Applied Mathematics and Computation 216 (2010) 2988–3001

Contents lists available at ScienceDirect

Applied Mathematics and Computation

journal homepage: www.elsevier.com/locate/amc

Enhancing PSO methods for global optimization

Ioannis G. Tsoulos [a,*], Athanassios Stavrakoudis [b]

[a] Technological Education Institute of Epiros, Department of Communications, Informatics and Management, University of Ioannina, Greece
[b] Department of Economics, University of Ioannina, Ioannina, Greece

ARTICLE INFO

Keywords:
Global optimization
Particle swarm optimization
Stochastic methods
Stopping rules

ABSTRACT

The Particle Swarm Optimization (PSO) method is a well-established technique for global optimization. During the past years several variations of the original PSO have been proposed in the relevant literature. Because of the increasing necessity in global optimization methods in almost all fields of science there is a great demand for efficient and fast implementations of relative algorithms. In this work we propose three modifications of the original PSO method in order to increase the speed and its efficiency that can be applied independently in almost every PSO variant. These modifications are: (a) a new stopping rule, (b) a similarity check and (c) a conditional application of some local search method. The proposed were tested using three popular PSO variants and a variety test functions. We have found that the application of these modifications resulted in significant gain in speed and efficiency.

1. Introduction

The problem of locating the global minimum of a continuous and differentiable function f can be formulated as: Determine

$$x^* = \arg\min_{x \in S}(x), \tag{1}$$

where the hyper box $S \subset R^n$ is defined as:

$$S = [a_1, b_1] \otimes [a_2, b_2] \otimes \cdots [a_n, b_n].$$

The above problem is applicable in many scientific fields such as chemistry [1,5,6], physics [2,4], architectural synthesis [3], economics [7,8] etc. During the past years many methods have been proposed for tackling the problem of global optimization. These methods can be divided in two main categories namely deterministic and stochastic. Methods belonging to the first category are more difficult to implement and they depend on a priory information about the objective function. Therefore, they are not further examined in this paper. On the other side, stochastic methods are implemented more easily and they do not require a priory information about the objective function. Among the stochastic methods for global optimization we refer to Random Line Search [9], Adaptive Random Search [10], Competitive Evolution [11], Controlled Random Search [12], Simulated Annealing [13–16], Genetic Algorithms [17,18], Differential Evolution [19,20], Tabu Search [21] etc. A stochastic method for global optimization which has recently attracted considerable attention by researchers is the Particle Swarm Optimization (PSO) algorithm. The PSO was initially suggested by Kennedy and Eberhart [22].

PSO is an evolutionary algorithm and it is based on population of candidate solutions (swarm of particles) which move in an n-dimensional search space. Every particle i is assigned its current position x_i and the so called velocity u_i. The two vectors are repeatedly updated, until a predefined convergence criterion is met (see Algorithm 1). The PSO method has been applied

* Corresponding author.
E-mail addresses: itsoulos@cs.uoi.gr, itsoulos@gmail.com (I.G. Tsoulos).

0096-3003/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.amc.2010.04.011

on a wide range of applications [23–26]. Recently many variations of the original method have been proposed in order to increase the speed and the efficiency of the method. Some of these methods aim to develop an automatic mechanism for the estimation of the parameters of the PSO method [27–29], while some others conjunct the PSO method with different type of stochastic techniques [30–32].

This article proposes three modifications of the PSO method: (a) a new stopping rule, (b) a similarity check and (c) a periodically application of a local optimization procedure. The generality of these modifications allows their application to any PSO variant.

The rest of this article is organized as follows: in Section 2 a general description of a typical PSO method is given as well as a detailed description of the proposed modifications. In Section 3 we describe the optimization problems used in our comparison and we give the results from the application of the proposed modifications to three PSO variants. Finally, in Section 4 a discussion is made about the experimental results.

2. Method description

In this section an outline of the general PSO method is given followed by the description of the three proposed modifications.

2.1. Description of the PSO method

The main steps of a generic PSO algorithm are presented in Algorithm 1. In **Initialization** step the algorithm: (a) sets the number of particles, (b) initializes the iteration counter and (c) assigns the initial positions (x_i) of the particles and their velocities (u_i) with uniformly distributed random numbers. The vector p_i holds the best visited position (the one with the lowest function value) for the particle i and the vector p_{best} is the best among $\{p_1, p_2, \ldots, p_m\}$. In the **Termination Check** step, the algorithm checks some predefined criteria such as the maximum number of iterations ($k \geqslant k_{\max}$) or how close is the best and the worst function values of the particles ($|f_{\max} - f_{\min}| < e, e > 0$) or some other stopping rules. In step 2 the main loop of the algorithm is performed: for every particle (a) the velocity is updated, (b) the current position is modified as a function of its associated velocity, (c) the fitness is calculated and (d) the best position p_i is updated if a better position is found. After the main loop the best position among p_i is assigned to p_{best}. The most common update mechanism for the current position of a particle is given by:

$$x_i = x_i + u_i, \tag{2}$$

where

$$u_{ij} = \omega u_{ij} + r_1 c_1 (p_{ij} - x_{ij}) + r_2 c_2 (p_{\text{best},j} - x_{ij}). \tag{3}$$

The j parameter denotes the jth element of the vector, where $j \in [1, \ldots, n]$. The parameters r_1 and $r_2$are random numbers in [0,1] and the constants c_1 and c_2 stands for the cognitive and the social parameters. Usually, the values for $c_1$and c_2 are in [1,2]. The parameter ω is called inertia with $0 \leqslant \omega \leqslant 1$. In [36] an update mechanism is proposed for the inertia following the formula:

$$\omega = \omega_{\max} - \frac{k}{k_{\max}} (\omega_{\max} - \omega_{\min}), \tag{4}$$

where $k_{\max}$ is the maximum number of iterations allowed and $\omega_{\min}$, $\omega_{\max}$ are defined by the user. Common values for these parameters are $\omega_{\min} = 0.4$ and $\omega_{\max} = 0.9$.

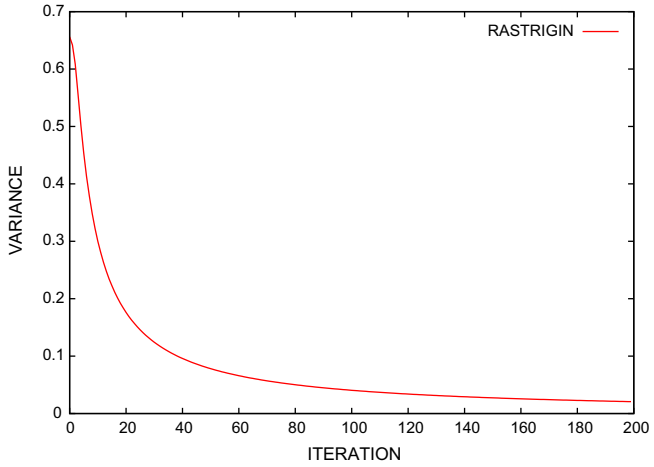

Fig. 1. Plot of variance of best value for the function Rastrigin.