Selecting Decision Variables for Artificial Bee Colony using a Self-adaptive Variable Matrix

Marco A. F. Mollinetti[1], Mário T. R. S. Neto[2], Bernardo B. Gatto[3], Takahito Kuno[1]

[1]Systems Optimization Laboratory – University of Tsukuba
Tsukuba, Japan

[2]Department of Computer Science – University of Porto
Porto, Portugal.

[3]Center for Artificial Intelligence Research (C-AIR)
Tsukuba, Japan

mollinetti@syou.cs.tsukuba.ac.jp, mariotrsn@gmail.com

bernardo@cvlab.cs.tsukuba.ac.jp, takahito@cs.tsukuba.ac.jp

Abstract. *Artificial Bee Colony (ABC) is a Swarm Intelligence optimization algorithm well-know for its versatility. The selection of decision variables to update is purely stochastic, incurring in several issues to the local search capability of the ABC. To address these issues, a self-adaptive decision variable selection mechanism is proposed with the goal of balancing the degree of exploration and exploitation throughout the execution of the algorithm. This selection, named Adaptive Decision Variable Matrix (ADVM) represents both stochastic and deterministic parameter selection in a binary matrix and regulates the extent of how much each selection is employed based on the estimation of the sparsity of the solutions in the search space. Influence of the proposed approach to performance and robustness of the original algorithm is validated by experimenting on fifteen highly multimodal benchmark optimization problems. Numerical comparison on fifteen highly multimodal benchmark optimization problems is made against the ABC and their variants and prominent population-based algorithms (e.g., Particle Swarm Optimization and Differential Evolution). Results show an improvement of the performance of the algorithms with the ADVM in the most difficult instances.*

1. Introduction

Artificial Bee Colony (ABC) is a swarm intelligence algorithm inspired by the foraging behavior of swarms of honey bees, initially designed to solve box-constrained optimization problems [Karaboga 2005a]. Many modifications to the original ABC were proposed by a multitude of researchers. Their main focus is changes to the solution update procedure, initialization and randomization of solutions [Aydın et al. 2017]. For every variant, decision variables are selected to update following a random uniform distribution where each variable has equal probability of being chosen. Clearly, this allows solutions to conduct global search in the search space. However, this selection scheme may cause several issues to the convergence and robustness of the ABC.

To address these issues, we propose a self-adaptive decision variable selection procedure named ADVM (Adaptive Decision Variable Matrix), an adaptation of the deterministic solution variable scheme developed by Mollinetti et al. [Mollinetti et al. 2018]. ADVM automatically regulates the deterministic and stochastic decision variable selection to maintain good levels of exploitation of solutions in the early stages and exploration in later stages. Levels of exploration and exploitation are monitored by estimating the population spread in the search space by calculating the Δ measure [Morrison 2013], which provides a reliable assessment of solution coverage in the search space. To validate the proposed approach, the selection mechanism is incorporated into the original ABC and several successful variants and evaluated in 15 multimodal unconstrained benchmarks problems. Results are compared to the original versions of the algorithms, as well as to some well-established optimization algorithms such as the Particle Swarm Optimization (PSO) and Differential Evolution (DE). This paper is organized as follows: Section 2 describes the original ABC while Section 3 explains the idea behind ADVM. Section 4 reports the experiments, while results are discussed in Section 5. Lastly, Section 6 outlines the conclusion of the paper and some future directions.

2. Artificial Bee Colony

Artificial Bee Colony is a Swarm Intelligence (SI) algorithm developed by Karaboga [Karaboga 2005a] based on the mathematical model of the foraging and information sharing behavior of honey bees. Initially developed to solve bound-constrained continuous optimization problems, ABC stands as a prominent SI algorithm due to its efficiency and versatility [Akay and Karaboga 2015].

The algorithm consists of four main steps: initialization, employed bees step, onlooker bees step, and scout bees step. In the first step, initial solutions are generated and specific parameters for the algorithm and problem are set. For the remaining steps, solutions are sampled and improved by local and global search procedures cyclically, until a stopping criterion is met. Three parameters regulate the algorithm: number of solutions SN; the maximum number of iterations MCN; and solution stagnation threshold Lit. A brief description of each step is explained as follows. Let $X = \{x^1, x^2, \ldots, x^n\}, n = SN$, be solution of the algorithm and $x^i_j, j = 1, \ldots, D$ be the j-th component (decision variable) of the i-th solution.

2.1. Initialization

When no partial information is provided, solution initialization is performed by drawing values from a uniform distribution (denoted by $U(\alpha, \beta)$) ranging between the feasible bounds of each decision variable max and min.

$$x^{new} = x^i_{\min_j} + \mathrm{U}(0,1)\left(x^i_{\max_j} - x^i_{\min_j}\right). \tag{1}$$

where $1 \leq j \leq D$. For each solution, a counter for unsuccessful updates is initialized.

