Deriving and Improving CMA-ES with Information Geometric Trust Regions

Abbas Abdolmaleki
PARC, Palo Alto Research Center
IEETA, University of Aveiro
LIACC, University of Porto
abbas.a@ua.pt

Bob Price
PARC, Palo Alto Research Center
bob.price@parc.com

Nuno Lau
DETI, IEETA, University of Aveiro
nunolau@ua.pt

Luis Paulo Reis
DSI, University of Minho
LIACC, University of Porto
lpreis@dsi.uminho.pt

Gerhard Neumann
CLAS, TU Darmstadt, Darmstadt
University of Lincoln
neumann@ias.tu-darmstadt.de

ABSTRACT

CMA-ES is one of the most popular stochastic search algorithms. It performs favourably in many tasks without the need of extensive parameter tuning. The algorithm has many beneficial properties, including automatic step-size adaptation, efficient covariance updates that incorporates the current samples as well as the evolution path and its invariance properties. Its update rules are composed of well established heuristics where the theoretical foundations of some of these rules are also well understood. In this paper we will fully derive all CMA-ES update rules within the framework of expectation-maximisation-based stochastic search algorithms using information-geometric trust regions. We show that the use of the trust region results in similar updates to CMA-ES for the *mean* and the *covariance* matrix while it allows for the derivation of an improved update rule for the step-size. Our new algorithm, Trust-Region Covariance Matrix Adaptation Evolution Strategy (TR-CMA-ES) is fully derived from first order optimization principles and performs favourably in compare to standard CMA-ES algorithm.

KEYWORDS

Stochastic Search, Expectation Maximisation, Trust Regions

ACM Reference format:
Abbas Abdolmaleki, Bob Price, Nuno Lau, Luis Paulo Reis, and Gerhard Neumann. 2017. Deriving and Improving CMA-ES
with Information Geometric Trust Regions . In *Proceedings of the Genetic and Evolutionary Computation Conference 2017, Berlin, Germany, July 15–19, 2017 (GECCO '17),* 8 pages.
DOI: 10.475/123_4

1 INTRODUCTION

Stochastic search algorithms [10, 19] are black box optimizers of a fitness function that is either unknown or too complex to be modelled explicitly. These algorithms only use the fitness values and don't require gradients or higher derivatives of the fitness function. Stochastic search algorithms typically maintain a search distribution over individuals or candidate solutions, which is typically a Gaussian distribution. This search distribution is used to generate a population of individuals which are evaluated by their corresponding fitness values. Subsequently, a new search distribution is computed by either computing gradient based updates [19], expectation-maximisation-based updates [7, 12], evolutionary strategies [10], the cross-entropy method [14] or information-theoretic policy updates [1], such that the individuals with higher fitness will have better selection probability. The Covariance Matrix Adaptation - Evolutionary Strategy (CMA-ES) is one of the most popular stochastic search algorithms [10]. It performs well in many tasks and does not require extensive parameter tuning. There are three ingredients for the success of CMA-ES. Firstly, the covariance matrix update efficiently combines the old covariance matrix with the sample covariance. Secondly, the evolution path, which stores the average update direction is also incorporated in the covariance matrix to shape future update directions. Lastly, the step-size adaptation of CMA-ES scales the distribution efficiently, increasing it when subsequent updates are correlated while decreasing it otherwise. All these update rules are well established heuristics. The foundations of some of these heuristics are also already theoretically well understood. However, a unique mathematical framework that explains all these update rules is so far still missing. In contrast, expectation maximisation-based algorithms [7, 12, 14] (Section 2.2) optimize a clearly defined objective, i.e., the maximization of a lower-bound. The maximisation of lower bound in each iteration is equivalent to weighted maximum likelihood estimation (MLE) of the distribution. It is well-known that ML estimation of the covariance matrix yields a degenerate, over-fitted distribution [2] which typically results in premature convergence of the algorithm. We extend the EM-based framework by incorporating information geometric trust regions. Using trust regions will restrict the change of the search distribution in each update. We implement such trust regions by bounding the Kullback-Leibler (KL) divergence[1] of the search distribution (Section 3). Using these trust regions, we can recover exactly the same form of covariance update as in CMA-ES. And the mean update of CMA-ES is a degenerate case of the trust region update. Furthermore, we can derive

GECCO '17, Berlin, Germany

DOI: 10.475/123_4

[1] For simplicity we use 'KL' and 'KL divergence' interchangeably

an improved step-size update rule that is based on the same theoretical foundation as the remaining update rules. Our new algorithm, Trust-Region Covariance Matrix Adaptation Evolution Strategy (TR-CMA-ES) works favourably in compare to CMA-ES in all tested scenarios, which includes typical standard benchmark functions and complex policy optimization tasks from robotics.

1.1 Related Work

We will review CMA-ES and Expectation-Maximisation-based algorithms in the preliminaries section. In this section, we will briefly review other existing stochastic search algorithms.

The Natural Evolution Strategy (NES) uses the natural gradient to optimize the expected fitness value under the search distribution [19]. The natural gradient has been shown to outperform the standard gradient in many applications in machine learning [6]. The intuition of the natural gradient is that we want to obtain an update vector of the parameters of the search distribution that optimises the value of expected fitness while the KL-divergence between new and current search distributions is bounded. To obtain this update vector, a second order approximation of the KL, which is equivalent to the Fisher information matrix, is used. The resulting natural gradient is obtained by multiplying the standard gradient with the inverse of the Fisher matrix. Yet, in contrast to our algorithm, NES family algorithms do not exactly enforce a desired bound of the KL-divergence but use a constant learning rate [19].

The use of trust regions is a common approach in policy search to obtain a stable policy update and to avoid premature convergence [1, 17, 18]. The model-based relative entropy stochastic search algorithm (MORE)[1] uses a local quadratic surrogate to update it's Gaussian search distribution. As these surrogates can be inaccurate, MORE doesn't exploit the surrogate greedily, but uses a trust region in form of a KL-bound. This trust region defines how much the algorithm can exploit the quadratic model. Moreover, MORE explicitly bounds the entropy loss to avoid premature convergence. This method has been shown very competitive when the hyper parameters are set correctly. Similar trust regions have been used in other policy search methods such as Relative Entropy Policy Search [17].

Similar to these methods, our framework also uses a KL bound to define a trust region. However, in difference to all these policy search methods, firstly we use this bound in an EM-based framework and secondly we use the reverse KL bound. As the KL is not symmetric, this is a very important difference. For a more detailed discussion please see Section 3.

