UMDAs for Dynamic Optimization Problems

Carlos M. Fernandes[1,3]
cfernandes@laseeb.org
[1]LaSEEB, Technical Univ. of Lisbon
Av. Rovisco Pais, 1, TN 6.21, 1049-001, Lisbon, PORTUGAL

Claudio F. Lima[2]
clima.research@gmail.com
[2]Informatics Laboratory, University of Algarve
Campus de Gambelas, 8000-117, Faro, PORTUGAL

Agostinho C. Rosa[1]
acrosa@ laseeb.org
[3]Departamento ATC, University of Granada
c/ periodista Daniel Saucedo, sn, 18071, Granada, SPAIN

ABSTRACT
This paper investigates how the *Univariate Marginal Distribution Algorithm* (UMDA) behaves in non-stationary environments when engaging in sampling and selection strategies designed to correct diversity loss. Although their performance when solving *Dynamic Optimization Problems* (DOP) is less studied than population-based *Evolutionary Algorithms*, UMDA and other *Estimation of Distribution Algorithms* may follow similar schemes when tracking moving optima: genetic diversity maintenance, memory schemes, niching methods, and even reinicialization of the probability vectors. This study is focused on diversity maintenance schemes. A new update strategy for UMDA's probability model, based on *Ant Colony Optimization* transition probability equations, is presented and empirically compared with other strategies recently published that aim to correct diversity loss in UMDA. Results demonstrate that loss correction strategies delay or avoid full convergence, thus increasing UMDA's adaptability to changing environments. However, the strategy proposed in this paper achieves a higher performance on the DOP test set when compared with other methods. In addition, the new strategy incorporates two parameters that control the diversity of the probability model.

Categories and Subject Descriptors
I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods and Search.

General Terms
Algorithms, Experimentation.

Keywords
UMDA, Dynamic Optimization Problems.

1. INTRODUCTION
The crucial and delicate equilibrium needed between exploration and exploitation in static environments and the premature convergence issue becomes even more important and complex when dealing with *Dynamic Optimization Problems* (DOPs). In DOPs, the fitness function and problem constraints are not steady. When changes occur, solutions already found may be no longer valuable and the process must engage in a new search effort (check [6] for enhanced analysis). Traditional *Evolutionary Algorithms* (EAs) [3], for instance, may encounter difficulties while solving dynamic problems if the first convergence stage reduces population diversity, thus decreasing its capability to react to sudden changes. When solving a DOP, an EA must not only maintain diversity through the first convergence stage, but also be able to escape the old solution when the fitness function changes.

Estimation of Distribution Algorithms (EDAs) [18, 23] is a class of EAs that replaces the standard crossover and mutation operators by building a probabilistic model of promising solutions and sampling from the corresponding probability distribution. During the optimization process, an EDA makes use of these models to build possible solutions to the problem (*sampling*). The probability model is then updated in a way that reflects the quality of those solutions (*selection*). Different strategies may be used at sampling and selection steps and the diversity of the model (and resulting algorithm's convergence) is strongly dependent on the chosen schemes. Traditional EDAs, like the *Univariate Marginal Distribution Algorithm* (UMDA) [21], have no means to restore diversity once it is lost. Since the variance loss occurs at sampling and selection steps, it is crucial to devise methods to slow down or avoid full convergence of the probability model. Dynamic problems, due to its traits, are particularly demanding, requiring constant diversity maintenance to avoid full convergence.

Although EDAs build probabilistic models of the genotype and ACO [10] build probabilistic models for construction paths on the phenotype, there are attempts to unify the two paradigms into a single framework [33], together with *stochastic gradient ascent* [25] and *cross-entropy* [26]. ACO and EDAs are closely related and the strategy proposed in this paper explores the similarities between the two classes of metaheuristics: a new update strategy for UMDA's probability model, based on ACO equations. The strategy is compared with the loss correction techniques presented in [8] in a DOP test set.

The present work is organized as follows. Next section provides an overview of UMDA and describes previously proposed update strategies later included in the test set. Section 3 deals with DOPs topic and describes previous work on EAs and DOPs. This paper's proposal is described in section 4. Section 5 presents and discusses the results. Finally, Section 6 concludes the paper and outlines future research.

2. UMDA
2.1 Simple UMDA
The UMDA [21] is a discrete EDA with independent variables. It starts by initializing the probability model, assigning 0.5 to each parameter γ_i of the model, meaning that the first population is randomly generated. Parameters γ_i are defined as the probability that each component takes the value 1:

GECCO'08, July 12-16, 2008, Atlanta, Georgia, USA.

$$\gamma_i \equiv P(x_i = 1) \quad (1)$$

where i = 1,…L, and L is the string length (this work considers binary strings). After initializing the model, UMDA generates N strings according to equation 2, which defines the probability of generating a string x^μ:

$$P(x_i^\mu) = \prod_{i=1}^{L} \left[\left(\gamma_i x_i^\mu + (1-\gamma_i)(1-x_i^\mu) \right) \right] \quad (2)$$

where x_i^μ is the ith component of string x^μ. The $f{\times}N$ fittest strings are then selected by truncation or any other method. The chosen solutions update the model in the following manner:

$$\gamma_i \leftarrow \frac{1}{(f \times N)} \sum_{\mu \in D_S} x_i^\mu \quad (3)$$

where D_s is the selected population. The algorithm repeats these procedures until a stop criterion is met.

Figure 1 shows UMDA's pseudo-code. Please note that once a parameter of the model loses diversity, that is, γ_i = 0 or γ_i = 1, UMDA has no means to regain it. For that reason, it is of extreme importance to avoid diversity loss, or at least to delay it, especially when dealing with DOPs. Since the loss occurs at two steps of the algorithm - sampling and selection - those are the components that must be addressed in order to build UMDAs capable of dealing with dynamic landscapes. A number of strategies have already been proposed in the past to deal with EDAs diversity issue ([19], [29] and [32], for instance). The present study is focused on those presented in [8] (and described in the next subsection).

2.2 Reducing Diversity Loss

The *permutation sampling* aims at reducing the diversity loss due to sampling. It works by simply ensuring that the generated population has exactly $N{\times}\gamma_i$ solutions with bit i equal to 1. Permutation sampling does not bias the algorithm in any way and it should be always beneficial, although it is more useful when using small populations. Following recommendations in [8], permutation sampling was used in all UMDAs tested for the present study.

