Contextual Covariance Matrix Adaptation Evolutionary Strategies

Abbas Abdolmaleki[1,2,3], **Bob Price**[5], **Nuno Lau**[1], **Luis Paulo Reis**[2,3], **Gerhard Neumann**[4,6*]
1: IEETA, DETI, University of Aveiro
2: DSI, University of Minho
3: LIACC, University of Porto, Portugal
4: CLAS, TU Darmstadt, Germany
5: PARC, A Xerox Company, USA
6: University of Lincoln

Abstract

Many stochastic search algorithms are designed to optimize a fixed objective function to learn a task, i.e., if the objective function changes slightly, for example, due to a change in the situation or context of the task, relearning is required to adapt to the new context. For instance, if we want to learn a kicking movement for a soccer robot, we have to relearn the movement for different ball locations. Such relearning is undesired as it is highly inefficient and many applications require a fast adaptation to a new context/situation. Therefore, we investigate contextual stochastic search algorithms that can learn multiple, similar tasks simultaneously. Current contextual stochastic search methods are based on policy search algorithms and suffer from premature convergence and the need for parameter tuning. In this paper, we extend the well known CMA-ES algorithm to the contextual setting and illustrate its performance on several contextual tasks. Our new algorithm, called contextual CMA-ES, leverages from contextual learning while it preserves all the features of standard CMA-ES such as stability, avoidance of premature convergence, step size control and a minimal amount of parameter tuning.

1 Introduction

The notion of multi-task learning[1] has been stablished in the machine learning community for at least the past two decades [Caruana, 1997]. The main motivation for contextual learning is the potential for exploiting relevant information available in related tasks by concurrent learning using a shared representation. Therefore, instead of learning one task at a time, we would like to learn multiple tasks at once and exploit the correlations between related tasks. We use a context vector to characterize a task which typically changes from one task execution to the next. For example, consider a humanoid soccer robot that needs to pass the ball to its team mates which are positioned on different locations on the field. Here, the soccer robot should learn to kick the ball to any given target location, which is specified by the context vector, on the field. In such cases, learning for every possible context is clearly inefficient or even infeasible. Therefore our goal is to generalize learned tasks from similar contexts to a new context. To do so, we learn a context-dependent policy for a continuous range of contexts without restarting the learning process. In this paper, we consider stochastic search algorithms for contextual learning. Stochastic search algorithms [Hansen *et al.*, 2003; Sun *et al.*, 2009; Rückstieß *et al.*, 2008] optimize an objective function in a continuous domain. These algorithms assume that the objective function is a black box function, i.e., they only use the objective values and don't require gradients or higher-order derivatives of the objective function. Contextual stochastic search algorithms have been investigated in the field of policy search for robotics [Kupcsik *et al.*, 2013; Kober *et al.*, 2010]. However, these policy search algorithms typically suffer from premature convergence and perform unfavourably in comparison to state of the art stochastic search methods [Stulp and Sigaud, 2012] such as the CMA-ES algorithm [Hansen *et al.*, 2003]. The CMA-ES algorithm is considered as the state of the art in stochastic optimization. CMA-ES 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. However, CMA-ES is lacking the important feature of contextual (multi-task) learning. Please note that standard CMA-ES can be used to optimize the parameters of a closed loop policy that depends on context. However, the dimension of parameter space would grow linearly with the number of context dimensions. Moreover, to evaluate a parameter vector we would need to call the objective function for many different contexts and average the results. Both facts limit the data efficiency as it has been shown by [Ha and Liu, 2016]. Therefore, in this paper, we extend the well known CMA-ES algorithm to contextual setting using inspiration from contextual policy search. Our new algorithm, called contextual CMA-ES, generalizes the learned solution to new, unseen contexts during the optimization process while it preserves all the features of standard CMA-ES such as stability, avoidance of pre-

*Contact email: abbas.a@ua.pt

[1]The terms "contextual learning" and "multi-task learning" are used interchangeably throughout this paper.

mature convergence, step size control, a minimal amount of parameter tuning and simple implementation. In our derivation of the algorithm, we also provide a new theoretical justification for the covariance matrix update rule of contextual CMA-ES algorithm that also applies to the non-contextual case and gives new insights into how the covariance update can be motivated. For illustration of the algorithm, we will use contextual standard functions and two contextual simulated robotic tasks which are robot table tennis, and a robot kick task. We show that our contextual CMA-ES algorithm performs favourably in comparison to other contextual learning algorithms.

2 Related Work

In order to generalize a learned parameter vector for a context to the other contexts, a standard approach is to optimize the parameters for several target contexts independently. Subsequently, regression methods are used to generalize the optimized contexts to a new, unseen context [Da Silva *et al.*, 2012; Stulp *et al.*, 2013]. Although such approaches have been used successfully, they are time consuming and inefficient in terms of the number of needed training samples as optimizing for different contexts and the generalization between optimized parameters for different contexts are two independent processes. Hence, we cannot reuse datapoints obtained from optimizing a task with context $\boldsymbol{s}$ to improve and accelerate the optimization of a task with another context $\boldsymbol{s}'$. Learning for multiple tasks without restarting the learning process is known as contextual (multi-task) policy search [Kupcsik *et al.*, 2013; Kober *et al.*, 2010; Deisenroth *et al.*, 2014]. In the area of contextual stochastic search algorithms, such a multi-task learning capability was established for information-theoretic policy search algorithms [Peters *et al.*, 2010], such as the Contextual Relative Entropy Policy Search (CREPS) algorithm [Kupcsik *et al.*, 2013]. However, it has been shown that REPS suffers from premature convergence [Abdolmaleki *et al.*, 2016; 2015a] as the update of the covariance matrix, which is based only on the current set of samples, is reducing the variance of the search distribution too quickly. In order to alleviate this problem, the authors of [Abdolmaleki *et al.*, 2016; 2015a] suggest to combine the sample covariance with the old covariance matrix, similar to the CMA-ES algorithm [Hansen *et al.*, 2003]. However this method does not take advantage of other features of CMA-ES such as step-size control. In [Ha and Liu, 2016] also, an evolutionary contextual learning algorithm for only single dimensional contextual problems was proposed. This method defines a set of discrete contexts and represents the mean of the search distribution of each segment of the space. They evolve the mean of each segment using the (1+1)-CMA-ES [Igel *et al.*, 2006] algorithm. The used discretization inherently limits the approach to one dimensional context variable.

