An Empirical Comparison of CMA-ES in Dynamic Environments

Chun-Kit Au and Ho-Fung Leung

Department of Computer Science and Engineering,
The Chinese University of Hong Kong, Shatin, Hong Kong
{auchunkit,lhf}@cuhk.edu.hk

Abstract. This paper empirically investigates the behavior of three variants of covariance matrix adaptation evolution strategies (CMA-ES) for dynamic optimization. These three strategies include the elitist (1+1)-CMA-ES, the non-elitist (μ, λ)-CMA-ES and sep-CMA-ES. To better understand the influence of covariance matrix adaptation methods and of the selection methods to the strategies in dynamic environments, we use the state-of-art dynamic optimization benchmark problems to evaluate the performance. We compare these CMA-ES variants with the traditional (1+1)-ES with the one-fifth success rule. Our experimental results show that the simple elitist strategies including the (1+1)-ES and the (1+1)-CMA-ES generally outperform those non-elitist CMA-ES variants on one out of the six dynamic functions. We also investigate the performance when the dynamic environments change with different severity and when the problems are in higher dimensions. The elitist strategies are robust to different severity of dynamic changes, but the performance is worse when the problem dimensions are increased. In high dimensions, the performance of the elitist and the non-elitist versions of CMA-ES are marginally the same.

Keywords: Dynamic Optimization, Evolution Strategies, Covariance Matrix Adaptation.

1 Introduction

In recent years, there has been a fair amount of research works that have contributed to the state-of-art covariance matrix adaptation evolution strategies (CMA-ES) [1,2,3,4] that is used to solve many black-box optimization problems. CMA-ES usually optimizes the real-valued objective functions $f : \mathbb{R}^n \to \mathbb{R}$ in the continuous domain. On ill-conditioned problems covariance matrix adaptation can accelerate the rate of convergence of evolution strategies by orders of magnitude. For example, a successful covariance matrix adaptation can enable strategies to generate candidate solutions predominantly in the direction of narrow valleys. The CMA-ES is able to learn the appropriate covariance matrix from the successful steps that the algorithm has taken. The covariance matrix is updated such that variances in the directions of the search space that have

C.A. Coello Coello et al. (Eds.): PPSN 2012, Part I, LNCS 7491, pp. 529–538, 2012.

previously been successful are increased while those in other directions are decreased. Even for a small population, the accumulation of information over a number of successful steps can reliably adapt the covariance matrix.

However, the problem classes that have been considered in most of these works are of a static nature. In contrast, many problems in engineering, computational and biological domains are dynamic in that the objective functions are not constant but vary with time. Examples of dynamic optimization problems arise in the context of online job scheduling, where new jobs arrive in the course of optimization. A complete list of survey and works on the evolutionary algorithms for dynamic optimization has been reviewed by [5,6].

There are a few works that focus on evolution strategies in dynamic optimization. The early work [7] has empirically studied the family of evolution strategies in dynamic rotating problems. It investigated the performance when evolution strategies employed different forms of mutation step size adaptation. The experimental results show that a simple mutation step size adaptation achieves the best results compared to other complicated adaptation mechanisms including covariance matrix adaptation. It also suggested to use small populations in evolution strategies because using large populations implies a higher degree of dynamism and this is undesirable in dynamic optimization. Another work [8] studies evolution strategies for the number of mutation step sizes required when the optima move in one or all n coordinates with different severity. The results showed that adapting all n mutation step sizes achieves a better performance than adapting a single mutation step size. The work [9] compares different variants of mutative self-adaptation and shows that the lognormal self-adaptation used in evolution strategies performs better than the variants of self-adaptation commonly used in evolutionary programming. Obviously all these works demonstrate the difficulty of understanding the behavior of evolution strategies and their operators and parameters for dynamic environments.

In this paper, we will empirically investigate the state-of-art CMA-ES variants in the literature and study their performance for dynamic optimization. In the next two sections, we will briefly recall the CMA-ES variants and the dynamic optimization benchmark problems. The empirical comparison is described in Section 4 followed by the conclusion in the last section.

