An Adaptive Random Search for Unconstrained Global Optimization

Jonás Velasco[1], Mario A. Saucedo-Espinosa[1], Hugo Jair Escalante[2], Karlo Mendoza[1], César Emilio Villarreal-Rodríguez[1], Óscar L. Chacón-Mondragón[1], and Arturo Berrones[1]

[1] Posgrado en Ingeniería de Sistemas,
Facultad de Ingeniería Mecánica y Eléctrica,
Universidad Autónoma de Nuevo León,
Mexico

[2] Departamento de Ciencias Computacionales,
Instituto Nacional de Astrofísica, Óptica y Electrónica,
Mexico

{jonasovich2, m.a.saucedo.e, hugo.jair, karlo.mendoza}@gmail.com,
{cesarevr, olchacon.uanl, arturo.berrones}@gmail.com

Abstract. Adaptive Gibbs Sampling (AGS) algorithm is a new heuristic for unconstrained global optimization. AGS algorithm is a population-based method that uses a random search strategy to generate a set of new potential solutions. Random search combines the one-dimensional Metropolis-Hastings algorithm with the multidimensional Gibbs sampler in such a way that the noise level can be adaptively controlled according to the landscape providing a good balance between exploration and exploitation over all search space. Local search strategies can be coupled to the random search methods in order to intensify in the promising regions. We have performed experiments on three well known test problems in a range of dimensions with a resulting testbed of 33 instances. We compare the AGS algorithm against two deterministic methods and three stochastic methods. Results show that the AGS algorithm is robust in problems that involve central aspects which is the main reason of global optimization problem difficulty including high-dimensionality, multi-modality and non-smoothness.

Keywords. Random search, Metropolis-Hastings algorithm, heuristics, global optimization.

Búsqueda aleatoria adaptiva para problemas de optimizacón global sin restricciones

Resumen. El algoritmo del Muestreador Adaptivo de Gibbs (MAG) es una nueva heurística para la optimización global irrestricta. El algoritmo MAG es un método basado en poblaciones que utiliza una estrategia de búsqueda aleatoria para generar un nuevo conjunto de soluciones potenciales. La búsqueda aleatoria combina el algoritmo unidimensional de Metrópolis-Hastings con el multidimensional muestreador de Gibbs, de tal manera que el nivel de ruido se puede controlar adaptativamente de acuerdo al panorama de la función. Existe un buen equilibrio entre la exploración y la explotación en todo el espacio de búsqueda. Una estrategía de búsqueda local puede acoplarse a la búsqueda aleatoria con el fin de intensificar en las regiones prometedoras. Los experimentos se desarrollaron sobre tres problemas conocidos en un rango de dimensiones, con un banco de prueba resultante de 33 instancias. El algoritmo MAG se comparó contra dos métodos deterministas y tres métodos estocásticos. Los resultados muestran que el algoritmo MAG es robusto en problemas que involucran aspectos centrales que determinan principalmente la dificultad de los problemas de optimización global, es decir, de alta dimensionalidad, multimodalidad y la no suavidad.

Palabras clave. Búsqueda aleatoria, algoritmo de Metrópolis-Hastings, heurísticas, optimización global.

1 Introduction

The inherent difficulty of global optimization problems lies in finding the very best minimum from a multitude of local minima. We consider the problem

Computación y Sistemas Vol. 18 No. 2, 2014 pp. 243-257
ISSN 1405-5546
http://dx.doi.org/10.13053/CyS-18-2-2014-030

of finding the global minimum of the unconstrained continuous optimization problem

$$\min f(x) \quad \text{such that} \quad x \in \Omega \subset \Re^n \tag{1}$$

where $f(x)$ is a nonlinear function and x is a vector of continuous and bounded variables. A global minimization algorithm aims at finding the global minimizer x^* of $f(x)$ such that

$$f^* = f(x^*) \leq f(x), \quad \forall x \in \Omega. \tag{2}$$

Such optimization problem arises in many practical fields of application, generally involving a large number of continuous variables, so there is a need for designing robust algorithms capable of solving problems with different characteristics within each field.

Random search is one of the pillars of most heuristic methods in global optimization. By introducing stochastic perturbations (e.g., mutations in a genetic algorithm) it is possible to explore large regions of a landscape and potentially escape from local minima, resulting in the exploration of different local minima points. The optimal magnitude of this perturbation, in order to achieve a good balance between exploration and exploitation, is a problem-dependent task. In general, this dependency makes the parameter selection a major issue of heuristic algorithms design. In this paper we propose a Markov Chain Monte Carlo (MCMC) procedure which, in combination with a local search strategy, can find very competitive solutions to large global optimization problems in comparison with both deterministic and stochastic established methods. During execution, our algorithm adaptively determines adequate exploration and exploitation rates. Due to the fact that the exploration stage is given by a clearly defined stochastic process, it is possible to have robust and meaningful control parameters.

In order to construct a global exploration strategy, the well known analogy between optimization problems and equilibrium in physical systems [12] is used. Consider a cost function $f(x_1, x_2, ..., x_n, ..., x_N)$. The probability density of a physical system at thermal equilibrium under the potential f is given by

$$p(x) = \left(\frac{1}{Z}\right) exp(-f/kT) \tag{3}$$

where T is the temperature and k is the Boltzmann constant. At small values of the kT term, sampling from the equilibrium density will generate points close to the global optimum. However, if the kT term is too small, most of MCMC methods will suffer a large risk of getting trapped in local regions. This evidence situates the need for an accurate selection of the step size parameters which dictate the amount of noise in the random search. A carefully tuned set of step size parameters for a given temperature may not be appropriate for a different temperature. Moreover, a logarithmic schedule should be imposed to avoid premature convergence [9].

An attractive alternative to usual Metropolis-Hastings based approaches, such as simulated annealing, is the use of Gibbs sampling. The main reason is that Gibbs sampling does not require the definition of any step size parameter and, in addition, the random search processes generated by it are capable of jumping out of large low probability regions [3, 4]. Furthermore, convergence to the correct density is geometric under general conditions [5,15]. However, a disadvantage of Gibbs sampling is that explicit expressions for the conditionals densities of interest are required. These conditionals can be provided only for particular density shapes. This drawback has been recently addressed by the Stationary Fokker-Planck (SFP) sampler which generalizes the Gibbs sampler for arbitrary densities at the cost of using some gradient information [3]. The SFP method has already been applied to global optimization as an exploration mechanism [3, 4]. Here, a similar approach is followed, but it has as implement the fact that no gradient information is required. Our method is based on the Metropolis within Gibbs (MG) algorithm proposed in [1]. The simplicity of MG algorithm makes it easy to define intensification strategies and adaptive simulated annealing type cooling schedules.