3 Preliminaries

In this section, we will first formulate the problem statement and subsequently explain contextual stochastic search algorithms in general. Afterwards, we will emphasize the contextual Relative Entropy Policy Search (REPS) algorithm [Kupcsik *et al.*, 2013], which will provide insights for the development of the new contextual CMA-ES algorithm.

3.1 Problem Statement

We consider contextual black-box optimization problems that are characterized by a n_s-dimensional context vector $\boldsymbol{s}$. The task is to find for each context vector $\boldsymbol{s}$, an optimal parameter vector $\boldsymbol{\theta}^*_{\boldsymbol{s}}$ that maximizes an objective function $\boldsymbol{R}(\boldsymbol{s},\boldsymbol{\theta}) : \{\mathbb{R}^{n_s} \times \mathbb{R}^{n_\theta}\} \rightarrow \mathbb{R}$. Note that the objective function is also dependent on the given context vector $\boldsymbol{s}$. We want to find an optimal context dependent policy $m^*(\boldsymbol{s})$ in form of

$$\boldsymbol{\theta}^*_{\boldsymbol{s}} = m^*(\boldsymbol{s}) = \boldsymbol{A}^*\boldsymbol{\varphi}(\boldsymbol{s}), \;\; m^*(\boldsymbol{s}) : \mathbb{R}^{n_s} \rightarrow \mathbb{R}^{n_{\boldsymbol{\theta}}}$$

that outputs the optimal parameter vector $\boldsymbol{\theta}^*_{\boldsymbol{s}}$ for the given context $\boldsymbol{s}$.The vector $\boldsymbol{\varphi}(\boldsymbol{s})$ is an arbitrary $n_{\boldsymbol{\varphi}}$-dimensional feature function of the context $\boldsymbol{s}$ and the gain matrix $\boldsymbol{A}$ is a $n_{\boldsymbol{\theta}} \times n_{\boldsymbol{\varphi}}$ matrix that models the dependence of parameters $\boldsymbol{\theta}$ on the context $\boldsymbol{s}$. Throughout this paper, we use $\boldsymbol{\varphi}(\boldsymbol{s}) = [1 \;\; \boldsymbol{s}]$, which results in linear generalization over contexts. However, other feature functions such as radial basis functions (RBF) for non-linear generalization over contexts [Abdolmaleki *et al.*, 2015b] can also be used. The only accessible information on objective function $\boldsymbol{R}(\boldsymbol{s},\boldsymbol{\theta})$ are returns $\{R_k\}_{k=1\ldots N}$ of context-parameters samples $\{\boldsymbol{s}_k, \boldsymbol{\theta}_k\}_{k=1\ldots N}$, where k is the index of the sample and N is number of samples.

Algorithm 1 Contextual Stochastic Search Algorithms

given $n_{\boldsymbol{\theta}},\ n_{\boldsymbol{s}}, N,\ \boldsymbol{\varphi}(\boldsymbol{s}) = [1 \;\; \boldsymbol{s}]$
initialize $\boldsymbol{A}^{t=0}_{n_a \times n_{\boldsymbol{\varphi}}}, \sigma^{t=0} > 0, \boldsymbol{\Sigma}^{t=0} = \boldsymbol{I}_{n_{\boldsymbol{\theta}} \times n_{\boldsymbol{\theta}}},\ 0 \leftarrow t$
repeat
  for k = 1,...,N **do**
    Observe $\boldsymbol{s}_k$
    $m(\boldsymbol{s}_k) = \boldsymbol{A}^{t^T}\boldsymbol{\varphi}(\boldsymbol{s}_k)$
    $\boldsymbol{\theta}_k = m(\boldsymbol{s}_k) + \sigma^t \times \mathcal{N}\left(\boldsymbol{0}, \boldsymbol{\Sigma}^t\right)$
    $R_k = \boldsymbol{R}(\boldsymbol{s}_k, \boldsymbol{\theta}_k)$
  end for
  $\boldsymbol{d}$ = **ComputeWeights**$(\{\boldsymbol{s}_k, \boldsymbol{\theta}_k, R_k\}_{k=1\ldots N})$
  $\boldsymbol{A}^{t+1}$ = **UpdateMean**$(\{\boldsymbol{s}_k, \boldsymbol{\theta}_k, d_k\}_{k=1\ldots N})$
  $\boldsymbol{\Sigma}^{t+1}$ = **UpdateCov**$(\{\boldsymbol{s}_k, \boldsymbol{\theta}_k, d_k\}_{k=1\ldots N}, \boldsymbol{\Sigma}^t, \boldsymbol{A}^{t+1}, \boldsymbol{A}^t)$
  σ^{t+1} = **UpdateStepSize**$(\sigma^t, \boldsymbol{A}^{t+1}, \boldsymbol{A}^t)$
  $t \leftarrow t + 1$
until stopping criterion is met

3.2 Contextual Stochastic Search

Contextual Stochastic search algorithms maintain a conditional search distribution $\pi(\boldsymbol{\theta}|\boldsymbol{s})$ over the parameter space $\boldsymbol{\theta}$ of the objective function $\boldsymbol{R}(\boldsymbol{s},\boldsymbol{\theta})$. The search distribution $\pi(\boldsymbol{\theta}|\boldsymbol{s})$ is often modeled as a linear Gaussian distribution, i.e.,

$$\pi(\boldsymbol{\theta}|\boldsymbol{s}) = \mathcal{N}\left(\boldsymbol{\theta}|m(\boldsymbol{s}) = \boldsymbol{A}\boldsymbol{\varphi}(\boldsymbol{s}), \sigma\boldsymbol{\Sigma}\right),$$

where $m(\boldsymbol{s})$ is a context dependent mean function that represents the context dependent policy we want to learn, $\boldsymbol{\Sigma}$ is a covariance matrix (shape of the distribution) and σ is the step size (magnitude). Covariance matrix and step size are used