Diversity loss due to selection may be reduced by changing the way the probability model is updated. In [8], the authors present four methods to correct or replace equation 3.

1) *Loss correction* (LC) intends to bias the model in order to generate a larger variance. For that purpose, probability γ_i is corrected by the following equation:

$$\gamma_i' = \begin{cases} \dfrac{1 - \sqrt{1 - 4(1-\gamma_i)\gamma_i / \mathcal{L}_S}}{2} & : \quad \gamma_i \le \dfrac{1}{2}\left(1 - \sqrt{1-\mathcal{L}_S}\right) \\ \dfrac{1 + \sqrt{1 - 4(1-\gamma_i)\gamma_i / \mathcal{L}_S}}{2} & : \quad \gamma_i > \dfrac{1}{2}\left(1 - \sqrt{1-\mathcal{L}_S}\right) \\ 0.5 & : \quad \text{otherwise} \end{cases} \quad (4)$$

where

$$\mathcal{L}_S = \frac{f \times N - 1}{f \times N - f} \quad (5)$$

is the variance loss due to sampling in a flat landscape. Figure 2a shows how loss correction changes the shape of $\gamma_i$curve. LC increases exploration in the early stage of search by setting γ_i' = 0.5 when γ_i is close to 0.5, but allows the model to converge: see how the correction diminishes when γ_i approaches 0 and 1.

Simple UMDA

Set $\gamma_i \leftarrow 1/2$ for all i = 1 . . .L;
repeat
  Sample N strings according to Eq. 2 to make a population D.
  Generate a new population Ds from D by selecting the $f{\times}N$ fittest strings.
  for i = 1 to L **do**
    update model:

$$\gamma_i \leftarrow \frac{1}{(fN)} \sum_{\mu \in D_S} x_i^\mu$$

  end for
until stop criterion

Figure 1. UMDA's pseudo-code.

2) *Laplace correction* is a Bayesian method of updating the probability model that takes into account a non-informative prior distribution: in this case, a beta distribution. Equation 3 is replaced by equation 6:

$$\gamma_i = \frac{\sum_{x^\mu \in D_S} x_i^\mu}{fN + 2\alpha} \quad (6)$$

where α determines the strength of prior's influence. Figure 2b shows the effect of *Laplace* correction on γ_i. Unlike loss correction, *Laplace* prevents UMDA from converging. As it is shown in section 5, this attribute is very important when solving DOPs, because, if all parameters γ_i converge, then the algorithm is no longer able to react to a change. On the other hand, *Laplace* correction may delay the algorithm. However, when solving DOPs, EAs speed of convergence is often sacrificed on behalf of exploration abilities; a slow algorithm is not necessarily a drawback in non-stationary environments.

3) *Incremental Laplace (iLaplace) correction* uses a beta distribution peaked at γ_i(t-1) as the prior distribution instead of assuming a prior peaked at 0.5, like Laplace correction. Equation 3 is replaced by:

$$\gamma_i(t) = \frac{\sum_{x^\mu \in D_S} x_i^\mu + 2\alpha\gamma_i(t-1)}{fN + 2\alpha} \quad (7)$$

Figure 2c shows the effect of *iLaplace* correction on γ_i. The graph exemplifies the correction by setting $\gamma_i(t-1)$ = 0.25 and $\gamma_i(t-1)$ = 0.05. Please note that when $\gamma_i(t-1)$ = 0.05, the parameter is very close to γ_i without correction when $\gamma_i \to 0$, that is, *iLaplace* does not prevent the model to converge.

4) Finally, *boundary correction* (BC) prevents UMDA from converging but it does not change the probability distribution except when the parameters are close to 0 or 1. This method corrects γ_i in the following manner:

$$\gamma_i' = \begin{cases} \beta & : \quad \gamma_i \le \beta \\ 1-\beta & : \quad \gamma_i > \beta \\ \gamma_i & : \quad \text{otherwise} \end{cases} \quad (8)$$

Parameter β is suggested to be set to 1/L, where L is the string length. This scheme avoids convergence by setting minimal and maximal values for γ_i. Boundary correction effect on γ_i is represented in figure 2d. (This strategy may be combined with other loss correction methods.) Like Laplace correction, which also prevents full converge, BC is very effective on DOPs – see section 5. Experiments made for this paper suggest that UMDA's full convergence must be avoided when solving DOPs.

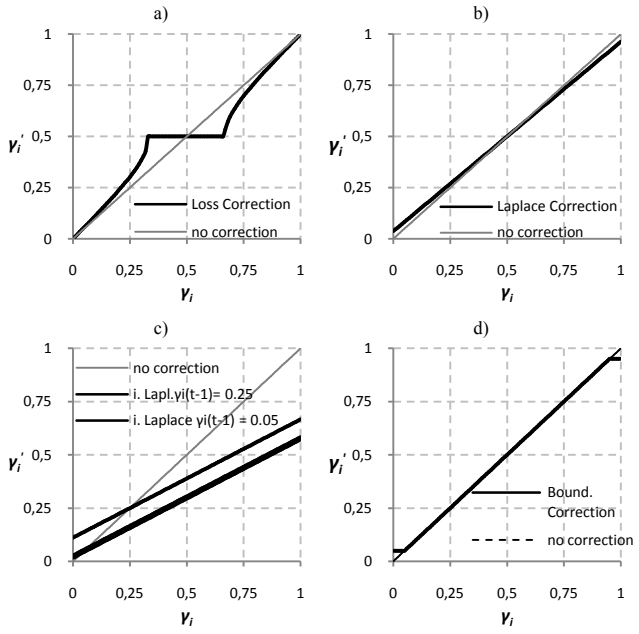

Figure 2. LC, *Laplace*, *iLaplace* ($\alpha = 2$) and BC ($\beta = 0.05$) effect on γ_i. $N = 10$.

3. EVOLUTIONARY ALGORITHMS IN DYNAMIC ENVIRONMENTS

3.1 Dynamic Optimization Problems

Optimization problems are said to be dynamic when there is a change in the fitness function, problem instance or restrictions, thus making the optimum change as well. Due to its adaptive characteristics, EAs seem to be a suitable candidate for solving DOPs, but they typically converge to an optimum and thereby lose the diversity necessary to adapt to a change in the environment when such a change occurs. Over the past two decades, a number of authors have addressed the problem of convergence and subsequent loss of adaptability in many different ways. However, most of these approaches may be grouped into one of the following three categories [6].

