ASSESSMENT ON STOCHASTIC ALGORITHMS FOR GLOBAL OPTIMIZATION

***Thiago André Weschenfelder*[1]**
***Márcio Schwaab*[2]**
***Marco di Luccio*[3]**
***Marcos Lúcio Corazza*[4]**
***Fernanda de Castilhos*[5]**

Resumo: Este trabalho compara algoritmos estocásticos diferentes aplicados a problemas de minimização de funções teste, para as quais técnicas matemáticas tradicionais costumam falhar. Os algoritmos estocásticos avaliados neste trabalho foram Colônia Artificial de Abelhas (CAB), Evolução Diferencial (ED), Enxame de Partículas (EP) e Recozimento Simulado (RS). Os parâmetros internos de cada algoritmo foram alterados e seus efeitos na performance do algoritmo foram avaliados e comparados na minimização de seis funções teste (Ackley, Griewank, Parabolic, Rastrigin, Rosenbrock and Scheffers). Os resultados permitiram concluir sobre a importância de uma definição adequada dos parâmetros internos. Além disso, a minimização da função de Rosenbrock com alta dimensão foi realizada com a melhor configuração de cada algoritmo. Os resultados mostraram que a maioria dos algoritmos conseguiu encontrar o mínimo global das funções multimodais. Entretanto, os algoritmos ED e CAB apresentaram os melhores resultados em termos de convergência e qualidade de resultados.

Palavras-Chave: Simulated Annealing, Differential Evolution, Artificial Bee Colony, Particle Swarm Optimization.

Abstract: This paper compares different stochastic algorithms applied to benchmark function minimization problems, for which traditional mathematical techniques might fail. The stochastic algorithms analyzed in this work were the Artificial Bee Colony (ABC), Differential Evolution (DE), Particle Swarm Optimization (PSO) and Simulated Annealing (SA). Internal parameter settings of each algorithm were changed and its effects in the algorithms performance were evaluated and compared in minimization of six test functions (Ackley, Griewank, Parabolic, Rastrigin, Rosenbrock and Scheffers). The results allowed concluding about the importance of an appropriate definition of the internal parameter values of each algorithm. Besides, the minimization of the Rosenbrock function with high dimension was carried out with the best configuration of each algorithm. Based on a comparative analysis, the results showed that most of the algorithms could find global minima of the multimodal functions. However, the DE and ABC algorithms presented the best results in terms of convergence speed and quality of the results.

Keywords: Simulated Annealing, Differential Evolution, Artificial Bee Colony, Particle Swarm Optimization.

[1]Chemical Engineering Department, Federal University of Parana. E-mail: thiagoandrew@yahoo.com.br
[2]Chemical Engineering Department, Federal University of Santa Maria. E-mail: schwaab@smail.ufsm.br
[3]Chemical Engineering Department, Federal University of Santa Catarina. E-mail: diluccio@enq.ufsc.br
[4]Chemical Engineering Department, Federal University of Parana. E-mail: coraza@ufpr.br
[5]Chemical Engineering Department, Federal University of Santa Maria. E-mail: fernanda.castilhos@ufsm.br

1. INTRODUCTION

Optimization problems are present in several fields of expertise, like in mathematics, physics, statistics, chemistry, engineering and economy. With the development of high-speed computers, researchers have focused on solving harder optimization problems and, consequently, have developed optimization algorithms that are able to solve these hard problems. A hard optimization problem can be characterized as an optimization of a nonlinear objective function that depends on a high number of variables and presents a high number of local optima and function discontinuities. According to Singh et al. (2005), objective functions with these characteristics can be found in chemical and biochemical catalysis, molecular design, liquid-liquid extraction processes and polymerization reactors.

In this scenario, the use of the traditional optimization algorithms, such as Newton-based methods and the direct search methods, are discouraged (Schwaab et al, 2008). Although they are able to find a minimum, they do not assure that the global one is found. Furthermore, the traditional methods are dependent on a good initial guess of the optimum solution. In the case of Newton-based methods, the derivatives of the objective functions are necessary, constituting a strong limitation when dealing with discontinuous objective functions.

In order to overcome these difficulties, a class of numerical procedures called stochastic or population-based optimization algorithms has been developed (Kennedy and Eberhart, 1995; Corana et al, 1987; Price and Storn, 2005). The expression 'population-based' is used because some of these optimization procedures are executed with a set of possible solutions, instead of only one that converges to the optimum, like the traditional optimization methods. The expression stochastic is used due to the presence of a strong random component that associates an extensive number of objective function evaluations and a population-based search approach, introducing a global search characteristic, eliminating the need for good initial guesses of the optimal solution and increasing the probability of finding the global optimum. These algorithms also do not evaluate derivatives of the objective function and can be employed in the optimization of non-continuous objective functions without any additional effort.

The features of the stochastic optimization algorithms have gained a lot of attention and they have been used in many different optimization problems, such as dynamic optimization, multiobjective optimization, parameter estimation, nonlinear dynamic analysis of chemical processes, clustering analysis, constrained and global optimizations (Durand et al, 2009; Babu et al, 2005; Schwaab et al ,2008; Ourique et al, 2002; Karaboga e Ozturk, 2011; Karaboga e Akay, 2011).

Several stochastic optimization algorithms can be found in literature, such as Genetic Algorithm (GA), Simulated Annealing (SA), Particle Swarm Optimization (PSO), Differential Evolution (DE) and Artificial Bee Colony (ABC) (Holland, 1975; Kirkpatrick et al, 1983; Kennedy e Eberhart, 1985; Storn e Price, 1997; Karaboga e Basturk, 2007) . The number of available algorithms becomes very high if the modifications in the original algorithms (Karaboga e Akay, 2011; Tsallis e Stariolo, 1996; Chen e Chi, 2010; Liu e Sun, 2011), as well as the combination of different optimization algorithms (Da e Xiurun, 2005; Kao e Zahara, 2008) are considered.

Due to the high number of stochastic algorithms, a question that always arises is about the relative performance of such algorithms. The high number of stochastic algorithms is the first difficulty when making a comparison among them. A second difficulty is associated with the differing performance of the stochastic algorithms in different optimization problems, like problems with many local optima or problems with discontinuous objective function. It should also be emphasized that the stochastic algorithms contain some search parameters that must be well defined to allow a good optimization performance. Despite the fact that the influence on stochastic algorithms performance of the search parameter values

are rarely evaluated, it is important to point out that the proper definition of the search parameter values can contribute to the control of the efficiency and robustness of the optimization procedure, and an inappropriate definition of search parameter values can lead to very poor results.

In the literature, it can be found several works providing comparisons among different stochastic algorithms (Karaboga e Basturk, 2007; Karaboga e Basturk, 2008; Karaboga e Akay, 2009). More recently, Karaboga and Akay (2009) presented an assessment among four stochastic algorithms, that is, ABC, PSO, GA and DE, and they showed that the ABC algorithm led to better results and also presented fewer search parameters to be defined. However, only one set of search parameter values for each stochastic algorithm was used. Consequently, the search procedures cannot be precisely compared since better performances could be obtained with a different set of search parameter values for used in each stochastic algorithm.

In this work, a comparison among four stochastic optimization algorithms, ABC, DE, PSO and SA, is presented with six benchmark test functions. The main contribution of this work is the comparison of the algorithms with different parameters, which is unseen in the literature until now. The parameter values and some specific configurations of the algorithms were changed, in order to evaluate their effects on the performance of each algorithm, providing a general comparison of the algorithms. Also the number of optimization variables was varied, to show how the performance of each algorithm was influenced by the problem dimension. Therefore, the main contributions of this work are to assess on the performance of theses algorithms with different configurations and with high dimensional problems. In the following section, each one of the stochastic optimization algorithms is presented. In Section 3, the benchmark test functions are presented, and the optimization results are presented in Section 4. Finally, in Section 5, we present the conclusions on the obtained results.

2. STOCHASTIC ALGORITHMS

Among stochastic algorithms available in literature, four algorithms were used in this work and their characteristic are briefly described below. These algorithms were chosen since they are commonly used for solving optimization problems in the engineering field (Babu et al, 2005; Ourique et al, 2002; Li et al, 2000). Particle Swarm Optimization, Differential Evolution and Artificial Bee Colony are population-based algorithms where a solution set is interacted in order to find the global optimum. On the other hand, Simulated Annealing deals with only one solution that evolves along iterations. Despite this main difference, all these algorithms present stochastic features and carry out a global optimization.

2.1. SIMULATED ANNEALING

The concept of the simulated annealing is based on the behavior of a metal re-crystallization in the process of annealing. As the temperature decreases, the system becomes more ordered and approaches a “frozen” ground state at T equal to 0.

The Simulated Annealing is a stochastic algorithm with heuristic characteristics of random search, where the parameters are changed following probabilistic rules. It is important to say that several works are available in the literature, which proposed modifications to this algorithm originally reported by Kirkpatrick *et al*. (1983). Among them, the proposal of Li *et al.* (2000) is one of the most representative, where the reduction of the acceptance of degenerated states is associated to the temperature decrease.