for exploration and are independent of the context in most setups. In each iteration, N context-parameter-return samples are generated with the current contextual policy. To do so, the context vectors $\boldsymbol{s}_k$ are drawn from a possibly unknown context distribution $\mu(\boldsymbol{s})$ [2]. Subsequently, the current search distribution $\pi^t(\boldsymbol{\theta}|\boldsymbol{s})$ is used to generate the parameter $\boldsymbol{\theta}_k$ for the corresponding context $\boldsymbol{s}_k$. For each sample k, the return R_k of $\{\boldsymbol{s}_k, \boldsymbol{\theta}_k\}$ is obtained by querying the objective function $\boldsymbol{R}(\boldsymbol{s}, \boldsymbol{\theta})$. Typically, the samples $\{\boldsymbol{s}_k, \boldsymbol{\theta}_k, R_k\}_{k=1...N}$ are used to compute a weight or pseudo probability d_k for each sample k. Subsequently, using $\{\boldsymbol{s}_k, \boldsymbol{\theta}_k, d_k\}_{k=1...N}$, a new conditional Gaussian search distribution $\pi^{t+1}(\boldsymbol{\theta}|\boldsymbol{s})$ is estimated by updating the gain matrix $\boldsymbol{A}^{t+1}$ of the context-dependent mean function, the covariance matrix $\boldsymbol{\Sigma}^{t+1}$ and step size σ^{t+1}. This process is run iteratively until the algorithm converges to a solution. The final solution is the mean function $m^*(\boldsymbol{s})$ with the estimate of the optimal gain matrix $\boldsymbol{A}^*$ in the last iteration. Please note that, if the context vector is fixed, then the explained algorithm reduces to standard stochastic search where the mean function is a constant. Algorithm 1 shows a compact representation of contextual policy search methods.

3.3 Contextual REPS

Contextual REPS [Kupcsik *et al.*, 2013] is an instance of the general stochastic search algorithms introduced in the previous section where the weight computation and the distribution update are performed in a specific way.

Computing the Weights. In order to obtain new weights for the context-parameters-return samples in the data set, contextual REPS, optimizes for the joint probabilities $p(\boldsymbol{s}_k, \boldsymbol{\theta}_k)$. The key idea behind contextual REPS is to find a joint search distribution $p(\boldsymbol{s}, \boldsymbol{\theta})$ that maximizes the expected return i.e., $\max_p \iint p(\boldsymbol{s}, \boldsymbol{\theta}) R_s(\boldsymbol{\theta}) d\boldsymbol{s} d\boldsymbol{\theta}$, while it ensures a smooth and stable learning process by bounding the Kullback-Leibler divergence between the old search distribution q and the newly estimated search distribution p, i.e, $\epsilon \geq \mathrm{KL}(p(\boldsymbol{s}, \boldsymbol{\theta}) || q(\boldsymbol{s}, \boldsymbol{\theta}))$. Please see [Kupcsik *et al.*, 2013] for full description of optimization program of contextual REPS. The solution of the contextual REPS optimization program results in a weight

$$d_k = \exp\left(\frac{R_k - V(\boldsymbol{s})}{\eta}\right) / Z \ , \ Z = \sum_{k=1}^{N} d_k, \qquad (1)$$

for each context-parameter sample $[\boldsymbol{s}_k, \boldsymbol{\theta}_k]$. The function $V(\boldsymbol{s}) = \boldsymbol{\phi}(\boldsymbol{s})^T \boldsymbol{w}$ is a context-dependent baseline, similar to a value function, that depends linearly on features $\boldsymbol{\phi}(\boldsymbol{s})$ of the context vector $\boldsymbol{s}$. It is subtracted from the return R. Intuitively, this subtraction allows us to assess the quality of the samples independently of the experienced context. In this paper, we use a quadratic feature function for the baseline, for example, in a one dimensional contextual problem, we use $\boldsymbol{\phi}(s) = [s, \ s^2]$. The parameters $\boldsymbol{w}$ and η are Lagrangian multipliers that can be obtained by optimizing the convex dual function of the REPS optimisation program, i.e.,

$$\min_{\eta, \boldsymbol{w}} g(\eta, \boldsymbol{w}) = \eta\epsilon + \hat{\boldsymbol{\phi}}^T \boldsymbol{w} + \eta \log\left(\sum_{K=1}^{N} \frac{1}{N} \exp\left(\frac{R^{[k]} - \boldsymbol{\phi}(\boldsymbol{s}^{[k]})^T \boldsymbol{w}}{\eta}\right)\right), \qquad (2)$$

where $\hat{\boldsymbol{\phi}} = \sum_{k=1}^{N} \frac{1}{N} \boldsymbol{\phi}(\boldsymbol{s}^{[k]})$ is the observed context feature expectation. ϵ is the maximum desired KL-divergence. The dual-function is non-linear but convex, and, hence, can be optimized efficiently by any non-linear optimization algorithm. For example we use fmincon tool in matlab.

Updating the Search Distribution. In contextual REPS, the step size is fixed to 1, i.e., $\sigma^t = 1$. In order to obtain a new Gaussian search distribution π^{t+1}, contextual REPS directly uses a weighted maximum likelihood estimate, i.e,

$$\underset{\boldsymbol{\Sigma}^{t+1}, \boldsymbol{A}^{t+1}}{\operatorname{argmax}} \sum_{k=1}^{N} d_k \log \pi(\boldsymbol{\theta}_k | \boldsymbol{s}_k; \boldsymbol{\Sigma}^{t+1}, \boldsymbol{A}^{t+1}). \qquad (3)$$

Mean Function Update Rule. We can efficiently solve the optimization program in equation 3 for $\boldsymbol{A}^{t+1}$ in closed form. The solution for $\boldsymbol{A}^{t+1}$ is given by

$$\boldsymbol{A}^{t+1} = \left(\boldsymbol{\Phi}^T \boldsymbol{D} \boldsymbol{\Phi} + \lambda \boldsymbol{I}\right)^{-1} \boldsymbol{\Phi}^T \boldsymbol{D} \boldsymbol{U}, \qquad (4)$$

where $\boldsymbol{\Phi}^T = [\boldsymbol{\varphi}_1, ..., \boldsymbol{\varphi}_N]$ contains the feature vector for the policy for all sample contexts (see preliminary), $\boldsymbol{U} = [\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_N]$ contains all the sample parameters and $\boldsymbol{D}$ is the diagonal weighting matrix containing the weightings d_k. The term $\lambda \boldsymbol{I}$ is a regularization term.

