Evolution Strategies with q-Gaussian Mutation for Dynamic Optimization Problems

Renato Tinós
Grupo de Informática Biomédica, Depto. de Física e Matemática
FFCLRP - Universidade de São Paulo (USP), Brazil
rtinos@ffclrp.usp.br

Shengxiang Yang
Department of Information Systems and Computing
Brunel University, U.K.
shengxiang.yang@brunel.ac.uk

***Abstract*—Evolution strategies with q-Gaussian mutation, which allows the self-adaptation of the mutation distribution shape, is proposed for dynamic optimization problems in this paper. In the proposed method, a real parameter q, which allows to smoothly control the shape of the mutation distribution, is encoded in the chromosome of the individuals and is allowed to evolve. In the experimental study, the q-Gaussian mutation is compared to Gaussian and Cauchy mutation on four experiments generated from the simulation of evolutionary robots.**

***Keywords*-Evolution strategies, q-Gaussian mutation, evolutionary algorithm, dynamic environments, robotics**

I. Introduction

Evolution strategies (ESs) have been succesfully employed in complex optimization problems in recent years. ESs were proposed in the 1960's to optimize candidate solutions composed of real-valued parameters [2], and are a good choice when gradient based methods present bad performance due to rugged fitness landscapes in continuous optimization problems. Similar to other evolutionary algorithms (EAs), new candidate solutions are generated in ESs by using a stochastic mutation operator. The parameters of the mutation operator can be modified by self-adaptation during the evolutionary process, which provides an intrinsic mechanism for adaptation to eventual changes in the problem and makes the use of ESs interesting for *dynamic optimization problems* (DOPs) [1], [7], [12].

DOPs have attracted increasing attention from the EA community as many optimization problems in real world are DOPs. In DOPs, the evaluation function, dimension of the search space, and/or the constraints of the problem may not be fixed [3]. The simplest approach when a problem changes is to restart the optimization process. However, the optimization procedure may require a substantial computational effort and/or be slow, e.g., in the optimization of control laws in evolutionary robots [10]. When the new solution after the change in the problem is related to the previous solution, the search procedure based on previous solutions can save substantial processing time.

In ESs, new candidate solutions are traditionally generated by mutation generated from Gaussian distribution [2]. However, in recent years, researchers have argued that the use of mutation distributions with longer tails and infinite second moment in ESs can be useful in allowing the population escape from local optima in multimodal problems. For example, in [13], the Cauchy distribution was employed to generate new candidate solutions. The use of mutation taken from heavy tail distributions implies jumps of scale-free sizes, eventually allowing to reach distant regions of the search space faster. This property is interesting for DOPs too, as it can allow the population to escape faster from local optima located close to the best solution before the change. However, when mutation distributions with longer tails are employed, less local candidate solutions are generated, and the convergence to the new optima can be slower. Thus, the use of one or other mutation distribution may result in very different performance on a DOP. Here, the performance of mutation generated from two different distributions (Gaussian and Cauchy) are compared in experiments with DOPs.

The main contribution of this paper is the investigation of the use of the q-Gaussian mutation in ESs to address DOPs. In the q-Gaussian mutation, which was previously used in evolutionary programming [11], self-adaptation is employed, not only to control the mutation strength parameter, but also to control the mutation distribution. The q-Gaussian distribution allows to smoothly control the shape of the distribution by setting a real parameter q and can reproduce either finite second moment distributions, like the Gaussian distribution, or infinite second moment distributions, like the Cauchy distribution. Here, the real parameter q is encoded in the chromosome of the individuals and is allowed to evolve. This way, in the q-Gaussian mutation, the decision on which mutation distribution shape should be used (and when) is made by self-adaptation.

The rest of this paper is organized as follows. The q-Gaussian mutation is briefly discussed in Secion II. The ES with q-Gaussian mutation is presented in Section III. The experimental study with DOPs generated from the simulation of evolutionary robots is presented in Section IV. In the experimental study, ESs with q-Gaussian, Gaussian, and Cauchy mutation are compared. Finally, the conclusions of the paper are presented in Section V.