This paper is organized as follows. In the next section, the Adaptive Gibbs Sampling (AGS) algorithm is described. Sections 3 and 4 introduce a number of test problems and several comparative methods, respectively. In Section 5 the algorithm is empirically evaluated and the results are analyzed, while emphasizing those aspects that are more difficult to tackle for any global or local optimization

method, namely, the increase of dimensionality and the presence of very rough landscapes. Section 6 highlights some future work and concludes the paper.

2 Adaptive Gibbs Sampling (AGS) Algorithm

In this section, we introduce our optimization algorithm based on the Metropolis within Gibbs algorithm [1]. In the proposed algorithm, random search combines the one-dimensional Metropolis-Hastings algorithm with the multidimensional Gibbs sampler in such a way that the noise level can be adaptively controlled according to the landscape. Noise intensity allows exploring all search space and escaping from local optima. Cooling schedules or noise reduction allows exploitation in the promising regions where local optima exist.

Local search strategies can be coupled to the random search in order to intensify in the promising regions. The main steps of the AGS method is given in Algorithm 1 and are described below.

Algorithm 1 General framework for the proposed AGS algorithm

```
Initialization
repeat
    Sampling
    Selection
    Update
    Mutation
    Intensification
until (termination criteria are not met)
```

2.1 Initialization

In AGS algorithm, an initial solution vector $x = (x_1, x_2, ..., x_n, ..., x_N)$ is randomly created over the search space.

2.2 Sampling

In order to generate a set of new potential solutions, candidate points for each variable are generated by

$$x_n^* = x_n^t + c_n Z \tag{4}$$

where Z is a standard normal variance and c_n are scale parameters. The candidate point will be accepted as the next value ($x_n^{t+1} = x_n^*$) with probability

$$P = \min\left\{1, \frac{p(x_1, x_2, ..., x_n^*, ..., x_N)}{p(x_1, x_2, ..., x_n^t, ..., x_N)}\right\}. \tag{5}$$

If the candidate point is not accepted, then the current value of x is retained: $x_n^{t+1} = x_n^t$. Therefore, at sufficiently large values of c_n the acceptance rates should be low, and as c_n tend to zero, the acceptance rates will tend to 1. This feature permits not only to define cooling schedules, but more importantly, to give criteria for the exploration of the landscape at the single variable level.

In the Gibbs sampler, a Markov chain that converges to the density of interest $p(x)$ is constructed by sampling from the conditionals $p(x_n|\{x_{-n}\})$. Simulating one value in turn for each individual variable from these conditionals is called one cycle of Gibbs sampling. We can draw a population of M solutions by performing M Gibbs cycles. Under general conditions, draws from this simulation algorithm will converge to the target density at a geometric rate [5, 15].

The output of the sampling step is a population $X_{N,M}$ of potential solutions and a θ_N vector of acceptance rates.

2.3 Selection

For each variable over the whole population of solutions generated, we estimate a mode vector x like a promising solution in the search process. The mode vector extracts information from more likely variable values of the population.

2.4 Update

We have chosen cooling schedules of the form

$$c_n = c_n^o \tau^{-\lambda}, \lambda > 0 \quad (6)$$

where c_n^o is a constant chosen so that initially the acceptance rates are close to zero. The variable τ represents the actual iteration number. The actual iteration number τ is initialized at the beginning with $\tau = 1$ and will be increased iteratively by $\tau = \tau + 1$.

2.5 Mutation

In order to escape from local minima and explore large regions of a landscape at the single variable level, we introduce a simple mutation mechanism. This mechanism involves replacing the variable value for a random value within the search space according to the following rule:

$$\text{if } \theta_n > \varepsilon, \text{ then } x_n = rand,\ c_n = c_n^o \text{ and } \tau = 1 \quad (7)$$

where $rand$ is a random value for nth variable of x. Note that for the values close to zero of ε, there is a high noise level in the search process; therefore, we would have a blind search process. If ε criterion is reached, the acceptance rates and τ iteration number are initialized.

2.6 Intensification

In order to improve the proposed solution, a local search strategy is introduced. Therefore, if $\langle\theta\rangle > \beta$, the proposed solution x is improved via local search strategy. $\langle\theta\rangle$ is an average of acceptance rates vector over all variables. We can use an arbitrary local search method. In this paper, we use the Nelder-Mead method as a local search strategy [13]. Note that for β values close to 1, a pure local search exist. The β value must be suitable for the random search process in order to intensify in the promising regions.

Note that the proposed method does not converge to a temperature of zero, but it adapts through the β parameter such that the temperature has the conditions for which the literature on MCMC methods suggests to have a sampling that adequately covers all search space [16]. Therefore it is necessary to couple a local search method. Other stochastic methods, which are comparable with the AGS algorithm, converge to a point, thus it is analogous to bring the temperature of the stochastic search to zero.

2.7 Parameter Settings

The AGS parameters used in this paper are chosen such as to be a robust setting and therefore, in our experience, applicable to a wide range of functions to be optimized. The parameters used are population size M=100, initial scale parameters $c^o = (0.1, ..., 0.1)$, $\beta = 0.7$, $\varepsilon = 0.95$ and $\lambda = 2$.

3 Test Problems

In order to empirically evaluate the AGS algorithm, we selected some well known problems which act as performance tests for global optimization algorithms. These test problems were selected for testing the robustness of the AGS against stochastic and deterministic methods in three aspects which, even individually, decrease the performance of many global optimization algorithms. These aspects are the increase in dimensionality, the multimodal function optimization and the optimization of non-smooth functions. The selected test problems are now introduced.

3.1 Rosenbrock Problem

The Rosenbrock function is a well known test problem for optimization algorithms. Fig. 1 exhibits the Rosenbrock function for two variables, where it can be seen that the global minimum is inside a long, narrow, parabolic shaped flat valley. Finding the valley is a trivial task, but convergence to the global minimum is difficult. For this reason it has been reported in the literature as a very difficult task for stochastic heuristics [10] and is very well suited to study the behavior of algorithms while increasing the problem dimension. The problem is defined as

$$\min \sum_{n=1}^{N} 100(x_{n+1} - x_n^2)^2 + (x_n - 1)^2 \quad (8)$$
$$-10 \leq x_n \leq 10$$

and it has the known global solution $x^* = (1, ..., 1)$ for which the cost function value is zero.