Covariance Matrix Update Rule. We can also solve the optimization program in equation 3 for $\boldsymbol{\Sigma}^{t+1}$ in closed form. The solution for $\boldsymbol{\Sigma}^{t+1} = \boldsymbol{S}$ which is also known as sample covariance $\boldsymbol{S}$ matrix and is given by

$$\boldsymbol{S} = \sum_{k=1}^{N} d_k \left(\boldsymbol{\theta}_k - \boldsymbol{A}^{\boldsymbol{t+1}} \boldsymbol{\varphi}_k\right) \left(\boldsymbol{\theta}_k - \boldsymbol{A}^{\boldsymbol{t+1}} \boldsymbol{\varphi}_k\right)^T. \qquad (5)$$

As contextual REPS, most contextual policy search algorithms only use the current set of samples to estimate the new search distribution. It has been already noted by several authors [Abdolmaleki *et al.*, 2015a; Stulp and Sigaud, 2012] that such approach causes problems with premature convergence as the covariance matrix is overfitted to the data and, consequently, the covariance update reduces the variance too quickly.

4 Contextual Covariance Matrix Adaptation Evolutionary Strategies

Current contextual stochastic search algorithms are lacking the beneficial features from CMA-ES such as pre-mature convergence avoidance, step-size control and a minimal set of tuning parameters. To this end, we will contextualize the CMA-ES algorithm and hence inherit all its beneficial features. We will now explain the contextual CMA-ES rules for computing the weights as well as the distribution updates.

[2] Please note that the context samples depends on the task in an uncontrollable manner. However, throughout this paper, we use a uniform distribution to sample contexts for simplicity.

Algorithm 2 Contextual CMA-ES

1: **given** n, $n_{\boldsymbol{s}}$, $n_c = n + n_{\boldsymbol{s}}$, $N = 4 + \lfloor 3 \ln n_c \rfloor (1 + 2n_{\boldsymbol{s}})$
2: **initialize** $\boldsymbol{A}^{t=0}, \sigma^{t=0} > 0, p_\sigma^{t=0} = 0, p_c^{t=0} = 0, \Sigma^{t=0} = \boldsymbol{I}, 0 \leftarrow t$
3: **repeat**
4: **for** k = 1,...,N **do**
5: **Observe** $\boldsymbol{s}_k$
6: $m(\boldsymbol{s}_k) = \boldsymbol{A^t}\boldsymbol{\varphi}(\boldsymbol{s}_k)$
7: $\boldsymbol{\theta}_k = m(\boldsymbol{s}_k) + \sigma^t \times \mathcal{N}\left(0, \boldsymbol{\Sigma}^t\right)$
8: $R_k = \boldsymbol{R}_{\boldsymbol{s}_k}(\boldsymbol{\theta}_k)$
9: **end for**
10: $\boldsymbol{d}$ = **ComputeWeights**$(\{\boldsymbol{s}_k, \boldsymbol{\theta}_k, R_k\}_{k=1\ldots N})$ (Eq. 6)
11: **Set Hyper Parameters:**
$\mu_w = \frac{1}{\sum_{i=1}^N d_k^2}$(Number of effective samples)
Covariance Hyper Parameters
$c_1 = \frac{2\min(1, \lambda/6)}{(n_c+1.3)^2+\mu_w}, c_\mu = \frac{2(\mu_w - 2 + 1/\mu_w)}{(n_c+2)^2+\mu_w}, c_c = \frac{4}{4+n_c}$
Step Size Hyper Parameters
$c_\sigma = \frac{\mu_w+2}{n_c+\mu_w+3}, d_\sigma = 1 + c_\sigma + 2\sqrt{\frac{\mu_w-1}{n_c+1}} - 2 + \log(1+2n_{\boldsymbol{s}})$
12: **Update Mean Function**(Eq. 3)
13: **Update Evolution Path**
$\hat{\boldsymbol{\varphi}} = \sum_{k=1}^N \frac{1}{N}\boldsymbol{\varphi}(\boldsymbol{s}_k)$, $\boldsymbol{y} = \frac{\boldsymbol{A^{t+1}}\hat{\boldsymbol{\varphi}} - \boldsymbol{A^t}\hat{\boldsymbol{\varphi}}}{\sigma^t}$
$\boldsymbol{p}_c^{t+1} \leftarrow (1-c_c)\boldsymbol{p}_c^t + h_\sigma\sqrt{c_c(2-c_c)}\sqrt{\mu_w}\boldsymbol{y}$
$\boldsymbol{p}_\sigma^{t+1} \leftarrow (1-c_\sigma)\boldsymbol{p}_\sigma^t + \sqrt{c_\sigma(2-c_\sigma)}\sqrt{\mu_w}(\boldsymbol{\Sigma}^t)^{\frac{-1}{2}}\boldsymbol{y}$
14: **Update Covariance Matrix**
$\boldsymbol{S} = \frac{1}{(\sigma^t)^2}\sum_{k=1}^N d_k(\boldsymbol{\theta}_k - \boldsymbol{A^t}^T\boldsymbol{\varphi}(\boldsymbol{s}_k))(\boldsymbol{\theta}_k - \boldsymbol{A^t}^T\boldsymbol{\varphi}(\boldsymbol{s}_k))^T$
$\boldsymbol{\Sigma}^{t+1} = (1 - c_1 - c_\mu)\boldsymbol{\Sigma}^t + \underbrace{c_\mu \boldsymbol{S}}_{\text{rank-}\mu\text{ update}} + \underbrace{c1\ \boldsymbol{p}_c^{t+1}\boldsymbol{p}_c^{t+1^T}}_{\text{rank-one update}}$
15: **Update Step Size:** $\sigma^{t+1} \leftarrow \sigma^t \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|\mathbf{p}_\sigma^{t+1}\|}{\mathbb{E}\|\mathcal{N}(\mathbf{0}, \boldsymbol{I})\|}\right)\right)$
16: $t \leftarrow t + 1$
17: **until** stopping criterion is met

4.1 Computing the Weights

CMA-ES originally ranks the samples based on their returns and, subsequently, it weights the samples based on this ranking such that better samples get higher weights. However before we rank the samples, we need to correct the returns from their context-dependent part so that we can judge the quality of the parameter vector $\boldsymbol{\theta}$ independently of the quality of the context $\boldsymbol{s}$. To do so, inspired by the contextual REPS, we compute a context-dependent baseline $V(\boldsymbol{s})$ which we subtract from the returns R_k to compute advantages A_k, i.e.,