II. The q-Gaussian Mutation

When mutation is applied in ESs, the i-th candidate solution $\vec{x}_i$ is generated from an m-dimensional solution $\vec{x}_i$

978-0-7695-4210-2/10 $26.00 © 2010 IEEE
DOI 10.1109/SBRN.2010.46

according to:

$$\vec{\tilde{x}}_i = \vec{x}_i + \mathbf{C}\ \vec{z}, \tag{1}$$

where $\vec{z}$ is an m-dimensional vector generated from a random distribution with zero mean and the matrix $\mathbf{C}$, in the standard ES, is a diagonal matrix composed of the elements of vector $\vec{\sigma} = [\sigma(1)\ \sigma(2) \ldots \sigma(m)]^{\mathrm{T}}$, which defines the mutation strength in each coordinate of the search space.

For the q-Gaussian mutation generated from anisotropic distribution ([11] proposes the use of a mutation generated from isotropic distribution), the vector $\vec{z}$ is generated by sampling m independent q-Gaussian random variables, i.e., the q-Gaussian distribution is employed instead of the Gaussian distribution (Gaussian mutation) [2] or the Cauchy distribution (Cauchy mutation) [13]. In the q-Gaussian distribution, the real parameter q controls the shape of the random distribution, which allows to smoothly and continuously change the shape of the distribution. For $q < 5/3$, the second order moment is finite and for $q = 1$, the q-Gaussian distribution reproduces the usual Gaussian distribution. For $q < 1$, the q-Gaussian distribution has a compact form, and decays asymptotically according to a power law for $1 < q < 3$. When $q = 2$, the q-Gaussian distribution reproduces the Cauchy distribution [8]. In this paper, the generalized Box-Müller method proposed in [9] is employed to generate the q-Gaussian random variables.

In this work, based on the mutation strength self-adaptation [2] (see next section), the parameter q_i of each individual i is added to its chromosome and is multiplicatively updated as follows:

$$\tilde{q}_i = q_i \exp\left(\tau_q \mathcal{N}(0,1)\right), \tag{2}$$

where τ_q denotes the standard deviation of the Gaussian distribution and $\mathcal{N}(0,1)$ denotes a sample variable taken from the Gaussian distribution with zero mean and standard deviation one. This way, different distributions can be reproduced during the evolutionary process.

III. ES WITH q-GAUSSIAN MUTATION

In ESs, two main selection procedures are usually employed. In the (μ, λ)-ES, a population of μ parents creates $\lambda > \mu$ offspring. The best μ offspring are then selected to compose the next population. In the $(\mu + \lambda)$-ES, the new population is composed of the best μ individuals obtained from the union of the μ parents and λ offspring. It can be observed that while the $(\mu + \lambda)$-ES is elitism-based, i.e., it always preserves the best individuals from one generation to the next one, the (μ, λ)-ES is not elitism-based. Thus, for DOPs, a procedure to detect the changes in the problem should be used when the $(\mu + \lambda)$-ES is employed.

In this paper, the (μ, λ)-ES with q-Gaussian mutation, as described in the previous section, is used. While the q-parameter is updated according to Eq. (2), the mutation strength parameter of each element $j = 1, \ldots, m$ of the vector $\vec{\sigma}_i$ is updated according to:

$$\tilde{\sigma}_i(j) = \sigma_i(j) e^{\tau_b \mathcal{N}(0,1)_i + \tau_c \mathcal{N}(0,1)}, \tag{3}$$

where τ_b denotes the standard deviation of the Gaussian distribution used to generate the random deviate $\mathcal{N}(0,1)_i$, which is common for all elements of the vector $\vec{x}_i$, and τ_c is the standard deviation of the Gaussian distribution used to generate the separated random deviate $\mathcal{N}(0,1)$ for each element $j = 1, \ldots, m$. The parameters τ_b and τ_c, as suggested by the theoretical and empirical work in [2], are defined by $\tau_b = \frac{b}{\sqrt{2m}}$ and $\tau_c = \frac{c}{\sqrt{2\sqrt{m}}}$, where b and c are positive real numbers. Here, τ_q (Eq. 2) is given by:

$$\tau_q = \frac{a}{\sqrt{2m}}, \tag{4}$$

where a is a positive real number.