2 PRELIMINARIES

In stochastic search, we want to maximize a fitness function $f(\boldsymbol{x}) : \mathbb{R}^n \rightarrow \mathbb{R}$. The goal is to find one or more individuals $\boldsymbol{x} \in \mathbb{R}^n$ which have the highest possible fitness value. The only accessible information is the fitness values of the individuals. Typically stochastic search algorithms, maintain a Gaussian distribution on individuals,

$$\boldsymbol{x} \sim \mathcal{N}\left(\boldsymbol{x}|\boldsymbol{\mu}, \sigma\Sigma\right),$$

where $\boldsymbol{\mu} \in \mathbb{R}^n$, $\Sigma \in \mathbb{R}^{n\times n}$ and $\sigma \in \mathbb{R}^+$ respectively are the mean(our current guess of optimum), the covariance matrix and step size which together define the exploration of the search distribution.[2] We seek to find a distribution over individuals $\boldsymbol{x}$, denoted $\pi(\boldsymbol{x};\boldsymbol{\theta})$, that minimises the expected fitness

$$\mathbb{E}[f(\boldsymbol{x})] = \int_{\boldsymbol{x}} \pi(\boldsymbol{x};\boldsymbol{\theta}) f(\boldsymbol{x}) d\boldsymbol{x}. \tag{1}$$

At each new iteration $t+1$, N individuals $\boldsymbol{x}$ are drawn from the current Gaussian distribution $\pi(\boldsymbol{x}, \boldsymbol{\theta}_t)$, with $\boldsymbol{\theta}_t = \{\boldsymbol{\mu}_t, \sigma_t, \Sigma_t\}$. These individuals are evaluated and, with their fitness values, construct a dataset $\{\boldsymbol{x}_i, f(\boldsymbol{x}_i)\}_{i=1\ldots N}$ that is subsequently used to compute a new Gaussian search distribution $\pi_{\boldsymbol{\theta}_{t+1}}$, with $\boldsymbol{\theta} = \{\boldsymbol{\mu}_{t+1}, \sigma_{t+1}, \Sigma_{t+1}\}$, such that better individuals will have higher selection probability.

2.1 Covariance Matrix Adaptation - Evolutionary Strategy

As CMA-ES will be our main baseline for comparisons, in this section we briefly explain the update rules of the CMA-ES algorithm to obtain a new search distribution.

Fitness Transformation. CMA-ES uses a rank preserving transformation of fitness values which makes it invariant under monotone transformations of the fitness function [9]. In particular, CMA-ES gives zero weights to the worse half of the population and the weight of the jth best individuals is proportional to

$$w_j \propto \ln(N/2 + 0.5) - \ln(j). \tag{2}$$

We will use the same weighting method in our algorithm.

Mean Update Rule. Using the weights, the next distribution mean $\boldsymbol{\mu}_{t+1}$ is set to the weighted sum of the individuals, i.e,

$$\boldsymbol{\mu}_{t+1} = \frac{\sum_{i=1}^{N/2} w_i \boldsymbol{x}}{Z}, \quad Z = \sum_{i=1}^{N/2} w_i.$$

Evolution Path. The evolution path records the sum of consecutive update steps, i.e, $\boldsymbol{\mu}_{t+1} - \boldsymbol{\mu}_t$. If consecutive update steps are towards the same direction, i.e., they are correlated, their updates will sum up. In contrast, if they are decorrelated, the update directions cancel each other out. Using the information from the evolution path leads to significant improvements in terms of convergence speed, as it enables the algorithm to exploit correlations between consecutive steps.

Covariance Matrix Update Rule. The covariance matrix update is computed using the local information about the fitness function from the individuals, called the rank-μ update, and the information about the evolution of the distribution contained in the evolution path $\boldsymbol{p}_{c\,t+1}$, called the rank-1 update. The covariance update is defined as

$$\Sigma_{t+1} = (1 - c_{\boldsymbol{\mu}} - c_1)\Sigma_t + \underbrace{c_{\mu} \sum_{i=1}^{N} \frac{w_i(\boldsymbol{x}_i - \boldsymbol{\mu}_t)(\boldsymbol{x}_i - \boldsymbol{\mu}_t)^T}{\sigma_t}}_{\text{Rank-}\boldsymbol{\mu}\text{ update}} + \underbrace{c_1 \boldsymbol{p}_{c\,t+1} \boldsymbol{p}_{c\,t+1}^T}_{\text{Rank-one update}},$$

where $c_1 \geq 0$ and $c_{\boldsymbol{\mu}} \geq 0$ are learning rates which are set by well established heuristics.

Step Size Update Rule. Subsequently, the evolution path is also used to find a new step size σ_{t+1} such that the difference between the distributions of the actual evolution path and an evolution path under random selection is minimised, see [9] for more details.

[2] In this paper σ is used to denote the variance which we refer to it as step size.

Theoretical Foundation. Recently there are theoretical studies to justify the update rules of CMA-ES. For example, In [16] and [4] it has been shown that the CMA-ES mean and rank-μ update rules can be derived by computing the approximate natural gradient of the expected fitness under the search distribution. Moreover, Akimoto et al. [4] nicely sketches the relation between CMA-ES and the EM-based stochastic search framework conceptually. However, it does not define a concrete mathematical framework for driving the search distribution update rules. While these theoretical studies provide a great understanding of CMA-ES, to the best of our knowledge, they do not lead to an improvement over the original CMA-ES. One reason is that, these frameworks [16], [4] do not derive the rank-1 of CMA-ES and a step-size update rule, which are crucial features of CMA-ES. Our new algorithm will have all the features of CMA-ES while it will be fully derived from a single principle.

2.2 Stochastic Search by Expectation-Maximisation

An alternative view to the formulation in Equation 1 is to convert the problem into an inference problem [15]. In order to use inference, the common method is to define a binary fitness event R as observed variable. To simplify notation we will always write R when we mean $R = 1$. The probability of this fitness event is given by $p(R|\boldsymbol{x}) \propto C(f(\boldsymbol{x}))$, where C is a monotonically increasing (i.e., rank preserving) transformation of the fitness values. Intuitively $p(R|\boldsymbol{x})$ defines the probability that individual $\boldsymbol{x}$ is the optimum solution. Hence now we optimise the objective

$$p(R;\boldsymbol{\theta}) = \int_{\boldsymbol{x}} p(R|\boldsymbol{x})\pi(\boldsymbol{x};\boldsymbol{\theta})d\boldsymbol{x}, \tag{3}$$

where $p(R;\boldsymbol{\theta})$ is the marginal distribution of the event R. We would now like to find the distribution parameters $\boldsymbol{\theta}$, that maximises $p(R;\boldsymbol{\theta})$. We can also maximize any strictly increasing function of $p(R;\boldsymbol{\theta})$. In particular, the derivations will be simpler if instead we maximise the log probability of the event R, i.e,

$$\log p(R;\boldsymbol{\theta}) = \log \int_{\boldsymbol{x}} p(R|\boldsymbol{x})\pi(\boldsymbol{x};\boldsymbol{\theta})d\boldsymbol{x} \tag{4}$$

To optimise $\log p(R;\boldsymbol{\theta})$, we resort to an iterative expectation maximization (EM) algorithm [7, 15]. In each iteration, the EM algorithm iteratively constructs a lower bound on $\log p(R;\boldsymbol{\theta})$ (E-step) and then optimizes that lower bound to obtain a new search distribution $\pi_{t+1} = \pi(\boldsymbol{x};\boldsymbol{\theta}_{t+1})$ (M-step).