$$A_k = R_k - V(\boldsymbol{s}_k).$$

The baseline function $V(\boldsymbol{s})$ is estimated from the current dataset $\{\boldsymbol{s}_k, R_k\}_{k=1\ldots N}$ and captures the average return of the samples for context $\boldsymbol{s}$. I.e., $V(\boldsymbol{s})$ is a value function that captures the expected return for a given context using the current search distribution. In order to learn the baseline, we can use ridge linear regression to fit a function of the form $V(\boldsymbol{s}) = \boldsymbol{\beta}^T\boldsymbol{\phi}(\boldsymbol{s})$, where $\boldsymbol{\phi}(\boldsymbol{s})$ defines the context features and $\boldsymbol{\beta}$ can be obtained using linear regression. In this paper, the feature function $\boldsymbol{\phi}(\boldsymbol{s})$ is similarly defined as the feature function $\boldsymbol{\phi}$ used for contextual REPS, but it contains an additional bias term, i.e., $\boldsymbol{\phi}(\boldsymbol{s}) = [1 \quad \boldsymbol{\phi}(\boldsymbol{s})^T]^T$. After computing the new dataset $\{\boldsymbol{s}_k, \boldsymbol{\theta}_k, A_k\}_{k=1\ldots N}$, we can now use the CMA-ES ranking to compute the weights d_k for each context-parameter sample pair $[\boldsymbol{s}_k, \boldsymbol{\theta}_k]$. We first sort the dataset $\{\boldsymbol{s}_k, \boldsymbol{\theta}_k, A_k\}_{k=1\ldots N}$ in ascending order with respect to the advantage values A_k. Subsequently, the weight of the jth best sample in the list is set to

$$d^j = \ln(N + 0.5) - \ln(j) \tag{6}$$

, which will give us the new dataset $\{\boldsymbol{s}_k, \boldsymbol{\theta}_k, d_k\}_{k=1\ldots N}$ that will be used to update the search distribution. Without loss of generality, we will assume that the weights sum to 1.

4.2 Search Distribution Update Rule

Next, we will explain the update rules for contextual CMA-ES and give a new mathematical interpretation for covariance matrix update rule. We will explain the parts that are relevant for contextualizing the standard CMA-ES. For further explanations regarding the tricks used in standard CMA-ES we refer to [Hansen, 2016]. The complete contextual CMA-ES algorithm including all parameter settings are outlined in Algorithm 2. We will refer to the lines of the Algorithm 2. *Mean Function Update Rule.* In standard CMA-ES, the mean of the distribution is a constant and not a context-dependent function. To update the constant mean, standard CMA-ES uses weighted average of samples which is also the solution of weighted maximum likelihood estimate. In order to update the context-dependent mean of contextual CMA-ES, we directly use the update rule we obtained for contextual REPS from Equation 4 to obtain the context-dependent mean function of our distribution.

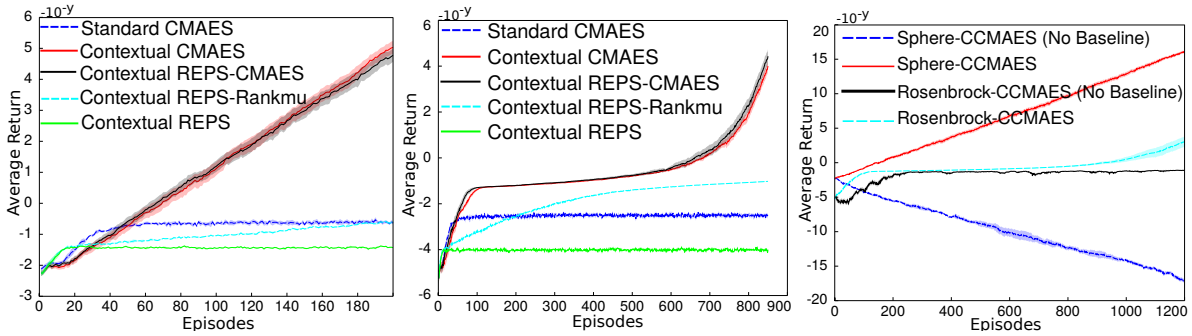
Covariance Matrix Update Rule. The covariance matrix update of CMA-ES consists of two parts (Line 14) which are the rank-μ and the rank-one updates.

Rank-μ Update The rank-μ update in the standard CMA-ES algorithm incorporates the information about the current successful steps (Line 14). This information is stored in the sample covariance matrix $\boldsymbol{S}$ (Line 14). The sample covariance in the CMA-ES algorithm is computed differently than in REPS. While REPS uses the new context dependent mean function $\boldsymbol{m}^{t+1}(\boldsymbol{s})$ to compute the covariance matrix $\boldsymbol{\Sigma}^{t+1}$(Equation 5), CMA-ES uses the old context dependent mean function $\boldsymbol{m}^t(\boldsymbol{s})$, i.e.,

$$\boldsymbol{S}_{cma} = \frac{1}{\sigma^{t^2}}\sum_{k=1}^N d_k\big(\boldsymbol{\theta}_k - \boldsymbol{A^t}\boldsymbol{\varphi}(\boldsymbol{s}_k)\big)\big(\boldsymbol{\theta}_k - \boldsymbol{A^t}\boldsymbol{\varphi}(\boldsymbol{s}_k)\big)^T.$$

By using the old mean function $m^t(\boldsymbol{s})$, we are increasing the likelihood of successful steps instead of likelihood of successful samples. In the other word we are interested in repeating the mutations(or steps) that resulted in current good samples instead of repeating the current good samples themselves. This approach has been shown to be less prone to premature convergence [Hansen, 2016].

Rank-one Update In standard CMA-ES, the rank-one update uses a vector called evolution path $\boldsymbol{p}$. *Evolution path* records the sum of consecutive step updates of the mean



(a) Sphere (b) Rosenbrock (c) Baseline evaluation

Figure 1: The performance comparison of stochastic search methods for optimizing contextual version of standard functions (a)Sphere (b)Rosenbrock, The results show that while both contextual CMA-ES and contextual REPS-CMAES perform well, Contextual REPS suffers from premature convergence and contextual REPS-rankμ is very slow which shows the importance of step size control and incorporation of evolution path. We also compared with standard CMA-ES to show the importance of contextual version of CMA-ES. (c) Evaluation of influence of baseline in contextual CMA-ES for the Sphere and Rosenbrock functions. As the figure shows, the baseline is crucial for good performance of contextual CMA-ES. Please note that y in -10^{-y} is value on y axis.