2.2. Employed bees cycle

A randomly chosen decision variable of solution x^i is moved in a random step size in the direction of the distance between x^i and another randomly chosen solution x^k. For each solution $x^i,\ i = 1, 2, \ldots, n$, a decision variable j is updated by:

$$x_j^{i^{(t+1)'}} = x_j^{i^{(t)}} + \phi(x_j^{i^{(t)}} - x_j^{k^{(t)}}), \tag{2}$$

where t is the time step; $k(\neq i)$ is a randomly chosen index; and ϕ is a uniformly distributed real random number between $[-1, 1]$. After updating the solution by (2), a greedy selection takes place:

$$x_j^{(t+1)} = \begin{cases} x_j^{i(t+1)'} & \text{if } f(x^{i(t+1)'}) \leq f(x^{i(t)}) \\ x_j^{i(t)} & \text{otherwise} \end{cases}. \tag{3}$$

If the value of $f(x^{i(t+1)'})$ obtained by modifying decision variable x_j^i with (2) is worse than the original value $f(x^{i(t)})$, the update step is invalidated and the former decision variable is kept. In this case, no improvement was observed in that solution, so the number of unsuccessful iterations associated to this solution is increased by 1; otherwise it is reset to 0.

2.3. Onlooker bees phase

Instead of choosing every solution to be updated, each solution is selected in probability p_i by means of a weighted roulette and inputted into the same procedure of the employed bees phase (updated by (2) and (3)). The same solution can be chosen multiple times during this step, resulting in what Akay and Karaboga [Akay and Karaboga 2017] state as a positive feedback feature by intensifying local searches in the surroundings of promising solutions. Probability p_i is calculated for each solution as follows:

$$p_i = \frac{F(x^i)}{\sum_{i=1}^{SN} F(x^i)}, \tag{4}$$

where $F(x^i)$ is the objective function value of candidate solution x^i, obtained by:

$$F(x^i) = \begin{cases} \frac{1}{1+f(x^i)} & \text{if } f(x^i) \geq 0 \\ |1 + f(x^i)| & \text{otherwise.} \end{cases} \tag{5}$$

2.4. Scout bees phase

Solutions that converged to a point and no improvement in the objective function value of a candidate solution x^i ($f(x^{(t+1)}) \geq f(x^{(t)})$) is observed for more than Lit iterations consecutively are discarded to prevent premature convergence to bad local optima. If a solution is judged to be stagnated, a new candidate solution is sampled using (1). The value of Lit is commonly defined as $(SN * D)$, where D is the dimension of the problem. If multiple solutions surpassed Lit number of iterations without improving at the same iteration, the worst solution always chosen.

3. Proposed Approach

Modifications to the original ABC have been proposed by several authors whose common purpose is to address problems regarding convergence and lack of information on the neighborhood adjacent solutions. These changes consists of new update rules in the employed and onlooker bees steps, new methods for initialization or randomization of solutions in scout bees phase and inclusion of solution clustering schemes [Aydın et al.

2017]. It has been observed that the variants share a common feature among themselves, decision variable(s) x_j in the employed and onlooker steps are selected with equal probability following an uniform random distribution. Choosing components of the solution set in a purely random fashion is seen as an inherent feature of any population-based optimization method that allows solutions to "cover more ground" in the search space. Although an effortless and somehow effective measure, it causes some negative effects to the overall performance of the algorithm. Some of the most crucial issues are explained as follows. For the sake of clarity, we refer in accordance to [Locatelli and Schoen 2013] to a neighborhood $\mathcal{N}(\cdot)$ as the classical definition of an Euclidean ball centered at a point $\mathbf{x}^k$:

$$\mathcal{N}(\mathbf{x}^k) = \{\mathbf{x} \in \mathbb{R}^n : \left\|\mathbf{x} - \mathbf{x}^k\right\| \leq \epsilon\}. \tag{6}$$

Where $\|\cdot\|$ is any norm (e.g., the euclidean norm), and ϵ is a given positive quantity. Three main issues associated to stochastic decision variable choice are listed:

- **Chosen decision variable does not contribute to improvement of the solution:** Let x_w and x_z be candidate solutions in X. Let i and j ($i \neq j$) be indices of x_w and x_z, and $x_{w_i} \approx x_{z_i}$, while $x_{w_j} >> x_{z_j}$ (or vice-versa). Updates for index i would likely result in negligible updates $f(x')$ where $f(x) \leq f(x')$, while updates in j are unknown for a $f(x'')$ in neighborhood $\mathcal{N}(x'')$. Updates to i-th decision variable would result in a failed step during the greedy selection (3), contributing to a premature and needless displacement of the solution at the scout step.
- **Decision variables may never be chosen:** Let the probability of variable j to not be chosen per iteration be $P(\sim x_j) = 1 - 1/D$, then for the entire execution of the algorithm, the probability of j to not be chosen is $P_{MCN}(\sim x_j) = (1 - 1/D)^{MCN}$. Although it is clear that $P_{MCN}(\sim x_j)$ converges to 0 as MCN goes to infinity, letting the algorithm run for a small number of iterations results in a high probability for j to not be chosen, especially in high dimensional domains.
- **Updating a decision variable that deviates the solution from better local optima:** candidate solutions may converge towards deceptive points of attraction. Let x_w and x_z to be components of a candidate solution x in X. A successful update of x_z or x_w to x'_z and x'_w would lead x to accumulation points x^* and x', respectively. Furthermore, $|x_z - x'_z| \approx |x_w - x'_w|$ and $f(x^*) < f(x')$. Therefore, a successful update towards $\mathcal{N}(x'_z)$ would be beneficial, while an update towards $\mathcal{N}(x'_w)$ guides the solution towards a worse accumulation point.