React on Changes. The EA is run in standard fashion, but as soon as a change in the environment has been detected, explicit actions are taken to increase diversity and thus facilitating the shift to the new optimum. Techniques such as *Hypermutation* [9] pursue the first category, keeping the whole population after a change but increasing population diversity by drastically increasing the mutation rate for some number of generations.

Maintaining diversity. Convergence is avoided all the time and it is hoped that a spread-out population can adapt to changes more easily. The *Random Immigrants GA* (RIGA) [13] is an example of a strategy belonging to this category. Every generation, RIGA replaces part of the population by randomly generated individuals. This guarantees the introduction of new genetic material in every time step and avoids the convergence of the whole population to a narrow region of the search space. Other examples of diversity oriented EAs for DOPs may be found [17] and [22].

Memory schemes. The EA is supplied with a memory to recall useful information from past generations. Memory may be provided in two general ways: *implicitly* [12] by using redundant representations, or *explicitly* by introducing an extra memory and formulating strategies to deposit and retrieve solutions later [7].

Some recent proposals have been made using a Swarm Intelligence [5] approach to attempt to solve dynamic problems. Examples of Swarm Intelligence applied to dynamic environments may be found in [15] and [24]. In [11], Fernandes et al., developed the *Binary Ant Algorithm* (BAA), based on ACO [10], to take advantage of ACO's ability to solve combinatorial DOPs and generalize it to binary DOPs. However, this method may be also regarded as a type of EDA, because BAA creates the possible solutions to a problem via transition probability vectors.

3.2 EDAs on DOPs

Although DOPs have been a subject of EAs research for the last two decades, only recently the DOP issue has started to raise a strong interest on EDAs' researchers.

The *Population Based Incremental Learning* (PBIL) [4] − one of the first EDAs − was used in [31] to solve DOPs created by a problem generator proposed by the same authors. A comparison of several versions of PBIL with simple GAs and RIGAs is provided. In [30], Yang proposes the UMDA with enhanced memory and the results of the experiments show that memory is efficient in dynamic environments. In addition, a combination of memory and Random Immigrants [13] for the UMDA is studied. Abbass et al. [1] were the first to introduce the *Extended Compact Genetic Algorithm* (ECGA) [16] to solve problems in dynamic environments. Their approach is based on random restarts of the population at each change so that diversity in the population can be increased at the beginning of each new environment. Additionally, Abbass et al. proposed a slightly different approach that used the probabilistic model from the previous generation when the population was to be restarted. The ECGA with random restart was later extended [27] to include substructural niching [28]. In [14], Ghosh and Muehlenbein apply the UMDA to dynamic environments by introducing mutation whenever the fitness changes. This approach (like [1]) assumes that it is possible to detect environmental changes. Our proposal maintains diversity through the run so that UMDA is able to respond to a change without additional methods to detect changes in the environment.

4. NEW PROPOSAL

Recently, there have been attempts to unify ACO [10] and EDAs into the same framework [33]. These theoretical efforts are fundamental to fully understand and eventually bring together some features shared by the two classes of metaheuristics, but obvious similarities between the two paradigms arise from a close observation of some algorithms. For instance, PBIL [4] holds an update scheme very similar to the pheromone update and

RE UMDA

Set $\gamma_i \leftarrow 1/2$ for all i = 1 . . .*L*; Set $\tau_i^{0,1} \leftarrow 0$ for all i = 1 . . .*L*; Set α and β
repeat
  sample *N* strings according to Eq. 2 to make a population *D*.
  generate new population *Ds* by selecting the $f \times N$ fittest strings.
  update pheromone (equations 9 and 10)
  evaporate (equation 12)
  for i = 1 to *L* **do**
    update model (equation 11)
  end for
until stop criterion met

Figure 3. Reinforcement-Evaporation (RE) UMDA.

evaporation of ant algorithms. BAA [11], as referred in previous section, was inspired by ACO but it is clearly a kind of EDA. (For an exhaustive analysis of the similarities between ant algorithms and EDAs please refer to [33].) This paper introduces a new technique to reduce UMDA's variance loss due to selection, based on ACO [10] pheromone update and transition probability equations: the *Reinforcement-Evaporation* loss correction (figure 2), or RE correction as it will be designated in the remaining of this work.

Consider two vectors τ_i^0 and τ_i^1 that are updated in each time step as defined by equations 9 and 10:

$$\tau_i^1\ (t) \leftarrow \tau_i^1\ (t) + \frac{1}{(fN)} \sum_{\mu \in D_S} x_i^\mu \tag{9}$$

$$\tau_i^0\ (t) \leftarrow \tau_i^0\ (t) + \left(1 - \frac{1}{(fN)} \sum_{\mu \in D_S} x_i^\mu\right) \tag{10}$$

with

$$\tau_i^0\ (0) = 0 \qquad \text{and} \qquad \tau_i^1\ (0) = 0$$

where $i = 1,\ldots L$, and $\tau_i^0(0) = 0$ and $\tau_i^1(0) = 0$. These vectors emulate ACO's pheromone maps and act as kind of memory, allowing UMDA to incorporate information from prior distributions into the current parameters. The parameters γ_i are then updated in the following manner:

$$\gamma_i \leftarrow \frac{(\tau_i^1)^\alpha}{(\tau_i^0 + \tau_i^1)^\alpha} \tag{11}$$

where α is a parameter that controls the relative weights of τ_i^0 and τ_i^1 on the probability γ_i. Before the update stage, vectors $\tau_i^{0,1}$ are "evaporated" according to equation 12:

$$\tau_i^{0,1}(t) = \tau_i^{0,1}(t-1) \times (1 - \beta) \tag{12}$$

where β is the evaporation rate. Please note that setting $\beta = 1$ implies that the vectors $\tau_i^{0,1}$ are set to 0 at the beginning of each time step, thus meaning that the previous equations are reduced to equation 13:

$$\gamma_i \leftarrow \frac{\left(\sum_{\mu \in D_S} x_i^\mu\right)^\alpha}{(f \times N)^\alpha} \tag{13}$$

Figure 4 shows RE correction with different α values (β is set to 1*)*. With low α values, the method approaches random search, since γ_i is kept close to 0.5 (with $\alpha = 0$, UMDA performs random search: γ_i is always 0.5). Increasing α cause the model to relax its exploration efforts and the diversity is reduced. When α approaches 1, the model becomes very close to the standard update strategy (equation 3). Parameter β also controls diversity: with $\beta < 1$, UMDA incorporates prior distributions in the probability model. Please note that setting $\alpha = 1$ and $\beta = 1$ will result in the simple UMDA depicted in figure 1.