The Simulated Annealing optimization algorithm is based on the Monte Carlo method and replaces the energy by a difference between objective function values at two sequential points, the current position $\mathbf{x}_k$ and the next solution candidate $\mathbf{x}_{k+1}$, where $\mathbf{x}$ is a vector that defines a particular position in the N-dimensional space of the optimization variables. Therefore, through this adaptation, the probability of passing from

the state $\mathbf{x}_k$ to a state $\mathbf{x}_{k+1}$, in a minimization problem, is defined as:

$$P(\mathbf{x}_k \rightarrow \mathbf{x}_{k+1}) = \begin{cases} 1 & \text{if } F(\mathbf{x}_{k+1}) \leq F(\mathbf{x}_k) \\ \exp\left(-\dfrac{F(\mathbf{x}_{k+1}) - F(\mathbf{x}_k)}{T_A}\right) & \text{if } F(\mathbf{x}_{k+1}) > F(\mathbf{x}_k) \end{cases} \quad (1)$$

where $F(\mathbf{x}_k)$ is the objective function at position $\mathbf{x}_k$ and T_A is the current annealing temperature. The probability calculated in Equation (1) is compared with a random number with uniform distribution in the range [0,1]. If the probability defined in Equation (1) is higher than *r*, the transition from $\mathbf{x}_k$ to $\mathbf{x}_{k+1}$ is accepted, despite the fact that the objective function can be increased during this transition. This behavior allows the optimization search to escape from local minima and to locate the global minimum of the objective function.

To induce convergence to a minimum value, the annealing temperature is decreased along the search process.The temperature reduction can be defined according to Equation (2).

$$T_A = \alpha T_A, \quad 0 < \alpha < 1 \quad (2)$$

Nevertheless, as a fast cooling of a metal results in a structure of higher energetic content, an inadequate variation in the annealing temperature (α value near 0) can result in a premature convergence to a local minimum. It is also important initiate the search from a sufficiently high initial temperature, in order to allow a good exploration of the solution-space problem.

In the original proposal (Kirkpatrick et al, 1983), the disturbances in the optimization variables are done by the following equation:

$$x_{k+1,j} = x_{k,j} + r_j \Delta x_j, \quad j = 1, ..., N \quad (3)$$

where r_j is a random number with uniform distribution in the range [-1, 1] and Δx_j is the increment to the j^{th} variable of the *N*-dimensional search space. Corana *et al*. (1987) presented a proposal where the disturbance size Δx_j should decrease in order to keep the ratio between accepted and rejected transition around 1, that is, about 50 % of the transitions should be accepted. According to this method, namely success ratio method, after *NS* iterations, the step Δx_j is updated as the following relation:

$$\Delta x_j = \begin{cases} \Delta x_j \left(1 + c\dfrac{q - 0.6}{0.4}\right) & \text{if } q > 0.6 \\ \Delta x_j & \text{if } 0.4 \leq q \leq 0.6 \\ \Delta x_j \left(1 + c\dfrac{0.4 - q}{0.4}\right)^{-1} & \text{if } q < 0.4 \end{cases} \quad (4)$$

where *c* is a constant (usually *c*is equal to 2) and *q* is the number ratio of accepted configurations to the generated configurations number in the last *NS* iteration. This method can also be applied in problems with domain restriction to every variable, where the step Δx_j is limited to $\beta\left(x_j^{high} - x_j^{low}\right)$, with $0 < \beta \leq 1$, that is limited to a fraction of the search space.

The pseudo-code used in this work to the Simulated Annealing algorithm is presented in the Appendix of this work and it is based on the works of Corana *et al*. (1987) and Goffe *et al*. (1994).

2.2.PARTICLE SWARM OPTIMIZATION

The Particle Swarm Optimization (PSO) algorithm, originally proposed by Kennedy and Eberhart (1985), is also a stochastic algorithm with heuristic features of random search, but that takes into account the best result generated in the current state and also the best global result. As it is a relatively recent algorithm, many reports have been pondered in the investigation and implementation of the algorithm, focusing in the process optimization and parameters estimation. Eberhart and Shi (2000) make a comparison between inertial weight and constriction factor. Brits *et al*. (2007) propose a strategy to PSO with an assured convergence. The investigated and implemented alternatives in the last years have attributed to PSO the potential to be applied in the global optimization in a wide diversity of problems.

In the PSO algorithm, each individual solution is named as particle and the solution population is called swarm.

According to Jiao *et al*. (2008), a particle *i* (*i* = 1,..,*NP*), in a iteration *k*, is characterized by two features: (1) its position, represented by the array $X_i^k = \left(x_{i,1}^k,\ldots,x_{i,N}^k\right)$ in the problem *N*-dimensional space limited according to $x_j^{low} \le x_j \le x_j^{high}$, where x_j^{low} and x_j^{high} are the bounds of the *j*search direction (*j* = 1,2,…,*N*); (2) its velocity, represented by the array $V_i^k = \left(v_{i,1}^k,\ldots,v_{i,N}^k\right)$ in this same *N*-dimensional space of the problem.

The position and pseudo-velocity of the particles are updated along the search iterations by the following equations:

$$V_{i,j}^{k+1} = wV_{i,j}^k + c_1 r_1\left(P_{i,j}^k - X_{i,j}^k\right) + c_2 r_2\left(P_{g,j}^k - X_{i,j}^k\right) \quad (5)$$

$$X_{i,j}^{k+1} = X_{i,j}^k + X_{i,j}^{k+1} \quad (6)$$

where r_1 and r_2 are random numbers uniformly distributed between [0,1] and w, c_1 and c_2 are search parameters, called inertial weight, cognitive and social parameters respectively, that must be properly defined in order to control convergence of the particles and, consequently, the efficiency and robustness of the algorithm. The term $c_1 r_1\left(P_{i,j}^k - X_{i,j}^k\right)$ represents the distance of the particle *i* to the best position reached by itself up to the *k*-th iteration. Similarly, the term $c_2 r_2\left(P_{g,j}^k - X_{i,j}^k\right)$ represents the distance of the particle *i* to the best position found by the whole group up to the *k*-th iteration. The performance of the PSO algorithm in solving optimization problems is closely related to the definition of the search parameters w, c_1 and c_2. These parameters can be defined as constants along the search iterations or as varying parameters as a function of iteration number. Nonlinear parameter estimation through PSO is presented by Schwaab et al. (2008).

The pseudo-code to the PSO algorithm implemented in this work is based on the work of Schwaab *et al.* (2008) and is presented in the Appendix.

2.3. DIFFERENTIAL EVOLUTION

Classic DE (Price et al, 2005) begins by initializing a population of *NP* points represented by *D*-dimensional vectors with parameter values that are distributed with random uniformity between pre-specified lower and upper parameter bounds (x_j^{low} and x_j^{high}), according to Equation (7):

$$\begin{aligned} &x_{i,j,g} = x_j^{low} + rand(0,1).\left(x_j^{high} - x_j^{low}\right), \\ &i = (1,2,...,NP),\ j = (1,2,...,N),\ g = 0 \end{aligned} \quad (7)$$

The subscript g is the generation (or iteration) index, *j* and *i* are the parameter and population index, respectively. Hence, $x_{j,i}$ is the j^{th} parameter of the i^{th} population vector. The random number generator *rand*(0,1) returns a uniformly distributed value in the range [0, 1]. In DE, parameter values are encoded as ordinary floating-point numbers and are manipulated with standard floating-point operators like those available in high level languages like C and FORTRAN (Price et al, 2005).

To generate a trial solution, DE first mutates a vector form of the current population. Mutation operation can be performed through different strategies, according to Equations (8a-e).

$$\mathbf{v}_{i,g} = \mathbf{x}_{best,g} + F\left(\mathbf{x}_{p_1,g} - \mathbf{x}_{p_2,g}\right) \quad (8a)$$

$$\mathbf{v}_{i,g} = \mathbf{x}_{p_3,g} + F\left(\mathbf{x}_{p_1,g} - \mathbf{x}_{p_2,g}\right) \quad (8b)$$

$$\mathbf{v}_{i,g} = \mathbf{x}_{i,g} + F\left(\left(\mathbf{x}_{best,g} - \mathbf{x}_{i,g}\right) + \left(\mathbf{x}_{p_1,g} - \mathbf{x}_{p_2,g}\right)\right) \quad (8c)$$

$$\mathbf{v}_{i,g} = \mathbf{x}_{best,g} + F\left(\left(\mathbf{x}_{p_1,g} - \mathbf{x}_{p_2,g}\right) + \left(\mathbf{x}_{p_3,g} - \mathbf{x}_{p_4,g}\right)\right) \quad (8d)$$

$$\mathbf{v}_{i,g} = \mathbf{x}_{p_5,g} + F\left(\left(\mathbf{x}_{p_1,g} - \mathbf{x}_{p_2,g}\right) + \left(\mathbf{x}_{p_3,g} - \mathbf{x}_{p_4,g}\right)\right) \quad (8e)$$

Vector indices p_1, p_2, p_3, p_4, and p_5 are randomly selected from the *NP* solution candidates, except those whose indices are distinct and different from the population index *i*, that is, $p_1 \neq p_2 \neq p_3 \neq p_4 \neq p_5 \neq i$. The *mutation scale factor, F,* is a positive real number that is typically between 0 and 1.0.