An initial attempt to address these issues was made by Mollinetti et al. [Mollinetti et al. 2018], who proposed a deterministic solution selection method based on diagonals and superdiagonals of rectangular matrices. The authors showed that eliminating the randomness in the choice of index j in the movement rule boosted the performance of the ABC in multimodal domains featuring 30 or more decision variables. However, it has been observed that such improvement was obtained not without a cost: diversity of solutions in the search space has been compromised because algorithm now completely emphasizes exploitation over exploration. From this outcome, we hypothesize that a fully deterministic parameter selection biases the algorithm towards exploitation, therefore reintroducing a small degree of randomness would recover the global search capabilities of the algorithm while preserving the improvement in multimodal domains brought by the deterministic selection.

3.1. A self-adaptive decision variable selection procedure (A-DVM)

We propose an extension of the decision variable selection procedure of Mollinetti et al. [Mollinetti et al. 2018] named ADVM (Adaptive Decision Variable Matrix) that can be included in the employed and/or onlooker bees phase. As opposed to a fully deterministic selection, we reintroduce an adaptive degree of stochasticity throughout iterations by measuring the diversity of the population and using an augmented binary decision matrix. The goal of the ADVM is to improve the overall performance of the state-of-the-art of ABC for multimodal and higher dimensional problems, since it can be effectively incorporated into any ABC variant because it does not interfere with any modification.

The selection of a decision variable for each solution in the onlooker or employed step can be represented as a binary matrix P_r where each column is a solution of the solution set X arranged into a $D \times n$ matrix. Entries with 1 represent the chosen j variables of each solution to be updated by the movement rule. The deterministic selection rule of Mollinetti et al. [Mollinetti et al. 2018] constructs a binary matrix P_d where the 1's are in the main diagonal and superdiagonals offset by the dimension of the problem.

$$P_r = \begin{pmatrix} 0 & 0 & 0 & \dots & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 & \dots & 0 \\ 1 & 0 & 0 & \dots & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & 1 \\ 0 & 1 & 0 & \dots & 0 & 0 & \dots & 0 \end{pmatrix}, \qquad P_d = \begin{pmatrix} 1 & 0 & 0 & \dots & 0 & 1 & \dots & 1 \\ 0 & 1 & 0 & \dots & 0 & 0 & \dots & 1 \\ 0 & 0 & 1 & \dots & 0 & 0 & \dots & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 & 0 & \dots & 1 \end{pmatrix}.$$

The main idea behind the ADVM consists of employing random variable selection to a portion of the population while selecting the remaining variables using the deterministic scheme. ADVM constructs a binary matrix P_{am} by replacing a portion of the columns of P_r with α columns from P_d. This operation is represented by the $\oplus$ operator:

$$P_{am} = P_r \oplus \alpha P_d = \begin{pmatrix} 0 & 0 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 & \dots & 0 \\ 1 & 0 & 1 & \dots & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 & \vdots & \ddots & 0 \\ 0 & 0 & 0 & \dots & 1 & 0 & \dots & 1 \end{pmatrix}.$$

The degree of how much P_d is favored over P_r is represented by the coefficient α that is iteratively adjusted to balance the degree of exploitation and exploration while maintaining a healthy population diversity. Computation of α is given by:

$$\alpha = (1 - \Delta)K_1 + \Delta K_2, \tag{7}$$

where $\Delta \in [0, 1]$ is the measure of dispersion of the population at the current iteration and scaling parameters K_1 and K_2 are set to 0.3 and 0.7 in accordance to McGinley et al. [Mc Ginley et al. 2011]. Values of α close to 1 signify high population diversity and activate exploitation by the deterministic selection. On the other hand, values close to 0 boost exploration by random values. Since solutions in most population-based algorithms tend to concentrate around accumulation points on later stages [Locatelli and Schoen 2013], the parameter α is penalized in each iteration according to an exponential decay function as follows,

$$\alpha = \alpha e^{-\lambda(t-t')}. \tag{8}$$

In (8), λ is a decay coefficient, set to 0.01 to indicate slow decay. The amount t' is defined for fully stochastic solution selection to be performed in t' iterations. The rationale behind t' is that an initial random global search is necessary so that solution diversity during that amount of iterations is achieved. t' is given by:

$$t' = \min\left(\frac{SN * d}{\lambda_t MCN},\ \lambda_t MCN\right), \tag{9}$$

where λ_t is chosen to be 0.1.

Estimation of the population diversity Δ is given by the computation of Δ_1 and Δ_2, detailed in the following section. The steps of the ADVM are outlined in Algorithm 1.