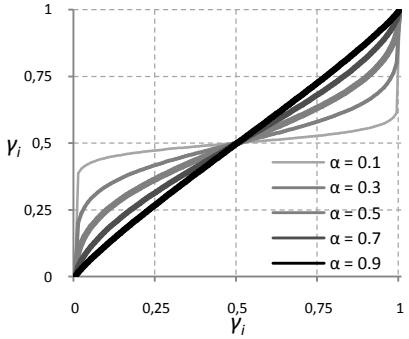

Figure 4. RE correction effect with different α values ($\beta = 1$).

RE correction does not prevent UMDA's complete convergence. However, this method may be combined with BC without changing its general behavior except when the parameters γ_i become closer to 0 or 1. Results will show that the UMDA combining RE and BC correction outperforms not only BC alone, but also other strategies and combinations – see next section.

5. EXPERIMENTS

5.1 Experimental Setup

The test environment proposed in [31] was chosen to test the efficiency of the different diversity correction techniques on DOPs. Given a stationary problem $f(\boldsymbol{x})$ ($\boldsymbol{x} \in \{0,1\}^l$) where l is the chromosome length, dynamic environments may be constructed by applying a binary mask $\mathbf{M} \in \{0,1\}^l$ to each solution before its evaluation in the following manner:

$$f(\boldsymbol{x}, t) = f\left(\boldsymbol{x}\ \text{XOR}\ \mathbf{M}(k)\right) \tag{14}$$

where t is the generation index, $k = t/\tau$ is the period index and $f(\boldsymbol{x}, t)$ is the fitness of solution $\boldsymbol{x}$. $\mathbf{M}(k)$ can be incrementally generated as follows:

$$\mathbf{M}(k) = \mathbf{M}(k-1)\ \text{XOR}\ \mathbf{T}(k) \tag{15}$$

where $\mathbf{T}(k)$ is an intermediate binary mask for every period k. This mask $\mathbf{T}(k)$ has $\rho \times l$ ones, where ρ is a value between 0 and 1.0 which controls the intensity or severity of change. Notice that $\rho = 0$ corresponds to a stationary problem since T vectors will carry only 0's and no change will occur in the environment. On the other hand, $\rho = 1$ guarantees the highest degree of change: for instance, if a solution to a problem is a string of 1's, then the dynamic solution oscillates between a string of 1's and a string of 0's. Therefore, by changing ρ and τ in the previous set of equations it is possible to control two of the most important features when testing algorithms on DOPs: severity (ρ) and speed (τ) of change [2]. *Onemax* and *Royal Road R1* were chosen to test UMDA's correction strategies.

The function *Onemax* counts the number of 1's in a string and it may be formalized by the following equation:

$$f(\boldsymbol{x}) = u(\boldsymbol{x}) \tag{14}$$

where u(x) is the unitation function.

Royal Road functions [20] were specifically designed to study GAs' performance on the level of building blocks interactions, and are widely used in GAs test and analysis. From the set of Royal Road functions, function R1 was selected. This Royal Road problem is defined as:

$$f(\boldsymbol{x}) = \sum_{i=1}^{q} c_s \delta_s(\boldsymbol{x}) \tag{15}$$

where q is the number of schemata $S = \{s_1,\ldots, s_q\}$ and, for the function used in this study, $\delta_s(x)$ is set as 1 if $\boldsymbol{x}$ is an instance of S

and 0 otherwise, and $c_s = 8$ for all s; a 64-bit string was used and each schema is composed of 8 contiguous bits.

5.2 Diversity on Flat Landscapes

Before proceeding into an investigation of UMDAs on DOPs, a brief analysis of RE diversity on a flat landscape is provided. The flat landscape exemplifies UMDA's diversity loss. Since all strings have the same fitness, there is no bias in the landscape leading the population towards a particular direction. However, after a certain number of generations, the model fully converges, generating identical solutions. Figure 5 shows how standard UMDA converges on a flat landscape ($\alpha = 1$). When reducing α, diversity increases and convergence is delayed, as expected.

Plotting RE diversity with $\alpha = 1$ and different evaporation rates β results in the graphic depicted in figure 6. As parameter α, β value influences the diversity of the model. However, it is of much importance how the diversity is generated, and not only the diversity itself. Decreasing α and β has the same general effect on the model's variance, but may cause rather distinct effects when optimizing a function. While α increases diversity by "pulling" γ_i towards 0.5, β act as a kind of memory, moving the model towards diversity or convergence, depending on previous states. Nevertheless, the general idea to retain at this point is that both α and β may be tuned in order to control UMDA's variance loss.

5.3 Dynamic Optimization Problems

For each DOP several degrees of severity (τ) and speed (ρ) were set: $\tau = 10$ and $\tau = 100$; $\rho = 0.05$, $\rho = 0.6$ *and* $\rho = 0.95$. Each algorithm was executed for 10 periods of environmental changes and 30 runs were performed for each configuration of (τ, ρ) and each algorithm. The fitness was measured and averaged over the 30 runs. Population size was set to $N = 120$ and parameter $f = 0.1$ in all the experiments. String length is $L = 300$ (*Onemax*) and $L = 64$ (*Royal Road*). Optimal values are $f(\mathbf{x}) = 300$ (*Onemax*) and $f(\mathbf{x}) = 64$ (*Royal Road*).

As stated before, RE correction holds no means to avoid full convergence. For that reason, the first study investigates RE behavior with different α and β values and compares the proposed strategy with *LC* and *iLaplace*, which are loss correction techniques that allow UMDA to converge. Parameter α of *iLaplace* correction is suggested in [8] to be set $\alpha = 2$. However, since this study deals with DOPs, the optimal parameter may be different and other values were tested. As matter of fact, increasing α leads to a better performance and a general analysis of the results suggest $\alpha = 10$ to be used in dynamic *Onemax* and *Royal Road* functions.