2 CMA-ES Variants

The standard (μ, λ)-CMA-ES: In the standard (μ, λ)-CMA-ES [1,2], in each iteration g, λ number of candidate solutions are generated by sampling a multivariate normal distribution $\boldsymbol{\mathcal{N}}(\mathbf{0}, \mathbf{C})$ with mean $\mathbf{0}$ and $n \times n$ covariance matrix $\mathbf{C}$. The μ best solutions are selected to update the distribution parameters for the next iteration step $g + 1$. The standard CMA-ES employs the concept of cumulative step adaptation (CSA). There are two evolution paths $\mathbf{p}_\sigma$ and $\mathbf{p}_c$ and they are two n-dimensional vectors that are used to accumulate the information about the recent steps of the strategy. The learning of the accumulating information is controlled by three independent learning rates c_σ, c_1 and c_c that

change the global step size σ and the covariance matrix $\mathbf{C}$. The standard (μ, λ)-CMA-ES in this paper is identical to that described by [2] and is summarized in Table 1.

Table 1. Update equations in the standard (μ, λ)-CMA-ES with iteration index $g = 1, 2, 3, \ldots$. The symbol $\mathbf{x}_{i:\lambda}$ represents the i-th best of the candidate solutions $\mathbf{x}_1, \ldots, \mathbf{x}_\lambda$. The values of learning parameters $c_1, c_c, c_\mu, c_\sigma$ are set to the same values as in [1,2].

Given $g \in \mathbb{N} \cup \{0\}, \mathbf{m}^g \in \mathbb{R}^n, \sigma^g \in \mathbb{R}, \mathbf{C}^g \in \mathbb{R}^{n \times n}, \mathbf{p}_\sigma^g, \mathbf{p}_c^g \in \mathbb{R}^n$ and $\mathbf{p}_\sigma^{g=0} = \mathbf{p}_c^{g=0} = \mathbf{0}, \mathbf{C}^{g=0} = \mathbf{I}$
$\mathbf{x}_i \sim \mathbf{m}^g + \sigma^g \times \boldsymbol{\mathcal{N}}_i(\mathbf{0}, \mathbf{C}^g)$ is normally distributed for $i = 1, \ldots, \lambda$
$\mathbf{m}^{g+1} = \sum_{i=1}^{\mu} w_i \mathbf{x}_{i:\lambda}$ where $f(\mathbf{x}_{1:\lambda}) \leq \cdots \leq f(\mathbf{x}_{\lambda:\lambda})$
$\mathbf{p}_\sigma^{g+1} = (1 - c_\sigma)\mathbf{p}_\sigma^g + \sqrt{c_\sigma(2 - c_\sigma)\mu_w}\mathbf{C}^{g-\frac{1}{2}}\frac{\mathbf{m}^{g+1} - \mathbf{m}^g}{\sigma^g}$ for $\mathbf{C}^{g-\frac{1}{2}} = \mathbf{B}\mathbf{D}^{-\frac{1}{2}}\mathbf{B}^T, \mathbf{B}\mathbf{D}\mathbf{B}^T = \mathbf{C}$
$h_\sigma = \begin{cases} 1 & \text{if } \|\mathbf{p}_\sigma^{g+1}\| < \sqrt{1 - (1 - c_\sigma)^{2(g+1)}}(1.4 + \frac{2}{n+1})\mathrm{E}\|\boldsymbol{\mathcal{N}}(\mathbf{0}, \mathbf{I})\| \\ 0 & \text{otherwise} \end{cases}$
$\mathbf{p}_c^{g+1} = (1 - c_c)\mathbf{p}_c^g + h_\sigma\sqrt{c_c(2 - c_c)\mu_w}\frac{\mathbf{m}^{g+1} - \mathbf{m}^g}{\sigma^g}$
$\mathbf{C}_\mu = \sum_{i=1}^{\mu} w_i \frac{\mathbf{x}_{i:\lambda} - \mathbf{m}^g}{\sigma_g} \times \frac{(\mathbf{x}_{i:\lambda} - \mathbf{m}^g)^T}{\sigma_g}$
$\mathbf{C}^{g+1} = (1 - c_1 - c_\mu)\mathbf{C}^g + c_1\mathbf{p}_c^{g+1}\mathbf{p}_c^{g+1^T} + c_\mu\mathbf{C}_\mu$
$\sigma^{g+1} = \sigma^g \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|\mathbf{p}_\sigma^{g+1}\|}{\mathrm{E}\|\boldsymbol{\mathcal{N}}(\mathbf{0},\mathbf{I})\|} - 1\right)\right)$ where $\mathrm{E}\|\boldsymbol{\mathcal{N}}(\mathbf{0}, \mathbf{I})\|$ is the expectation of the n-dimensional normal distributed vector.