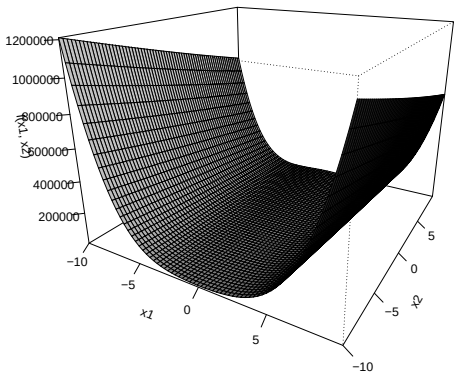


Fig. 1. The Rosenbrock function in 2D, $f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (x_1 - 1)^2$

3.2 Morse Clusters

An important application of global optimization techniques is the minimization of potential energy structures, which is relevant in the study of proteins and nanomaterials. The Morse potential is an adequate model for several atomic clusters and gives a challenging benchmark for global optimization algorithms [14]. The model consists in an expression for the pairwise atomic interactions:

$$V_{ij} = e^{2\rho(1-r_{ij})} - 2e^{\rho(1-r_{ij})} \tag{9}$$

where r_{ij} is the interatomic Euclidean distance and ρ is a parameter that represents the equilibrium pair separation.

The problem is to minimize the potential energy of the N atom cluster,

$$V = \sum_{i<j} V_{ij}. \tag{10}$$

Fitting to bulk data indicates that, by the Morse model, realistic predictions can be made for clusters like C_60 (using $\rho = 13.62$), sodium (with $\rho = 3.15$) and nickel ($\rho = 3.96$), just to mention a few. The minimum energy configurations are of fundamental importance in addressing the chemical and physical properties of a given system.

The putative global optima for each dimension of the considered clusters are given in Table 1. In Table 1, N denotes the number of atoms to cluster-ize in a three-dimensional space and S denotes the real problem size ($S = 3 \cdot N$).

Table 1. Putative known global optima for each Morse clusters instance

N	S	$f(x^*)$	N	S	$f(x^*)$
5	15	-9.04	35	105	-141.96
10	30	-27.47	40	120	-167.99
15	45	-49.75	45	135	-192.95
20	60	-72.51	50	150	-219.82
25	75	-95.13	55	165	-250.29
30	90	-118.43			

3.3 Fractal Function

One of the main interests in the development of heuristics is their use in problems for which an exact solution is not easily attainable. Functions with very rough landscapes are one of the most challenging problems for both exact and heuristic global optimization methods. Fractal function has strong similarities to real-world problems. Here at first instance we consider a test problem with a fractal landscape introduced in [2]:

$$\min f(x) = \sum_{n=1}^{N} C'(x_n) + x_n^2 - 1 \qquad -5 \leq x_n \leq 5 \tag{11}$$

$$C'(x) = \begin{cases} \frac{C(x)}{C(1)|x|^{2-D}}, & \text{if } x \neq 0 \\ 1, & \text{if } x = 0 \end{cases} \tag{12}$$

$$C(x) = \sum_{j=-\infty}^{\infty} \frac{1 - \cos(b^j x)}{b^{(2-D)j}} \tag{13}$$

where $C(x)$ is an approximation of the Weierstrass-Mandelbrot cosine fractal function. For this function, D is known to be a box dimension ($1 < D < 2$) and it represents a parameter that arbitrarily increases or decreases the complexity of the cost function.

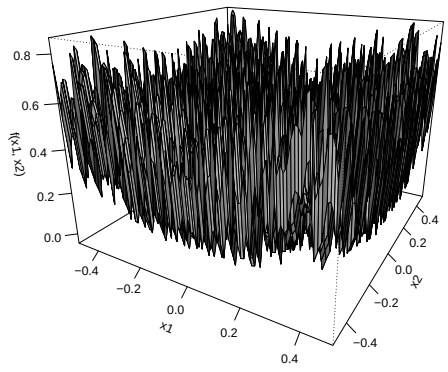


Fig. 2. Zoom on the fractal function in 2D with parameters D=1.85 and b=1.5

For this fractal function it is impossible to indicate the exact position of the global minimum. Due to the zigzagging peaks close to the origin, several local optima with function values smaller than zero exist. Fig. 2 introduces the fractal function with parameters $D = 1.85$ and $b = 1.5$. This figure illustrates the complexity of performing an optimization routine for this function, due to both the multiple locally optimal points within each region and the fact that gradient information cannot be used to determine the direction in which the function is decreasing. The parameters $D = 1.85$ and $b = 1.5$ are used in all the experiments reported in this paper.

4 Algorithms for Comparative Tests

Having selected suitable performance test problems for evaluating the AGS algorithm robustness, the task was then to select competitors for every single test problem. To achieve this, we chose some methods, both deterministic and stochastic, that have been proved to reach good quality solutions in at least one of the test problems. Now we present the methods selected for the comparative tests with AGS.

4.1 Stochastic Methods

In order to compare the proposed AGS algorithm with other population-based heuristic search methods, we considered three of the most popular strategies nowadays, namely, Genetic Algorithms (GAs), Particle Swarm Optimization (PSO) and Differential Evolution (DE). All three heuristics start with a set of randomly generated solutions which are updated throughout an iterative process using different mechanisms. GAs, PSO and DE methods have reported comparable results in highly complex problems such as some of the problems we used for evaluation in this work. Due to their proved performance, these techniques have been widely studied and, as a consequence, there exist many versions of each of these strategies. To avoid biasing our study towards specific implementations, we consider the standard/basic versions of GAs, PSO and DE methods.

GAs: The genetic algorithms are inspired from biological evolution, where solutions (chromosomes) are coded as binary vectors and new individuals are created or updated by a recombination of selected individuals and mutation rules. In this work we considered the canonical genetic algorithm with roulette wheel selection, one-point crossover and a standard mutation procedure, see [8] for details. We used the Matlab® GAs toolbox implementation.

PSO: A particle swarm optimization is inspired by the behavior of biological societies, such as flocks of birds and shoals of fishes, which present local and social behavior for achieving common goals [6]. Solutions are coded as numerical vectors (particles) and they are updated by combining information from global and local solutions that are found during the search process. For the comparison we implemented the standard PSO algorithm with adaptive inertia weight, which is one of the most used improvements of PSO for enhancing the rate of convergence of the algorithm [18].

DE: The differential evolution is the newest population based heuristic which we consider for the experimental comparison. DE updates solutions by combining existing solutions and adding randomness into this combination. The update of solutions is defined by simple rules for selection,

crossover and mutation. In this paper we considered the basic DE algorithm as described in [19]. We used the DE implementation available at http://www.icsi.berkeley.edu/∼storn/code.html.

4.2 Deterministic Methods

In addition to stochastic methods, we considered two classical deterministic algorithms which have shown their capabilities to achieve good quality solutions when implementing them to solve large optimization problems, namely, the Nelder-Mead Method (NMM) and the Conjugate Gradients Algorithm (CGA).