Constructing the lower bound (E-Step). Similar to prior work [15], we can now introduce a variational distribution $q(\boldsymbol{x})$ which is used to decompose the $\log p(R;\boldsymbol{\theta})$, i.e,

$$\log p(R;\boldsymbol{\theta}) = \mathcal{L}(q,\boldsymbol{\theta}) + \mathrm{KL}(q(\boldsymbol{x})||p(\boldsymbol{x}|R;\boldsymbol{\theta})), \tag{5}$$

where the first term

$$\mathcal{L}(q,\boldsymbol{\theta}) = \int_x q(\boldsymbol{x}) \log \frac{p(R|\boldsymbol{x})\pi(\boldsymbol{x};\boldsymbol{\theta})}{q(\boldsymbol{x})} d\boldsymbol{x} \tag{6}$$

is the lower bound of $\log p(R;\boldsymbol{\theta})$ and second term

$$\mathrm{KL}(q(\boldsymbol{x})||p(\boldsymbol{x}|R;\boldsymbol{\theta})) = -\int_{\boldsymbol{x}} q(\boldsymbol{x}) \log \frac{p(\boldsymbol{x}|R;\boldsymbol{\theta})}{q(\boldsymbol{x})} d\boldsymbol{x} \tag{7}$$

is the KL-divergence between the variational distribution $q(\boldsymbol{x})$ and the posterior

$$p(\boldsymbol{x}|R;\boldsymbol{\theta}) = \frac{p(R|\boldsymbol{x})\pi(\boldsymbol{x};\boldsymbol{\theta})}{\int p(R|\boldsymbol{x})\pi(\boldsymbol{x};\boldsymbol{\theta})d\boldsymbol{x}}, \tag{8}$$

which is proportional to the search distribution $\pi(\boldsymbol{x};\boldsymbol{\theta})$ weighted by fitness event probabilities $p(R|\boldsymbol{x})$. Equation 5 holds for any variational distribution $q(\boldsymbol{x})$ and parameters $\boldsymbol{\theta}$, and hence,

$$\log p(R;\boldsymbol{\theta}) \geq \mathcal{L}(q(\boldsymbol{x}),\boldsymbol{\theta}) \tag{9}$$

as the KL term is always positive. Note that the lower bound will be tight at our current distribution parameters $\boldsymbol{\theta}_t$, i.e., $\log p(R;\boldsymbol{\theta}_t) = \mathcal{L}(q(\boldsymbol{x}),\boldsymbol{\theta}_t)$, if we choose a variational distribution $q(\boldsymbol{x})$ such that $\mathrm{KL}(q(\boldsymbol{x})||p(\boldsymbol{x}|R;\boldsymbol{\theta}_t)) = 0$, which is equivalent to choosing

$$q(\boldsymbol{x}) = p(\boldsymbol{x}|R;\boldsymbol{\theta}_t) = \frac{p(R|\boldsymbol{x})\pi(\boldsymbol{x};\boldsymbol{\theta}_t)}{\int_{\boldsymbol{x}} p(R|\boldsymbol{x})\pi(\boldsymbol{x};\boldsymbol{\theta}_t)d\boldsymbol{x}}.$$

This choice of the $q(\boldsymbol{x})$ gives us a lower-bound $\mathcal{L}(p(\boldsymbol{x}|R;\boldsymbol{\theta}_t),\boldsymbol{\theta})$ on the $\log p(R;\boldsymbol{\theta})$ such that $p(R;\boldsymbol{\theta}_t) = \mathcal{L}(p(\boldsymbol{x}|R;\boldsymbol{\theta}_t),\boldsymbol{\theta}_t)$. Constructing this lower-bound corresponds to the E-step.

Optimising the lower bound (M-Step). In the M-step, we then maximise $\mathcal{L}(p(\boldsymbol{x}|R;\boldsymbol{\theta}_t),\boldsymbol{\theta})$ with respect to the parameters $\boldsymbol{\theta}$ to obtain a new parameters $\boldsymbol{\theta}_{t+1}$. Typically, $p(R|\boldsymbol{x})$ is unknown and we only have access to sample evaluations for the individual $\boldsymbol{x}$ generated from the search distribution $\pi(\boldsymbol{x};\boldsymbol{\theta}_t)$. In this case, the lower-bound can be approximated by

$$\begin{aligned}\mathcal{L}(q(\boldsymbol{x}),\boldsymbol{\theta}) &\approx 1/N \sum_i \frac{q(\boldsymbol{x}_i)}{\pi(\boldsymbol{x}_i;\boldsymbol{\theta}_t)} \log \frac{p(R|\boldsymbol{x}_i)\pi(\boldsymbol{x}_i;\boldsymbol{\theta})}{q(\boldsymbol{x}_i)} d\boldsymbol{x} \\ &\approx 1/N \sum_i w_i \log \pi(\boldsymbol{x}_i;\boldsymbol{\theta}) + \text{const.},\end{aligned} \tag{10}$$

where $w_i \propto p(R|\boldsymbol{x}_i)$. This corresponds to a weighted maximum likelihood estimate. Please note that in Equation 10, we focus on terms depending on $\boldsymbol{\theta}$ and the others are treated as constant.

Monotonic improvement guarantee. Expectation-Maximization stochastic search methods inherit a monotonic improvement guarantee from the EM algorithm, i.e., $\log p(R;\boldsymbol{\theta}_t)$ is always increased (or stays the same) at each iteration. This is easy to see by noting that the lower-bound $\mathcal{L}$ is tight after the E-step, i.e., $\log p(R;\boldsymbol{\theta}_t) = \mathcal{L}(q,\boldsymbol{\theta}_t)$. Therefore, optimising the lower-bound at the M-step, can only improve $\log p(R;\boldsymbol{\theta}_{t+1})$ over $\log p(R;\boldsymbol{\theta}_t)$.

Disadvantage of EM-based algorithms. While this algorithm can improve the search distribution, it can quickly result in a degenerated distribution which stops exploration and would lead to premature convergence, which is a problem in many of these methods [12, 14]. The cause of this limitation is mainly the maximum likelihood estimate (Equation 10) which overfits the current individuals and change the current search distribution drastically[2].

3 TRUST REGIONS FOR COVARIANCE MATRIX ADAPTATION

In order to remedy the problem of premature convergence in EM-based algorithms, we will add information-geometric trust region to the optimization objective of the lower bound. The trust region regularizes the maximization step and bound the entropy loss of the new distribution with respect to the current distribution. As we do not

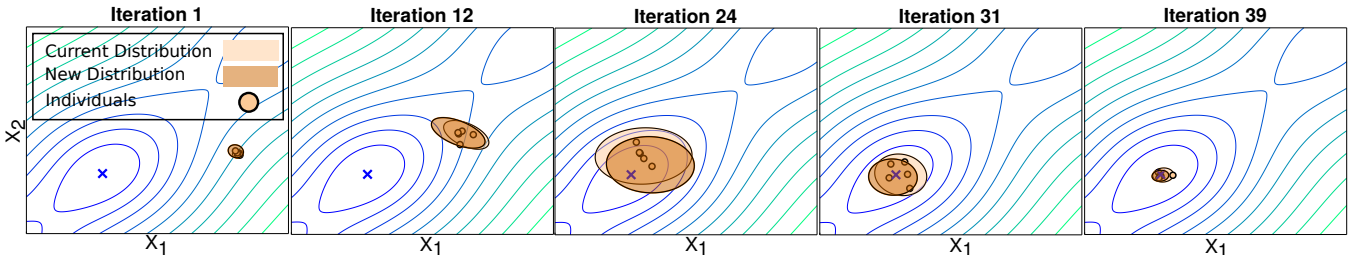


Figure 1: In this illustration, we use a slightly modified version of the McCormick function with two dimensions. The blue cross shows the optimum and blue contours have higher value than green contours. In iteration one, the algorithm starts with a small distribution. In each iteration we sample five individuals. The figures show that our step size strategy effectively controls growing and shrinking of the distribution. At the beginning, it naturally grows and when it has found the optimum, it naturally shrinks (iteration 24-39). Please note that the step size strategy is derived from a well defined consistent principle.

fully maximize the lower-bound any more, our algorithm belongs to the class of generalized Expectation-Maximisation algorithms [8] which also holds the expectation improvement guarantee of conventional EM algorithms.