Algorithm 1: Steps of the ADVM

if $t > t'$ **then**
  $\Delta_1 \leftarrow 0.75 - \mathbb{S}_1$
  $\Delta_2 \leftarrow 1 - \mathbb{S}_2$
  $\Delta \leftarrow \Delta_1 + \Delta_2$
  $\alpha \leftarrow \Delta * (K_2 - K_1) + K_1$
  $\alpha \leftarrow \alpha * e^{-\lambda(t-t')}$ // penalize α by (8)
  $\beta \leftarrow 1 - \alpha$
else
  $\alpha \leftarrow 0$
  $\beta \leftarrow 1$
$P_r \leftarrow BuildRandomMatrix(\beta)$
$P_d \leftarrow BuildDeterministicMatrix(\alpha)$
$P \leftarrow \alpha P_d \oplus P_r$
$X' \leftarrow$UpdateSolutions(P)

3.2. Measuring Population Dispersion

Measuring population dispersion is helpful for population-based algorithms to estimate how much the solutions are far from each other when solving highly multimodal problems. Significant contributions related to this subject are present in [Ursem 2002, Bäck and Hoffmeister 1991] who introduced the measure *SPD*, the degree of variation in a population by measuring the distance from each solution regarding as the population centroid. Moreover, McGinley et al. [Mc Ginley et al. 2011] created the concept of individual contribution to the calculation of *SPD*, which results in a new measure called *HPD*.

Diversity metrics like *SPD* and *HPD* may accurately and inexpensively identify disparities between population members. However, they do not take into account the distribution of the population throughout the search space. Many population distributions may have the same amount in their diversity metrics but different search-space coverage. Such a thing can be misleading when translating from the search-space to the solution landscape of functions. Therefore, a more robust indication for search-space coverage is desired. For the above purpose, the dispersion measure Δ designed by Morrisson [Morrison 2013] is employed:

$$\Delta = \Delta_1 + \Delta_2 = \frac{1.75 - \mathbb{S}}{1.75}, \tag{10}$$

where Δ_1, Δ_2 and $\mathbb{S}$ are given as, respectively $\Delta_1 = 0.75 - \mathbb{S}_1$, $\Delta_2 = 1 - \mathbb{S}_2$ and $\mathbb{S} = \mathbb{S}_1 + \mathbb{S}_2$. The values of $\mathbb{S}_1$ and $\mathbb{S}_2$ are obtained by measuring the moment of inertia of the solution centroid in relation to each solution. Moment of inertia is a term used in engineering problems to denote the relationship between torque and angular acceleration [Morrison 2013]:

$$I = mr^2. \tag{11}$$

For P solutions, the centroid c_i and the moment of inertia I_c of centroid c_i is:

$$c_i = \frac{\sum_{j=1}^{P} x_{ij}}{P}, \tag{12}$$

$$I_c = \sum_{j=1}^{P} (x_{ij} - c_i)^2. \tag{13}$$

The first measure $\mathbb{S}_1$ involves a quantitative assessment of the solutions around the distribution centroid. If the distribution were uniform, $\mathbb{S}_1$ is computed as follows:

$$\mathbb{S}_1 = \max_i \frac{\left[\left| I_{U_o} - I_{c_i} + P c_i^2 \right| \right]}{P}, \tag{14}$$

where the inertia of a uniform distribution is:

$$I_{U_o} = \sum_{j=1}^{P} \left(\frac{j}{P+1} \right)^2. \tag{15}$$

The measure Δ_2 indicates how much the calculation of Δ_1 is misleading when the distribution is not uniform in the search-space, since Δ_1 only verifies non-uniformity along the principal diagonal of the search space. Therefore, $\mathbb{S}_2$ is measured as follows:

$$\mathbb{S}_2 = \max \left[\frac{\left| \sum_P \chi_{c+} - \left\lceil \left(\frac{\prod_j (1-c_j)}{\prod_j \rho_j} \right) P \right\rceil \right|}{P}, \frac{\left| \sum_P \chi_{c-} - \left\lceil \left(\frac{\prod_j (c_j)}{\prod_j \rho_j} \right) P \right\rceil \right|}{P} \right]. \tag{16}$$

where $c- = \{x_j \in X | x_{ij} \leq c_i\}, c+ = \{x_j \in X | x_{ij} \geq c_i\}$, and χ is the characteristic function that returns either 0 or 1 depending on whether a solution belongs to $c-$ or $c+$

Table 1. Benchmark functions and its definitions.

Name	Dim	Range	Opt.	Name	Dim	Range	Opt.
Bukin06	10	[-10, 0]	0.0	Rosenbrock	30	[-30, 30]	0.0
Cola	17	[-4, 4]	11.7464	Schwefel06	30	[-500,500]	0.0
CrossLegTable	2	[-10,10]	-1.0	SineEnvelope	20	[-500, 500]	0.0
CrownedCross	2	[-10,10]	0.0001	Trefethen	2	[-10,10]	-3.0
Damavandi	2	[0,14]	0.0	Whitley	2	[-10.24, 10.24]	0.0
DeVilliersGlasser02	5	[1, 60]	0.0	XinSheYang03	20	[-500, 500]	0.0
Griewank	30	[-100, 100]	0.0	Zimmerman	2	[0, 100]	0.0
Rastrigin	30	[-5.12, 5.12]	0.0	–	–	–	–

4. Experiment

To verify whether the proposed approach exert any influence to the overall performance of the ABC, we experiment on 15 instances of benchmark functions designed to validate the capability of metaheuristics to handle multimodality and ruggedness. The instances are ranked in the top 30 hardest continuous optimization functions in the Global Optimization Benchmarks suite [Gavana 2019]. Each algorithm was executed 30 times with the same seed. The number of dimensions, range, and global optimum of each instance is listed in Table 1.