Algorithm 1 ES(μ,λ) with q-Gaussian mutation (qGES)

1: Initialize the population of individuals ($\vec{x}_k$, $\vec{\sigma}_k$, q_k) for $k = 1, \ldots, \mu$
2: Evaluate the individuals ($\vec{x}_k$, $\vec{\sigma}_k$, q_k) for $k = 1, \ldots, \mu$
3: **while** (stop criteria are not satisfied) **do**
4: Use recombination to generate the individuals ($\vec{\tilde{x}}_i$, $\vec{\tilde{\sigma}}_i$, $\tilde{q}_i$) for $i = 1, \ldots, \lambda$ from the individuals ($\vec{x}_k$, $\vec{\sigma}_k$, q_k) for $k = 1, \ldots, \mu$
5: **for** $i \leftarrow 1$ to λ **do**
6: **if** $rand(0,1) \geq r_q$ **then**
7: Update the mutation strength vector $\vec{\tilde{\sigma}}_i$ according to Eq. (3).
8: **else**
9: Update the parameter $\tilde{q}_i$ according to Eq. (2)
10: **end if**
11: $\vec{\tilde{x}}_i \leftarrow \vec{\tilde{x}}_i + \mathbf{C}_i\ \vec{z}$, where $\vec{z}$ is a q-Gaussian vector generated from distribution with parameter $\tilde{q}_i$ and $\mathbf{C}_i = \mathrm{diag}(\vec{\tilde{\sigma}}_i^{\mathrm{T}})$
12: **end for**
13: Evaluate the offspring ($\vec{\tilde{x}}_i$, $\vec{\tilde{\sigma}}_i$, $\tilde{q}_i$) for $i = 1, \ldots, \lambda$
14: Select, to compose the new population with individuals ($\vec{x}_k$, $\vec{\sigma}_k$, q_k), the μ individuals with best fitness from the population composed of the offspring ($\vec{\tilde{x}}_i$, $\vec{\tilde{\sigma}}_i$, $\tilde{q}_i$) for $i = 1, \ldots, \lambda$
15: **end while**

The algorithm ES(μ,λ) with q-Gaussian mutation, called $qGES$, is presented in Algorithm 1. The main difference of the ES presented in Algorithm 1 from the standard ES and the fast ES [13] lies in that, in qGES, the q-Gaussian mutation is employed (step 11) instead of the Gaussian mutation (in the standard ES) or Cauchy mutation (in the fast ES), and a procedure to update the parameter q is adopted (steps 6, 8, 9 and 10).

IV. EXPERIMENTAL STUDY

In the experiments presented here, DOPs are generated through the simulation of evolurionary mobile robots navigating in dynamic environments or with faults. In evolutionary robotics, artificial evolution is the fundamental force in the adaptation and design of robots and their control laws. Particularly here, ESs are employed to adjust the synaptic weights in an Elman *artificial neural network* (ANN) used to control simulated mobile robots. In the experiments, the robots are simulated in DOPs using a modified version of the Evorobot simulator developed by Nolfi [6].

The four experiments presented in this section are generated from the experiment proposed in [4], where a Khepera robot with eight infrared distance sensors (six sensors in one side and two in another side of the robot), two ambient light sensors, and one floor brightness sensor navigates in an arena. The robot has a measurable limited energy, which is recharged every time the robot crosses a battery recharge area. The battery recharge area is indicated by a different color of the floor and by a light source mounted in a tower inside the area.

In the experiments, the fitness function is given by the accumulated averaged rotation speed of the two wheels of the robot during its life time, i.e., while the battery has energy and while the robot does not crash into a wall or an obstacle, considering a maximum limit of 60 seconds. A fully charged battery allows the robot to move for 20 seconds. The fitness is not computed while the robot remains in the battery recharge area. Although the fitness function does not specify that the robot should return to the battery recharge area, the individuals that develop the ability to find it and periodically return to it while exploring the arena without hitting the obstacles accumulate more fitness. The ANN used to control the robots has 17 inputs (8 infrared sensors, 2 light sensors, 1 floor brightness sensor, 1 sensor for the battery energy, and 5 recurrent units), 5 hidden neurons, and 2 outputs (2 motors in the wheels of the robot).