The sep-CMA-ES: In the standard CMA-ES, the full learning task scales roughly with n^2 and can dominate most of the search cost. This is one of the major limitations in the standard CMA-ES because of the high degree of freedom $\frac{n^2+n}{2}$ in the covariance matrix. One of the solutions to this is to reduce the degree of freedom from $\frac{n^2+n}{2}$ to n where only the diagonal of the covariance matrix is adapted. The resulting algorithm is called "sep-CMA-ES" [3]. There are two simple changes undertaken in sep-CMA-ES. First, the covariance matrix $\mathbf{C}$ is constrained to be diagonal. Second, the learning rate c_μ is increased. This means that the mutation distribution is sampled independently in the given coordinate system using n individual variances. For sep-CMA-ES, the changes to the update equations in Table 1 are:

1. $\mathbf{D} = \sqrt{\mathrm{diag}(\mathbf{C})}$ where $\mathrm{diag}(\mathbf{C})$ is a diagonal matrix with the same diagonal elements as $\mathbf{C}$. The matrix $\mathbf{B}$ remains $\mathbf{I}$ for all iterations.

2. $c_\mu^{\text{sep-CMA-ES}} = \frac{n+2}{3} \cdot c_\mu$

(1+1)-CMA-ES: The (1+1)-CMA-ES is a new variant that has been recently proposed by [4] as an extension of (1+1)-ES with the one-fifth success rule [10]. It differs from the standard CMA-ES variant in that: (1) it is an elitist algorithm, and (2) not only the step size but also a covariance matrix associated to the search distribution is adapted. The experimental results in [4] shows that it is about 1.5 times faster than the standard CMA-ES on unimodal functions. We follow the principles introduced in [4] and the (1+1)-CMA-ES is summarized in Table 2. A candidate solution $\mathbf{y}^g$ is sampled by perturbing the current solution $\mathbf{x}^g$ by adding a normal distributed vector with mean vector $\mathbf{0}$ and covariance matrix $\mathbf{C}^g$ and scaled by the mutation step-size σ^g. This candidate solution is accepted only if $f(\mathbf{y}^g) < f(\mathbf{x}^g)$. The mutation step-size is adapted using the averaged success rate p_{succ} such that it is increased if the success rate is strictly larger than the target probability p_{succ}^{target}, and decreased if it is strictly smaller. If $f(\mathbf{y}^g) < f(\mathbf{x}^g)$, the covariance matrix is adapted by adding to a multiple of $\mathbf{C}^g$ the rank-one update matrix $\mathbf{p}^{g+1}\mathbf{p}^{g+1^T}$ where $\mathbf{p}^{g+1^T}$ is the transpose of $\mathbf{p}^{g+1}$. We will use the same default settings as in [4] for all strategy parameters.

Table 2. Update equations in the $(1+1)$-CMA-ES with iteration index $g = 1, 2, 3, \ldots$