Next, one or more parameter values of this *mutant vector*, $\mathbf{v}_{i,g}$, are uniformly crossed with those belonging to the i^{th}

population vector, $\mathbf{x}_{i,g}$, (the *target vector*). The result is the *trial vector,* $\mathbf{u}_{i,g}$

$$u_{j,i,g} = \begin{cases} v_{j,i,g} \text{ if } rand(0,1) \le Cr \text{ or } j = j_{rand}; \\ x_{j,i,g} \text{ otherwise} \end{cases} \quad (9)$$
$$j_{rand} \in \{1,2,...,N\}.$$

The *crossover constant*, $0.0 \le Cr \le 1.0$ controls the parameters fraction that the mutant vector contributes to the trial vector. In addition, the trial vector always inherits the mutant vector parameter with the randomly chosen index j_{rand} to ensure that the trial vector differs by at least one parameter from the vector with which it will be compared (that is, the target vector, $\mathbf{x}_{i,g}$).

To keep solutions feasible when problems are bound-constrained, trial parameters that violate boundary constraints are set back to the violated bound, according to Equation (10).

$$u_{j,i,g} = \begin{cases} u_{j,i,g} = x_j^{low} & \text{if } u_{j,i,g} < x_j^{low} \\ u_{j,i,g} = x_j^{high} & \text{if } u_{j,i,g} < x_j^{high} \end{cases} \quad (10)$$

If the trial vector's function value is less than or equal to that of the target vector, the trial vector replaces the target vector in the next generation. Otherwise, the target vector remains in the population for at least one more generation.

$$\mathbf{x}_{i,g+1} = \begin{cases} \mathbf{u}_{i,g} & \text{if } f\left(\mathbf{u}_{i,g}\right) \le \left(\mathbf{x}_{i,g}\right) \\ \mathbf{x}_{i,g} & \text{otherwise} \end{cases} \quad (11)$$

In DE algorithm, the selection is global in the sense that the base vector is any randomly chosen population vector other than the target vector. Mutation operation strategies defined in Equations (8b) and (8e) are global strategies. Global selection allows underperforming population vectors to be replaced with variants of the population's better solutions. If, however, the base and target vectors are the same, as in mutation operation strategy defined in Equation (8c), then the selection is *local* in the sense that each vector in the current population is compared to its own mutant. When the selection is local, a vector's evolution only depends on the current set of vector *differences* and not directly on the parameter values of vectors other than those of the target. In effect, local selection partitions the population into *NP* niches, each of which is inhabited by a single vector that evolves in isolation without the benefit of access to the population's better solutions.

2.4. ARTIFICIAL BEE COLONY

The Artificial Bee Colony algorithm (ABC) is an optimization algorithm based on the peculiar intelligence of a bee swarm. In the ABC algorithm, the artificial bee colony has three bee groups: employer, scout and onlooker bees. The first half of the colony consists of employer bees and the second half of onlooker bees. For every food source, there is only one employer bee. In others reports, the number of employer bees is equal to the food sources. The employer bee of a discarded food source becomes a scout bee (Karaboga e Basturk, 2008). The search performed by artificial bees can be described as:

- Employer bees establish a food source close to its neighborhood in their memories.
- Employer bees share their information with the neighborhood of food sources in their memories.
- Onlooker bees chose a food source in the neighborhood of food sources chosen by them.
- An employer bee whose source was discarded becomes a scout bee and the random search starts for a new food source.

Each sequence of the search consists of three steps: to move the onlooker and employer bees to the food sources, to evaluate their nectar amounts, to determine the scout bees and then to move them on the food sources in a random way. A food source represents a possible solution to the problem to be optimized. The nectar amount of a food source corresponds to the solution quality represented by that food source. The employer bees are placed on the food using the wheel roulette selection method. Each bee colony has scout bees that walk around the colony. They are

firstly assigned to find any food source. As a result of their behavior, the scout bees are characterized by the low cost of search and a low quality of the food sources. Sometimes, the scout bees can accidentally find out rich food sources and completely unknown. In the case of artificial bees, the artificial scout could have a fast discovery of the viable solution group as a task. In the ABC algorithm, one of the employer bees is selected and classified as a scout bee. The classification is controlled by a control parameter called by "limit". If a solution that represents a food source is not improved in a specific number of trials, then this food source is left by the employer bee and this employer bee becomes a scout bee. The number of trials is the same as the limit value, which is an important control parameter of the ABC algorithm (Karaboga e Basturk, 2008).

As others social foragers, the bees search for food sources in a way that maximizes the rate E/T (where E is the obtained energy and T is the time spent to the feeding). In the case of bee swarm, E is proportional to the nectar amount of the food sources found out by the bees and the bee swarm works to maximize the honey stored in the hive. In a maximization problem, the key point is to find the maximum of the objective function $F(\theta)$, $\theta \in R^p$ .If θ_i is the position of the i-th food source, $F(\theta_i)$ represents the nectar amount of the food source placed in θ_i and is proportional to the energy $E(\theta_i)$.The term $P(c)=\{\theta_i(c)|i=1,2,...,S\}$ (c: cycle, S: number of food source around the hive) represents the population of food sources that are being visited by the bees (Karaboga e Basturk, 2008).

As it was mentioned previously, the preference for a food source by an employer bee depends on the nectar amount $F(\theta)$ present in this food source. The nectar amount of a food source increases the probability of this source to become the favorite source by an employer bee in a proportional manner. Then, the probability of a food source placed in θ_i is chosen by a bee can be expressed as:

$$P_i = \frac{F(\theta_i)}{\sum_{k=1}^{S} F(\theta_k)} \qquad (14)$$

After observing the employer bees movement, an onlooker bee selects a food source based on a comparison among the food sources around θ_i . The position of the neighbor food source selected is evaluated as:

$$\theta_i(c+1) = \theta_i(c) \pm \phi_i(c) \qquad (15)$$

where $\phi_i(c)$ is a random step used to find a food source with more nectar around θ_i . $\phi(c)$ is evaluated with the difference of the same parts of $\phi_i(c)$ and $\phi_k(c)$ (k is a random parameter produced) food positions. If the nectar amount $F(\theta_i(c+1))$ in $\theta_i(c+1)$ is higher than in $\theta_i(c)$, than the bee goes to the hive and shares its information with the others and the food source position $\theta_i(c)$ is changed to $\theta_i(c+1)$, however $\theta_i(c)$ is preserved.

3. NUMERICAL EXPERIMENTS

In order to evaluate the performance of stochastic algorithms, some classical benchmark functions were tested. The expression, parameter search ranges and dimension of each benchmark function are presented in Table 1. All benchmark functions have zero as global optimum value. Parabolic function is continue, convex and unimodal. Rosenbrock function is commonly used in problems of function minimization, since it presents a high difficulty in convergence due to its gradients do not point to the optimum and also to the interdependence of its variables.

Table 1. Benchmark functions which the search range, minimum value and problem dimension.

Function name	Function equation	Search Range	Problem Dimension
Parabolic	$f_1(x) = \sum_{i=1}^{n} x_i^2$	$-100 \le x_i \le 100$	30
Rosenbrock	$f_2(x) = \sum_{i=1}^{n-1} 100\left(x_{i+1} - x_i^2\right)^2 + \left(x_i - 1\right)^2$	$-50 \le x_i \le 50$	50
Rastringin	$f_3(x) = \sum_{i=1}^{n} \left(x_i^2 - 10\cos\left(2\pi x_i\right) + 10\right)$	$-5.12 \le x_i \le 5.12$	50
Griewank	$f_4(x) = \frac{1}{400}\left(\sum_{i=1}^{n}(x_i - 100)^2\right) - \left(\prod_{i=1}^{n}\cos\left(\frac{x_i - 100}{\sqrt{i}}\right)\right) + 1$	$-600 \le x_i \le 600$	50
Schaffer	$f_5(x) = 0.5 + \frac{\sin^2\left(\sqrt{x_1^2 + x_2^2}\right) - 0.5}{\left(1 + 0.001\left(x_1^2 + x_2^2\right)\right)^2}$	$-100 \le x_i \le 100$	2
Ackley	$f_6(x) = 20 + e - 20e^{\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right)} - e^{\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)}$	$-30 \le x_i \le 30$	30

Rastringin, Schaffer and Griewank functions present many local minima that are regularly distributed. Ackley function presents many local minimum due to a exponential term.

To assess the influence of the parameters of the stochastic algorithms used in this study, different configurations for each algorithm were used, as described below. The population number was also varied for all the algorithms, to assess its influence in their performance. For all the algorithms, a stopping criterion (maximum evaluation number) of 500,000 evaluations of the objective function was utilized.

According to Table 2, the ratio between the number of onlooker and employers bees R and presence (wScout) or absence (wtScout) of scout bees were evaluated in this study for the ABC algorithm. All configurations were evaluated for a population size (Nbee) of 20, 60 and 100 and the limit parameter was set as the number of employers bees times the problem dimension.

In DE algorithm, the scale mutant factor F is a real constant, which affects the differential variations between two solutions. Higher F means larger and smaller exploitation ability. Therefore, a value of F which is too small can cause the population to converge to the minimum too soon and it also causes the difficulty to jump out of a local minimum.

Table 2. Evaluated configurations for the ABC algorithm.

Configuration Number	Evaluated Parameters	
	R	Scout
1	0.5	wScout
2	0.5	wtScout
3	1.0	wScout
4	1.0	wtScout
5	2.0	wScout
6	2.0	wtScout

So, as suggested by Wang and Huang (2010), F was equal to 0.5 in this study. The crossover rate constant value (CR), which controls the change of the population diversity, was set to 0.8. It is used to adjust the weight between the history and the current selection operation. Higher CR may fasten the convergence speed and make the evolution process be trapped in a local minimum (Wang e Huang, 2010). In this study, all configurations were evaluated for a population size (NP) of 20, 50 and 100. Different mutation strategies, as it can be

seen in Table 3, were evaluated for this algorithm.