The ADVM was incorporated in the onlooker and employed bees phase of the following versions of the ABC: the original ABC from Karaboga [Karaboga 2005b] (ABC+ADVM), two versions of the global best guided ABC (gbestABC) from Zhu et al. [Zhu and Kwong 2010] (GBESTABC+ADVM, GBESTABC2+ADVM). These three algorithms are compared to their original counterparts (ABC, GBESTABC, GBESTABC2) and the modified ABC for multidimensional functions (MABC) from Akay and Karaboga [Akay and Karaboga 2012], as well as against well-established population-based algorithms, such as the Particle Swarm Optimization from Kennedy and Eberhart [Kennedy and Eberhart 1995], Evolutionary Particle Swarm Optimization by Miranda and Fonseca [Miranda and Fonseca 2002] and Differential Evolution (DE) [Storn and Price 1997].

The Stopping criteria for each algorithm is set as 10^5 function evaluations (FE's) or if the difference between of the best value found so far and the global optimum $f(x^*)$ value is less than 10^{-8}. Shared among all algorithms, population size is fixed at 30. For PSO, the inertia factor (w_1) is set to 0.6 and both cognitive and social parameters (w_2, w_3) as 1.8. For Differential Evolution (DE) [Storn and Price 1997] with *best1bin* strategy, *F* value was 0.5 and *CR* 0.9. For each version of the ABC: $limit$ is set to $SN * D$. For MABC, MR, SF and m are 0.4, 1, and 2.5% of maximum FE's, respectively. Lastly, selection parameters λ and λ_t of the ADVM are set to 0.001 and 0.1 respectively.

5. Results

Table 2 show the results obtained from the experiment. The statistics used for comparison are the mean, standard deviation, median, and best-worst results obtained by 30 distinct runs with different random seeds. Statistical significance between pairs is verified by the Mann-Whitney U test for non-parametric data, with confidence interval α set to 0.95. For better legibility, precision of decimals are set to 5 digits and values lower than 10^{-6} are rounded to 0. Furthermore, p-values for individual comparisons of the U test are not supplemented for the sake of brevity. Therefore, if the performance of any algorithm for a particular instance is statistically significant, it means that its p-value in the U test is less than 0.05 in the pairwise comparison against all other algorithms.

Firstly, the results show that the inclusion of the ADVM in the Cola and Rosenbrock instances resulted in a strictly worse performance than the original ABC, whose results was shown to be significantly better than all others. The poor performance of the ADVM can be attributed to the nature of the problem instances which rewards solution exploration. Additionally, we can state this case is a classical affirmation of the no-free-lunch theorem of Wolpert [Wolpert and Macready 1997]. Inferior results of the ADVM are also seen at the Bukin06 and Schewefel 06 instances. Causes of such behavior can be due to intensification of the local search mechanism that forced solutions to stay far from the sparse ridges of the surface of the functions.

Strong evidence of robustness of the ADVM for multimodal and deceptive surfaces was found in the Damavandi and DeVilliersGlasser02 instances, labeled as the first and second hardest function in the benchmark suite. Both functions feature large basins of attraction for bad local optima that is directly proportional to the dimensionality of the problem. A possible cause to the success of the ADVM in these instances can be attributed to the balance between solution exploration and exploitation that allowed the solutions to escape from the basins.

For the rest of the instances, no statistic significance that corroborated that the inclusion of the ADVM improved or worsened the performance of the original algorithm was found. However, statistical significance indicating that the ADVM was better than the PSO, EPSO and DE was found in the Rastrigin and Zimmerman instances.

6. Conclusion

In this paper, ADVM, self-adaptive decision variable selection method was proposed. The selection takes place in the employed and or onlooker bees phase and can be integrated to any variant of the ABC. ADVM attempts to balance exploration and exploitation throughout the execution of the algorithm by constructing an augmented binary matrix that represents the choice of components of the solution set to be updated. The binary matrix is obtained by a composition of portions of a binary matrix that follows the proposal of [Mollinetti et al. 2018] with another binary matrix with random 1 entries for each column. The number of columns to be used from the deterministic matrix is determined by an adaptive parameter that is calculated each iteration by estimating the Δ value, a measure of the sparsity of solutions in the search space.

Potential benefits of the ADVM to overall performance of the ABC and variants in multimodal domains were investigated by integrating the ADVM to the ABC and some of its variants and comparing against the original versions in several instances. Results indicate that the ADVM enhances the ABC capabilities of adapting itself to highly multimodal function landscapes. However, the elimination of the full global search of the stochastic selection resulted in solutions not converging towards accumulation points that are located in extreme points or ridges in the few instances where it performed poorly. Integration with ABC variants with smart restart procedures in the scout bees phase may ameliorate this issue.