Our constraint optimization problem for optimizing the lower bound now is given by

$$\boldsymbol{\theta}_{t+1} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \sum_i w_i \log \pi(\boldsymbol{x}_i;\boldsymbol{\theta})$$
$$\text{s.t. } \mathrm{KL}\Big(\pi(\boldsymbol{x};\boldsymbol{\theta}_t)||\pi(\boldsymbol{x};\boldsymbol{\theta})\Big) \leq \epsilon, \tag{11}$$

where ϵ defines the bound on the KL divergence. With this bound we can define a trust region for our updates to avoid a degenerated distribution. In next section, we will show how to solve this constraint optimisation problem efficiently in closed form for a Gaussian search distribution. The resulting update strategies match the update strategies of CMA-ES except for small, but important differences as we will discuss later in this section. While bounding the KL-divergence is a well established method [1, 13, 17, 18], it has so far not been applied to expectation-maximisation stochastic search algorithms. Another interesting observation is that, in difference to many existing policy search methods that use a trust region [1, 17], we employ the reverse KL divergence measure, i.e., instead of bounding $\mathrm{KL}(\pi||\pi_t)$ we bound $\mathrm{KL}(\pi_t||\pi)$. Hence, our derivation reveals an interesting connection of CMA-ES to many trust region optimization algorithms that use the forward KL, $\mathrm{KL}(\pi||\pi_t)$ which was so far unknown. Note that in the case of a Gaussian search distribution, we can solve the optimisation program in equation 11 in closed form only if we use the reverse KL.

3.1 Update Rules for Multivariate Normal Distributions

In each iteration, we solve the optimization program given in Equation 11 with a Gaussian distribution

$$\pi(\boldsymbol{x};\boldsymbol{\theta}) = (2\pi)^{-n/2}|\sigma\Sigma|^{-0.5}\exp(-0.5(\boldsymbol{x}-\boldsymbol{\mu})^T\sigma^{-1}\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu}))$$

and the KL divergence between two Gaussian distributions

$$\mathrm{KL}\Big(\pi_t||\pi\Big) = 0.5\Bigg(\mathrm{tr}(\sigma^{-1}\Sigma^{-1}\sigma_t\Sigma_t) + (\boldsymbol{\mu}-\boldsymbol{\mu}_t)^T\sigma^{-1}\Sigma^{-1}(\boldsymbol{\mu}-\boldsymbol{\mu}_t) - n + \ln\frac{|\sigma\Sigma|}{|\sigma_t\Sigma_t|}\Bigg). \tag{12}$$

Where $|.|$ and $\mathrm{tr}(.)$ are determinant and trace operators respectively. In general, to solve this constraint optimisation problem, we first construct the Lagrangian equation, i.e,

$$L(\eta,\boldsymbol{\theta}) = \sum_i w_i \log\pi(\boldsymbol{x}_i;\boldsymbol{\theta}) + \eta(\epsilon - \mathrm{KL}\Big(\pi_t||\pi\Big)),$$

where η is a Lagrangian multiplier. Subsequently, we differentiate $L(\eta,\boldsymbol{\theta})$ we respect to $\boldsymbol{\theta}$ and set it to zero to obtain $\boldsymbol{\theta}^*$ which will depend on the Lagrangian multiplier η. To compute the optimal value for the Lagrangian multiplier η^*, we can optimize the dual function $L(\eta,\boldsymbol{\theta}^*)$ which is obtained by setting the optimal $\boldsymbol{\theta}^*$ back into the Lagrangian. In order to perform an efficient optimization and to obtain the CMA-ES updates, we employ a coordinate descent strategy where we optimize for each parameter, i.e, the new mean $\boldsymbol{\mu}_{t+1}$, the covariance matrix Σ_{t+1} and the step size σ_{t+1}, independently. I.e., when we optimize, for example, for the covariance matrix Σ, we use the current mean $\boldsymbol{\mu}_t$ and step size σ_t for the remaining parameters. This decoupling have three advantages. Firstly, it renders the problem to a concave optimisation problem for each search distribution component. This is easy to verify by observing that the second derivative of the Lagrangian $L(\eta,\boldsymbol{\theta})$ with respect to each parameter is negative definite. Moreover, this decoupling allows us to set different ϵ values for each component, i.e., $\epsilon_{\boldsymbol{\mu}},\epsilon_{\Sigma},\epsilon_{\sigma}$ for the mean, the covariance matrix and the step size respectively. Different ϵ lead to different learning rates. We can set a higher learning rate for the mean and step size than for the covariance as the covariance matrix has many more parameters which makes it more frail to overfit. Natural evolutionary strategy algorithms also use different learning rates for mean, step size, and covariance matrix [19]. Finally, by setting the mean to the current mean when optimising for covariance matrix and step size, we find a covariance matrix Σ_{t+1} and step size σ_{t+1} that increases the likelihood of successful steps instead of likelihood of successful individuals. CMA-ES also uses the current mean when obtaining the new covariance matrix. This approach has been shown to be less prone to premature convergence [9].

3.1.1 Mean Update Rule. Now, we construct the Lagrangian for obtaining the new mean, i.e,

$$L(\boldsymbol{\mu},\eta) = 2\epsilon_{\mu}\eta - \mathrm{tr}({\sigma_t}^{-1}{\Sigma_t}^{-1}F), \tag{13}$$

where

$$F = \sum_i w_i(\boldsymbol{x}_i - \boldsymbol{\mu})(\boldsymbol{x}_i - \boldsymbol{\mu})^T + \eta(\boldsymbol{\mu} - \boldsymbol{\mu}_t)(\boldsymbol{\mu} - \boldsymbol{\mu}_t)^T.$$

To obtain the new mean $\boldsymbol{\mu}_{t+1} = \text{argmax}_{\boldsymbol{\mu}} L(\boldsymbol{\mu}, \eta)$, we differentiate the Lagrangian with respect to $\boldsymbol{\mu}$ and set it to zero to obtain

$$\boldsymbol{\mu}_{t+1} = \frac{\eta^*_\mu \boldsymbol{\mu}_t + \sum w_i \boldsymbol{x}_i}{\sum w_i + \eta^*_\mu}, \tag{14}$$

where η^*_μ is the optimum Lagrangian multiplier which can be obtained by minimising the convex dual function

$$g_{\boldsymbol{\mu}}(\eta) = \text{supp}_{\boldsymbol{\mu}} L(\eta, \boldsymbol{\mu}) = L(\eta, \boldsymbol{\mu}_{t+1}),$$

i.e., $\eta^*_\mu = \text{argmin}_\eta\, g_{\boldsymbol{\mu}}(\eta)$. As indicated in the equation, the dual function is obtained by setting the optimal solution for the mean back into the Lagrangian. The dual function is convex and can be efficiently optimised by any arbitrary non-linear optimiser. We use fmincon tool in matlab (please see the matlab code for implementation). If we set a large KL bound for the mean, i.e., $\epsilon_\mu \to \infty$, η will go to 0 and we obtain the original CMA-ES mean update $\boldsymbol{\mu}_{t+1} = \sum w_i \boldsymbol{x}_i / \sum w_i$, which is equivalent to the solution of unregularized weighted maximum likelihood estimate. In this paper we always set a large learning rate for mean which results in similar mean update as CMA-ES. Investigating the effect of different learning rates for mean is part of our future work.