Table 3. Evaluated configurations for the DE algorithm

Configuration Number	Mutation Strategy
1	Equation (8a)
2	Equation (8b)
3	Equation (8c)
4	Equation (8d)
5	Equation (8e)

For the SA algorithm all configurations were evaluated for a population size (NT) of the 20 and 100. The temperature reduction rate α and the annealing temperature T_A were assessed according to Table 4. Both these parameters are relevant in the SA algorithm performance. The annealing temperature influences the probability of acceptance of a new state, as it can be seen in equation (2). As it was pointed out in section 2.1, a higher temperature reduction rate can make the convergence faster but sometimes it can result in a premature convergence to a local minimum.

Table 4. Evaluated configurations for the SA algorithm.

Configuration Number	Evaluated Parameters	
	α	T_A
1	0.50	0.1
2	0.50	10
3	0.50	1000
4	0.85	0.1
5	0.85	10
6	0.85	1000

In PSO algorithm, the cognitive (C_1) and social (C_2) components are constants that can be used to change the influence between the individual and population experience, respectively (Kennedy and Eberhart, 1995). Inertia weight is used to determine how the previous velocity of the particle influences the velocity in the next iteration. Since the performance of the algorithm is strongly influenced by its values, these parameters were chosen to be investigated in this work. A list of the different configurations tested for this algorithm can be found in Table 5. All configurations were evaluated with a particle number (Npt) of 20, 60 and 100.

Each algorithm configuration was simulated 40 times and in order to compare them, the median of objective function values was used since it is statistically more robust than the mean value, when some outliers values are present.

Table 5. Evaluated configurations for the PSO algorithm.

Configuration Number	Evaluated Parameters	
	W	C_1,C_2
1	0.60	1.0
2	0.60	1.5
3	0.60	2.0
4	0.75	1.0
5	0.75	1.5
6	0.75	2.0
7	0.90	1.0
8	0.90	1.5
9	0.90	2.0

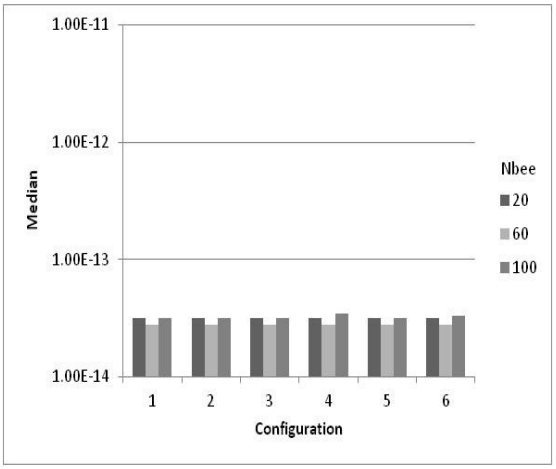
3. RESULTS AND DISCUSSION

This section presents the minimization results obtained with the four stochastic algorithms used with different parameter configurations, followed by a comparison among them with their best parameter values. Finally, performance algorithms in the minimization of a high dimension problem is also presented and discussed.

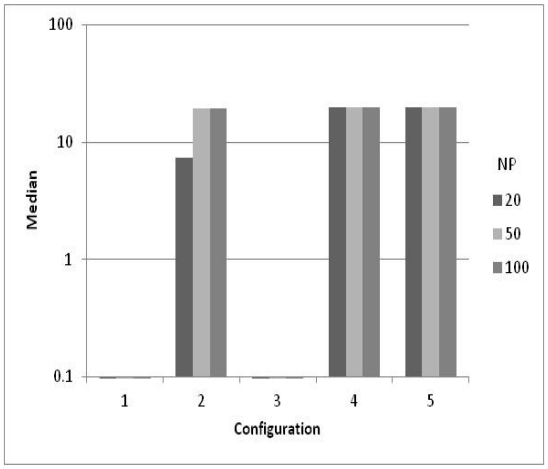
3.1 BENCHMARK FUNCTION RESULTS

Figure 1 shows the results of the stochastic algorithms studied for the optimization of the Ackley function. The ABC algorithm achieved median values for the objective function near to zero for all their configurations, independent of the initial population. It may be an indicative that random initial population, the algorithm is able to find the global minimum. For the DE algorithm, the strategies 1 and 3 achieved the best values, reaching values equal to zero for the

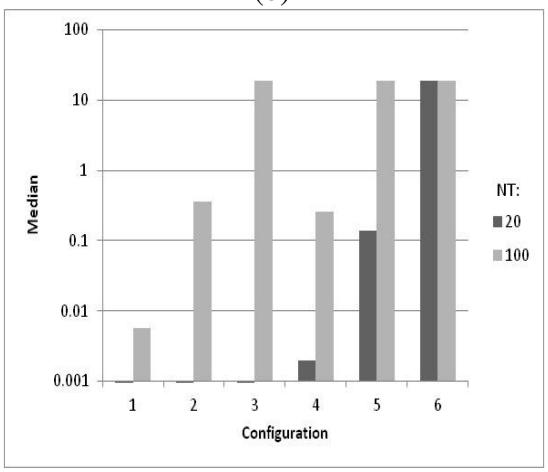
objective function. The minimization with other strategies showed worse results, independent of the population number. This can be due to a greater influence of mutation strategy compared to the population number. The SA algorithm with configurations 1, 2 and 3 showed satisfactory results with NT=20. However, with NT=100, the objective function values were significantly higher. In configuration 4, with NT=20, values between $1x10^{-1}$ and $1x10^{-2}$were achieved; while for NT=100, the objective function values were more distant from the global optimum (zero). The results obtained in configurations 5 and 6 were worse than the other configurations. It can be pointed out that a lower α (configuration 1, 2 and 3) provided better results. However, for high values of T_A and α, the algorithm faced difficulties to converge to the global optimum. In a general way, we can say that the temperature reduction rate was more influent in the SA performance in the minimization of the Ackley function. For the PSO algorithm, the configurations 3 and 7 provided values of the objective function equal to zero. With configuration 5, the global optimum was achieved with Npt = 60 and Npt = 100. It can be noted that the values reached with configurations 3 ,5 and 7 are associated with a balance between the C_1, C_2 and w parameters of the algorithm, that is, increasing the value of w would be associated with a decrease in the values of C_1 and C_2.



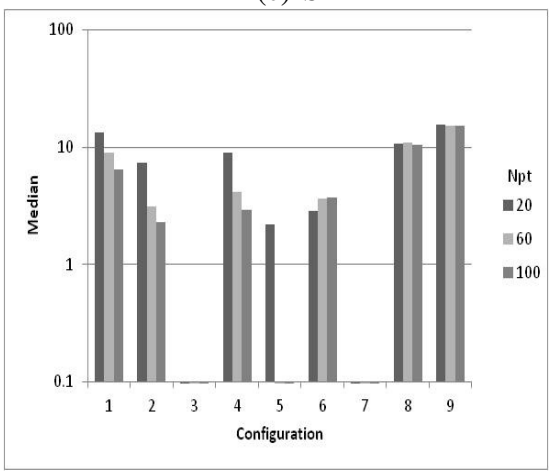
(a) ABC



(b) DE



(c) SA

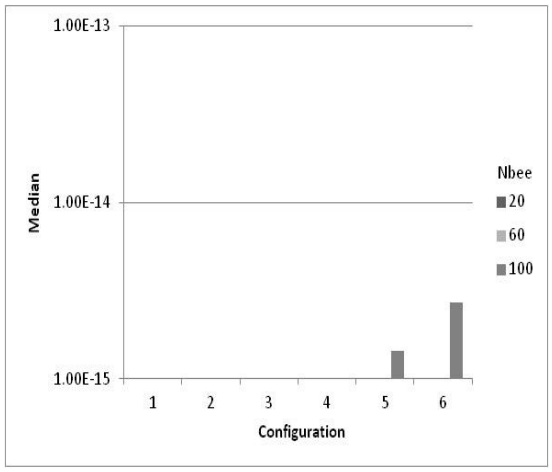


(d) PSO

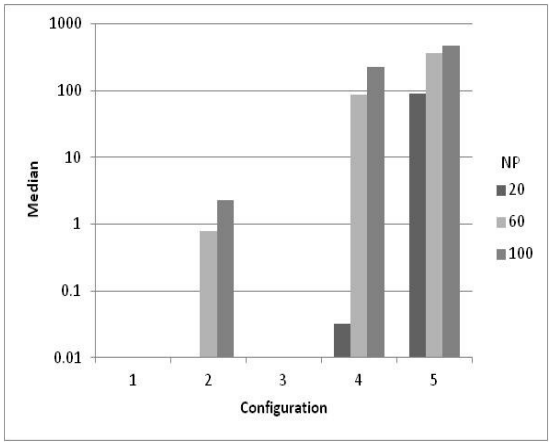
Figure 1. Ackley function median for the stochastic algorithms: (a) ABC; (b) DE; (c) SA; (d) PSO.