Future works include in-depth sensitivity analysis and integration of the selection

Table 2. Results of the experiment for all problem instances

Problem	Algorithm	Mean	Median	Std. Dev	Best	Worst	Problem	Algorithm	Mean	Median	Std. Dev	Best	Worst	Problem	Algorithm	Mean	Median	Std. Dev	Best	Worst
Bukin06	DE	0.97893	0.91922	0.52393	0.33	2.60682	DeVilliersGlasser02	DE	352.642	64.37420	767.121	0.39534	3182.44	SineEnvelope	DE	0.47414	0.47354	0.09202	0.3289	0.69363
	PSO	0.03759	0.04883	0.01611	0.00431	0.05000		PSO	7108.73	9377.49	4258.06	**0.00000**	10647.4		PSO	7.09257	7.26654	0.57119	6.09679	7.86571
	EPSO	**0.00901**	**0.00715**	**0.00690**	**0.00057**	**0.02419**		EPSO	799.059	6.52004	2538.22	**0.00000**	10467.8		EPSO	4.74872	4.8902	0.89428	3.04664	6.09835
	ABC	0.06535	0.05000	0.03345	0.01909	0.15020		ABC	5.71168	3.20875	6.02096	0.34329	21.70150		ABC	**0.25965**	**0.22372**	0.07091	0.18175	0.39453
	MABC	0.05800	0.05000	0.046600	0.01611	0.21930		MABC	4.12467	2.74194	4.63619	0.24565	15.8204		MABC	3.01514	3.09304	0.30318	2.47955	3.57755
	GBESTABC	0.18781	0.18014	0.07922	0.05044	0.34700		GBESTABC	8.13107	4.71285	11.61610	0.84988	53.06630		GBESTABC	0.25933	0.23034	0.06796	**0.16192**	0.38841
	GBESTABC2	0.35783	0.33525	0.17333	0.12420	0.63954		GBESTABC2	5.56375	4.42650	4.64582	0.65744	21.59900		GBESTABC2	0.30696	0.30139	0.07077	0.19638	0.43203
	ABC+ADVM	0.05949	0.04990	0.04087	0.00340	0.15985		ABC+ADVM	**2.53581**	**1.93048**	**2.00250**	**0.05803**	**7.09280**		ABC+ADVM	0.30694	0.29637	0.09403	0.19931	0.56087
	GBESTABC+ADVM	0.17330	0.13761	0.08993	0.04872	0.33291		GBESTABC+ADVM	6.89254	4.77318	5.30292	1.06621	23.2227		GBESTABC+ADVM	0.26499	0.25378	**0.05339**	0.18960	**0.37096**
	GBESTABC2+ADVM	0.32394	0.34000	0.17225	0.09892	0.69858		GBESTABC2+ADVM	8.36485	7.13058	8.01541	0.40725	32.1687		GBESTABC2+ADVM	0.29561	0.29025	0.05581	0.19367	0.39693
Cola	DE	12.44390	12.39020	0.502836	11.77570	13.82660	Griewank	DE	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**	Trefethen	DE	-3.29379	**-3.30687**	0.04156	**-3.30687**	-3.14408
	PSO	16.13410	15.30270	2.44014	12.9697	22.06270		PSO	1.34282	1.30004	0.17168	1.10178	1.80049		PSO	-3.08315	-3.17611	0.21692	**-3.30687**	-2.64262
	EPSO	13.42210	13.60100	1.06166	**11.7481**	15.48070		EPSO	0.00910	**0.00000**	0.01379	**0.00000**	0.04426		EPSO	-3.27985	**-3.30687**	0.06253	**-3.30687**	-3.06263
	ABC	**12.05440**	**11.95500**	**0.22993**	11.75370	**12.54730**		ABC	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		ABC	**-3.30687**	**-3.30687**	**0.00000**	**-3.30687**	**-3.30687**
	MABC	12.83970	12.84790	0.443416	12.11390	13.61120		MABC	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		MABC	**-3.30687**	**-3.30687**	**0.00000**	**-3.30687**	**-3.30687**
	GBESTABC	12.22790	12.15260	0.30889	11.79990	13.08850		GBESTABC	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		GBESTABC	**-3.30687**	**-3.30687**	**0.00000**	**-3.30687**	**-3.30687**
	GBESTABC2	12.23520	12.25250	0.28235	11.80430	12.76960		GBESTABC2	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		GBESTABC2	**-3.30687**	**-3.30687**	**0.00000**	**-3.30687**	**-3.30687**
	ABC+ADVM	12.15250	12.08500	0.232509	11.84320	12.65390		ABC+ADVM	0.00041	**0.00000**	0.00186	**0.00000**	0.00834		ABC+ADVM	**-3.30687**	**-3.30687**	**0.00000**	**-3.30687**	**-3.30687**
	GBESTABC+ADVM	12.28230	12.18300	0.32599	11.81830	13.12510		GBESTABC+ADVM	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		GBESTABC+ADVM	**-3.30682**	**-3.30687**	0.00022	**-3.30687**	-3.30588
	GBESTABC2+ADVM	12.23630	12.21250	0.29413	11.82890	12.96570		GBESTABC2+ADVM	**0.00000**	**0.00000**	**0.00000**	**0.00000**	1.43E-05		GBESTABC2+ADVM	**-3.30687**	**-3.30687**	**0.00000**	**-3.30687**	**-3.30687**
CrossLegTable	DE	**-0.26326**	-0.08477	0.37832	**-1.00000**	-0.00611	Rastrigin	DE	0.54722	0.49748	0.60175	**0.00000**	1.98992	XinSheYang03	DE	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	PSO	-0.09723	**-0.08283**	0.21626	**-1.00000**	-0.00255		PSO	125.84700	126.20500	21.16520	88.1972	160.206		PSO	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	EPSO	-0.17534	-0.08477	0.28203	**-1.00000**	-0.07959		EPSO	45.76950	51.25510	23.05110	6.96471	99.49550		EPSO	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	ABC	-0.13062	-0.08493	0.20463	**-1.00000**	**-0.08477**		ABC	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		ABC	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	