3.1.2 Incorporating the Evolution Path. The evolution path is a crucial ingredient of the performance of CMA-ES [9]. It summarizes the path taken by the recent distribution updates and can be interpreted as sort of momentum term. In order to exploit the evolution path in our formulation, similar to [3] we treat the evolution path $\boldsymbol{p}_c$ as an individual sample, i.e., our optimisation problem is now given by

$$\max_{\boldsymbol{\theta}} \sum_i w_i \log \pi(\boldsymbol{x}_i; \boldsymbol{\theta}) + \lambda \log \pi(\boldsymbol{\mu}_t + \boldsymbol{p}_c; \boldsymbol{\theta}) \tag{15}$$
$$\text{s.t. } \text{KL}\Big(\pi(\boldsymbol{x}; \boldsymbol{\theta}_t) || \pi(\boldsymbol{x}; \boldsymbol{\theta})\Big) \leq \epsilon,$$

where $\lambda > 0$ is a user-defined weight that specifies the importance of the evolution path $\boldsymbol{p}_c$ with respect to other individuals. Empirically, we found it is better to use different λ coefficients for the covariance matrix and the step size, i.e, λ_Σ, λ_σ. Please note that we also tried to use the evolution path to adapt the mean as well. However, we observed that the learning process gets considerably unstable due to overshooting the optimum.

3.1.3 Covariance Matrix Update Rule. We construct the covariance matrix Lagrangian for the optimisation program in Equation 15, i.e.,

$$L(\eta, \Sigma) = -(\sum_i w_i + \lambda_\Sigma + \eta) \ln |\Sigma| + \eta(2\epsilon_\Sigma + n + \ln |\Sigma_t|)$$
$$- \text{tr}(\Sigma^{-1}(\Sigma_s/\sigma_t + \eta\Sigma_t + \lambda_\Sigma \boldsymbol{p}_c \boldsymbol{p}_c{}^T/\sigma_t)), \tag{16}$$

where

$$\Sigma_s = \sum_i w_i(\boldsymbol{x}_i - \boldsymbol{\mu}_t)(\boldsymbol{x}_i - \boldsymbol{\mu}_t)^T. \tag{17}$$

To obtain the new covariance matrix $\Sigma_{t+1} = \text{argmax}_\Sigma L(\Sigma, \eta)$, we differentiate the Lagrangian with respect to Σ^{-1} and set it to zero to obtain

$$\Sigma_{t+1} = \frac{\eta^*_\Sigma \Sigma_t + \Sigma_s/\sigma_t + \lambda_\Sigma \boldsymbol{p}_c \boldsymbol{p}_c^T/\sigma_t}{\sum_i w_i + \lambda_\Sigma + \eta^*_\Sigma} \tag{18}$$

in closed form. The coefficient η^*_Σ is again the optimum Lagrangian multiplier. It can be obtained by minimising the dual function $\eta^*_\Sigma = \text{argmin}_\eta\, g_\Sigma(\eta)$. Where $g_\Sigma(\eta) = L(\eta, \Sigma_{t+1})$. Now, by changing variables, i.e.,

$$1 - \alpha - \gamma = \frac{\eta}{\sum_i w_i + \lambda_\Sigma + \eta}\ ,\ \gamma = \frac{\lambda_\Sigma}{\sum_i w_i + \lambda_\Sigma + \eta} \tag{19}$$
$$\alpha = \frac{1}{\sum_i w_i + \lambda_\Sigma + \eta},$$

we can rewrite the equation for Σ as

$$\Sigma_{t+1} = (1 - \alpha - \gamma)\Sigma_t + \alpha \frac{\Sigma_s}{\sigma_t} + \gamma \frac{\boldsymbol{p}_c \boldsymbol{p}_c{}^T}{\sigma_t}.$$

Note that as $\eta > 0$ and $\lambda_\Sigma > 0$ we can infer that $0 \leq \alpha + \gamma \leq 1$. Hence, we obtained the exact form of covariance matrix update of CMA-ES where the second and third terms are called rank-μ and rank-1 updates respectively. The only difference is that α and γ are constant coefficients in CMA-ES. Please note that CMA-ES uses constant coefficients to combine the old covariance matrix with rank-1 and rank-μ while TR-CMA-ES compute these coefficients based on satisfying an exact KL bound. We observed that this exact KL-bound results in different coefficients in each iteration.

3.1.4 Step Size Update Rule. We also construct the step size Lagrangian for the optimisation program in Equation 15, i.e.,

$$L(\eta, \sigma) = -(\sum_i w_i + \lambda_\sigma + \eta) \ln \sigma^n + \eta(2\epsilon_\sigma + n + \ln({\sigma_t}^n))$$
$$- \sigma^{-1} \text{tr}({\Sigma_t}^{-1}(\lambda_\sigma \boldsymbol{p}_c \boldsymbol{p}_c{}^T + \Sigma_s) + \eta\sigma_t I). \tag{20}$$

To obtain the new step size $\sigma_{t+1} = \text{argmax}_\sigma\, L(\sigma, \eta)$ we differentiate the Lagrangian with respect to σ^{-1} and set it to zero to obtain

$$\sigma_{t+1} = \frac{n\eta^*_\sigma \sigma_t + \text{tr}(\Sigma_t^{-1}(\Sigma_s + \lambda_\sigma \boldsymbol{p}_c \boldsymbol{p}_c^T))}{n(\sum w_i + \lambda_\sigma + \eta^*_\sigma)}. \tag{21}$$

Where Σ_s is given in equation 17 and the coefficient η^*_σ is the optimal Lagrangian multiplier. It is again obtained by minimising the dual function $\eta^*_\sigma = \text{argmin}_\eta\, g_\sigma(\eta)$. Where $g_\sigma(\eta) = L(\eta, \sigma_{t+1})$. In contrast to the current step size control approaches, our algorithm naturally increases or decreases the step size σ based on the the current individuals and the evolution path without explicitly defining any heuristic. More formally, the step size is adapted to increase the likelihood of successful steps and the evolution path using the same mathematical foundations as the updates for the mean and covariance matrix. CMA-ES only uses the evolution path without using the current individuals [5]. Figure 1 illustrates the effectiveness of our search distribution step size update rule. Now, all the updates follow the same principle.