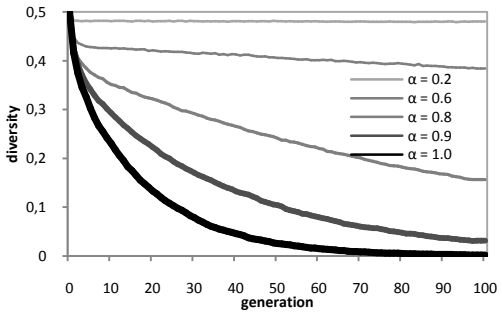

Figure 5. Diversity loss in a flat landscape. Comparing RE correction with different α values and $\beta = 1$ (meaning total evaporation). Parameters: $N = 20$, $f = 0.5$ and $L = 100$.

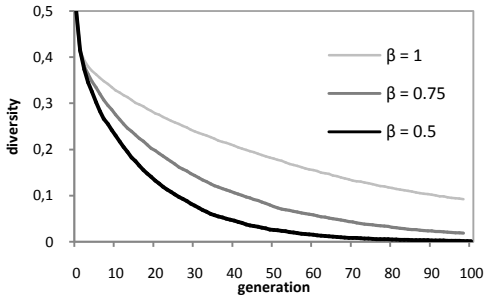

Figure 6. Diversity loss in flat landscape. Comparing RE with different β values ($\alpha = 1$). Parameters: $N = 20$, $f = 0.5$, $L = 100$.

EDAs (and other EAs) performance analysis on DOPs must be addressed in a distinct manner from static environments' usual procedure. Dynamic behavior throughout the run must be examined, rather than the final convergence. For that purpose, the evaluation of UMDA's performance is done by measuring the mean *best-of-generation* values (this is the standard procedure for DOPs). In addition, the progression of *best-of-generation* values may be plotted in a graph, thus helping to understand how the algorithm reacts to changes in the environment.

Figure 7 shows RE performance on *Royal Road* and indicates that introducing *RE* and varying α and β may improve standard UMDA performance (remember that *RE* correction is equivalent to the standard UMDA update strategy if $\alpha = 1$ and $\beta = 1$). In addition, decreasing evaporation (β) appears to result in a higher optimal α value – compare both graphics. Please note also that the effect varies with DOP conditions (τ, ρ values), which is an expected behavior since varying the speed and severity of change results in rather distinct problem conditions.

The results (mean *best_of_generation*) attained by *iLaplace*, *LC* and *RE* UMDAs are shown in table 1. Two configurations of *RE* were tested: *RE*1, with no evaporation ($\beta = 1$) and $\alpha = 0.6$; and *RE*2, with $\alpha = 0.8$ and $\beta = 0.5$. A statistical comparison was

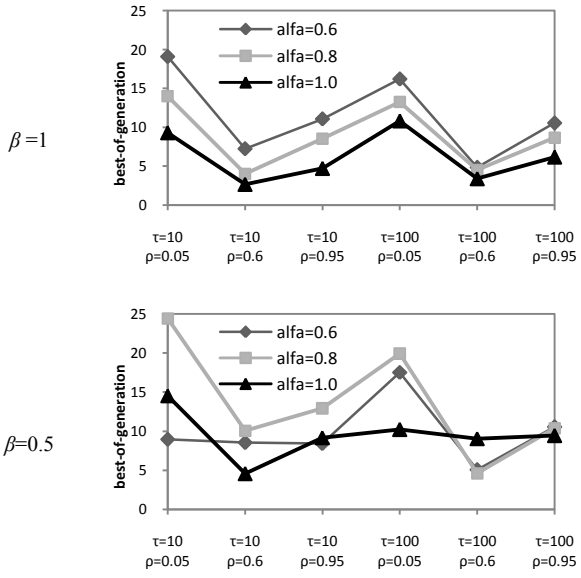

Figure 7. *RE* performance on Royal Road. Comparison of *RE* strategy with different α and β values.

Table 1. Comparison of *iLaplace* (α = 10), LC, *RE*1 ($\alpha = 0.6, \beta = 1$) and *RE*2 ($\alpha = 0.8, \beta = 0.5$). Best results in bold.

τ		Onemax				Royal Road			
ρ		iLap.	LC	RE1	RE2	iLap.	LC	RE1	RE2
10	0.05	**249.32** **±2.08**	239.28 ±2.91	235.26 ±3.32	240.37 ±1.81	21.86 ±4.97	11.71 ±5.49	19.07 ±4.89	**24.42** **±3.52**
10	0.60	176.05 ±1.52	159.93 ±2.23	179.30 ±1.89	**186.86** **±1.00**	6.52 ±0.87	3.60 ±0.79	7.26 ±1.35	**10.07** **±0.60**
10	0.95	166.47 ±0.55	154.42 ±0.71	173.54 ±0.90	**175.26** **±0.47**	11.81 ±2.10	6.96 ±2.42	11.08 ±1.99	**12.91** **±1.58**
100	0.05	**246.98** **±3.18**	244.91 ±3.10	242.90 ±3.75	245.67 ±2.65	16.56 ±4.17	13.23 ±3.62	16.20 ±3.50	**19.90** **±3.19**
100	0.60	160.40 ±1.85	161.57 ±2.24	158.53 ±2.01	**160.88** **±1.78**	**4.66** **±0.82**	3.67 ±0.95	4.86 ±0.74	4.63 ±0.42
100	0.95	153.40 ±0.52	153.73 ±0.59	151.02 ±0.70	**154.59** **±1.24**	**10.97** **±2.30**	6.60 ±2.55	10.56 ±1.92	10.38 ±1.80

carried out by *t*-tests with 58 degrees of freedom at a 0.05 level of significance and the result is depicted on table 2. Symbols +, − and ~ were used to illustrate *t*-test results: + means that strategy 1 is significantly better than strategy 2, ~ means that performance is statistically equivalent and − means that strategy 1 is significantly worse. The results demonstrate that, in general, *RE* strategies are more efficient than *iLaplace* and LC when solving dynamic *Onemax* and *Royal Road* functions, especially with medium and high severity ($\rho = 0.6$ *and* $\rho = 0.95$). It is also clear that *iLaplace* outperforms LC which means that the latter strategy may be converging to quickly after leaving the $\gamma_i = 0.5$ plateau. The difference between *iLaplace* and *RE* performance is larger when evaporation is introduced (*RE*2). Please note that *iLaplace* also holds a memory scheme, since the model incorporates the information from the previous distribution. However, *RE* appears to be more effective in generating diversity by keeping memory in a kind of pheromone field.