Given $g \in \mathbb{N} \cup \{0\}\,, \mathbf{x}^g \in \mathbb{R}^n, \sigma^g \in \mathbb{R}, \mathbf{C}^g \in \mathbb{R}^{n\times n}, p_{\text{succ}}^g \in \mathbb{R}, \mathbf{p}^{g=0} = \mathbf{0}, \mathbf{C}^{g=0} = \mathbf{I}$ and $p_{\text{succ}}^{g=0} = p_{succ}^{target} = \frac{2}{11}, c_p = \frac{1}{12}, c_c = \frac{2}{N+2}, c_\mu = \frac{2}{N^2+6}, p_{thresh} = 0.44$
$\mathbf{A}^g = \text{chol}(\mathbf{C}^g)$ where $\text{chol}(\cdot)$ is the Cholesky decompositions such that $\mathbf{C} = \mathbf{A}\mathbf{A}^T$
$\mathbf{z}^g \sim \boldsymbol{\mathcal{N}}(\mathbf{0}, \mathbf{I})$
$\mathbf{y}^g \sim \mathbf{x}^g + \sigma^g \mathbf{A}^g \mathbf{z}^g$
$p_{\text{succ}}^{g+1} = (1 - c_p) p_{\text{succ}}^g + c_p \mathbf{1}_{\{f(\mathbf{y}^g) \le f(\mathbf{x}^g)\}}$
$\sigma^{g+1} = \sigma^g \exp\left(\frac{p_{\text{succ}}^{g+1} - p_{succ}^{target}}{n \cdot (1 - p_{succ}^{target})}\right)$
$\mathbf{p}^{g+1} = \begin{cases} (1 - c_c)\mathbf{p}^g + \mathbf{1}_{\{p_{succ}^{g+1} < p_{thresh}\}} \sqrt{c_c(2 - c_c)}\, \mathbf{A}^g \mathbf{z}^g & \text{if } f(\mathbf{y}^g) \le f(\mathbf{x}^g) \\ \mathbf{p}^g & \text{otherwise} \end{cases}$
$\mathbf{C}^{g+1} = \begin{cases} \left(1 - c_\mu + c_\mu \mathbf{1}_{\{p_{succ}^{g+1} > p_{thresh}\}} c_c(2 - c_c)\right) \mathbf{C}^g & \\ \quad + c_\mu \mathbf{p}^{g+1} \mathbf{p}^{g+1^T} & \text{if } f(\mathbf{y}^g) \le f(\mathbf{x}^g) \\ \mathbf{C}^g & \text{otherwise} \end{cases}$
$\mathbf{x}^{g+1} = \begin{cases} \mathbf{y}^g & \text{if } f(\mathbf{y}^g) \le f(\mathbf{x}^g) \\ \mathbf{x}^g & \text{otherwise} \end{cases}$

3 Dynamic Optimization Benchmark

In dynamic optimization, the objective function changes during the course of optimization. At any given time $t \in \mathbb{T}$, one needs to find the solutions $\mathbf{x}^*$ such that $\forall \mathbf{x}, \mathbf{x}^* \in \mathbb{R}^n, f(\mathbf{x}^*, t) \le f(\mathbf{x}, t)$ where $f : \mathbb{R}^n \times \mathbb{T} \to \mathbb{R}$ is the objective function of a minimization problem and n is the problem dimension. For dynamic optimization problems, the fitness functions, design variables and environmental conditions change from time to time. The simplest way to solve dynamic

optimization problems is to consider each change as an arrival of a new static optimization problem if the time and computational resources are sufficient. However, time and resources given are always limited and the explicit restart approach may not be feasible.

In this paper, we use the generalized dynamic benchmark generator (GDBG)[1]. [11] to evaluate the performance of the CMA-ES variants. This benchmark is a problem generator that can construct dynamic environments in the continuous space. It differs from the benchmarks in the literature that it uses the rotation method instead of shifting the positions of the peaks. Using the rotation method can prevent the unequal challenge per change for the algorithms when the positions of the peaks bounce back from the boundary of the search space.

Types of Environment Changes: We focus on the non-dimensional changes in GDBG. In non-dimensional changes, the values of variables within the problem constraints are changed. One of the examples is to increase or decrease the number of peaks during the course of optimization. Formally, we can describe dynamic changes as: $\phi(t+1) = \phi(t) \oplus \Delta\phi$ where $\phi(t)$ is the system control parameters and $\Delta\phi$ is a deviation from the current system control parameters. At time t, the new environment at time $t+1$ can be expressed as: $f(x, \phi, t+1) = f(x, \phi(t) \oplus \Delta\phi, t)$. There are six types of the non-dimensional changes, including the small step change, the large step change, the random change, the chaotic change, the recurrent change, and the recurrent change with noisy. We name them C_1 to C_6 for our easy reference:

- C_1 Small step change: $\Delta\phi = \alpha \cdot ||\phi|| \cdot r \cdot \phi_{severity}$
- C_2 Large step change: $\Delta\phi = ||\phi|| \cdot (\alpha \cdot sgn(r) + (\alpha_{max} - \alpha) \cdot r) \cdot \phi_{severity}$
- C_3 Random step change: $\Delta\phi = \mathcal{N}(0,1) \cdot \phi_{severity}$
- C_4 Chaotic change: $\phi(t+1) = A \cdot \phi(t) \cdot (1 - \frac{\phi(t)}{||\phi||})$
- C_5 Recurrent change: $\phi(t+1) = \phi_{min} + ||\phi|| \cdot \frac{\left(sin(\frac{2\pi t}{P} + \varphi) + 1\right)}{2}$
- C_6 Recurrent change step with noise:

$$\phi(t+1) = \phi_{min} + ||\phi|| \cdot \frac{\left(sin(\frac{2\pi t}{P} + \varphi) + 1\right)}{2} + \mathcal{N}(0,1) \cdot \phi_{noisyseverity}$$

where $||\phi||$ is the range of ϕ, $\phi_{severity} \in (0,1)$ is the change severity of ϕ, ϕ_{min} is the minimum value of ϕ, $\phi_{noisyseverity} \in (0,1)$ is the noisy severity in the recurrent change change with noise. The parameters $\alpha \in (0,1)$ and $\alpha_{max} \in (0,1)$ are constant values in C_1 Small step change and C_2 Large step change. A logistics function is used in the C_4 Chaotic change, where A is a positive constant between $(1.0, 4.0)$, if ϕ is a vector, the initial values of the items in ϕ should be different within $||\phi||$ in C_4 chaotic change. P is the period in the C_5 Recurrent step change and the C_6 Recurrent step change with noise, φ is the initial phase, r is a random

[1] Due to pages constraints, we outline the key equations of GDBG. For complete details, please read [11].

number drawn uniformly from -1 and 1. The function $sgn(x)$ returns 1 when x is greater than 0, returns -1 when x is less than 0, otherwise returns 0. Finally, $\mathcal{N}(0,1)$ returns a normal distributed one dimensional random number with mean zero and standard derivation one.

Rotation DBG F_1: There are two instances in the GBDB benchmark: Rotation DBG and Composition DBG. In the Rotation DBG, the fitness landscape consists of multiple peaks that can be controlled by tuning the system control parameters. The height, the width and the position of each peak are changed in the six change types described above. If the dynamic problem is $f(x,\phi,t)$, then the set of system control parameters is $\phi = (\mathbf{H},\mathbf{W},\mathbf{X})$, where $\mathbf{H}$,$\mathbf{W}$ and $\mathbf{X}$ are the peak height, the peak width and the peak position respectively. Formally, the function $f(x,\phi,t)$ is

$$f(x,\phi,t) = \min\left\{\mathbf{H}_i(t)+\mathbf{W}_i(t)\left(\exp\left(\sqrt{\sum_{j=1}^{n}\frac{(x_j-\mathbf{X}_j^i(t))^2}{n}}\right)-1\right)\right\}_{i=1}^{m}$$

where m is the number of peaks, n is the problem dimension. The height and the width of the peaks are changed: $\mathbf{H}(t+1) = \text{DynamicChanges}\left(\mathbf{H}(t),\phi_{h_{severity}},||\phi_h||\right)$ and $\mathbf{W}(t+1) = \text{DynamicChanges}\left(\mathbf{W}(t),\phi_{h_{severity}},||\phi_h||\right)$ where the change severity of the height and the width are $\phi_{h_{severity}}$ and $\phi_{w_{severity}}$ respectively. The ranges of the height and the width are denoted by $||\phi_h||$ and $||\phi_w|$.

Composition DBG F_2 to F_6: Another instance of the GDBG benchmark is the Composition DBG. The basic idea is to construct more challenging benchmark functions with randomly located global and local optima. By shifting, rotating and composing the global optima of the standard functions, more challenging test functions that possesses many desirable properties can be obtained. Formally, the Composition DBG can be described as follows:

$$F(x,\phi,t) = \sum_{i=1}^{m}\left\{w_i^{'}\cdot\left(f_i^{'}\left(\frac{(x-\mathbf{O}_i(t)+\mathbf{O}_{iold})\cdot\mathbf{M}_i}{\lambda_i}\right)+\mathbf{H}_i(t)\right)\right\}$$

where the system control parameter $\phi = (\mathbf{O},\mathbf{M},\mathbf{H})$, $F(x)$ is the composition function, $f_i(\mathbf{x})$ is the i-th basic function used to construct the composition function, m is the number of the basic functions, $\mathbf{M}_i$ is the orthogonal rotation matrix for each $f_i(\mathbf{x})$, $\mathbf{O}_i$ and $\mathbf{O}_{iold}$ are the shifted and the old optimum position respectively for each basic function $f_i(\mathbf{x})$.

