max}}]$, as described in the previous chapter. When a large number of subsequent experiments are conducted, re-initialization of the pseudo-random generator with a different seed may be occasionally necessary, in order to obtain unbiased experimental results.

Mathematically speaking, the particles produced by the aforementioned procedure do not exactly follow the multi-dimensional uniform distribution over $A$. Despite this theoretical deficiency, random uniform initialization became the technique one of choice; a popularity that can be attributed to the following properties:

1. It can be implemented easily in any computer system and programming language.
2. In many applications, except from some complex constrained optimization problems, the feasible search space can be given in the form of a bounded box or approximated by a sequence of such boxes.
3. It is suitable for time-critical applications as the generation of numbers is adequately fast, requiring only minor computational effort.
4. It does not have a partiality for any region of the search space.