In the first three experiments (Exp. A, Exp. B, and Exp. C), we are interested in investigating the reconfiguration of the robot after faults [10]. In these experiments, the environment where the robot evolves is switched every $\tau = 25$ or $\tau = 50$ generations (called the change cycle duration) between two configurations. In both configurations, the size of the arena is 40cm$\times$45cm. The first configuration is free of obstacles (default arena). The position of the light source and recharge area are changed in the second configuration, and a cylindrical obstacle is added.

In Exp. A (faults in the light sensors), the responses of the light sensors are reduced by a factor which is changed every τ generations. In Exp. B (faults in motor 2), the power of the second motor of the robot is reduced by a factor changed in each τ generations. The factor applied in the response of the light sensors (Exp. A) and in the power of the second motor (Exp. B) in each one of the 10 change cycles (including the first τ generations) is given by $\vec{v}_f = \{1.0, 0.5, 0.2, 0.9, 0.7, 0.4, 0.8, 0.6, 0.1, 0.7\}$, e.g., only 50% of the power computed by the respective ANN's output is applied in the second motor in the second change cycle (between generations $\tau + 1$ and 2τ) of Exp. B. In Exp. C (faults in the infrared sensors), we are interested in investigating the reconfiguration after intermittent faults in the infrared sensors. During the evolutionary process, the responses of the infrared sensors of the robots are affected by two faults, which are switched every $\tau = 25$ or $\tau = 50$ generations. In the first fault, the responses of the six infrared sensors located in one side of the robot are set to zero when it is affected by the fault. In the second fault, the responses of the remaining two sensors (located in the other side) are set to zero. This way, the robot should learn how to navigate using different sets of sensors in each change cycle.

The last experiment (Exp. D) was carried out to investigate how a changing environment affects the learning process. Environmental changes frequently occur in real world problems, where some aspects of the environment are frequently modified. Besides, robots are frequently evolved in simulations to avoid damage, and, when a satisfactory behaviour is reached, the ANN employed to control the simulated robot are transferred to the real ones. In Exp. D (changing environment), the environment where the robot is evolving is changed after $\tau = 25$ or $\tau = 50$ generations. The robot evolves for the first τ generations in the default arena, which is changed in its dimensions and in the number of cylindrical obstacles present in the environment every τ generations.

For all experiments, the number of changes during the evolutionary process was set to ten. In the runs, the individuals of the initial population were randomly chosen. The evolving robot always starts in a fixed position on the arena, but the initial orientation was randomly varied in a range of 10 degrees. A white noise with a range equal to 0.05 was added to the measures generated by the infrared and light sensors. The individuals are represented by a vector of integer values corresponding to the synaptic weights of the ANN. Following [4], in each generation, the 20 best individuals (μ) are selected and each one generates 5 children ($\lambda = 100$) by mutation (recombination was not used).

In order to compare the three types of mutation, three ESs were executed 50 times (with different random seeds) in each experiment with $\tau = 25$ and $\tau = 50$ generations. In the first algorithm, qGES, the parameter q of the q-Gaussian mutation is allowed to evolve during the optimization process (Algorithm 1). In the other two algorithms, the parameter q is fixed, i.e., it starts with a given value and is not modified during the evolutionary process. In algorithm GES, $q = 1$, i.e., the q-Gaussian distribution reproduces the Gaussian distribution. In algorithm CES, $q = 2$, i.e., the Cauchy distribution is reproduced by the q-Gaussian distribution. Thus, the three types of mutation, q-Gaussian, Gaussian, and Cauchy are compared in the experiments.

In all experiments, the initial q-Gaussian parameter q in qGES was set to 1.0 (a value where the Gaussian distribution is reproduced), $r_q = 0.8$, and the minimum and maximum values of the q-Gaussian parameter q were set to 0.8 and $0.8e$, respectively, i.e., values respectively smaller and higher than the values of q where the Gaussian and Cauchy mutations are reproduced. The parameters a and b, respectively used to compute τ_b and τ_c, were set to 1, while in Eq. (4), $a = 1.5$.