3.2 Algorithm

Algorithm 1 shows a compact representation of the new stochastic search method. In each iteration, we generate N individuals $\boldsymbol{x} \in \mathbb{R}^n$ and evaluate their fitness values (Lines 1-7). Subsequently, we compute a weight for each individual based on the fitness value (Line

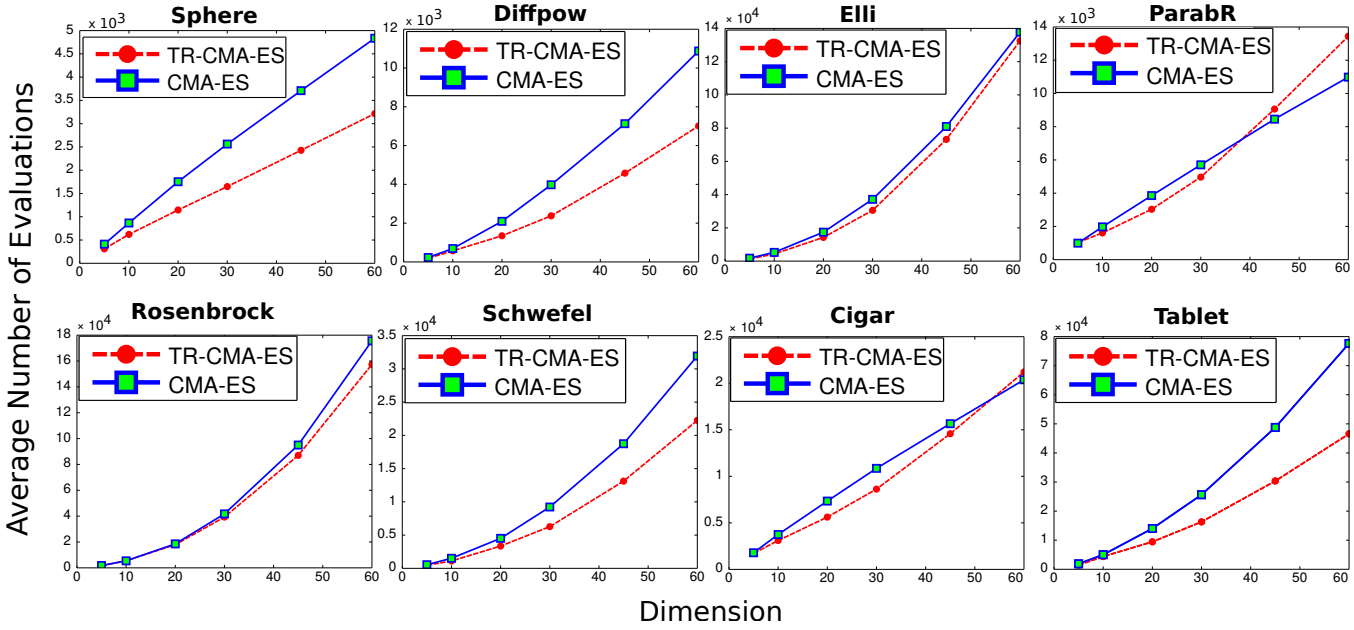


Figure 2: Plot of the average number of required fitness evaluations over 20 trials to reach the target fitness value of -10^{-5} (10^3 for the unbounded function ParabR) for 8 different benchmark functions with 5 to 60 dimensions. TR-CMA-ES outperformed CMA-ES in most benchmark functions. In fact only in some dimensions of ParabR and Cigar functions CMA-ES performs better than TR-CMA-ES.

8). For ease of comparison we use the same rank-based weighting as CMA-ES. Please see preliminary section for the weighting method. We compute the number of effective samples (Line 9). Similar to CMA-ES, we empirically obtained a default setting for hyper parameters of the algorithm which scales with the dimension of the problem and size of the population (Line 10). Subsequently we compute the new mean and new evolution path (Line 11-15). Finally, we obtain the new covariance matrix and new step size (Line 16-21). Note that the η^* are obtained by optimising the dual functions we explained in the previous section.

4 EXPERIMENTS

In this section, we present our experimental evaluations of TR-CMA-ES in comparison to CMA-ES. We use two sets of benchmarks, the standard functions and three simulated robotics tasks where we compare against CMA-ES algorithm. For all experiments we use the same hyper parameter setting as given in Algorithm 1. For comparison, we use the CMA-ES Matlab source code released in CMA-ES official website [3]. We run both algorithms for several trials for each single experiment. For both algorithms, we used the same population size and both algorithms will start with a same initial distribution in each trial. However, initial distribution for each trial is varied slightly. Also, both algorithms use fixed hyper parameter settings, i.e, throughout the experiments for both algorithms, we don't set any hyper parameters by hand[4].

[3] https://www.lri.fr/~hansen/cmaes_inmatlab.html

[4] **Matlab source code of TR-CMA-ES (with examples) as well as videos regarding the robotics experiments are available at** https://goo.gl/9u15Wf

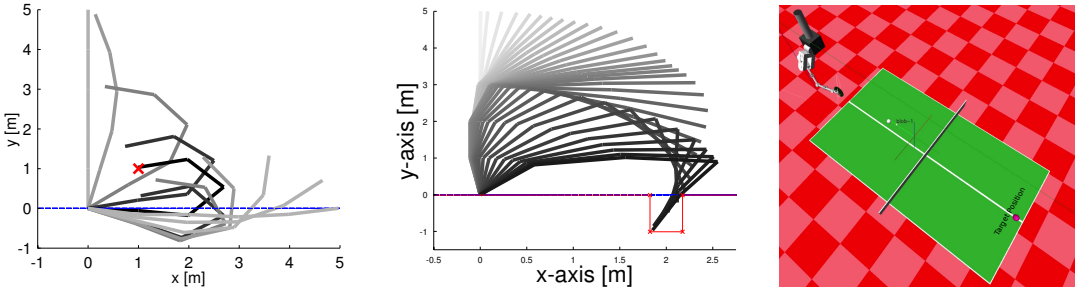
4.1 Standard Functions

We empirically evaluate TR-CMA-ES on 8 benchmark functions from [10]. As TR-CMA-ES is a maximiser we multiply all the functions values by -1 to turn their minimum to a maximum. Now, except for PrabR function that is unbounded, all other functions have a global maximum of zero. For both TR-CMA-ES and CMA-ES, we use the formula given in algorithm 1 to define the population size. For each trial, we randomly generate the mean of the initial distribution from a normal distribution with zero mean. We set the initial covariance matrix as a identity matrix with initial $\sigma = 1$. We ran TR-CMA-ES and CMA-ES on these functions with dimensions from 5 to 60 and a target fitness of -10^{-5} (10^3 for ParabR). We perform 20 trials for each dimension and report the average number of fitness evaluations to reach the target fitness.

Results. Figure 2 provides the full results on the set of eight benchmark functions. The results show that TR-CMA-ES in most of the cases, outperforms CMA-ES in terms of number of evaluations to reach the target fitness. In fact only in some dimensions of ParabR and Cigar functions CMA-ES performs better than TR-CMA-ES.