$\boldsymbol{y} = \frac{\boldsymbol{m}^{t+1} - \boldsymbol{m}^t}{\sigma^t}$ of the search distribution. 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. For a full explanation see [Hansen, 2016]. However in contextual CMA-ES, the mean is now a context dependent function and not a constant. Therefore, to compute the evolution path $\boldsymbol{p}_c$ (Line 13), we use the expected mean update of the search distribution over the context distribution $\mu^t(\boldsymbol{s})$, i.e,

$$\boldsymbol{y} = \frac{\mathbb{E}_{s\sim\mu^t(\boldsymbol{s})}[m^{t+1}(\boldsymbol{s}) - m^t(\boldsymbol{s})]}{\sigma^t}.$$

Computing $\boldsymbol{y}$ for our samples reads

$$\boldsymbol{y} = \frac{(\boldsymbol{A^{t+1}} - \boldsymbol{A^t})\hat{\boldsymbol{\varphi}}}{\sigma^t}\ ,\ \hat{\boldsymbol{\varphi}} = \sum_{k=1}^{N} \frac{1}{N}\boldsymbol{\varphi}(\boldsymbol{s}_k). \tag{7}$$

The final covariance matrix update rule combines the old covariance matrix, sample covariance matrix and the evolution path information matrix, i.e,

$$\boldsymbol{\Sigma}^{t+1} = (1 - c_1 - c_\mu)\boldsymbol{\Sigma}^t + \underbrace{c_\mu \boldsymbol{S}_{cma}}_{\text{rank-}\mu\text{ update}} + \underbrace{c_1\, \boldsymbol{p}_c^{t+1}\boldsymbol{p}_c^{t+1^T}}_{\text{rank-one update}}.$$

The factors c_1 and c_μ are the corresponding factors for rank-one and rank-μ updates such that $c_1 + c_\mu \leq 1$.

Interpretation of the CMA-ES Covariance Update Rule. Originally, the CMA-ES update rules have been obtained from intuitive, well-defined heuristics. Recently, it has been shown that the rank-μ update of covariance follows an approximate natural gradient of the expected returns [Akimoto *et al.*, 2012]. In this section, we give a new mathematical interpretation for contextual CMA-ES covariance matrix update rule which also applies to standard CMA-ES. The covariance update rule in Line 14 has been shown very effective for reproducing past successful steps while avoiding premature convergence. In fact, this update rule can be obtained by maximizing the likelihood of weighted steps as well as the weighted evolution path-step while minimizing the KL-divergence between new and old search distribution to avoid over-fitting and premature convergence, i.e,

$$\underset{\boldsymbol{\Sigma}^{t+1}|m=m^t}{\operatorname{argmax}} \underbrace{\sum_{k=1}^{N} d_k \log \pi^{t+1}(\boldsymbol{\theta}_k|\boldsymbol{s}_k) +}_{\text{successful steps}}$$
$$\underbrace{\lambda \log \pi^{t+1}(\boldsymbol{p}_c^{t+1} + m^t(\hat{\boldsymbol{s}})|\hat{\boldsymbol{s}})}_{\text{evolution path}} - \underbrace{\gamma \mathrm{KL}(\pi^t|\pi^{t+1})}_{\text{Avoids overfitting}}.$$

Where $\hat{\boldsymbol{s}} = \sum_{k=1}^{N} \frac{1}{N}\boldsymbol{s}_k$. The notation $\{\boldsymbol{\Sigma}^{t+1}|m = m^t\}$ means that we optimize for $\boldsymbol{\Sigma}^{t+1}$ while the mean function is set to the old mean function m^t which results in an optimized $\boldsymbol{\Sigma}^{t+1}$ for successful steps. $\lambda > 0$ and $\gamma > 0$ define the trade off between maximizing the likelihood of successful steps and keeping the KL-divergence of the new and old search distribution small. By setting λ and γ to zero and setting the mean function to the new one i.e., $m = m^{t+1}$, we obtain the sample covariance matrix $\boldsymbol{S}$ used by REPS. Considering a Gaussian search distribution, we can solve this optimization program in closed form and obtain the exact form of covariance matrix update rule as shown in Line 14 (Please see supplement material for derivation details)[3]. This derivation allows for the first time to formulate a clearly defined objective to obtain the full CMAES update rules which gives us a better understanding of the algorithm.

4.3 Step Size Update Rule

The step-size adaptation control of CMAES also uses the evolution path vector $\boldsymbol{p}_\sigma$ (line 13) to correct the step size. The intuition is that if the successful steps between consecutive search distributions are towards the same direction, i.e., they

[3] https://goo.gl/MLzKsW

are correlated, the updates will sum up and the evolution path will have a high magnitude. In this case, the step size should be increased. If the update directions cancel each other out, the evolution path will have a small magnitude and the step size should be decreased. In the contextual case, similar to the rank-one update, we use the Equation 7 to compute the mean update between two consecutive search distribution. Line 15 shows the step size update rule. For a full explanation about CMA-ES step size control, see [Hansen, 2016].

4.4 Algorithm

Algorithm 2 shows a compact representation of the contextual CMA-ES algorithm. In each iteration, we generate N context-parameters samples and evaluate their return (Lines 3-9). Subsequently, we compute a weight for each individual based on the return (Line 10). We compute the number of effective samples (Line 11). Similar to standard CMA-ES, we empirically obtained a default setting for hyper parameters of the algorithm which scales with the dimension of context and parameter space and size of the population (Line 11). Subsequently we compute the new mean function and new expected evolution path (Line 12-13). Please note that similar to standard CMA-ES we also use different evolution path vector for step size and covariance updates. Finally, we obtain the new expected covariance matrix and new expected step size (Line 14-15)[4].

5 Hybrid Algorithms

From the general algorithmic description in Algorithm 1 we can see that the weight computation and the distribution update are mostly independent for the algorithms and can be exchanged. Hence, we can create hybrid versions of those, for example by combining the weight computation of REPS with the distribution update of contextual CMAES. We will denote this algorithm contextual REPS-CMAES.

6 Experiments

In this section, we evaluate four contextual algorithms such as CMA-ES, REPS, REPS-CMAES and REPS-Rankμ [Abdolmaleki *et al.*, 2016][5]. The REPS-CMAES algorithm uses the weighting method of REPS and the distribution update rule of CMA-ES. REPS-Rankμ also uses the weighting method of REPS but it uses only the rank-μ update rule of CMA-ES without step size adaptation. We also evaluate simultaneous multi task learning versus learning tasks in isolation. We chose two series of optimization tasks for comparisons. In the first series, we use the standard optimization test functions [Molga and Smutnicki, 2005] , and for the second series of optimization tasks, we use a robot table tennis task. For all experiments, the KL-bound ϵ for REPS is set to 1 and for all experiments we use the default hyper parameter settings given in Algorithm 1 without further tuning. Please note that if the context dimension n_s is set to zero, we get the parameter setting for standard CMA-ES.