NMM: The Nelder-Mead Method is a simplex method for finding a local minimum of a function of several variables that has been devised by Nelder and Mead [13]. The NMM requires only function evaluations, not derivatives. In the N-dimensional space, a simplex is a polyhedron with $N + 1$ points (or vertices). We chose the $N+1$ points and defined an initial simplex. The method iteratively updates the worst point by four operations: reflection, expansion, one-dimensional contraction and multiple contraction. The NMM uses a small number of function evaluations per iteration and it is one of the most widely used direct search methods for multidimensional nonlinear optimization problems that have a unique optimal solution. A big disadvantage of the NMM is that it can converge to non-stationary points [11]. For the experimental comparison we used the package `neldermead` available in the CRAN package repositories (http://cran.r-project.org/).

CGA: The Conjugate Gradient Algorithm is a method for finding the nearest local minimum of a function of n variables. This method presupposes that the gradient of the function can be computed. It uses conjugate directions instead of the local gradient for going downhill [7]. The CGA combines the information from all previous directions in such a way as to create a subsequent search direction that is independent (or conjugated) to all previous directions. For the experimental comparison we used the package `Rcgmin` available in the CRAN package repositories.

5 Experimental Results

In this section we evaluate the AGS performance to solve several well known test problems for unconstrained global optimization. These test problems were selected for testing the robustness of the AGS algorithm against stochastic and deterministic methods in three aspects that individually decrease the performance of many global optimization algorithms, namely, the increase in dimensionality, the multimodal function optimization and the optimization of non-smooth functions. The empirical evaluation consisted in assessing the performance of the methods based on five independent executions of each method and for each problem size, from 5 to 55 variables. Both the cost function average value and the average number of cost function evaluations performed are reported for each problem dimension. The algorithm that reached the best cost function average value for each problem dimension is marked in bold face. Every test problem implementation has its own main stopping criterion, but in order to make the comparisons as fair as possible, we have set an additional stopping criterion consisting in the number of cost function evaluations (FEs), which is set to 500 000 for all experiments in this paper. Furthermore, we make use of statistics to validate the results of our empirical evaluation and we present a ranking for comparing the overall performance of each algorithm when applying them to the selected test problems.

5.1 Rosenbrock Problem

Given that the optimal cost function value for the Rosenbrock problem is zero, we consider as the main stopping criterion reaching a cost function value of 0.001 or lower. Defining such threshold for acceptable solutions is necessary as we are dealing with heuristic search methods that do not guarantee obtaining the global optimum (at least for a finite number of iterations).

Table 2 displays the cost function average value ($f(\hat{x})$ column) and the average number of cost function evaluations needed (# of FEs column) by each algorithm when applied to the Rosenbrock problem while varying the problem dimensionality.

The standard deviation for both measures is reported as well (Std. Dev. columns). It is noteworthy that all tables within this paper follow this format.

It can be seen that, for all the dimensions, the method with the best performance is the CGA, as it finds the objective function values close to the global optimum while using a few cost function evaluations. Note that the number of cost function evaluations for the CGA method is an estimate of the number of evaluations necessary to compute the gradient and the Hessian throughout the process. For the dimensions of 5 to 30 in Table 2, the AGS algorithm achieves solutions close to zero. For the rest of the dimensions, we observed that AGS needs more than 500 000 cost function evaluations to achieve high quality solutions. Nevertheless, the results obtained by AGS are the closest, in comparison to the rest of the comparative methods, to the results achieved by the CGA method. The behavior of the NMM method is acceptable for problems of low dimensionality (5-20), although its performance drops significantly for dimensions larger than 20. Among the stochastic optimization techniques, the DE technique obtained the best results for the dimensionalities of 5-30, followed by PSO and GA; however, the performance of DE is rather poor for dimensionalities larger than 30. PSO and GA methods showed a more stable behavior across dimensionalities.

5.2 Morse Clusters

For the Morse cluster problem we allow the methods under evaluation to run until either their own default stopping criterion is met (related to a number of successive iterations with no improvement in the solution) or the maximum number of function evaluations is reached. For the comparisons we have set the following error function, $|f(x^*) - f(\hat{x})|/|f(x^*)| < 0.1$, where $f(x^*)$ is the putative known global optima and $f(\hat{x})$ is the average cost function.

Despite the fact that there exist very effective heuristics for Morse cluster optimization [14], they are particularly designed for this problem while our interest in this paper is on the development of a general purpose method.

Table 3 shows both the average value of the cost function and the average number of cost function evaluations needed by every algorithm when applied to the Morse cluster problem while varying the problem dimensionality. It is important to point out that N denotes the number of atoms to clusterize in a three-dimensional space, so the real problem size is $S = 3 \cdot N$, where S is the overall number of variables to optimize. From our experiments it can be seen that for N = 15, 20 and 25 the method with the best performance is the CGA, while for the rest of dimensions this distinction belongs to the NMM method. It should be noted that, for large dimensions ($3 \cdot 30$–$3 \cdot 55$), the solutions achieved by AGS are closer to the solutions provided by the best performance method than the rest of algorithms under comparison, while for low dimensions ($3 \cdot 5$ – $3 \cdot 10$) most of the compared methods seem to achieve solutions with almost equal quality. For N = 40 to 55 the quality of the solution of the CGA method diverges. Regarding the stochastic techniques, the three methods obtained lower performances for dimensionalities larger than $3 \cdot 15$.

5.3 Fractal Function

When solving the non-smooth fractal function, we have adopted similar stopping criteria as those considered in the Morse cluster problem: the method stops either when its own default stopping criterion is met or when the maximum number of function evaluations is reached.

Table 4 reports the average of the cost function values and the number of cost function evaluations needed by every algorithm when applied to the non-smooth fractal function problem while varying the problem dimensionality. It can be seen that for all dimensions in this problem the best performance is obtained by the GA method, reaching cost function values smaller than zero while evaluating the cost function 500 000 times. For all dimensions, the AGS algorithm achieves cost function values smaller than zero. The AGS solutions are comparable to those of GA method in all dimensions. The behavior of the DE method is similar to the behavior of both the GA and the AGS algorithm for low dimensions (5-30), although its performance drops significantly for dimensions larger than 30, where the AGS is the only method among all algorithms under