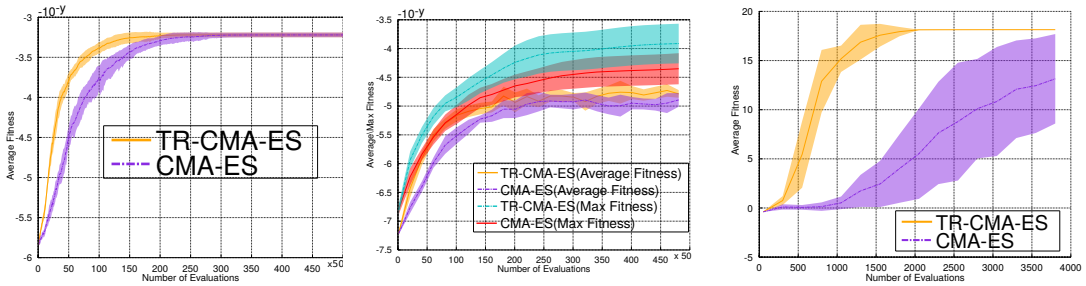
4.2 Simulated Robotics Tasks

We use a 5-link planar robot that has to reach a given point in task space as a toy task for the comparisons, see Figure 3(a). We subsequently made the task more difficult by introducing hard obstacles, which results in a discontinuous fitness function. We denote this task as hole-reaching task, see Figure 3(b). Finally, we evaluate our algorithm on a physical simulation of a robot playing table tennis (Figure 3(c)). For all tasks, we use dynamic motor primitives (DMPs) [11] as trajectory generator and we optimise the parameters of the DMPs to



(a) Reaching Task (b) Hole Reaching Task (c) Table Tennis

Figure 3: (a) Planar reaching task: a 5-link planar robot has to reach a via-point $\boldsymbol{v}_{50} = [1, 1]$ in task space. The via-point is indicated by the red cross. The postures of the resulting motion are shown as overlay, where darker postures indicate a posture which is close in time to the via-point. (b) Planar hole reaching task: A 5-link planar robot has to reach the bottom of a hole centring at point $[2\ 0]$ in task space while avoiding any collision with the walls. The hole is indicated by the red lines. The postures of the resulting motion are shown as overlay, where darker postures indicate a posture which is close in time to reach the bottom of the hole. (c) The table tennis task: The incoming ball has a fixed initial velocity. The goal of the robot is to learn forehand strokes to return the ball to a fixed target position.



(a) Reaching Task (b) Hole Reaching Task (c) Table Tennis

Figure 4: Comparisons for the three simulated robotics tasks which were used for comparison. (a) Reaching task (b) Hole reaching task and (c) Table Tennis task. Results show that TR-CMA-ES outperforms CMA-ES in terms of number of needed evaluations on all task. In the hole reaching task as well as in the table tennis task, TR-CMA-ES has a better average final performance than CMA-ES. Please note that y in -10^{-y} is value on y axis.

achieve optimum movements. For all tasks, we generate 50 samples in each iterations. We ran 10 trials and we report the average fitness in each iteration and the standard deviations over all trials. All other settings are set as before if not stated otherwise.

Planar Reaching Task. For completing the reaching task, the robot has to reach a via-point $\boldsymbol{v}_{50} = [1, 1]$ at time step 50 with its end-effector and at the final time step $T = 100$ the point $\boldsymbol{v}_{100} = [5, 0]$. The reward was given by a quadratic cost term for the distance from two via-points as well as quadratic costs for high accelerations. The DMPs goal attractor for reaching the final state was assumed to be known. Hence, the parameter vector $\boldsymbol{x}$ for a 5-link robot with 5 basis function for each degree of freedom had 25 dimensions. Figure 4(c) shows the learning progress. TR-CMA-ES again outperform CMA-ES, while both algorithms could find the optimal solution.

Planar Hole Reaching Task. For completing the hole reaching task, the robot's end effector has to reach the bottom of a hole ($35cm$ wide and $1m$ deep) centered point $[2, 0]$ without any collision with the ground or the hole wall. The reward was given by a quadratic cost term for the distance to bottom of the hole, quadratic costs for high accelerations and quadratic costs for collisions with the environment. Note that this objective function is highly discontinuous due to the quadratic costs for the collisions. The DMP goal attractor for reaching the final state in this task is unknown and also need to be learned. Hence, our lower level policy for a 5-link robot with 5 basis functions for each degree of freedom had 30 dimensions. Figure 4(b) shows the results. For this task we report both the maximum fitness and average fitness in each iteration. In both cases TR-CMA-ES illustrates a better performance.

Table Tennis Task. In this task, we use a simulated robot arm (see Figure 3(c)) to learn a forehand hitting stroke in a table tennis game. The robot is mounted on a floating base and has 8 actuated joints including the floating base. The goal of the robot is to return the incoming ball at a target position on the opponent's side of the table. The ball is always served with same initial velocities in the x,y and z directions. To learn the task, we initialize the mean of the distribution with a initial DMP trajectory obtained by kinaesthetic teaching, such that the movement generates a single forehand stroke. We only learn the final positions and final velocities of the DMP

Algorithm 1 TR-CMA-ES

given n(Dimension), $N = 4 + \lfloor 3\log(n) \rfloor$(Number of Individuals)
initialize $\boldsymbol{\mu}_{t=0}$, $\sigma_{t=0} > 0$, $\boldsymbol{p}_{c,t=0} = 0$, $\Sigma_{t=0} = \boldsymbol{I}$, $t = 0$
repeat
  for i = 1,...,N **do**
    $\boldsymbol{x}_i = \boldsymbol{\mu}_t + \sigma_t \times \mathcal{N}(0, \Sigma_t)$
    $f_i = f(\boldsymbol{x}_i)$
  end for
  $\boldsymbol{w}$ = **Compute Weights**$(\{\boldsymbol{x}_i, f_i\}_{i=1\ldots N})$ (See preliminaries section)
  Compute variance effective selection mass: $\mu_w = \frac{\sum_{i=1}^{N} w_i}{\sum_{i=1}^{N} w_i^2}$
  Set Hyper Parameters
  $\lambda_\sigma = 1$, $\lambda_\Sigma = \frac{4n}{(n+1.3)^2+\mu_w}$, $\epsilon_\Sigma = \min(0.2, 1.5\frac{\mu_w+\frac{1}{\mu_w}}{(n+2)^2+\mu_w})$
  $\epsilon_\sigma = \frac{\mu_w^2}{2n}$, $\epsilon_{\boldsymbol{\mu}} = 1000$, $c_c = \frac{\mu_w+2}{n+\mu_w+5}$
  Update Mean:
  $\eta_{\boldsymbol{\mu}}^* = \text{argmin}_\eta\, g_{\boldsymbol{\mu}}(\eta)$
  $\boldsymbol{\mu}_{t+1} = \frac{\eta_{\boldsymbol{\mu}}^* \boldsymbol{\mu}_t + \sum w_i \boldsymbol{x}_i}{\sum w_i + \eta_{\boldsymbol{\mu}}^*}$
  Compute Evolution Path:
  $\boldsymbol{p}_{c\,t+1} = (1-c_c)\boldsymbol{p}_{c\,t} + \sqrt{c_c(2-c_c)\mu_w}(\boldsymbol{\mu}_{t+1} - \boldsymbol{\mu}_t)$
  Update Covariance:
  $\eta_\Sigma^* = \text{argmin}_\eta\, g_\Sigma(\eta)$
  $\Sigma_{t+1} = \frac{\eta_\Sigma^* \Sigma_t + \sum_i w_i \frac{(\boldsymbol{x}_i-\boldsymbol{\mu}_t)(\boldsymbol{x}_i-\boldsymbol{\mu}_t)^T}{\sigma_t} + \lambda_\Sigma \frac{\boldsymbol{p}_c \boldsymbol{p}_c{}^T}{\sigma_t}}{\sum w_i + \lambda_\Sigma + \eta_\Sigma^*}$
  Update Step-Size:
  $\eta_\sigma^* = \text{argmin}_\eta\, g_\sigma(\eta)$
  $\sigma_{t+1} = \frac{n\eta_\sigma^* \sigma_t + \text{tr}(\Sigma_t{}^{-1}(\sum_i w_i(\boldsymbol{x}_i-\boldsymbol{\mu}_t)(\boldsymbol{x}_i-\boldsymbol{\mu}_t)^T + \lambda_\sigma \boldsymbol{p}_c \boldsymbol{p}_c{}^T))}{n(\sum w_i + \lambda_\sigma + \eta_\sigma^*)}$
  $t = t + 1$
until stopping criterion is met