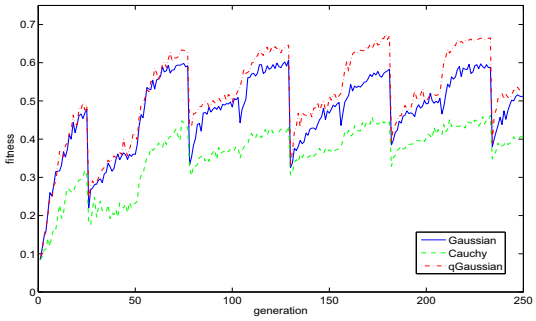

Figure 1. Averaged fitness of the best individual in Experiment A (faults in the light sensors of the evolutionary robot) for $\tau = 25$ for GES (where the Gaussian mutation is reproduced), CES (where the Cauchy mutation is reproduced), and qGES (q-Gaussian mutation).

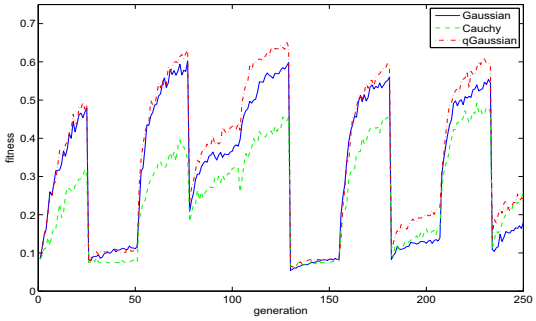

Figure 2. Averaged fitness of the best individual in Experiment B (faults in the motor 2 of the evolutionary robot) for $\tau = 25$.

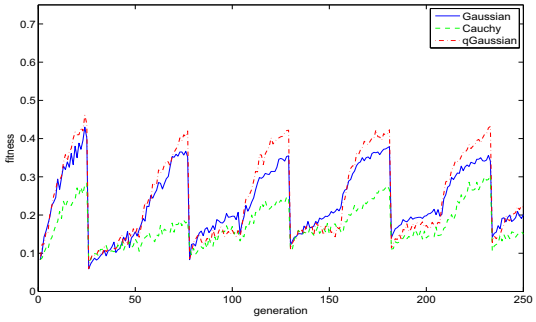

Figure 3. Averaged fitness of the best individual in Experiment C (faults in the infrared sensors of the evolutionary robot) for $\tau = 25$.

Figures 1, 2, 3, and 4 respectively show the mean best fitness found in each change cycle averaged over 50 runs for the four experiments with $\tau = 25$ generations, while Figures 5, 6, 7, and 8 respectively show the mean best fitness found in each change cycle averaged over 50 runs for the four experiments with $\tau = 50$ generations.

Table I presents the experimental results with respect to the mean best fitness found in each change cycle (offline

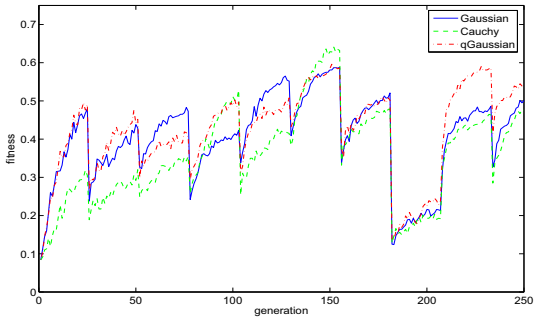

Figure 4. Averaged fitness of the best individual in Experiment D (changing environment) for $\tau = 25$.

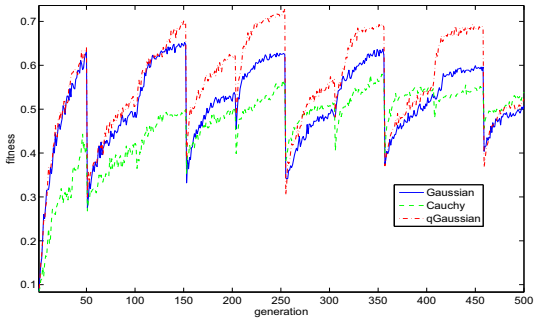

Figure 5. Averaged fitness of the best individual in Exp. A for $\tau = 50$.