When testing strategies that avoid full convergence (*Laplace* and BC), UMDA's performance on dynamic *Onemax* and *Royal Road* is greatly improved when compared to *iLaplace*, *LC* and even *RE*. In order to make fairly comparisons it is necessary to supply all the strategies with abilities to avoid convergence: BC provides such tool. Tables 3 show the results of *iLaplace*, LC and *RE* combined with BC. BC alone and *Laplace* were also tested (*Laplace* was tested in combination with BC, but the results were statistically similar; adding BC to *Laplace* appears redundant, at least in this test set.)

A general analysis of the results presented in table 3 (and the statistically tests in table 4) concludes that both *RE* 1 and 2 improve other strategies' performance in a significant number of (τ, ρ) configurations, especially *RE*1, which appears to outperform

Table 2. Statistical analysis of the results in table 1.

		Onemax						Royal Road					
τ		10	10	10	100	100	100	10	10	10	100	100	100
ρ		0.05	0.6	0.95	0.05	0.6	0.95	0.05	0.6	0.95	0.05	0.6	0.95
1	2												
RE1	iLap	−	+	+	−	−	−	−	+	~	~	~	~
RE1	LC	+	+	+	−	−	−	+	+	+	+	+	+
RE2	iLap	−	+	+	~	~	+	+	+	+	+	~	~
RE2	LC	~	+	+	~	~	+	+	+	+	+	+	+
iLap	LC	+	+	+	~	~	~	+	+	+	+	+	+

*RE*2 in these tests. It seems that evaporation is not so useful when adding BC to *RE* correction. Please note that α was increased in both *RE*, because BC already guarantees diversity. Another aspect worth notice is that the difference between *RE* and other strategies is more obvious when speed is lower and severity is higher. For *Onemax* and (τ, ρ) = (10, 0.05), for instance, *RE*1 and *RE*2 are outperformed by other strategies; but when ρ increases both *RE* are statistically better than all other strategies – see table 4. Since low severity rates do not require as much diversity as higher ρ values, *RE* is not as useful as when the functions experience dramatic changes. Furthermore, lower speed means that the algorithm is given more time to converge to regions near the optimal solution, and, if that happens, it will require stronger diversity mechanisms to escape previous optimal regions of the landscape. *RE* strategy, if correctly tuned, provides the means to escape those regions.

Although different parameter values have been tested, a fine-tuning of the *RE* strategy was not performed. Further research is needed in order to understand how α and β affect UMDAs with *RE*. In addition, a wider range of problems is required in the test bench if the aim is investigating optimal α and β values. However, a few hints are given by the experiments performed for this section. Apparently, a strong diversity maintenance by drastically reducing α and β values from those corresponding to standard strategy ($\alpha = 1$; $\beta = 1$) is not required for DOPs when adding BC to *RE*. *RE* seems to perform well when decreasing α to values around 0.8, maintaining β close to 1 (as a matter of fact, best results were achieved with $\beta = 1$ but, as already stated, no fine-tuning was performed and β values between 0.5 and 1 may provide better performance).

Table 3. Results on *Onemax* and *Royal Road*. *iLaplace* (α = 2). *RE*1($\alpha = 0.8; \beta = 1$) *RE* 2($\alpha = 0.9; \beta = 0.5$). Best results in bold.

τ		Onemax					
ρ		iLap+BC	Laplace	LC+BC	BC	RE1+BC	RE2+BC
10	0.05	**275.55** **±1.11**	278.96 ±1.09	275.14 ±0.97	279.22 ±0.95	274.81 ±0.88	265.71 ±1.22
10	0.60	176.80 ±1.38	177.77 ±1.87	179.50 ±1.99	178.81 ±1.64	**184.53** **±1.70**	184.38 ±1.28
10	0.95	161.05 ±0.81	160.05 ±0.65	162.72 ±0.63	159.85 ±0.65	168.79 ±0.80	**170.89** **±0.59**
100	0.05	297.88 ±0.07	298.13 ±0.04	297.75 ±0.06	**298.15** **±0.08**	297.67 ±0.07	296.93 ±0.08
100	0.60	260.90 ±0.36	267.89 ±0.41	267.82 ±0.33	267.42 ±0.34	**269.32** **±0.38**	254.24 ±0.39
100	0.95	224.56 ±0.69	238.87 ±0.55	239.37 ±0.54	237.78 ±0.42	**244.27** **±0.48**	214.68 ±0.77
τ		Royal Road					
ρ		iLap+BC	Laplace	LC+BC	BC	RE1+BC	RE2+BC
10	0.05	36.78 ±5.90	39.36 ±6.86	40.20 ±5.28	35.58 ±6.20	**45.19** **±4.00**	40.31 ±3.39
10	0.60	8.72 1.26	10.26 ±1.88	10.70 ±1.32	8.78 ±1.38	**13.27** **±1.24**	11.06 ±1.07
10	0.95	17.37 ±2.44	18.67 ±2.08	18.84 ±2.40	17.01 3.87	**19.12** **±1.55**	18.27 ±2.00
100	0.05	56.64 ±3.30	59.30 ±1.95	60.43 ±1.50	57.50 ±2.66	**61.68** **±0.79**	60.70 ±1.24
100	0.60	18.98 ±2.19	26.49 ±2.33	30.89 ±1.49	22.14 ±2.26	**39.90** **±1.08**	28.61 ±1.30
100	0.95	21.75 ±1.80	23.05 ±1.38	26.08 ±1.27	20.53 ±1.96	**34.87** **±1.11**	25.66 ±1.14

Table 4. Statistical analysis of the results in table 3.