Table 2. Experimental results when applying the selected algorithms to the Rosenbrock problem for 5 to 55 dimensions. All results have been averaged over five independent runs

method	N	$f(\hat{x})$	Std. Dev.	# of FEs	Std. Dev.	N	$f(\hat{x})$	Std. Dev.	# of FEs	Std. Dev.
AGS		2.34E-14	5.18E-14	1000	0		3.21E+00	1.13E+00	496986	1299
CGA		**9.19E-19**	1.38E-18	357	85		**7.28E-18**	1.13E-17	1480	201
NMM	5	8.60E-03	1.00E-03	587	196	35	1.02E+02	1.13E+01	346808	313588
PSO		5.30E-03	3.50E-03	487720	27458		5.69E+01	1.13E+01	500000	0
DE		7.00E-04	1.00E-04	99221	10052		1.88E+02	1.13E+02	500000	0
GA		5.77E-01	9.09E-01	500000	0		8.09E+01	1.13E+01	500000	0
AGS		1.92E-08	3.69E-08	10200	3834		1.02E+01	1.13E+00	497846	2657
CGA		**4.04E-18**	5.66E-18	506	116		**1.52E-18**	1.38E-18	1550	254
NMM	10	9.50E-03	0.00E+00	4551	2173	40	1.92E+02	1.13E+02	457433	211480
PSO		6.18E-01	3.12E-01	500000	0		6.52E+01	1.13E+01	500000	0
DE		8.00E-04	1.00E-04	339241	7759		1.03E+03	1.13E+02	500000	0
GA		2.23E+00	3.06E+00	500000	0		1.28E+02	1.13E+01	500000	0
AGS		7.19E-06	1.11E-05	41062	9293		1.92E+01	1.13E+00	498498	1402
CGA		**4.77E-18**	8.03E-18	749	93		**2.44E-18**	2.49E-18	1706	326
NMM	15	9.50E-03	0.00E+00	31345	12606	45	3.67E+02	1.13E+02	433512	91419
PSO		3.22E+00	3.10E+00	488480	25759		7.32E+01	1.13E+01	500000	0
DE		1.54E-01	3.82E-02	500000	0		8.86E+03	1.13E+03	500000	0
GA		1.85E+01	3.11E+01	500000	0		1.55E+02	1.13E+01	500000	0
AGS		7.88E-05	9.53E-05	122518	17303		3.17E+01	1.13E+00	497710	1817
CGA		**3.13E-18**	6.66E-18	941	83		**3.35E-18**	3.26E-18	1863	376
NMM	20	9.90E-03	0.00E+00	120235	69874	50	1.43E+02	1.13E+01	500000	0
PSO		7.63E+00	6.60E+00	500000	0		8.37E+01	1.13E+01	500000	0
DE		7.16E+00	2.10E-01	500000	0		6.44E+04	1.13E+04	500000	0
GA		4.73E+01	3.99E+01	500000	0		1.53E+02	1.13E+01	500000	0
AGS		1.56E-04	1.18E-04	278530	35925		4.28E+01	1.13E+00	497479	1712
CGA		**4.25E-18**	6.01E-18	990	134		**2.76E-18**	4.18E-18	2084	264
NMM	25	1.58E+01	1.00E+01	205129	146579	55	2.19E+02	1.13E+02	495701	9613
PSO		1.48E+01	8.17E+00	500000	0		1.20E+02	1.13E+01	500000	0
DE		1.59E+01	3.08E-01	500000	0		3.48E+05	1.13E+04	500000	0
GA		5.50E+01	3.88E+01	500000	0		1.65E+02	1.13E+01	500000	0
AGS		3.24E-04	1.68E-04	496824	2310					
CGA		**1.29E-17**	1.07E-17	1258	260					
NMM	30	5.73E+01	5.75E+01	447232	391524					
PSO		3.25E+01	2.55E+01	500000	0					
DE		2.55E+01	1.08E+00	500000	0					
GA		7.34E+01	3.41E+01	500000	0					

comparison that remains close to the performance of the most effective method.

Roughly speaking, the gradient free approaches (AGS, NMM, PSO, DE and GA) have similar computation times while the CGA algorithm is much faster in the problems to which it is applicable, since it is a method that requires fewer evaluations of the cost function to achieve a good solution.

5.4 Ranking Comparison

In order to illustrate the overall performance of each algorithm with respect to the others, we have constructed a simple ranking by taking into consideration the information shown in Tables 2 to 4. The best function value has a rank 1, the second best function value has a rank 2, etc., so the worst method has a rank 6. For the case where

Table 3. Experimental results when applying the selected algorithms to the Morse cluster problem for 5 to 55 dimensions. All results have been averaged over five independent runs

method	N	$f(\hat{x})$	Std. Dev.	# of FEs	Std. Dev.	N	$f(\hat{x})$	Std. Dev.	# of FEs	Std. Dev.
AGS		-9.04E+00	0.00E+00	3603	226		-1.29E+02	7.37E-01	172786	58764
CGA		**-9.04E+00**	0.00E+00	**3125**	326		-2.21E+01	2.28E+01	772548	248565
NMM	5	-9.04E+00	0.00E+00	5156	2700	35	**-1.34E+02**	1.62E+00	500000	0
PSO		-9.04E+00	0.00E+00	500000	0		-4.32E+01	1.27E+01	500000	0
DE		-7.40E+00	3.67E-01	500000	0		-8.98E+00	2.55E+00	500000	0
GA		-9.04E+00	1.94E-05	500000	0		-7.38E+01	7.35E+00	500000	0
AGS		-2.61E+01	9.11E-01	14801	4025		-1.52E+02	1.28E-01	311787	73343
CGA		-2.54E+01	4.76E-01	81216	42044		1.48E+02	3.06E+01	970632	44211
NMM	10	**-2.65E+01**	7.86E-01	22429	6487	40	**-1.55E+02**	2.31E+00	500000	0
PSO		-2.49E+01	3.14E+00	500000	0		-4.71E+01	6.91E+00	500000	0
DE		-7.97E+00	1.49E+00	500000	0		-1.08E+01	3.63E+00	500000	0
GA		-2.55E+01	1.13E+00	500000	0		-7.02E+01	7.09E+00	500000	0
AGS		-4.60E+01	8.29E-01	55829	28601		-1.75E+02	2.90E-01	490773	8739
CGA		**-4.67E+01**	1.79E+00	197148	57527		5.74E+02	2.79E+01	1001880	187917
NMM	15	-4.65E+01	1.86E+00	149418	49892	45	**-1.78E+02**	3.87E+00	500000	0
PSO		-3.10E+01	6.16E+00	500000	0		-3.88E+01	5.93E+00	500000	0
DE		-9.40E+00	2.29E+00	500000	0		-1.22E+01	4.73E+00	500000	0
GA		-3.72E+01	5.44E+00	500000	0		-7.46E+01	9.42E+00	500000	0
AGS		-6.66E+01	1.14E+00	63203	12458		-1.84E+02	3.50E+00	488098	7098
CGA		**-6.82E+01**	1.99E+00	361000	111125		1.28E+03	4.37E+01	1289680	403702
NMM	20	-6.65E+01	1.24E+00	301985	44931	50	**-1.99E+02**	5.81E+00	500000	0
PSO		-3.58E+01	3.23E+00	500000	0		-4.89E+01	1.10E+01	500000	0
DE		-8.44E+00	1.31E+00	500000	0		-9.73E+00	1.81E+00	500000	0
GA		-4.51E+01	4.13E+00	500000	0		-7.83E+01	4.58E+00	500000	0
AGS		-8.63E+01	5.01E-01	79769	20075		-1.95E+02	1.36E+01	485588	151
CGA		**-8.90E+01**	2.68E+00	811800	312243		2.42E+03	4.78E+01	1485000	792375
NMM	25	-8.88E+01	1.46E+00	500000	0	55	**-2.22E+02**	6.54E+00	500000	0
PSO		-4.85E+01	5.55E+00	500000	0		-4.70E+01	6.59E+00	500000	0
DE		-9.41E+00	8.70E-01	500000	0		-1.01E+01	5.69E-01	500000	0
GA		-5.50E+01	5.20E+00	500000	0		-6.96E+01	6.08E+00	500000	0
AGS		-1.07E+02	4.27E-01	184306	61085					
CGA		-8.67E+01	9.05E+00	877830	497173					
NMM	30	**-1.09E+02**	3.50E+00	500000	0					
PSO		-5.24E+01	5.08E+00	500000	0					
DE		-1.04E+01	1.57E+00	500000	0					
GA		-6.54E+01	5.13E+00	500000	0					