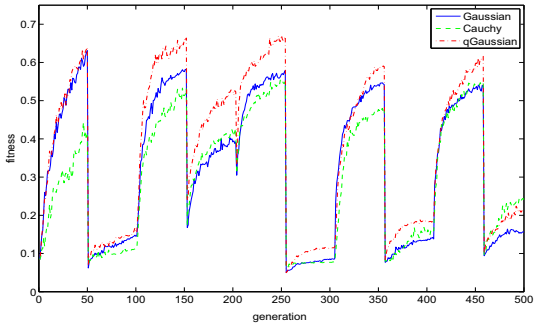

Figure 6. Averaged fitness of the best individual in Exp. B for $\tau = 50$.

performance) averaged over 50 runs. The mean best fitness reached in each change cycle for run j is given by:

$$\bar{f}_j = \frac{1}{\tau} \sum_{i=1}^{n_c} f_{ij}^* \tag{5}$$

where τ is the change cycle duration, $n_c = 10$ is the number of changes in run j, and f_{ij}^* is the best fitness found in change cycle i in run j.

In Table II, the statistic comparison of the algorihtms regarding the mean best fitness found in each change cycle (Eq. (5)) is carried out by the Wilcoxon Signed Rank Test

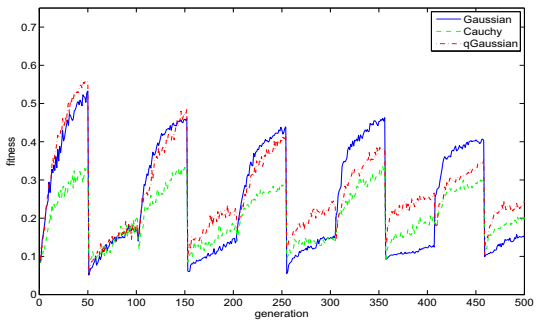

Figure 7. Averaged fitness of the best individual in Exp. C for $\tau = 50$.

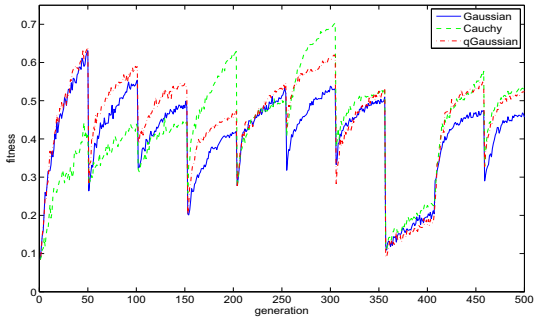

Figure 8. Averaged fitness of the best individual in Exp. D for $\tau = 50$.

Table I
RESULTS OF THE MEAN BEST-OF-GENERATION FITNESS FOUND IN EACH CHANGE CYCLE FOR THE EXPERIMENTS WITH EVOLUTIONARY ROBOTS.

τ	Exp.		GES	CES	qGES
25	A	median	7.39E-001	6.46E-001	7.57E-001
		mean	6.65E-001	5.62E-001	6.96E-001
		std	1.84E-001	2.15E-001	1.72E-001
	B	median	6.95E-001	6.49E-001	7.25E-001
		mean	6.22E-001	5.50E-001	6.70E-001
		std	1.93E-001	2.20E-001	1.57E-001
	C	median	5.83E-001	4.25E-001	6.33E-001
		mean	5.06E-001	4.61E-001	5.47E-001
		std	2.17E-001	1.69E-001	2.07E-001
	D	median	6.46E-001	6.72E-001	6.80E-001
		mean	6.34E-001	6.63E-001	6.69E-001
		std	1.14E-001	8.15E-002	6.50E-002
50	A	median	7.43E-001	7.20E-001	7.70E-001
		mean	6.66E-001	6.06E-001	7.22E-001
		std	1.93E-001	2.18E-001	1.49E-001
	B	median	6.59E-001	6.06E-001	6.97E-001
		mean	5.55E-001	5.42E-001	6.32E-001
		std	2.08E-001	2.14E-001	1.63E-001
	C	median	5.47E-001	3.87E-001	5.85E-001
		mean	4.82E-001	4.48E-001	5.23E-001
		std	2.03E-001	1.89E-001	1.70E-001
	D	median	6.43E-001	6.78E-001	6.77E-001
		mean	6.08E-001	6.40E-001	6.50E-001
		std	1.47E-001	1.20E-001	9.50E-002

[5]. Table II shows the p-value of the Wilcoxon Signed Rank Test for each experiment, which indicates the significance for testing the null hypothesis that the difference between

Table II
STATISTICAL COMPARISON OF ALGORITHMS GES, CES, AND QGES REGARDING THE MEAN BEST-OF-GENERATION FITNESS FOUND IN EACH CHANGE CYCLE.