		Onemax						Royal Road					
τ		10	10	10	100	100	100	10	10	10	100	100	100
ρ		0.05	0.6	0.95	0.05	0.6	0.95	0.05	0.6	0.95	0.05	0.6	0.95
1	2												
RE1+BC	iLap+BC	∼	+	+	∼	+	+	+	+	+	+	+	+
RE1+BC	Laplace	−	+	+	−	+	+	+	+	∼	+	+	+
RE1+BC	LC+BC	−	+	+	∼	+	+	+	+	∼	+	+	+
RE1+BC	BC	−	+	+	∼	+	+	+	+	+	+	+	+
RE2+BC	iLap+BC	−	+	+	−	−	−	+	+	∼	+	+	+
RE2+BC	Laplace	−	+	+	−	−	−	∼	+	∼	+	+	+
RE2+BC	LC+BC	−	+	+	−	−	−	∼	∼	∼	∼	−	∼
RE2+BC	BC	−	+	+	−	−	−	+	+	∼	+	+	+

Figure 8 shows the effect of α on the dynamics of UMDA ($\beta = 1$) when solving the DOP Royal Road with different severity rates and $\tau = 100$. Starting with $\alpha = 1$ (standard UMDA), it is clear that reducing α leads to an improved adaptation to changing environments. When $\alpha = 0.8$, the system is more able to approach *Royal Road*'s optimal value and, for $\rho = 0.6$ and $\rho = 0.95$, the curves generated at each period are very similar. When $\rho = 0.05$, *RE* with $\alpha = 0.8$ is faster at tracking and acquiring the optimum.

Figure 9 compares *RE* and *Laplace* dynamic behavior. *RE* correction is faster than *Laplace* at tracking the optimum when $\rho = 0.05$. With higher severity, *RE* behaves similarly in all the environments, while *Laplace* clearly looses diversity during the first environment, and the following performance is affected.

6. SUMMARY AND CONCLUSIONS

This paper presents a new diversity loss correction for UMDA to tackle *Dynamic Optimization Problems* (DOP). The new method is based on the ACO equations and thus it was named *Reinforcement-Evaporation* (RE) correction. The strategy has been tested and compared to other methods on two DOPs with different speed and severity settings. Results show that *RE* correction outperforms standard UMDA on the proposed test set. In addition, *RE* was compared with *loss correction* and *incremental Laplace correction*, outperforming both in the majority of speed/severity configurations. *RE* was also combined with *boundary correction* in order to avoid the full convergence of the model. Again, comparisons of different correct strategies show that *RE* behaves better in the majority of the test set. Further research will be focused on the parameters in order to determine potential optimal regions of the parameter space. It is important to understand if the diversity generated by α and β has different characteristics, and if that difference affects performance and dynamic behavior. Tests on deceptive and dynamic functions will also be performed in order to investigate how the model behaves in such a test environment. Finally, possible extensions of *RE* to other EDAs will be considered.

7. ACKNOWLEDGMENTS

Authors wish to thank FCT, *Ministério da Ciência e Tecnologia*, the Research Fellowships SFRH/BD/18868/2004 (partially supported by *Fundação para a Ciência e a Tecnologia*, ISR/IST plurianual funding, through the POS_Conhecimento Program that includes FEDER funds), SFRH/BD/16980/2004, PTDC/EIA /67776-2006. This work also been supported by the Spanish MICYT project TIN2007-68083-C02-01.

8. REFERENCES

[1] Abbass, H. A., Sastry K., and Goldberg, D. E. 2004. Oiling the wheels of change: The role of adaptive automatic problem decomposition in non-stationary environments. Technical Report 2004029, Illigal, University of Illinois at Urbana-Champaign, IL, USA

[2] Angeline, P. 1997. Tracking Extrema in Dynamic Environments. Proceedings of the 6th International Conf. on Evolutionary Programming, Springer, 335-345.

[3] Back, T. 1996. Evolutionary Algorithms in Theory and Practice. Oxford University Press.

[4] Baluja, S. 1994. Population-Based Incremental Learning: A Method for Integrating Genetic Search Based Function Optimization and Competitive Learning. Technical Report CMU-CS-94-163, Carnegie Mellon University, USA.

[5] Bonabeau, E., Dorigo, M., and Threraulaz, G. 1999. Swarm intelligence: from natural to artificial systems. Oxford University Press.

[6] Branke J. 2002. Evolutionary optimization in dynamic environments. Kluver Academic Publishers.

[7] Branke, J. 1999. Memory enhanced evolutionary algorithms for changing optimization problems. Proceedings of the 1999 Congress on Evolutionary Computation, IEEE, 1875-1882.

[8] Branke, J., Lode, C., and Shapiro, J. 2007. Adressing sampling errors and diversity loss in UMDA. Proceedings of the 2007 Genetic and Evolutionary Computation Conference, ACM, 508-515.

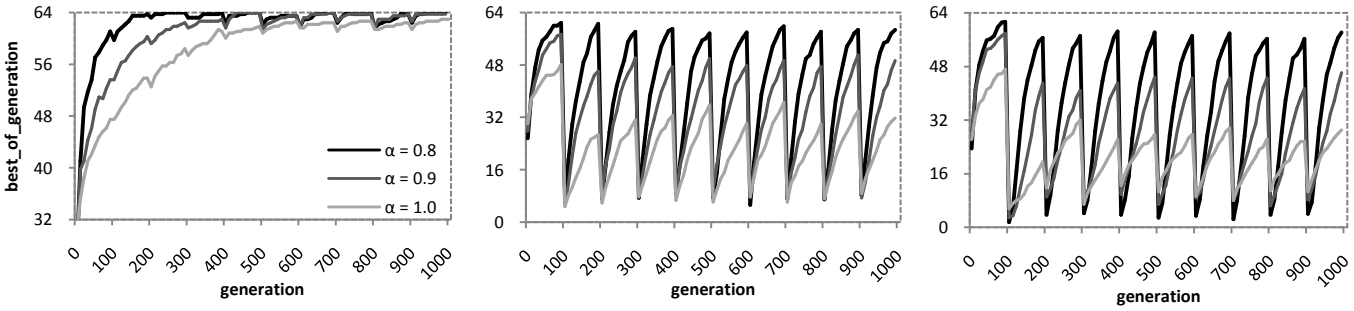

Figure 8. Dynamic behavior of UMDA with *RE* correction for different α values. Dynamic Royal Road with $\tau = 100$. Parameter β was set to 1, meaning that $\alpha = 1$ curves correspond to the standard UMDA update strategy.