The results of the stochastic algorithms studied for optimization of the Griewank function are presented in Figure 2. It can be verified that the ABC algorithm reached objective function values equal to zero for all its configurations, with only a little difficulty in convergence with a population number of 100. The DE

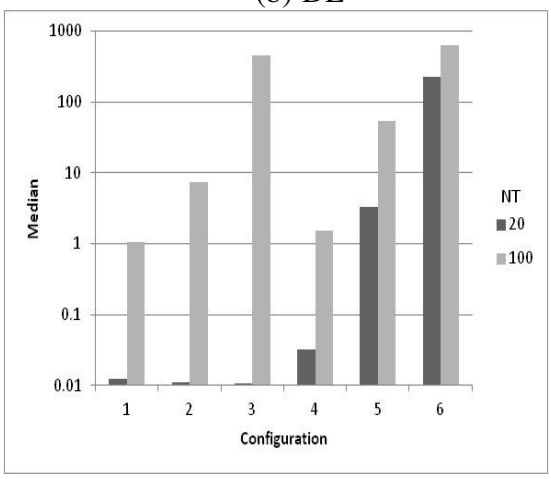
algorithm with the mutation strategies 1 and 3 achieved the global minimum of the function. The configurations 2, 4 and 5 provided worse results. In all of these cases, it can be said that the DE performance was generally disfavored by the population number, since in order to kept total number of function evaluations constant, the number of iterations is decreased. Consequently, it seems that an increase of the number of iterations provides better results than an increase of population number. For the SA algorithm, the best performances were obtained with configurations 1, 2 and 3, which corresponds to the lowest value of the temperature reduction parameter α. This can be explained by the fast cooling associated with the low value of this parameter, a similar result obtained with the Ackley function. It can also be noted from Figure 2 that for NT = 100, the minimum achieved by this algorithm is higher than with the annealing temperature. The results obtained with the PSO algorithm for the Griewank function are similar to the results obtained with the Ackley function. The best results for PSO were obtained with configurations 3, 5 and 7. It is worth to point out that the best performance was reached with a lower inertial weight factor and high values for the cognitive and social parameters, which means that the individual and collective memory of the particles seems to be more relevant than the influence of the previous velocity.



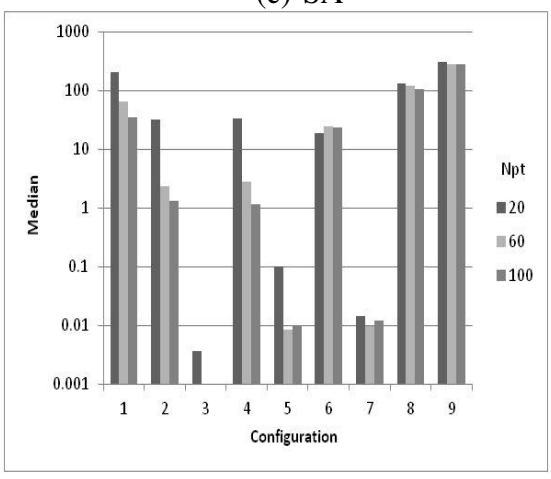
(a) ABC



(b) DE



(c) SA

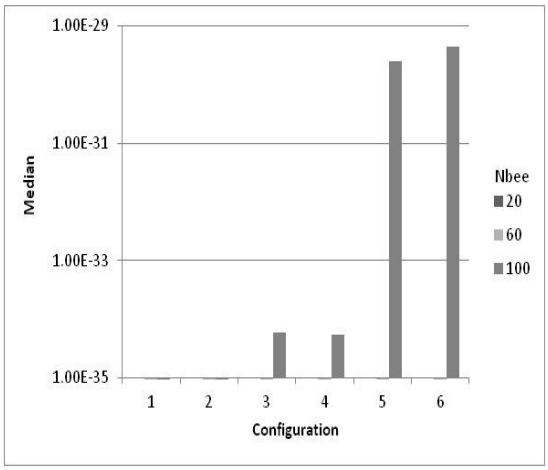


(d) PSO

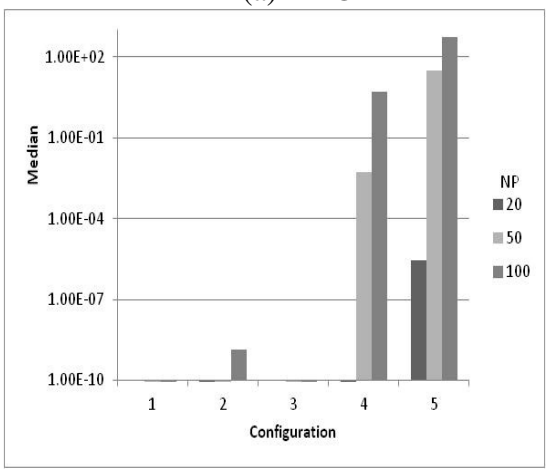
Figure 2. Griewank function median for the stochastic algorithm a) ABC; b) DE; c) SA; d) PSO with different configurations and population number.

Figure 3 illustrates the results for the use of the stochastic algorithms for optimization of the parabolic function. The results obtained with ABC algorithm were near to zero for all configurations tested. It can only be noted from Figure 2 that for some configurations, the population size of 20 provided the worst results. The best

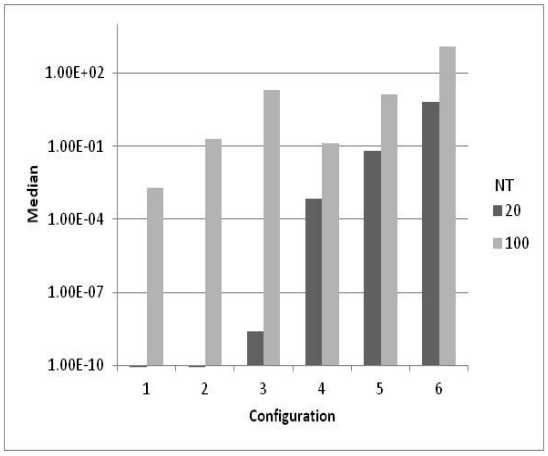
performance of DE algorithm was verified in configurations 1 and 3. The influence of the population number can be seen in configuration 5 that presented the highest value of the objective function median. For the SA algorithm, the results indicate that the results are improved with a low α.However, when T increases, the results worsen. This is the same behavior for the Ackley function. With better values of α, the efficiency rises while the robustness decreases, even though the function presents only one minimum. As it can be noted from the previous Figures, the configurations for the PSO algorithm that provided the best results were the configurations 1, 3 and 7.



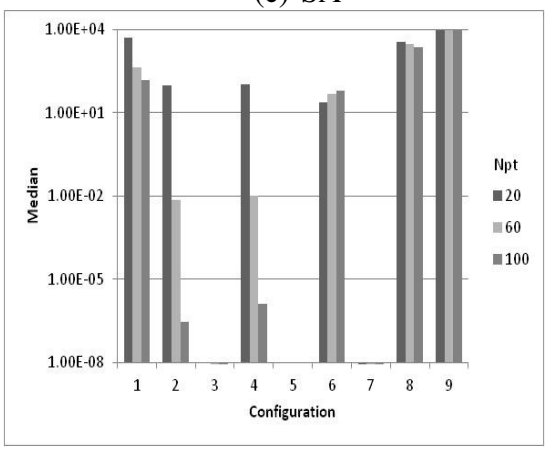
(a) ABC



(b) DE



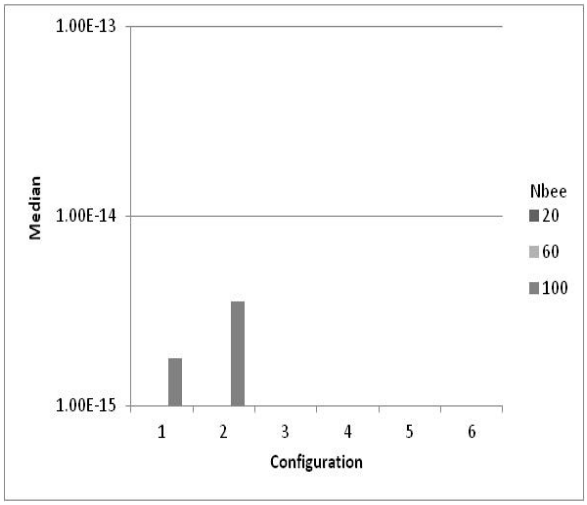
(c) SA



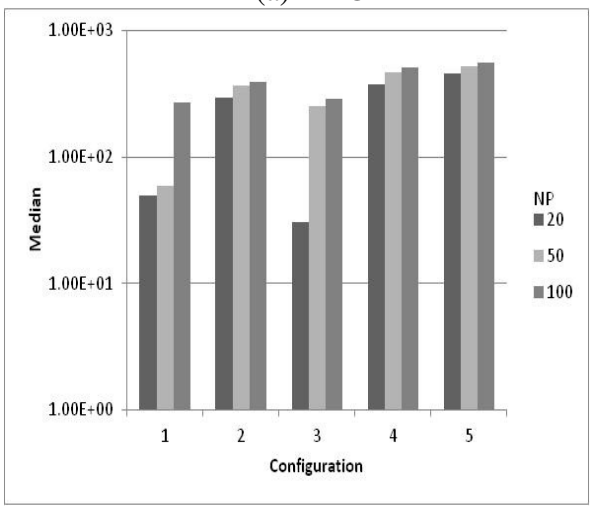
(d) PSO

Figure 3. Parabolic function median for the stochastic algorithm a) ABC; b) DE; c) SA; d) PSO with different configurations and population number.