4 Experimental Study

Setup: In our setup, we use the same set of problems in [11]. A total of six dynamic problems F_1 to F_6 are tested[2] . All six problems are multi-modal, scalable, rotated and have a large number of local optima. Unless stated otherwise,

[2] For details of functions please reference to [11].

a change will occur only after $1e^2 \cdot n$ number of functions evaluations are used. 50 independent runs are executed per each problem and per each change. All problems have the global optimum within the given bounds and there is no need to perform search outside of the given bounds for these problems. All algorithms will be terminated when the number of changes reaches 60. To evaluate the performance of the algorithms for maximization problems, we record the relative function error value $E^{last}(t) = \frac{f(\mathbf{x}_{best}(t))}{f(\mathbf{x}^*(t))}$ after each change. The vector $\mathbf{x}_{best}(t)$ is the best solutions found by the algorithm at time t and the vector $\mathbf{x}^*(t)$ is the location of the global optimum at time t. In our experiments, all three variants of CMA-ES and the (1+1)-ES with the one-fifth success rule are compared on all six benchmark problems. All strategy parameters of evolution strategies are set to the default values in their original works [10,1,2,3,4]. No parameters tuning has been conducted.

Results. The experimental results are shown in Figure 1. The graphs show the relative function error values against the change types. The whiskers in the graphs mark the 10th and 90th percentiles. We first take a look at the first problem F_1 Rotation Peak Problem. The performances of the (1+1)-ES and the (1+1)-CMA-ES generally outperform the standard CMA-ES and the sep-CMA-ES. The results are consistent for all 6 types of dynamic changes. An elitist evolution strategies using a small population size leads to the best results compared to all other strategies that are non-elitist and are in large population sizes. Both the (1+1)-ES and the (1+1)-CMA-ES are statistically indistinguishable for all 6 types of dynamic changes. Comparing the standard CMA-ES with the sep-CMA-ES, their performance are also statistically equivalent. None of our statistical tests are able to show any significance in these two strategies. Obviously the simple adaptation technique like the one-fifth success rule can adapt quickly to the dynamically changing environments. The more complicated mechanisms that produce very good results in static optimization, are not adapting very well for dynamic optimization. If we look into the graph for F_2, the results are consistent with those in F_1. The (1+1)-ES and the (1+1)-CMA-ES achieve the best performance. From functions F_3 to F_6, the performance of all strategies becomes worser since the fitness landscapes are more rugged than the functions F_1 and F_2. However, the performance differences between the elitist and non-elitist versions become smaller. In some of the cases in functions F_3 and F_5, all four variants are statistically equivalent. In functions F_4 and F_6, there are a few cases where the CMA-ES and the sep-CMA-ES outperform the (1+1) variants. Overall all strategies in most of the cases are indistinguishable in functions F_4 and F_6.

We next investigate how these strategies are robust to dynamic changes with different severity and to the problem dimensions. Figure 2 shows the median numbers of relative function errors against the severity when the underlying function is F_1 and the change type is C_3. The severity $\phi_{severity}$ is normalized such that severity is equal to $\frac{\phi_{severity}}{|\phi|}$ where $|\phi|$ is the range of the system control parameters. When we increase the severity, the performance generally becomes