τ	Exp.	qGES - GES	qGES - CES
25	A	1.95E-002 (s+)	2.56E-003 (s+)
	B	1.20E-001 (+)	1.05E-003 (s+)
	C	1.94E-001 (+)	2.10E-002 (s+)
	D	4.57E-002 (s+)	7.32E-001 (+)
50	A	1.01E-002 (s+)	2.99E-003 (s+)
	B	2.95E-002 (s+)	1.92E-002 (s+)
	C	1.44E-001 (+)	4.12E-002 (s+)
	D	5.91E-002 (+)	9.88E-001 (-)

the matched samples of the results regarding Alg. X and Alg. Y comes from a distribution with median equal to zero. For each experiment, the result regarding the comparison Alg. X - Alg. Y is shown, in parentheses, as "=" when the values of the median of Alg. X and Alg. Y are equal. When the values of the median are different but the p-value is higher than 0.05, i.e., the test indicates that the hypothesis that the median of the difference between the results are zero cannot be rejected at the 5% level, the result is respectively shown as "+" when the median of Alg. Y is smaller than the median of Alg. X and "−" when the median of Alg. Y is higher than the median of Alg. X. Otherwise, when the result is statistically significant, the result is respectively shown as "$s+$" or "$s-$" when the median of Alg. Y is smaller or higher than the median of Alg. X.

In the experiments, ESs find, in the first change cycle (before the first change) of most runs, weights of the ANNs (individuals) that allow the simulated robots navigate in the environment, and periodically return to the battery recharge area when the battery charge is low. It can be observed in the figures, that GES (where the Gaussian mutation is reproduced) presents better resuts than CES (where the Cauchy mutation is reproduced) in the first change cycle. In the simulated evolutionary robot, large modifications in the vector of synaptic weights cause a large change in the current navigation strategy found by the evolutionary process. This way, mutation distributions with smaller tails produce better results in the first change cycle as more solutions are generated close to the current best solution. In qGES, smaller values of q are selected by self-adaptation during the first change cycle. Hence, the performance of qGES is close to the performance of GES in the first change cycle.

When changes ocurr in the environment, new navigation strategies should be found. In Exp. A, the changes in the problem are, in general, small, as the changes in the light of the environment cause small modifications in the weights of the ANNs. As a result, it can be observed that GES presents better mean results than CES in most change cycles. However, it is possible to observe that CES presents better results that GES in some change cycles where the changes in the light were more drastic (see vector $\vec{v}_f$). It can be observed that qGES presents the best result in this

experiment, with the performance significantly better than that of other algorithms. In qGES, the value of q is modified according to the problem: small values are generally selected when the changes in the problem are small, while larger values of q are selected when the changes are drastic.

Similar results can be found in other experiments. In experiments B and C, the changes in the problems are severer, which explains the smaller values of mean final fitness found in each change cycle. Particularly in Exp. C, the reconfiguration of the navigation strategy after the change in the problem is very difficult. The robot learns to navigate in the direction that it has more sensors (called here front of the robot). When the problem changes, all the sensors in the front of the robot are affected by the fault. This way, the robot should learn how to navigate in the new environment after losing all infrared sensors in its front. It can be observed that it is difficult for the ESs to find new navigation strategies for change cycles 2, 4, 6, 8, and 10, as the robot tries to keep the navigation strategy learned in the first change cycle. In the figures, it is possible to observe that CES presented better results than GES in change cycles 2, 4, 6, 8, and 10 for $\tau = 50$. For $\tau = 50$, it is harder to find a new strategy for the even change cycles as the fitness value found in change cycle 1 is high and the diversity of the population is reduced due to the longer change cycle duration. This way, longer jumps ocasionally occured in order to allow scaping from the best solution before the change (local optimum), which explains the better results of CES. It can be observed that qGES presented the best results in experiments B and C because the shape of the distribution is self-adapted during the evolutionary process. Over the runs, the values of q are higher as mutation distributions with longer tails are eventually selected by the population in order to allow the individuals to escape from the local optima (solutions before the change) generated by changing the problem.