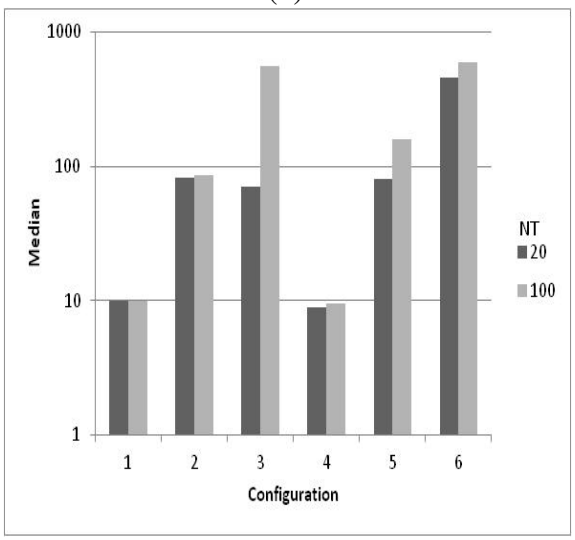
Figure 4 shows the results of the stochastic algorithms studied for the optimization of Rastrigin function. The ABC algorithm was the only one that achieved the global minimum of this function, which is hardly multimodal. As the results obtained in the previous functions, the DE algorithm presented better results with the mutation strategy 1 and 3. But, this algorithm presented more difficulty in the minimization of the Rastrigin function, resulting in a worse performance compared to the other functions. The performance of SA and PSO algorithms was significantly poor. However, it should be emphasized that the influence of the annealing temperature was higher than the temperature reduction factor for this function, which was the opposite behavior compared to the one observed in the previous benchmark functions.



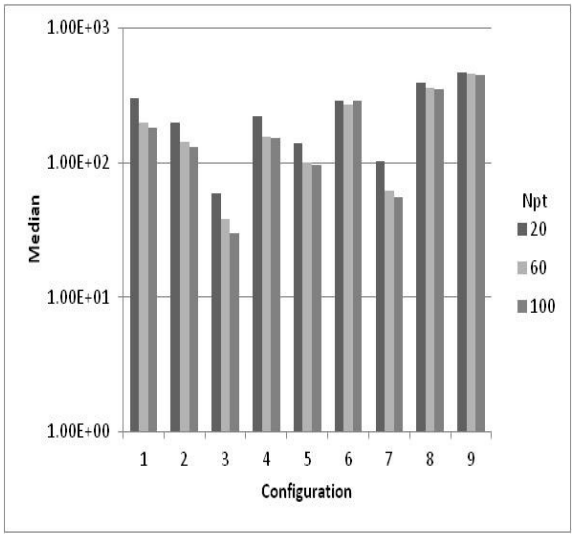
(a) ABC



(b) DE



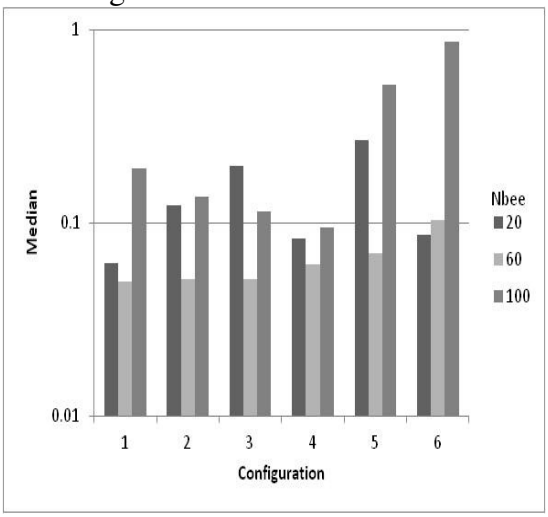
(c) SA



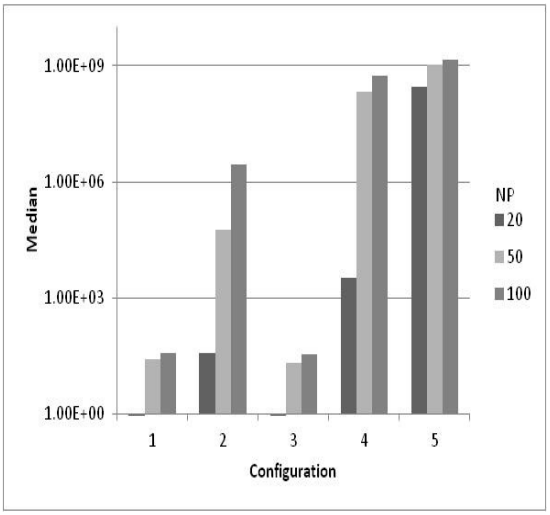
(d) PSO

Figure 4. Rastrigin function median for the stochastic algorithm a) ABC; b) DE; c) SA; d) PSO with different configurations and population number.

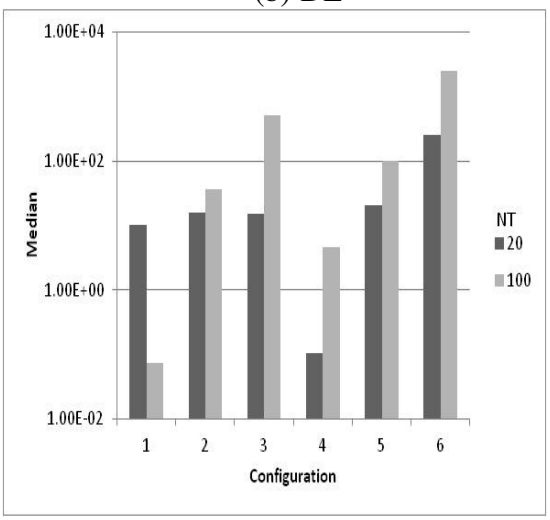
The results for the Rosenbrock function are shown in Figure 5. The ABC algorithm showed a good performance for all configurations. It should be pointed out that the population size of 60 used in this work achieved a better result than Karaboga and Basturk (2008) with a population size of 100. The better results obtained with the DE algorithm were with the configurations 1 and 3, with the population size of 20, reaching the global minimum of this function. The SA algorithm showed similar performance to the previous functions. With high values of α and low values of T_A the results were satisfactory. The results of the PSO algorithm for all configurations were not good.



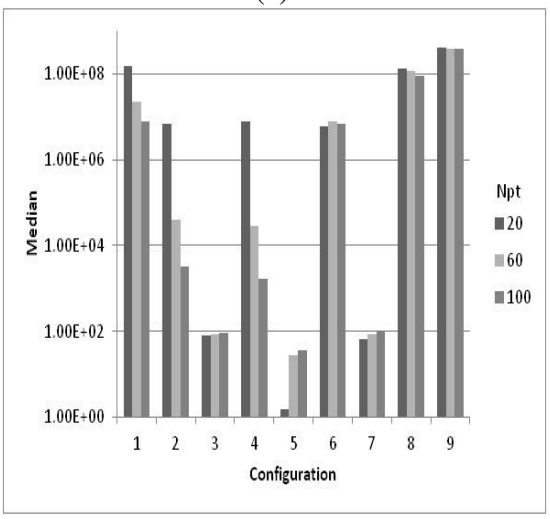
(a) ABC



(b) DE



(c) SA

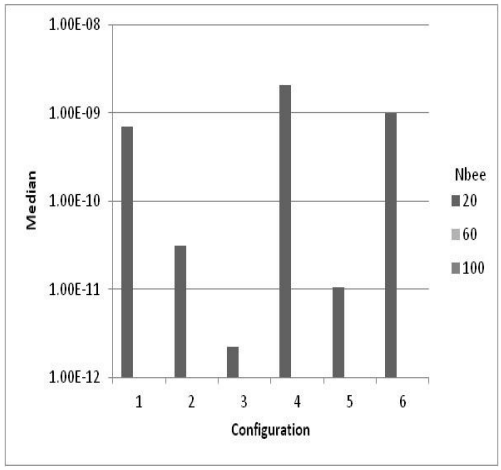


(d) PSO

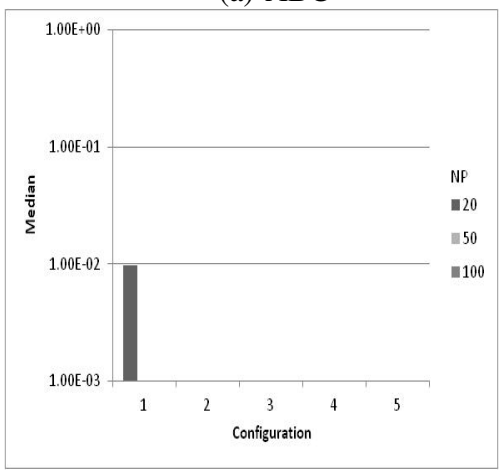
Figure 5. Rosenbrock function mean for the stochastic algorithm a) ABC; b) DE; c) SA; d) PSO with different configurations and population number.

Figure 6 shows the results of the stochastic algorithms studied for optimization of the Scheffer function. For the ABC, DE and PSO algorithms, almost all configurations reached the global minimum of this function. Only the SA algorithm was trapped in a local minimum for all of its configurations.