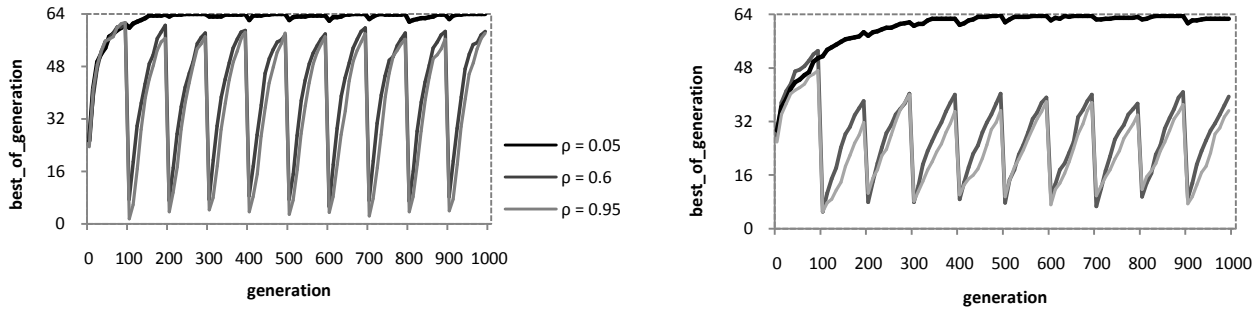

Figure 9. Comparing *RE*+BC and *Laplace*. Dynamic Royal Road with $\tau = 100$. *RE*: $\alpha = 0.8$; $\beta = 1$

[9] Cobb H.G. 1990. An investigation into the use of hypermutation as an adaptive operator in GAs having continuous, time-dependent non-stationary environments. Technical Report AIC-90-001, Naval Research Laboratory, Washington, USA.

[10] Colorni, A., Dorigo, M., and Maniezzo, V. 1992. Distributed Optimization by Ant Colonies. Proceedings of the 1st European Conference on Artificial Life, MIT Press, Cambridge, MA, 134-142.

[11] Fernandes, C.M., Rosa, A.C., and Ramos, V. 2007. Binary Ant Algorithm. Proceedings of the 2007 Genetic and Evolutionary Computation Conference, ACM, 41-48.

[12] Goldberg, D.E. and Smith, R.E. 1987. Nonstationary function optimization using genetic algorithms with dominance and diploidy. Proceedings of the 2nd International Conference on Genetic Algorithms, ACM, 59-68.

[13] Grefenstette, J.J. 1992. Genetic algorithms for changing environments. Proceedings of Parallel Problem Solving from Nature, North-Holland, 137-144.

[14] Ghosh, A. and Muehlenbein, H. 2004. Univariate Marginal Distribution Algorithms for non-stationary optimization problems, International Journal of Knowledge-based and Intelligent Engineering Systems 8(3), 129-138.

[15] Guntsch, M. and Middendorf, M. 2002. Applying population based ACO to dynamic optimization problems. Proceedings of 3rd International Workshop ANTS, LNCS 2463, Springer, 111-122.

[16] Harik, G. R. 1999. Linkage learning via probabilistic modeling in the ECGA, IlliGAL Report No. 99010, Illigal, University of Illinois at Urbana-Champaign, IL, USA.

[17] Huang, C., Kaur, J., Maguitman, A., and Rocha, L. 2007 Agent-based model of genotype editing. Evolutionary Computation 15(3), 253-289.

[18] Larrañaga, P. and Lozano, J.A. 2002. Estimation of distribution algorithms: A new tool for evolutionary computation. Kluwer Academic Publishers, Boston.

[19] Mahnig, T. and Muhlenbein, H. 2001. Optimal mutation rate using bayesian priors for estimation of distribution algorithms. Stochastic Algorithms: Foundations and Applications, LNCS 2264, 33-48.

[20] Mitchell, M. 1991. When will a GA outperform Hillclimbing? Advances in Neural Information Processing Systems 6, 51-58.

[21] Muehlenbein, H. 1996. From recombination of the genes to the estimation of distribution. Parallel Problem Solving from Nature I, North-Holland, 178-187.

[22] Ochoa, G., Madler-Kron, C., Rodriguez, R., and Jaffe, K. 2005. Assortative mating in genetic algorithms for dynamic problems. Proceedings of the 2005 EvoWorkshops, LNCS 3449, 617-622.

[23] Pelikan, M., Goldberg, D., and Lobo, F. 1999. A survey of optimization by building and using probabilistic models. Computational Optimization and Applications 21(1), 5-20.

[24] Ramos, V., Fernandes, C.M., Rosa, A.C. 2006. On self-regulated swarms, societal memory, speed and dynamics. Proceedings of ALifeX, MIT Press, 393-399.

[25] Robbins, H. and Monro, S. 1951. A stochastic approximation method. Annals of Mathematics Statistics 22, 400-407.

[26] Rubinstein, R.Y. 1999. The cross-entropy method for combinatorial and continuous optimization. Methodology and Computing in Applied Probability 1(2), 127-190.

[27] Sastry, K., Abbass, H. A., Goldberg, D. E. 2004. Sub-structural niching in non-stationary environments. Proceedings of the 17th Australian Joint Conference on Artificial Intelligence, LNAI 3339, 873-885.

[28] Sastry, K., Hussein, A., Abass, A., and Goldberg, D. E. 2005. Sub-structural niching in estimation of distribution algorithms. Proceedings of the 2005 Genetic and Evolutionary Computation Conference, ACM, 671-678.

[29] Shapiro, J. L. 2005. The detailed balance principle in estimation of distribution algorithms. Evolutionary Computation, 13(1), 99–124.

[30] Yang, S. 2005. Memory-enhanced univariate marginal distribution algorithms. Proceedings of the 2005 Congress on Evolutionary Computation, IEEE, 2560-2567.

[31] Yang, S. and Yao, X. 2005. Experimental study on PBIL algorithms for dynamic optimization problems. Soft Computing 9(11), 815-834.

[32] Wu, H. and Shapiro, J. L. 2006. Does overfitting affect performance in estimation of distribution algorithms? Proceedings of the 2006 Genetic and Evolutionary Computation Conference, ACM, 433–434.

[33] Zlochin, M., Birattari, M., Meuleau, N., and Dorigo, M. 2004. Model-based search for combinatorial optimization: A critical survey. Annals of Operations Research 131, 373-395.