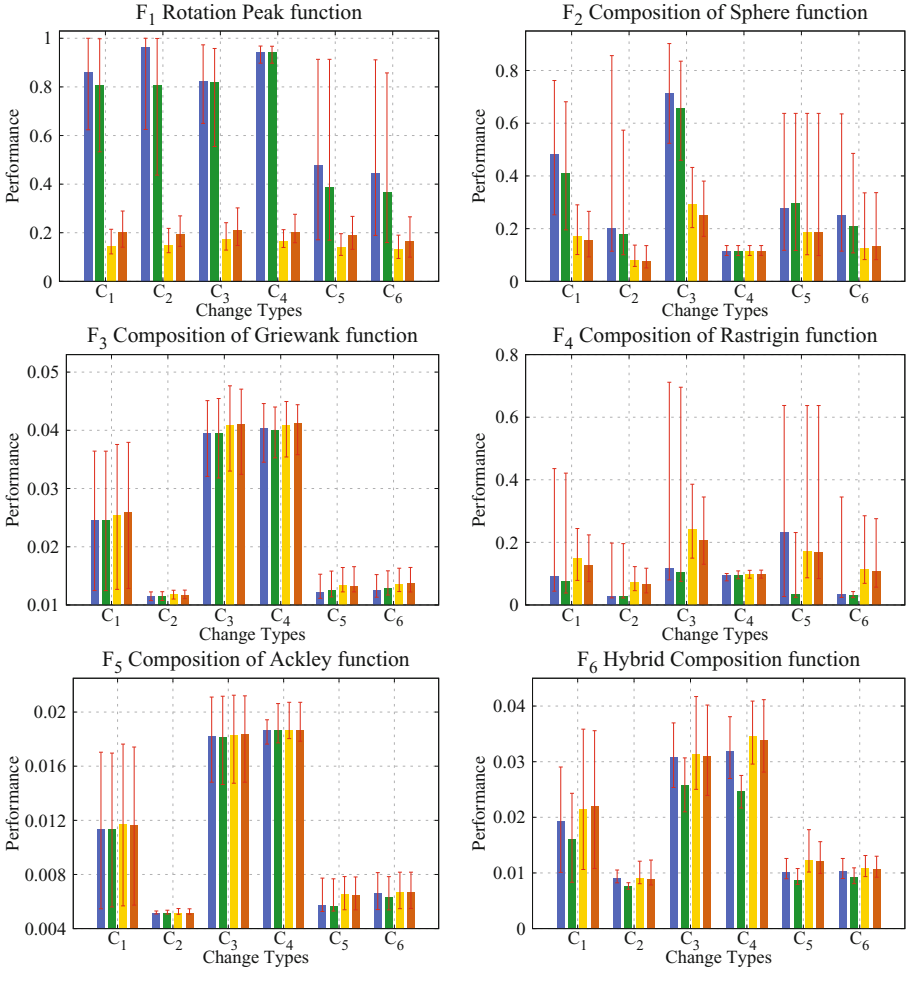

Fig. 1. The median performance of the CMA-ES variants and the (1+1)-ES with the one-fifth success rule on F_1 to F_6 over the trials of 50. The dynamic change types are C_1 Small step change, C_2 Large step change, C_3 Random step change, C_4 Chaotic change, C_5 Recurrent change and C_6 Recurrent change step with noise. For each change type, from left to right, the bars represent the (1+1)-ES with the one-fifth success rule (—), (1+1)-CMA-ES (—), the standard CMA-ES (—) and the sep-CMA-ES (—).

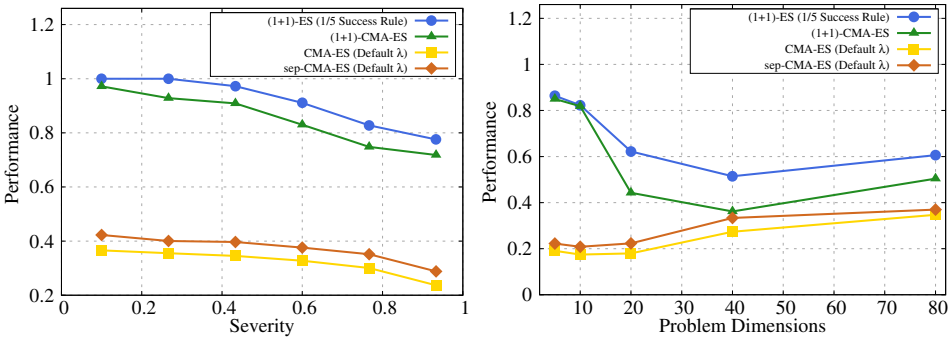

Fig. 2. The median performance of the CMA-ES variants and the (1+1)-ES with the one-fifth success rule on F_1 when the change type is C_3 random step change. The left graph shows the performance against the severity of the dynamic changes while the right graph shows the performance against the problem dimensions.