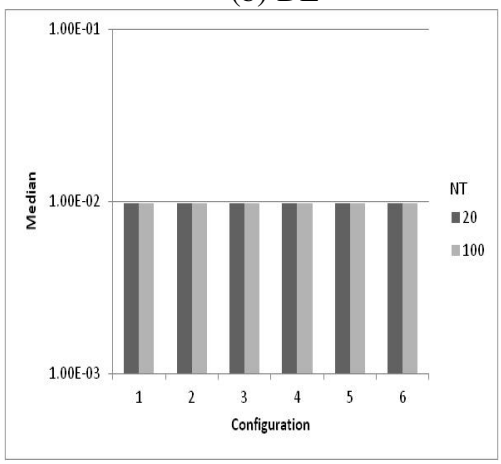
It can be pointed out that the algorithm performances are dependent on their parameters. If adequately used, they are generally able to reach the global minimum, independent of the benchmark function. The ABC and DE algorithms showed to be more robust and efficient than the others (SA and PSO).



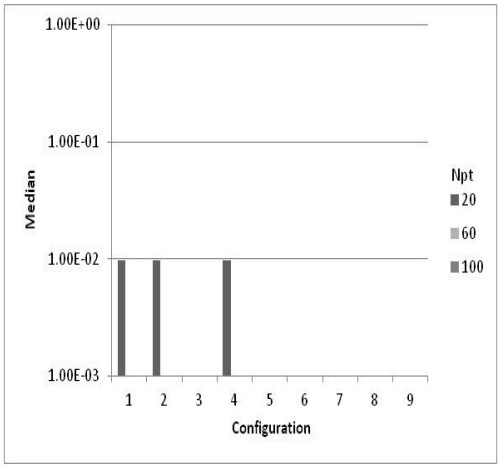
(a) ABC



(b) DE



(c) SA



(d) PSO

Figure 6. Scheffer function median for the stochastic algorithm a) ABC; b) DE; c) SA; d) PSO with different configurations and population number.

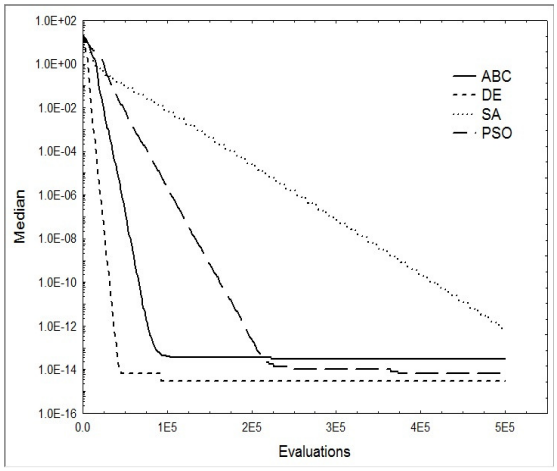
3.2 COMPARISON OF THE STOCHASTIC ALGORITHMS

As it can be seen in Figures 1-6, the performance of the algorithms configurations was similar for all the benchmark functions tested. From this observation, it was possible to determine the best configurations of the stochastic algorithms studied, which are listed in Table 6.

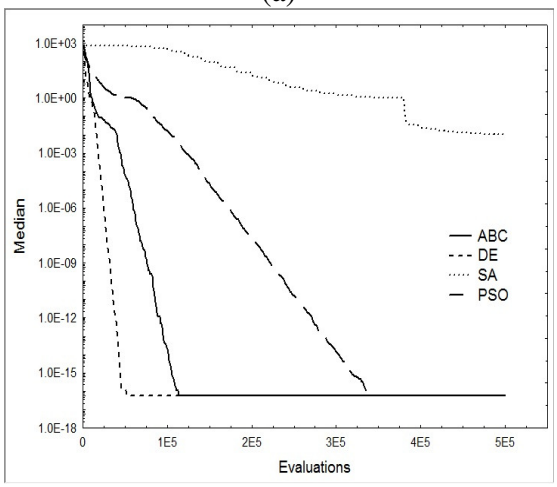
Table 6. Best configurations of the studied stochastic algorithms.

Algorithm	Best Configurations	
	Configuration Number	Population Size
ABC	3	60
DE	3	50
SA	3	20
PSO	5	60

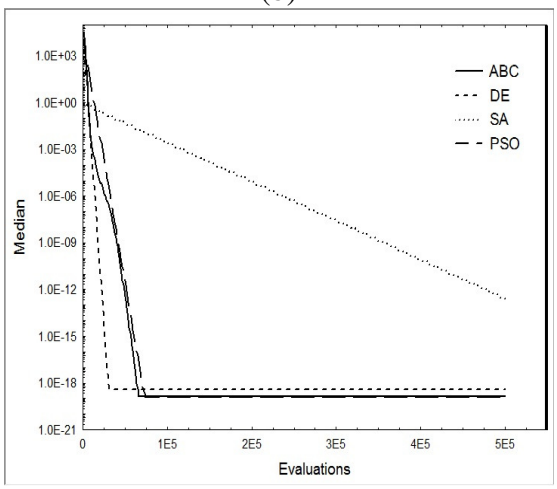
In order to make a more systematic comparison, the performance of the algorithms was compared, for each benchmark function, with a stop criterion of 500 000 evaluations of the objective function. The results of the median values of the objective function as a function of the number of evaluations are shown in Figure 7.



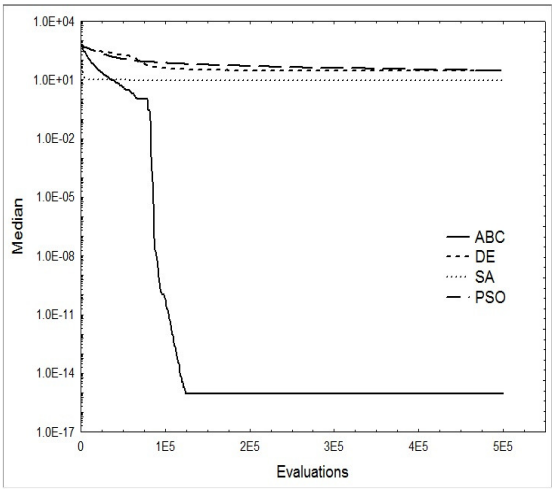
(a)



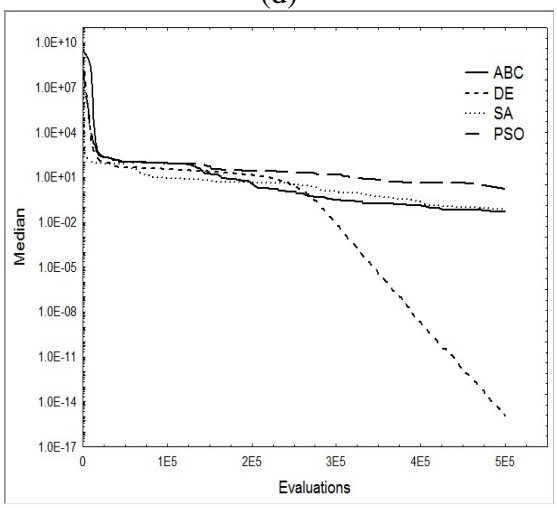
(b)



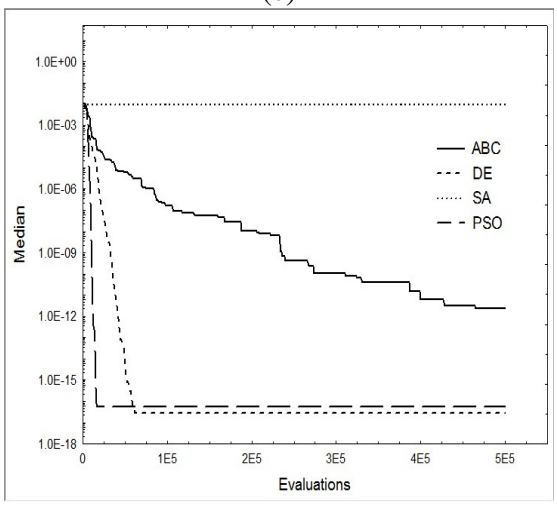
(c)



(d)



(e)



(f)

Figure 7. Profiles of the objective function minimization with the best configurations of stochastic algorithms with a) Ackley; b) Griewank; c) Parabolic; d) Rastrigin; e) Rosenbrock and f) Scheffer.

It can be seen from Figure 7a that DE algorithm could reach global optimum in less iterations compared to the other algorithms and that SA algorithm could not achieved the global optimum. A similar behavior can be observed Figures 7 (b) and (c). In Figure 7(d), PSO algorithm presented a better performance compared to the other algorithms, both in the number of iterations to achieve the global optimum and in the final value of objective function. The results in Figure 7(e) show that DE algorithm outperformed on the other algorithm in a similar manner. In Figure 7 (f), SA algorithm presented great difficult to minimize Scheffer function, while PSO and DE algorithms performed better.

Therefore, in a general way, as it can be seen in the Figure 7, the stochastic algorithm with best performance with the benchmark functions tested was the Differential Evolution algorithm with the mutation strategy 3 and population size equal to 50. It should be noted that only the DE algorithm was able to minimize the Rosenbrock function to the global minimum, as it is shown in Figure 7e. However, it is also worth to point out that the ABC algorithm showed a good ability to reach the global minimum of multimodal functions, like the Rastrigin function, while the other algorithms could not achieve the global minimum. The PSO algorithm reached good results when used with its best configuration, also showing ability to achieve the global minimum. The Simulated Annealing algorithm exhibited the worst performance, since it could only converge to the global minimum when the Ackley function was tested. From the results discussed above, the Rosenbrock function presented more difficulty to the evaluated algorithms, since only DE algorithm could achieve a value very close to the global optimum.