the cost function values are the same, we use the number of cost function evaluations to discriminate the ranking of the results of the compared algorithms. Tables 5 to 8 show the rank information for the Rosenbrock, the Morse cluster and the fractal problem, respectively. In each one of these tables, the rows exhibit the compared methods while the columns are related to the problem dimension, so each element of the table from columns 5 to 55 corresponds to the rank given to a specific method for that dimension, according to its performance when compared to the other methods. The lower the rank is, the better the performance of the method. Column R-Sum shows the sum of the ranks per method. Finally, the R-Rank column introduces the rank of the R-Sum column, which is an indicative of the overall performance for a given method in all dimensions for the solved problem. The overall

Table 4. Experimental results when applying the selected algorithms to the non-smooth fractal function problem for 5 to 55 dimensions. All results have been averaged over five independent runs

method	N	$f(\hat{x})$	Std. Dev.	# of FEs	Std. Dev.	N	$f(\hat{x})$	Std. Dev.	# of FEs	Std. Dev.
AGS		-4.30E-01	2.64E-02	69442	150		-2.23E-01	9.92E-01	496146	42
CGA		4.03E+01	1.61E+01	60	109		2.88E+02	4.77E+01	26	29
NMM	5	-9.60E-02	2.18E-01	807	92	35	1.62E+01	1.25E+01	204497	123646
PSO		2.52E+01	2.92E+00	500000	0		2.30E+02	8.19E+00	500000	0
DE		-8.75E-02	8.32E-02	500000	0		4.83E-02	2.10E-01	500000	0
GA		**-9.71E-01**	1.21E-01	500000	0		**-4.07E+00**	2.20E-01	500000	0
AGS		-2.35E-01	2.61E-01	137692	687		-2.95E-01	7.75E-01	496689	1275
CGA		7.94E+01	1.93E+01	13	1		3.26E+02	3.10E+01	41	31
NMM	10	3.65E+00	7.47E+00	6017	5958	40	2.09E+01	6.58E+00	237184	195298
PSO		5.72E+01	2.05E+00	500000	0		2.59E+02	3.52E+00	500000	0
DE		-1.06E-01	9.02E-02	500000	0		8.03E-01	6.91E-02	500000	0
GA		**-1.71E+00**	1.30E-01	500000	0		**-4.29E+00**	1.96E-01	500000	0
AGS		-5.46E-01	7.98E-02	203339	1329		-1.83E+00	5.49E-02	497508	86
CGA		1.23E+02	2.16E+01	19	14		3.89E+02	6.71E+01	12	1
NMM	15	6.39E+00	7.13E+00	8620	4080	45	1.30E+01	8.95E+00	335779	272687
PSO		9.00E+01	2.26E+00	500000	0		2.95E+02	2.74E+00	500000	0
DE		-9.96E-02	4.93E-02	500000	0		1.83E+00	3.26E-01	500000	0
GA		**-2.25E+00**	1.08E-01	500000	0		**-5.01E+00**	1.50E-01	500000	0
AGS		-2.64E-01	2.45E-01	268635	165		-1.89E-01	3.49E-01	495735	1350
CGA		1.73E+02	4.11E+01	22	16		4.13E+02	2.92E+01	11	1
NMM	20	4.28E+00	4.70E+00	17317	3963	50	1.81E+01	9.23E+00	300187	211061
PSO		1.20E+02	2.26E+00	500000	0		3.26E+02	4.74E+00	500000	0
DE		-8.84E-02	1.71E-02	500000	0		3.15E+00	3.56E-01	500000	0
GA		**-2.86E+00**	9.56E-02	500000	0		**-5.26E+00**	1.98E-01	500000	0
AGS		-1.37E-01	4.91E-01	409620	1832		-6.54E-01	9.73E-01	495544	926
CGA		2.14E+02	2.78E+01	33	34		4.54E+02	3.62E+01	17	13
NMM	25	5.69E+00	3.72E+00	23968	4114	55	2.51E+01	8.57E+00	500000	0
PSO		1.57E+02	4.76E+00	500000	0		3.68E+02	4.96E+00	500000	0
DE		-1.16E-01	1.16E-01	500000	0		8.76E+00	5.50E-01	500000	0
GA		**-3.34E+00**	8.55E-02	500000	0		**-5.22E+00**	2.32E-01	500000	0
AGS		-3.23E-01	4.79E-01	495054	2661					
CGA		3.09E+02	6.89E+01	14	3					
NMM	30	8.61E+00	6.37E+00	95174	33072					
PSO		1.90E+02	2.04E+00	500000	0					
DE		-1.03E-01	1.15E-01	500000	0					
GA		**-4.07E+00**	2.20E-01	500000	0					

performance is introduced in Table 8. The column R-Sum exhibits the sum of ranks obtained in columns R-Rank for all the problems. The Rank column is the overall rank, among all test problems. From this column, we can see that AGS obtained the best rank across the different problems and dimensions; this fact illustrates the robustness of our method while increasing the dimensionality and when optimizing multimodal and non-smooth functions. Despite the fact that other methods such as CGA outperformed the AGS in one problem, AGS consistently achieved good performance across the different tasks.