worser. The elitist (1+1)-ES and the (1+1)-CMA-ES are generally better than the non-elitist strategies in dynamic changes with different severity. Lastly we investigate the performance of the strategies when the dimensions are increased. The right graph in Figure 2 shows the median numbers against the problem dimensions. The performance of elitist strategies becomes worser when the problem dimensions are scaled up. We believe this is due to the small population sizes of these point-based (1+1) strategies. In contrast to the elitist strategies, the non-elitist version of the CMA-ES that are population-based improves gradually in higher dimensions. The performance gap between the elitist and non-elitist CMA-ES is getting smaller. Obviously when we increase the problem dimension, the dynamic problems become more challenging and using the population-based strategies are necessary in order to achieve a reasonable performance.

5 Conclusion

In this paper, we investigate the state-of-art CMA-ES variants for dynamic optimization and they include the elitist (1+1)-CMA-ES, the standard (μ, λ)-CMA-ES and the sep-(μ, λ)-CMA-ES. We first briefly review the CMA-ES variants in the context of static optimization and then we discuss the latest dynamic optimization benchmark problems that are used in our simulations. On one out of the six dynamic functions, the elitist (1+1)-ES with the one-fifth rule and the (1+1)-CMA-ES achieve the best performance. In most of our simulations, these two elitist strategies are statistically equivalent. The non-elitist strategies, including the standard (μ, λ)-CMA-ES and the sep-CMA-ES, are outperformed by the elitist variants. The results are consistent for dynamic changes with different severity. However, the performance of the elitist strategies, which are pointed-based search algorithms, becomes worser for higher dimensional problems. Using the population-based strategies like the standard (μ, λ)-CMA-ES and the sep-CMA-ES can achieve the equivalent performance as what the elitist (1+1) variants do.

In the future work, it would be interesting to introduce additional diversity into the CMA-ES variants. Concentrating the search near the the current optima in a dynamic environment could make the strategies missing the important changes in different region of the search space. Adding predictions mechanism and diversity control methods can be a promising way for CMA-ES to optimize the dynamic functions.

References

1. Hansen, N., Ostermeier, A.: Completely derandomized self-adaptation in evolution strategies. Evolutionary Computation 9(2), 159–195 (2001)
2. Hansen, N., Müller, S.D., Koumoutsakos, P.: Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). Evolutionary Computation 11(1), 1–18 (2003)
3. Ros, R., Hansen, N.: A Simple Modification in CMA-ES Achieving Linear Time and Space Complexity. In: Rudolph, G., Jansen, T., Lucas, S., Poloni, C., Beume, N. (eds.) PPSN 2008. LNCS, vol. 5199, pp. 296–305. Springer, Heidelberg (2008)
4. Igel, C., Suttorp, T., Hansen, N.: A computational efficient covariance matrix update and a (1+1)-cma for evolution strategies. In: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation, GECCO 2006, pp. 453–460 (2006)
5. Jin, Y., Branke, J.: Evolutionary optimization in uncertain environments-a survey. IEEE Transactions on Evolutionary Computation
6. Branke, J.: Evolutionary Optimization in Dynamic Environments (2001)
7. Weicker, K., Weicker, N.: On evolution strategy optimization in dynamic environments. In: Proceedings of the 1999 Congress on Evolutionary Computation, CEC 1999 vol.3., 3 vol. xxxvii+2348 (1999)
8. Schönemann, L.: Evolution Strategies in Dynamic Environments. In: Yang, S., Ong, Y.S., Jin, Y. (eds.) Evolutionary Computation in Dynamic and Uncertain Environments. SCI, vol. 51, pp. 51–77. Springer, Heidelberg (2007)
9. Back, T.: On the behavior of evolutionary algorithms in dynamic environments. In: IEEE World Congress on Computational Intelligence, The 1998 IEEE International Conference on Evolutionary Computation Proceedings, pp. 446–451 (May 1998)
10. Rechenberg, I.: Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution. PhD thesis, TU Berlin (1971)
11. Li, C., Yang, S., Nguyen, T.T., Yu, E.L., Yao, X., Jin, Y., Beyer, H.G., Suganthan, P.N.: Benchmark generator for cec 2009 competition on dynamic optimization (2008)