[4]Expectation here is over context distribution

[5]**Matlab code for reproducing the results on standard functions as well as videos regarding the experiments (table tennis and robot kick) at** `https://goo.gl/MLzKsW`

6.1 Standard Functions

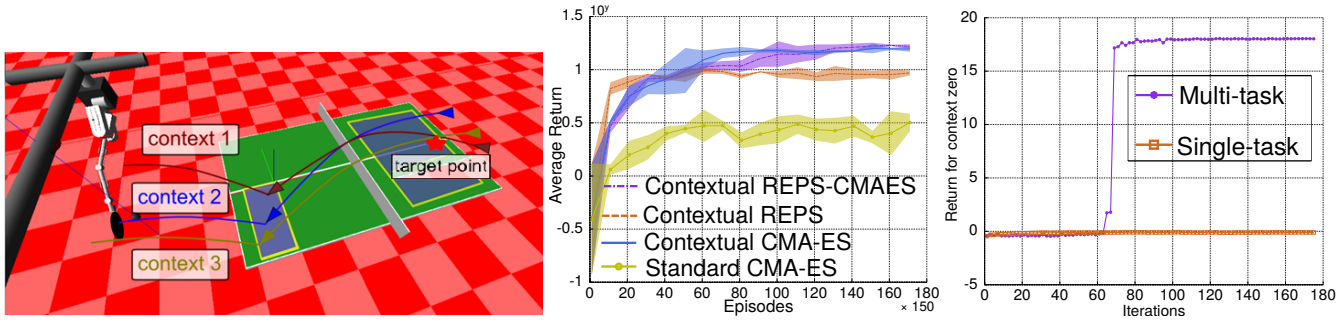
We chose two standard optimization functions which are the Sphere function $R_{\boldsymbol{s}}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \boldsymbol{x}_i^2$, and the Rosenbrock function $R_{\boldsymbol{s}}(\boldsymbol{\theta}) = \sum_{i=1}^{n-1}[100(\boldsymbol{x}_{i+1} - \boldsymbol{x}_i^2)^2 + (1 - \boldsymbol{x}_i)^2]$, where $\boldsymbol{x} = \boldsymbol{\theta} + \boldsymbol{G}\boldsymbol{s}$ adapts the sample $\boldsymbol{\theta}$ linearly with the context. The matrix $\boldsymbol{G}$ is a constant matrix and is sampled from a normal distribution. Our standard functions have a global minimum with a value of zero for every context. However our algorithm in the paper is a maximizer. Therefore, we multiply the functions by -1 such that now their maximum is 0. We want to find an optimum policy that for every context $\boldsymbol{s}$ outputs the optimal parameter $\boldsymbol{\theta}^*$. The optimum $\boldsymbol{\theta}_s^*$ for these functions is linearly dependent on the given context $\boldsymbol{s}$, hence, we initially test the performance of the algorithms under 'ideal contextual' conditions, i.e., the contextual policy is able to represent the optimal parameter vector $\boldsymbol{\theta}_{\boldsymbol{s}}^*$ for each context $\boldsymbol{s}$. We choose a linear policy $m(\boldsymbol{s}) = \boldsymbol{A}\boldsymbol{s} + b$. For the initial linear policy, $\boldsymbol{A}$ matrix is set to zero and b is sampled from a normal distribution. The initial covariance matrix is the identity matrix and the initial step size σ is 1. Moreover the contexts are sampled uniformly from interval $1 \leq \boldsymbol{s}_i \leq 2, i = 1, \ldots, n_s$ where n_s is dimension of the context space $\boldsymbol{s}$. For the Sphere we used a two dimensional context and 20 dimensional parameters while for the Rosenbrock function, we used a one dimensional context vector and 20 dimensional parameters. In each iteration we generate 50 context-parameters samples. We perform 20 trials for each experiment. And We report the average return of context-parameters samples in each iteration and the standard deviation over all 20 trials.

Algorithmic Comparison.

We compared contextual CMA-ES, contextual REPS, contextual REPS-CMAES, contextual REPS-Rank-μ and standard CMA-ES. The results in Figure 1(a) and Figure 1(b) show that contextual CMA-ES and contextual REPS-CMAES could successfully learn the contextual tasks while contextual REPS suffers from premature convergence and REPS-rankμ is too slow. As REPS-rankμ does not have step size control, this result shows the importance of step size control. Standard CMA-ES could not learn the task as it does not have any knowledge of the context. Please note that in this task standard CMA-ES which is a non-contextual algorithm has better performance than the contextual REPS. The reason is that contextual REPS suffers from premature convergence due to using only sample covariance. We already used more samples for contextual REPS to reduce the variance of covariance estimate and found a setting where contextual REPS found a better policy than standard CMA-ES. Yet, contextual REPS, needs much more samples to find such a policy. However contextual REPS with a proper covariance adaptation performs favourably(See Contextual REPS-CMAES).

Evaluation of the Baseline.

We also evaluated the influence of the baseline term that we use for contextual CMA-ES weighting. We use the sphere function with 3 dimensional context and 20 parameters. We also use the Rosenbrock with 1 dimensional context and 20 parameters. We generate 30 samples in each iteration. The



(a) Robot Table Tennis (b) Table tennis algorithm comparison (c) Multi-task VS Single-task

Figure 2: (a)The table tennis learning setup. The incoming ball has different initial velocity which is used as context vector. The goal of the robot is to learn forehand strokes to return the ball to a fixed target position.(b) Comparison of contextual algorithms on the table tennis task. Contextual REPS-CMAES achieves slightly better final performance. Please note that y in 10^y is value on y axis. (c) We trained the robot for a single, but hard context when the ball bounces at the middle of the table in the x-axis(red trajectory in Figure (a)) . In this case, the required solution is quite different from the initial solution. Due to the difficulty of the task, the robot could not learn the task and only found a locally optimal solution that hits the ball, but could not place it on the other side of the table. We also trained the robot in contextual setting where the context range includes the desired context but also easier tasks. The results show that the robot can learn even the complex task in this contextual setting as the easier tasks provide a guidance to the correct solution also for the difficult task.

results in Figure 1(c) show that without baseline term, i.e., $V(\boldsymbol{s}) = 0$, contextual CMA-ES can not find a good solution. Therefore, the baseline is a crucial part of the algorithm.