5.5 Statistical Comparison

In order to statistically validate the robustness of AGS algorithm, we use the Welch's t test (or unequal variance t test). The Welch's t test is a

Table 5. Ranking results for the Rosenbrock problem

Method/N	5	10	15	20	25	30	35	40	45	50	55	R-Sum	R-Rank
AGS	2	2	2	2	2	2	2	2	2	2	2	22	2
CGA	1	1	1	1	1	1	1	1	1	1	1	11	1
NMM	5	4	3	3	4	5	5	5	5	4	5	48	4
PSO	4	5	5	5	3	4	3	3	3	3	3	41	3
DE	3	3	4	4	5	3	6	6	6	6	6	52	5
GA	6	6	6	6	6	6	4	4	4	5	4	57	6

Table 6. Ranking results for the Morse problem

Method/N	5	10	15	20	25	30	35	40	45	50	55	R-Sum	R-Rank
AGS	2	2	3	2	3	2	2	2	2	2	2	24	2
CGA	1	4	1	1	1	3	5	6	6	6	6	40	4
NMM	3	1	2	3	2	1	1	1	1	1	1	17	1
PSO	4	5	5	5	5	5	4	4	4	4	4	49	5
DE	5	6	6	6	6	6	6	5	5	5	5	61	6
GA	4	3	4	4	4	4	3	3	3	3	3	38	3

Table 7. Ranking results for the Fractal problem

Method/N	5	10	15	20	25	30	35	40	45	50	55	R-Sum	R-Rank
AGS	2	2	2	2	2	2	2	2	2	2	2	24	2
CGA	6	6	6	6	6	6	6	6	6	6	6	72	6
NMM	3	4	4	4	4	4	4	4	4	4	4	46	4
PSO	5	5	5	5	5	5	5	5	5	5	5	60	5
DE	4	3	3	3	3	3	3	3	3	3	3	38	3
GA	1	1	1	1	1	1	1	1	1	1	1	12	1

Table 8. The final ranking results

Method	R-Sum	Rank
AGS	6	1
NMM	9	2
GA	10	3
CGA	11	4
PSO	13	5
DE	14	6

technique used to compare means of two samples when it cannot be safely assumed that population variances are equal [17]. On the one hand, by using this test we could infer if two means (average cost function values in this case) differ significantly and thus originate from different populations. Such a result will indicate that two methods are not likely to produce equal quality solutions.

On the other hand, we could construct and compare the means confidence intervals to determine how close the solution qualities of two methods are. The Welch's t test is performed under the assumption of normality. We use the samples (five independent runs) of the solutions obtained by the three best methods for each problem of the highest dimension ($N = 55$) for the statistical tests. GNU R was used for this statistical significance study.

The normality of the sample distribution is checked by using the Shapiro-Wilk test. Table 9 shows both the W statistic and the p-values computed. The null hypothesis for this test is that the data is normally distributed. The Rosenbrock problem is significant with a level of significance $\alpha = 0.01$ (W critical = 0.6859). The Morse and Fractal problems are significant with a level of significance $\alpha = 0.05$ (W critical = 0.7620). One would accept the null hypothesis, concluding that there is no information to discard normality in the data.

Table 10 shows the t statistic, the p-values and the confidence interval computed for all pairwise comparisons concerning AGS when applying the Welch's t test. The null hypothesis is that the two population means are the same, but the two population variances may differ. It can be seen for all comparisons that the resulting p-values obtained in this test clearly indicate statistically significant differences between every two methods, that is, all results are significant at the 5% significance level. One would reject the null hypothesis concluding that there is strong evidence that the expected values for all pairwise comparisons are different, which means that the three methods offer different quality solutions. For a further analysis, we make use of the confidence interval. In the rest of this section, when we refer to the difference between means, we stand for the difference between means predicted by the confidence interval.

For the Rosenbrock problem, the difference between the means of CGA (best method) against AGS seems to be smaller than the difference between the means of PSO (3rd best method) against AGS. Moreover, for the Morse cluster problem, the difference between the means of NMM (best method) against AGS is much smaller than the difference between the means of GA (2nd best) against AGS. Finally, for the Fractal problem, the difference between the means of GA (best method) against AGS is smaller than the difference between the means of DE (2nd best) against AGS. These results indicate that, on average, AGS solutions are the closest to the solutions provided by the best performance algorithm in each of the three test problems at the highest dimension. None of the other algorithms share this property. Therefore, we

conclude that there is strong evidence indicating that AGS is a robust approach for difficult high dimensional unconstrained global optimization tasks.

Table 9. The Shapiro-Wilk normality test results

Function	Method	W	p-value
Rosenbrock	CG	0.7362	0.0220
	AGS	0.7790	0.0540
	PSO	0.8893	0.3534
Morse	NMM	0.8849	0.3320
	AGS	0.8603	0.2293
	GA	0.9232	0.5508
Fractal	GA	0.9075	0.4525
	AGS	0.8741	0.2834
	DE	0.9361	0.6384

Table 10. Welch's t test results

Function	Comparison	Statistic t	p-value	Confidence interval
Rosenbrock	AGS versus CGA	60.77	4.38E-07	[40.82, 44.73]
	PSO versus AGS	4.94	0.0077	[33.80, 120.03]
Morse	AGS versus NMM	4.05	0.0072	[10.71, 44.11]
	GA versus AGS	18.75	3.18E-06	[108.40, 141.70]
Fractal	AGS versus GA	10.22	0.0002	[3.38, 5.76]
	DE versus AGS	18.85	8.75E-07	[8.21, 10.62]

6 Conclusions and Future Work

In this paper, a new optimization algorithm called Adaptive Gibbs Sampling (AGS) algorithm is introduced. AGS algorithm directly extracts global statistical information about the search space during the random search keeping a good compromise between exploration and exploitation. Local search strategy has been coupled to the random search process in order to intensify in promising regions. With both mechanisms of search, the AGS algorithm can find very competitive solutions to large global optimization problems, in comparison with deterministic and stochastic established methods. An adequate use of both the local information of solutions found and the global information about the search space improves the performance of the proposed method.

We have evaluated the performance of our method against deterministic and stochastic algorithms that are commonly employed for solving challenging well known test problems. For the selection of the test problems, we have focused on problems that involve three central aspects which mainly determine the difficulty of global optimization problems, namely, high-dimensionality, multimodality and non-smoothness. For comparison purposes, we selected three of the most popular heuristic strategies nowadays, namely, Genetic Algorithms (GAs), Particle Swarm Optimization (PSO) and Differential Evolution (DE), as well as two classical deterministic algorithms that have shown their capabilities to achieve good quality solutions when implementing them to solve large optimization problems, namely, the Nelder-Mead (NMM) method and the Conjugate Gradients Algorithm (CGA).