MABC	-0.10630	-0.08477	0.09595	-0.51398	**-0.08477**		MABC	74.55290	75.18170	8.78129	50.39630	87.9141		MABC	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	GBESTABC	-0.12994	-0.08477	0.20479	**-1.00000**	-0.07981		GBESTABC	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		GBESTABC	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	GBESTABC2	-0.08448	-0.08477	**0.00071**	-0.08477	-0.08283		GBESTABC2	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		GBESTABC2	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	ABC+ADVM	-0.13061	-0.08493	0.20463	**-1.00000**	**-0.08477**		ABC+ADVM	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		ABC+ADVM	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	GBESTABC+ADVM	-0.12355	-0.08477	0.20728	**-1.00000**	-0.00656		GBESTABC+ADVM	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		GBESTABC2+ADVM	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	GBESTABC2+ADVM	-0.13044	-0.08477	0.20467	**-1.00000**	-0.08283		GBESTABC2+ADVM	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		GBESTABC+ADVM	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
CrownedCross	DE	0.00173	0.00117	0.00347	**0.00010**	0.01635	Rosenbrock	DE	32.86880	25.58250	24.11440	7.49392	86.07950	Whitley	DE	0.01388	1.20E-05	0.01925	**0.00000**	0.03945
	PSO	0.01609	0.00120	0.01884	**0.00010**	0.03909		PSO	163741	122694	107137	31992.8	449777		PSO	0.03286	0.03945	0.04100	**0.00000**	0.15783
	EPSO	0.00108	0.00117	0.00033	**0.00010**	0.00125		EPSO	8.20397	8.59076	2.84563	3.16795	13.15980		EPSO	0.00591	**0.00000**	0.01445	**0.00000**	0.03945
	ABC	0.00117	**0.00010**	**0.00000**	0.00117	**0.00117**		ABC	**0.91220**	**0.15323**	**1.44928**	**0.01712**	**4.69119**		ABC	**3.02263e-05**	**0.00000**	0.00011	**0.00000**	0.00050
	MABC	0.00113	**0.00010**	0.0002	0.00027	**0.00117**		MABC	44.68880	27.22880	29.94650	23.93810	112.734		MABC	0.00203	**0.00000**	0.00881	**0.00000**	0.03945
	GBESTABC	0.00108	**0.00010**	0.00033	**0.00010**	0.00120		GBESTABC	1.70578	1.09851	1.85263	0.08547	6.11791		GBESTABC	0.00757	1.37E-05	0.01276	**0.00000**	0.03945
	GBESTABC2	0.00118	**0.00010**	1.13E-05	0.00117	0.00120		GBESTABC2	3.18224	2.5852	2.93224	0.35930	13.16040		GBESTABC2	0.00236	0.00015	0.00852	**0.00000**	0.03845
	ABC+ADVM	**0.00105**	**0.00010**	0.00032	**0.00010**	**0.00117**		ABC+ADVM	2.55989	1.04791	4.37552	0.02863	19.18300		ABC+ADVM	0.00064	**0.00000**	0.00226	**0.00000**	0.01009
	GBESTABC+ADVM	0.00113	**0.00010**	0.00024	**0.00010**	0.00125		GBESTABC+ADVM	3.46204	1.48614	4.40623	0.04360	13.87970		GBESTABC+ADVM	0.00618	**0.00000**	0.01435	**0.00000**	0.03945
	GBESTABC2+ADVM	0.00118	**0.00010**	**0.00000**	0.00117	0.00120		GBESTABC2+ADVM	4.55452	3.8697	3.73467	0.11879	14.8367		GBESTABC2+ADVM	0.00062	0.00022	**0.00095**	**0.00000**	**0.00379**
Damavandi	DE	2.00000	2.00000	**0.00000**	2.00000	2.00000	Schwefel06	DE	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**	Zimmerman	DE	0.38522	0.69869	0.35633	**0.00000**	0.70133
	PSO	2.00000	2.00000	**0.00000**	2.00000	2.00000		PSO	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		PSO	715.1750	1300	663.34500	**0.00000**	1300
	EPSO	1.90000	2.00000	0.44721	**0.00000**	2.00000		EPSO	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		EPSO	0.17464	**0.00000**	0.31035	**0.00000**	0.69858
	ABC	2.00000	2.00000	**0.00000**	2.00000	2.00000		ABC	0.17159	0.10386	0.17989	0.00056	0.59950		ABC	0.00037	**0.00000**	0.00126	**0.00000**	0.00569
	MABC	2.00000	2.00000	**0.00000**	2.00000	2.00000		MABC	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		MABC	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	GBESTABC	1.76059	2.00000	0.60220	0.00611	2.00000		GBESTABC	0.13196	0.11892	0.08783	0.01193	0.28331		GBESTABC	0.10626	0.00053	0.25554	**0.00000**	0.69923
	GBESTABC2	1.82772	2.00000	0.53654	0.02502	2.00000		GBESTABC2	0.13886	0.12197	0.10182	0.01223	0.42978		GBESTABC2	0.07062	0.00050	0.21487	5.96E-05	0.69904
	ABC+ADVM	1.80056	2.00000	0.61387	0.00258	2.00000		ABC+ADVM	0.11728	0.04898	0.17793	0.00272	0.72679		ABC+ADVM	0.00041	3.27E-05	0.00097	**0.00000**	0.00325
	GBESTABC+ADVM	1.81787	2.00000	0.56099	0.11335	2.00000		GBESTABC+ADVM	0.09749	0.10197	0.04416	0.02492	0.16431		GBESTABC+ADVM	0.10543	0.00035	0.25582	**0.00000**	0.69921
	GBESTABC2+ADVM	**1.74073**	2.00000	0.63782	0.00028	2.00000		GBESTABC2+ADVM	0.14695	0.09799	0.12129	0.02657	0.46299		GBESTABC2+ADVM	0.03531	0.00020	0.15612	1.40E-05	0.69862