3.3 EVALUATION OF THE STOCHASTIC ALGORITHMS AT HIGH DIMENSIONAL PROBLEMS

To assess the performance of the stochastic algorithms in high dimension problems, the Rosenbrock function with dimension of 100 and 500 was minimized with the four algorithms investigated in this work, with their best configurations. A stopping criterion of 5,000,000 objective

function evaluations was utilized for all the minimizations. The variation of the objective function median as a function of the number of evaluations for the different dimensions tested is shown in Figures 8 and 9.

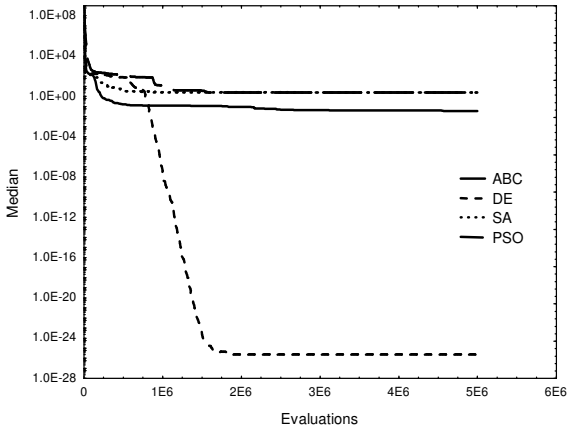


Figure 8. Profile of the Rosenbrock function minimization with dimension of 100.

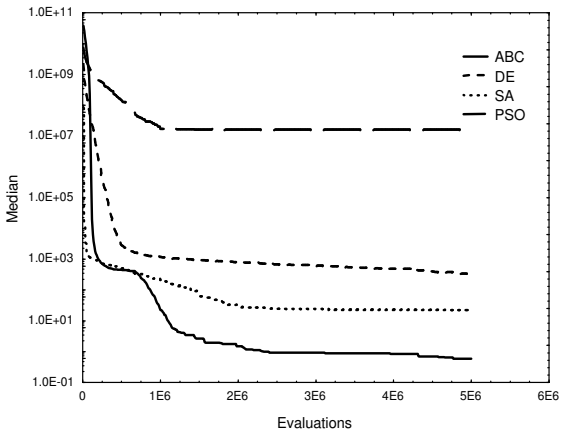


Figure 9. Profile of the Rosenbrock function minimization with dimension of 500.

The DE algorithm did not have difficulties in achieving the global minimum of the Rosenbrock function with dimension equal to 100, and it needed approximately 1,500,000 objective function evaluations to reach a 10^{-26} median value. The ABC algorithm reached values in order of 10^{-2}. The PSO and SA algorithms, even after the five million of evaluations of the objective function, failed to converge to the global minimum of the function.

Moreover, it can be seen in Figure 9 that all the stochastic algorithms failed to converge for the global minimum of the Rosenbrock function with dimension of 500. The only algorithm that could reach objective function values below $1x10^{0}$ was the ABC algorithm. Therefore, this algorithm showed greater potential to be successfully used in complex and multimodal problems, with high dimensions.

4. CONCLUSIONS

In this paper, four stochastic algorithms were implemented and evaluated: DE, PSO, SA, and ABC. Classical test functions presented in the literature were used to evaluate the performance of algorithms at different configurations.

In general, the performance of all the algorithms was satisfactory. The stochastic global optimization methods were excellent tools in minimizing unimodal and multimodal functions, presenting easy implementation with a few internal parameters to change and robustness.

The DE algorithm stands out for its superior performance when compared to the other algorithms. The results showed that there is a strong dependence on the mutation strategy, which is the main parameter procedure of this algorithm. The ABC algorithm was an efficient method of minimization of functions, although less efficient than the DE algorithm. It could minimize the Rastrigin function with only a few iterations. The main advantage is that it does not need configuration of too many internal parameters. In general, the SA and PSO algorithms presented worse performances compared to DE and ABC algorithms.

In relation to the high dimension problems, the ABC algorithm showed a higher probability of being successfully used in multimodal and complex problems of large dimensions, such as problems associated with dynamic optimization.

5. ACKNOWLEDGMENTS

The authors would like to thank CAPES, CNPq and FAPERGS for financial support and scholarships.

REFERENCES

BABU, B. V., CHAKOLE, P. G. & SYED MUBEEN, J. H. 2005. Multiobjective differential evolution (MODE) for optimization of adiabatic styrene reactor. *Chemical Engineering Science*, 60, 4822-4837.

BRITS, R., ENGELBRECHT, A.P., VAN DEN BERGH, F. 2007. Locating multiple optima using particle swarm optimization. *Applied Mathematics and Computation,* 189, 1859-1883.

CHEN, T.Y., CHI, T.M. 2010. On the improvements in the particle swarm optimization algorithm. *Advances in Engineering Software*, 41, 229-239.

CORANA, A., MARCHESI, M., MARTINI, C., RIDELLA, S. 1987. Minimizing Multimodal Functions of Continuous Variables with the Simulated Annealing Algorithm. *ACM Transactions on Mathematical Software*, 13, 262-280.

DA, Y., XIURUN, G. 2005. An improved PSO-based ANN with simulated annealing technique. *Neurocomputing*, 63, 527-533.

DURAND, G.A., CORAZZA, M. L., BLANCO, A.M., CORAZZA, F.C. 2009. Dynamic Optimization of the Mashing Process. *Food Control*, 29, 1127-1140.

EBERHART, R.C. & SHI, Y. Comparing inertia weights and construction factors in particle swarm optimization. Proceeding of the IEEE Congress on Evolutionary Computation, 2000, San Diego. USA. 84-88.

GOFFE, W.L., FERRIER, G.D., ROGERS, J. 1994. Global Optimization of Statistical Functions with Simulated Annealing. *Journal of Econometrics*, 60, 65-99.

HOLLAND, J.H. 1975. Adaptation in Natural and Artificial Systems.The University of Michigan Press.

JIAO, B., LIAN, Z., GU, X. 2008. A dynamic inertia weight particle swarm optimization algorithm. *Chaos Solitons& Fractals,* 37, 698-705.

KAO, Y.T. & ZAHARA, E. 2008. A hybrid genetic algorithm and particle swarm optimization for multimodal functions. *Applied Soft Computing*, 8 , 849-857.

KARABOGA, D. & BASTURK, B. 2007. A powerful and Efficient Algorithm for Numerical Function Optimization: Artificial Bee Colony (ABC) Algorithm. *Journal of Global Optimization*, 39, 459-471.

KARABOGA, D. & BASTURK, B. 2008. On the performance of artificial bee colony (ABC) algorithm. *Applied Soft Computing*, 8, 687-697.

KARABOGA, D. & AKAY, B. 2009. A comparative study of Artificial Bee Colony algorithm. *Applied Mathematics and Computation*, 214, 108-132.

KARABOGA, D. & OZTURK, C. 2011. A novel clustering approach: Artificial Bee Colony (ABC) algorithm. *Applied Soft Computing*, 11, 652-657.

KARABOGA, D. & AKAY, B. 2011. A modified Artificial Bee Colony (ABC) algorithm for constrained optimization. *Applied Soft Compunting*, 11, 3021-3031.

KENNEDY, J. & EBERHART, R. Particle swarm optimization. Proceedings of the IEEE International Conference on Neural Networks, 1995, Perth, Australia. 1942-1948.

KIRKPATRICK, S. & GELATT Jr., C.D. , Vecchi, M.P. 1983. Optimization by Simulated Annealing. *Science*, 220, 671-680.

LIU, Y. & SUN, F. 2011. A fast differential evolution algorithm using k-Nearest Neighbour predictor. *Expert Systems with Applications*, 38, 4254-4258.

LI, P., LOWE, K., ARELLANO-GARCIA, H. & WOZNY, G. 2000. Integration of simulated annealing to a simulation tool for dynamic optimization of chemical processes. *Chemical Engineering and Processing*, 39, 357-363.

OURIQUE, C.O. , BISCAIA JR., E.C. & PINTO, J.C. 2002. The use of particle swarm optimization for dynamical analysis in chemical processes, *Computers and Chemical Engineeering*, 26, 1783-1793.

PRICE, K.V., STORN, R.M. & LAMPINEN, J.A. 2005. *Differential Evolution - A Practical Approach to Global Optimization*. Berlin, Springer-Verlag.

SCHWAAB, M., BISCAIA JR., E.C., MONTEIRO, J.C. & PINTO, J.C. 2008. Nonlinear parameter estimation through particle swarm optimization. *Chemical Engineering Science*, 63, 1542-1552.

SINGH, M.K., BANERJEE, T. & KHANNA, A. 2005. Genetic algorithm to estimate interaction parameters of multicomponent system for liquid-liquid equilibria, *Computers and Chemical Engineering*, 29, 1712-1719.

STORN, R. & PRICE, K. . 1997. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11, 341-359.

TSALLIS, C. & STARIOLO, D.A. 1996. Generalized simulated annealing. *Physica A: Statistical Mechanics and its Applications*, 233, 395-406.

WANG, L. & HUANG, F. 2010. Parameter analysis based on stochastic model for differential evolution algorithm. *Applied Mathematics and Computation*, 217, 3263-3273.