6.2 Robot Table Tennis

In this task, we use a simulated robot arm (see Figure 2(a)) to learn forehand hitting strokes 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. However, the ball is always served differently towards the robot with different initial velocities in the x-direction. We use the initial velocity of the ball as context, i.e., $\boldsymbol{s} = [v_x]$. To learn the task we use a linear policy $m(\boldsymbol{s}) = \boldsymbol{A}\boldsymbol{s} + b$ which is also the mean function of distribution. We initialize b with a initial DMP trajectory obtained by kinesthetic teaching, such that the movement generates a single forehand stroke. Other parameters have the same initialization as we did for standard functions. We only learn the final positions and final velocities of the DMP trajectories as well as the τ time-scaling parameter and the starting time point of the DMP which results in 18 parameters vector $\boldsymbol{\theta}$. 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.

Algorithm Comparison

We compared contextual stochastic search methods on table tennis task. The results in Figure 2(b) shows that both the contextual CMA-ES and contextual REPS-CMA-ES can learn the task.However REPS-CMAES slightly outperforms contextual CMA-ES with the final solution. Contextual REPS again suffers from pre mature convergence. We also see that standard CMA-ES fails to learn this task.

Multi task learning versus Single task learning

In this experiment, we want to show that multi task learning can even facilitate learning of a single task. To do so, we choose a hard context to learn as it is shown in Figure 2(a) with a red trajectory. Here, the ball is served directly towards the robot and it lands close to the border of the table, which requires a quite different movement as the initial solution. We use the standard CMA-ES algorithm to learn this task, but as the results in Figure 2(c) show, the algorithm failed to learn it. However, when we use contextual CMA-ES with a context range that includes this desired context but also simpler tasks, it manages to learn this task 2(c). Hence, the simpler tasks guided the algorithm to find also a good solution for the hard task and avoid the local minimum found by the single task learner.

7 Conclusion and Future Work

Stochastic search methods such as CMA-ES have been employed extensively for black box optimization. However, these algorithms lack the important feature of contextual learning. Therefore we extended CMA-ES for contextual setting while we also provide a new theoretical justification for its covariance update rule. It turns out using baseline, the old covariance matrix and the step size control are crucial ingredients for a competitive performance. One interesting observation is that contextual learning also facilitates learning single tasks. The reason is that easier tasks can guide the optimisation for learning harder tasks. For the future work we will investigate the application of the contextual CMA-ES for full reinforcement learning problems where we need to find the optimal actions for task states.

Acknowledgements

This research was funded by European Union's FP7 under EuRoC grant agreement CP-IP 608849 and LIACC (UID/CEC/00027/2015) and IEETA (UID/CEC/00127/2015) and also partially was funded by PARC.

References

[Abdolmaleki *et al.*, 2015a] A. Abdolmaleki, N. Lua, L.P. Reis, and G. Neumann. Regularized covariance estimation for weighted maximum likelihood policy search methods. In *Proceedings of the International Conference on Humanoid Robots (HUMANOIDS)*, 2015.

[Abdolmaleki *et al.*, 2015b] A. Abdolmaleki, N. Lua, L.P. Reis, J. Peters, and G. Neumann. Contextual Policy Search for Generalizing a Parameterized Biped Walking Controller. In *IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, 2015.

[Abdolmaleki *et al.*, 2016] A. Abdolmaleki, D. Simoes, N. Lua, L.P. Reis, and G. Neumann. Learning a Humanoid Kick With Controlled Distance. In *Robocup Symposium*, 2016.

[Akimoto *et al.*, 2012] Y. Akimoto, Y. Nagata, I. Ono, and S. Kobayashi. Theoretical foundation for cma-es from information geometry perspective. *Algorithmica*, 2012.

[Caruana, 1997] Rich Caruana. *Multitask Learning*. PhD thesis, CMU, 1997.

[Da Silva *et al.*, 2012] Bruno Da Silva, George Konidaris, and Andrew Barto. Learning parameterized skills. *International Conference on Machine Learning (ICML)*, 2012.

[Deisenroth *et al.*, 2014] Marc Peter Deisenroth, Peter Englert, Jan Peters, and Dieter Fox. Multi-task policy search for robotics. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[Ha and Liu, 2016] S. Ha and C.K Liu. Evolutionary optimization for parameterized whole-body dynamic motor skills. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[Hansen *et al.*, 2003] N. Hansen, S.D. Muller, and P. Koumoutsakos. Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA-ES). *Evolutionary Computation*, 2003.

[Hansen, 2016] N. Hansen. The cma evolution strategy: A tutorial. Tutorial, 2016.

[Igel *et al.*, 2006] C. Igel, T. Suttorp, and N. Hansen. 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*, 2006.

[Kober *et al.*, 2010] J. Kober, E. Oztop, and J. Peters. Reinforcement Learning to adjust Robot Movements to New Situations. In *Proceedings of the Robotics: Science and Systems Conference (RSS)*, 2010.

[Kupcsik *et al.*, 2013] A. Kupcsik, M. P. Deisenroth, J. Peters, and G. Neumann. Data-Efficient Contextual Policy Search for Robot Movement Skills. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2013.

[Molga and Smutnicki, 2005] M. Molga and C. Smutnicki. Test Functions for Optimization Needs. In *http://www.zsd.ict.pwr.wroc.pl/files/docs/functions.pdf*, 2005.

[Peters *et al.*, 2010] J. Peters, K. Mülling, and Y. Altun. Relative Entropy Policy Search. In *Proceedings of the 24th National Conference on Artificial Intelligence (AAAI)*. AAAI Press, 2010.

[Rückstieß *et al.*, 2008] T. Rückstieß, M. Felder, and J. Schmidhuber. State-dependent Exploration for Policy Gradient Methods. In *Proceedings of the European Conference on Machine Learning (ECML)*, 2008.

[Stulp and Sigaud, 2012] F. Stulp and O. Sigaud. Path Integral Policy Improvement with Covariance Matrix Adaptation. In *International Conference on Machine Learning (ICML)*, 2012.

[Stulp *et al.*, 2013] Freek Stulp, Gennaro Raiola, Antoine Hoarau, Serena Ivaldi, and Olivier Sigaud. Learning compact parameterized skills with a single regression. In *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2013.

[Sun *et al.*, 2009] Y. Sun, D. Wierstra, T. Schaul, and J. Schmidhuber. Efficient Natural Evolution Strategies. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation(GECCO)*, 2009.