trajectories as well as the τ time-scaling parameter and the starting time point of the DMP which results in 18 parameters. The reward function is defined by the sum of quadratic penalties for missing the ball (minimum distance between ball and racket trajectory) and missing the target return position. Figure 4(c) shows the result. TR-CMA-ES robustly finds the solution in all trials while CMA-ES could not find good solution in 3 trials out of 10. And TR-CMA-ES was faster.

5 CONCLUSION

In this paper, we derived the full CMA-ES update equations for mean and covariance with an expectation-maximization based framework using information-geometric trust regions. The presented update for the covariance matrix share the same structure as the CMA-ES algorithm. However, CMA-ES is not using trust region (i.e., use constraint on KL), but a penalty on KL is used in the objective as regularizer. As a consequence, the optimum 'Lagrangian' multipliers in CMA-ES are set by hand and remain fixed during the learning. However they are well established and can be left constant throughout many applications. In the trust region formulation, the Lagrangian multipliers are optimized for the given bound ϵ. As we show, in addition to the theoretical foundation, this algorithm gives us an improved performance over the original algorithm. In contrast to the mean and the covariance update, our step-size update does not match the step-size update from CMA-ES. Both, the TR-CMA-ES and CMA-ES take advantage of evolution path to set step size σ. However our update rule for the step size is obtained from the same principle as used for the mean and the covariance matrix. Given the similarities in terms of the mean and covariance matrix update between CMA-ES and our algorithm, our step size control update rule is more consistent and a more principled than the update rule is used in standard CMA-ES. The new step-size update also performs favourably in our experiments and should also be preferred for CMA-ES due to the consistent derivation. Our algorithm also enjoys all the invariance properties of the CMA-ES. For future work, we will investigate the effect of regularized mean update over unregulated mean update. Moreover we will investigate other theoretical aspects of our framework such as the theoretical difference between the forward KL and inverse KL trust region. We will also extend our framework for solving contextual stochastic search problems and full reinforcement learning problems.

6 ACKNOWLEDGEMENTS

We thank the anonymous reviewers for their careful reading of our manuscript and their many insightful comments and suggestions. This research was funded by Palo Alto Research Centre (PARC) and also was funded by European Unions FP7 under EuRoC grant agreement CP-IP 608849 and by LIACC (UID/CEC/00027/2015) and IEETA (UID/CEC/00127/2015).

REFERENCES

[1] A. Abdolmaleki, R. Lioutikov, J. Peters, N. Lua, L.P. Reis, and G. Neumann. 2015. Model Based Relative Entropy Stochastic Search. In *NIPS*.
[2] A. Abdolmaleki, N. Lua, L.P. Reis, and G. Neumann. 2015. Regularized covariance estimation for weighted maximum likelihood policy search methods. In *Humanoids Conference*.
[3] Y. Akimoto, A. Auger, and N. Hansen. 2014. Comparison-based natural gradient optimization in high dimension. *GECCO* (2014).
[4] Y. Akimoto, Y. Nagata, I. Ono, and S. Kobayashi. 2012. Theoretical foundation for CMA-ES from information geometry perspective. *Algorithmica* (2012).
[5] Y. Akimoto, J. Sakuma, I. Ono, and S. Kobayashi. 2008. Functionally specialized CMA-ES: a modification of CMA-ES based on the specialization of the functions of covariance matrix adaptation and step size adaptation. *GECCO* (2008).
[6] S. Amari. 1998. Natural Gradient Works Efficiently in Learning. *Neural Computation* (1998). Issue 2. DOI: http://dx.doi.org/10.1162/089976698300017746
[7] P. Dayan and G.E. Hinton. 1997. Using expectation-maximization for reinforcement learning. *Neural Comput.* 9, 2 (1997), 271–278. DOI: http://dx.doi.org/10.1162/neco.1997.9.2.271
[8] A. Dempster, N. Laird, and D. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B* 39, 1 (1977), 1–38.
[9] N. Hansen. 2016. The CMA Evolution Strategy: A Tutorial. Tutorial. (2016).
[10] N. Hansen, S.D. Mller, and P. Koumoutsakos. 2003. Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA-ES). *Evolutionary Computation* (2003).
[11] A. Ijspeert and S. Schaal. 2003. Learning Attractor Landscapes for Learning Motor Primitives. In *Advances in Neural Information Processing Systems 15(NIPS)*.
[12] J. Kober, B.J. Mohler, and J. Peters. 2008. Learning Perceptual Coupling for Motor Primitives. In *Intelligent Robots and Systems (IROS)*. 834–839.
[13] I. Loshchilov, M. Schoenauer, and M. Sebag. 2013. KL-based Control of the Learning Schedule for Surrogate Black-Box Optimization. *CoRR* (2013).
[14] S. Mannor, R. Rubinstein, and Y. Gat. 2003. The Cross Entropy method for Fast Policy Search. In *Proceedings of the 20th International Conference on Machine Learning(ICML)*.
[15] G. Neumann. 2011. Variational Inference for Policy Search in Changing Situations. In *ICML*.
[16] Y. Ollivier, L. Arnold, A. Auger, and N. Hansen. 2011. Information-geometric optimization algorithms: A unifying picture via invariance principles.. In *arXiv preprint arXiv:1106.3708*.
[17] J. Peters, K. Mülling, and Y. Altun. 2010. Relative Entropy Policy Search. In *AAAI*. AAAI Press.
[18] J. Schulman, S. Levine, P. Abbeel, M. I. Jordan, and P. Moritz. 2015. Trust Region Policy Optimization. In *ICML*.
[19] D. Wierstra, T. Schaul, T. Glasmachers, Y. Sun, J. Peters, and J. Schmidhuber. 2014. Natural Evolution Strategies. *Journal of Machine Learning Research* (2014).