Experimental results showed that our approach is statistically robust, as it is capable of finding reasonable quality solutions for global optimization problems. By robust we refer to the fact that, on average, AGS algorithm performs better than other methods in high dimensional problems, as it achieves high quality solutions for all the test problems chosen.

In the future, we will extend these performance comparisons against other state-of-the-art methods for unconstrained global optimization, and we will extend the AGS algorithm to execute in parallel computing. For the AGS parallel approach, we will study problems with larger dimensions than the ones provided here, which conform a greater testing challenge for our method and other heuristics.

Acknowledgements

This work was supported in part by the Mexican National Council for Science and Technology (CONACyT), grant 206705, and by UANL-PAICYT program, grant "Inference based on density estimation".

References

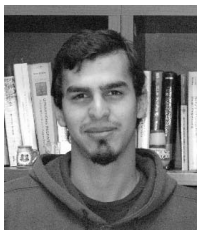
1. **Albert, J.** (**2009**). *Bayesian computation with R*. Springer.
2. **Bäck, T.** (**1996**). *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford University Press, Oxford, UK.

3. **Berrones, A.** (**2008**). Stationary probability density of stochastic search processes in global optimization. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(01), P01013.
4. **Berrones, A.** (**2010**). Bayesian inference based on stationary fokker–planck sampling. *Neural Computation*, 22(6), 1573–1596.
5. **Canty, A. J.** (**1999**). Hypothesis tests of convergence in markov chain monte carlo. *Journal of Computational and Graphical Statistics*, 8(1), 93–108.
6. **Engelbrecht, A. P.** (**2006**). *Fundamentals of Computational Swarm Intelligence*. John Wiley & Sons.
7. **Fletcher, R. & Reeves, C. M.** (**1964**). Function minimization by conjugate gradients. *The Computer Journal*, 7(2), 149–154.
8. **Goldberg, D. E.** (**1989**). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.
9. **Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P.** (**1983**). Optimization by simulated annealing. *Science*, 220(4598), 671–680.
10. **Laguna, M. & Martí, R.** (**2005**). Experimental testing of advanced scatter search designs for global optimization of multimodal functions. *Journal of Global Optimization*, 33(2), 235–255.
11. **Mckinnon, K. I. M.** (**1998**). Convergence of the nelder–mead simplex method to a nonstationary point. *SIAM Journal on Optimization*, 9(1), 148–158.
12. **Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E.** (**1953**). Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21, 1087–1092.
13. **Nelder, J. A. & Mead, R.** (**1965**). A simplex method for function minimization. *The Computer Journal*, 7(4), 308–313.
14. **Pintér, J. D.** (**2006**). *Global Optimization: Scientific and Engineering Case Studies*. Springer.
15. **Roberts, G. O. & Polson, N. G.** (**1994**). On the geometric convergence of the gibbs sampler. *Journal of the Royal Statistical Society B*, 56(2), 377–384.
16. **Roberts, G. O. & Rosenthal, J. S.** (**2001**). Optimal scaling for various metropolis-hastings algorithms. *Statistical Science*, 16(4), 351–367.
17. **Sawilowsky, S. S.** (**2002**). Fermat, schubert, einstein, and behrens–fisher: The probable difference between two means when $\sigma_1 \neq \sigma_2$. *Journal of Modern Applied Statistical Methods*, 1(2), 461–472.
18. **Shi, Y. & Eberhart, R. C.** (**1999**). Empirical study of particle swarm optimization. *Proceedings of the 1999 Congress on Evolutionary Computation, 1999. CEC 99*, 3, 1945–1950.
19. **Storn, R. & Price, K.** (**1997**). Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4), 341–359.

Jonás Velasco recieved his Ph.D. in Systems Engineering from Universidad Autónoma de Nuevo León (UANL), in 2013. He currently holds a postdoctoral research fellowship with the Center for Quality and Manufacturing of the School of Engineering at the Tecnológico de Monterrey, campus Monterrey. His research interests cover random walk simulation using general purpose computation in graphics processing units, evolutionary and bio-inspired algorithms, global optimization and its applications in diverse fields.

Mario A. Saucedo-Espinosa received his Bachelor degree in Chemical Engineering with honors, and a Master's degree in Systems Engineering from Universidad Autónoma de Nuevo León (UANL), specializing in machine learning and artificial intelligence. He received the 2012 Mexican National Award for the Best Master Thesis on Artificial Intelligence. Currently, he is completing his Doctoral studies at Rochester Institute of Technology, USA, as both a CONACyT and a Fulbright-García Robles scholar.

Hugo Jair Escalante obtained his Ph.D. in Computer Science from the Instituto Nacional de Astrofísica, Óptica y Electrónica at México, where he is now an associate researcher. Dr. Escalante has been a member of the Mexican System of Researchers (SNI) since 2011, and co-director of ChaLearn, the Challenges in Machine Learning Organization (2011–2013). Hugo Escalante obtained the 2010 Best Ph.D. Thesis on AI Award from the Mexican Society for Artificial intelligence (SMIA). His main research interests are machine learning and computational intelligence with applications to text mining and high-level computer vision.



Karlo Mendoza received his Bachelor degree in Mathematics and Master's degree in Systems Engineering from Universidad Autónoma de Nuevo León (UANL), in 2010 and 2012, respectively. His research interests cover machine learning, data mining and genetic programming for classification.

César Emilio Villarreal-Rodríguez received his Bachelor degree in Mathematics from Universidad Autónoma de Nuevo León (UANL) in 1987. He received his Ph.D. and Master's degree in Mathematics from the Center for Research and Advanced Studies of the National Polytechnic Institute (CINVESTAV-IPN), in 1991 and 1998, respectively. Dr. Villarreal is a member of the Mexican National System of Researchers (SNI). His research interests cover applied probability models, stochastic processes and queueing systems.

Óscar L. Chacón-Mondragón received his Bachelor degree in Chemical Engineering from Universidad Autónoma de Nuevo León (UANL), in 1968. He received his Master's degree in Chemical Engineering from University of Houston and Ph.D. degree in Chemical Engineering from University of Texas - Austin, USA, in 1976 and 1987, respectively. His research interests cover large-scale optimization, nonlinear programming, interior point methods, and applications to electricity industry and power systems.

Arturo Berrones received his Bachelor degree in Physics from Universidad Autónoma de Nuevo León (UANL), in 1997. He received his Ph.D. degree in Physics from Universidad Autónoma del Estado de Morelos, Mexico, in 2002. Dr. Berrones is a member of the Mexican National System of Researchers (SNI). His research interests cover stochastic systems, statistical aspects of complex systems and interdisciplinary applications (systems engineering, bioinformatics, economy, climate research).

Article received on 06/12/2012; accepted on 09/12/2013.