In Exp. D, small and larger changes are presented in the problem. As a consequence, GES presents better results than CES in some change cycles, while CES presents better results in others. In Exp. D, qGES presents good results too, despite of the worse result (compared to CES) for $\tau = 50$.

V. Conclusions

The use of self-adaptation of the mutation distribution in ESs is proposed for DOPs in this paper. The smooth change in the shape of the mutation distribution is obtained by using the q-Gaussian mutation. In the proposed method, the decision on which distribution is more indicated for a given problem and at a given moment of the evolutionary process is minimized by letting the proposed ES to decide which mutation distribution should be used. The experimental results indicate that this property can be useful for the ES for DOPs as, after the environmental changes, the parameter q can be increased resulting in a higher number of long jumps (like the Cauchy mutation), which can help the population to escape from local optima generated by the changes in the problem. In later stages, after the environmental changes, the parameter q reaches small values, which improves the local search (like the Gaussian mutation). In the future, other control methods for the q parameter should be investigated.

Acknowledgment

This work was supported by FAPESP, Brazil, and by the Engineering and Physical Sciences Research Council (EP/E060722/1), UK.

References

[1] D. V. Arnold and H.-G. Beyer. Random Dynamics Optimum Tracking with Evolution Strategies. In J.J. Merelo, P. Adamidis, H.-G. Beyer, J. L. Fernández-Villacañas, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature*, p. 3–12, 2002.

[2] H.-G. Beyer and H. S. Schwefel. Evolution strategies: a comprehensive introduction. *Natural Computing*, 1:3–52, 2002.

[3] J. Branke. *Evolutionary Optimization in Dynamic Environments*. Kluwer, 2001.

[4] D. Floreano and F. Mondada. Evolution of homing navigation in a real mobile robot. *IEEE Trans. on Syst., Man, and Cybern. B.*, 26(3):396–407, 1996.

[5] S. García, D. Molina, M. Lozano, and F. Herrera. A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 Special Session on Real Parameter Optimization. *Journal of Heuristics*, 15:617–644, 2009.

[6] S. Nolfi and D. Floreano. *Evolutionary robotics: the biology, intelligence, and technology of self-organizing machines*. MIT Press/Bradford Books: Cambridge, USA, 2000.

[7] L. Schönemann. Evolution Strategies in Dynamic Environments. In S. Yang, Y.-S. Ong, and Y. Jin, editors, *Evolutionary Computation in Dynamic and Uncertain Environments*, p. 51–77, Springer, 2007.

[8] A. M. C. Souza and C. Tsallis. Student's t- and r-distributions: Unified derivation from an entropic variational principle. *Physica A: Statistical Mech. and its Appl.*, 236(1-2):52–57, 1997.

[9] W. Thistleton, J. A. Marsh, K. Nelson, and C. Tsallis. Generalized Box-Muller method for generating q-Gaussian random deviates. *IEEE Trans. on Inform. Theory*, 53:4805–4810, 2007.

[10] R. Tinós and A. C. P. L. F. Carvalho. Use of gene dependent mutation probability in evolutionary neural networks for non-stationary problems. *Neurocomputing*, 70(1-3):44–54, 2006.

[11] R. Tinós and S. Yang. Self-adaptation of mutation distribution in evolutionary algorithms. In *Proc. of the 2007 IEEE Cong. on Evol. Comput.*, p. 79–86, 2007.

[12] K. Weicker and N. Weicker. On evolution strategy optimization in dynamic environments. In *Proc. of the 1999 Cong. on Evol. Comput.*, p. 2039–2046, 2000.

[13] X. Yao and Y. Liu. Fast evolution strategies. *Control and Cybernetics*, 26(3):467–496, 1997.