mechanism to the state-of-the-art ABC used for optimization competitions and testing on large scale problems, mechanical design and power systems to further investigate the performance of the selection. Another research direction includes applying the proposed method for weight tuning of shallow networks [Gatto and dos Santos 2017, Gatto et al. 2017]. Such networks may benefit from the proposed optimization mechanism since it tackles small sample size problems featuring rough fitness landscapes.

References

Akay, B. and Karaboga, D. (2012). A modified artificial bee colony algorithm for real-parameter optimization. *Information sciences*, 192:120–142.

Akay, B. and Karaboga, D. (2015). A survey on the applications of artificial bee colony in signal, image, and video processing. *Signal, Image and Video Processing*, 9(4):967–990.

Akay, B. B. and Karaboga, D. (2017). Artificial bee colony algorithm variants on constrained optimization. *An International Journal of Optimization and Control: Theories & Applications (IJOCTA)*, 7(1):98–111.

Aydın, D., Yavuz, G., and Stützle, T. (2017). Abc-x: a generalized, automatically configurable artificial bee colony framework. *Swarm Intelligence*, 11(1):1–38.

Bäck, T. and Hoffmeister, F. (1991). Extended selection mechanisms in genetic algorithms.

Gatto, B. B. and dos Santos, E. M. (2017). Discriminative canonical correlation analysis network for image classification. In *Image Processing (ICIP), 2017 IEEE International Conference on*. IEEE.

Gatto, B. B., dos Santos, E. M., and Fukui, K. (2017). Subspace-based convolutional network for handwritten character recognition. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 1. IEEE.

Gavana, A. (2019). Global optimization benchmarks and ampgo. *Accessed Apr*.

Karaboga, D. (2005a). An idea based on honey bee swarm for numerical optimization. Technical report, Erciyes University.

Karaboga, D. (2005b). An idea based on honey bee swarm for numerical optimization. Technical report, Technical report-tr06, Erciyes university, engineering faculty, computer engineering department.

Kennedy, J. and Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of the IEEE Int. Conf. on Neural Networks*, pages 1942–1948. IEEE, IEEE Press.

Locatelli, M. and Schoen, F. (2013). *Global optimization: theory, algorithms, and applications*, volume 15. Siam.

Mc Ginley, B., Maher, J., O'Riordan, C., and Morgan, F. (2011). Maintaining healthy population diversity using adaptive crossover, mutation, and selection. *IEEE Transactions on Evolutionary Computation*, 15(5):692–714.

Miranda, V. and Fonseca, N. (2002). Epso-evolutionary particle swarm optimization, a new algorithm with applications in power systems. In *Transmission and Distribution Conference and Exhibition 2002: Asia Pacific. IEEE/PES*, volume 2, pages 745–750. IEEE.

Mollinetti, M. A. F., Neto, M. T. R. S., and Kuno, T. (2018). Deterministic parameter selection of artificial bee colony based on diagonalization. In *International Conference on Hybrid Intelligent Systems*.

Morrison, R. W. (2013). *Designing evolutionary algorithms for dynamic environments*. Springer Science & Business Media.

Storn, R. and Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4):341–359.

Ursem, R. K. (2002). Diversity-guided evolutionary algorithms. In *International Conference on Parallel Problem Solving from Nature*, pages 462–471. Springer.

Wolpert, D. H. and Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82.

Zhu, G. and Kwong, S. (2010). Gbest-guided artificial bee colony algorithm for numerical function optimization. *Applied mathematics and computation*, 217(7):3166–3173.