Regularized Covariance Estimation for Weighted Maximum Likelihood Policy Search Methods

Abbas Abdolmaleki[1,2,3], Nuno Lau[1], Luis Paulo Reis[2,3], Gerhard Neumann[4]

***Abstract*— Many episode-based (or direct) policy search algorithms, maintain a multivariate Gaussian distribution as search distribution over the parameter space of some objective function. One class of algorithms, such as episodic REPS, PoWER or PI^2 uses, a weighted maximum likelihood estimate (WMLE) to update the mean and covariance matrix of this distribution in each iteration. However, due to high dimensionality of covariance matrices and limited number of samples, the WMLE is an unreliable estimator. The use of WMLE leads to over-fitted covariance estimates, and, hence the variance/entropy of the search distribution decreases too quickly, which may cause premature convergence. In order to alleviate this problem, the estimated covariance matrix can be regularized in different ways, for example by using a convex combination of the diagonal covariance estimate and the sample covariance estimate. In this paper, we propose a new covariance matrix regularization technique for policy search methods that uses the convex combination of the sample covariance matrix and the old covariance matrix used in last iteration. The combination weighting is determined by specifying the desired entropy of the new search distribution. With this mechanism, the entropy of the search distribution can be gradually decreased without damage from the maximum likelihood estimate.**

I. INTRODUCTION

Stochastic search algorithms are gradient-free black-box optimizers of some objective function $R_{\boldsymbol{\theta}}$ dependent on a high-dimensional parameter vector $\boldsymbol{\theta}$. Stochastic search algorithms do not put any assumption on the structure of the objective function, such as a Markov assumption. In this paper, we focus on episode-based policy search methods in robotics which are a special case of stochastic search methods. Due to its simplicity, episode based policy search is one of the most successful reinforcement learning approaches in robotics [1], [2], [3], [4]. Episode-based policy search methods address the continuous state-action problems in reinforcement learning by directly optimizing the parameters $\boldsymbol{\theta}$ of a control lower-level policy. Fourier series, splines and DMPs[5] has been commonly used as control lower-level policy in robotics. Policy search methods, directly search over the parameter space of the lower-level policy using an upper-level policy or search distribution which is typically implemented as a multivariate Gaussian distribution. Many state of art methods such as episodic REPS [4], CMA-PI^2 [3] and PoWER [6] estimate the Gaussian upper level policy (mean and covariance matrix) by a weighted maximum likelihood estimate (WMLE), see Equation 4. To do so, they generate samples from the current upper-level policy and use the return of the samples to estimate the quality of the samples. This quality estimate results in a weight for each sample that can be used to estimate a new mean and a new covariance matrix for the new Gaussian upper-level policy by using a WMLE. Yet, due to high dimensionality of a covariance matrix and limited number of samples, the WMLE estimate of the covariance matrix is an unreliable estimator with high variance. This over-fitted estimation of the covariance makes the upper-level policy highly biased to a specific region of the parameter space, which often causes premature convergence [7]. Instead, we can estimate only a diagonal covariance matrix with fewer parameters [8], yet, such a solution has a high bias and might result in a slow learning performance as we neglect the correlations between the parameters. One other solution is using regularization techniques for estimating the covariance matrix. Standard regularization techniques such as covariance shrinkage [9], [7] are based on a convex combination of different estimators, e.g., the high variance estimator of the sample covariance matrix and the high bias estimator of the diagonal covariance matrix. Yet, policy search algorithms have a big advantage when estimating the covariance matrix. They have access to the covariance with which the (unweighted) samples have been generated. Therefore we propose a new regularization technique that combines the sample covariance estimate with the covariance matrix of the generating distribution, i.e., the old upper-level policy, can be used as a prior in our estimation. Furthermore, we know that controlling the exploration rate in policy search is crucial. The variance/entropy of the upper-level policy should decrease slowly in order to give the algorithm enough time to converge to a (local) optimal solution. Hence, the combination factor of the prior (old covariance) and the sample covariance can be determined by an entropy reduction criterion. At each iteration, we want the entropy of the upper-level policy to decrease for a certain amount. We chose the combination factor between the two matrices such that the entropy of the resulting distribution is exactly at this desired level. We name our method Covariance Estimation with Controlled Entropy Reduction (CECER). Intuitively, our method can be seen as weighted averaging of covariance matrix estimates of all iterations, where the influence of the initial distribution is decreased at each iteration. Similar combinations of old and new covariance matrices have been used by other stochastic search algorithms such as CMA-ES [10]. We compare different covariance estimation techniques

[1]DETI/IEETA, University of Aveiro, Aveiro, Portugal
[2]LIACC, University of Porto, Porto, Portugal
[3]DSI, University of Minho, Braga, Portugal
[4]CLAS, IAS, TU Darmstadt, Darmstadt, Germany
{abbas.a,nunolau}@ua.pt, lpreis@dsi.uminho.pt, geri@robot-learning.de

including covariance shrinkage [9] to our new regularization technique based on entropy reduction in context of state of art episode-based policy search methods such as REPS [4] and an episode-based version of PI^2[3][1]. The resulting episode-based policy search algorithms are also compared to the Natural Evolution Strategy [11] and CMA-ES [10] on two simulated robotics tasks including a planar arm reaching task and a planar arm hole reaching task. Our algorithm performs favorably in our experiments.

II. Weighted Maximum Likelihood based policy search

We want to maximize an objective function $R(\boldsymbol{\theta})$,

$$R : \mathbb{R}^n \to \mathbb{R} \quad , \boldsymbol{\theta} \mapsto R(\boldsymbol{\theta}).$$

The goal is to find one or more parameter vectors, $\boldsymbol{\theta} \in \mathbb{R}^n$, with an objective value, $R(\boldsymbol{\theta})$, as big as possible. The only accessible information on $R(\boldsymbol{\theta})$, are function values $\{R^{[k]}\}_{k=1...N}$ of evaluated parameter vectors $\{\boldsymbol{\theta}^{[k]}\}_{k=1...N}$, where k is the index of the sample and N is number of samples. Episode-based policy search algorithms [1], [2], [3] typically maintain an upper-level policy or search distribution $\boldsymbol{\pi(\theta)}$, over the parameter space $\boldsymbol{\theta}$ of a parametrized lower-level policy. Typically, the upper-level policy $\pi(\boldsymbol{\theta})$ is implemented as a multivariate Gaussian distribution, i.e., $\pi(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$. In each iteration, the upper-level policy $\boldsymbol{\pi(\theta)}$ is used to create samples $\boldsymbol{\theta}^{[k]}$ of the parameter vector $\boldsymbol{\theta}$ of the lower-level policy. Subsequently, the return $R^{[k]}$ of $\boldsymbol{\theta}^{[k]}$ is obtained by evaluating the performance of the lower-level policy with the parameter vector $\boldsymbol{\theta}^{[k]}$. Using the samples and their returns $\{\boldsymbol{\theta}^{[k]}, R^{[k]}\}_{k=1...N}$, a new upper-level policy is computed [2] by either computing gradient based updates [2], [12], covariance matrix adaptation updates [10] or weighted MLE-based updates [3], [13], [4], [14]. We are particularly interested in weighted MLE-based policy search methods which have been shown to be able to outperform gradient-based methods such as Natural Actor-Critic [15]. WMLE-based policy search methods use the return $R^{[k]}$ to compute a weight $w^{[k]}$ for each sample $\boldsymbol{\theta}^{[k]}$ such that $\sum_{k=1}^{N} w^{[k]} = 1$ [3] and, subsequently, the mean and covariance matrix of the upper-level policy $\boldsymbol{\pi(\theta)}$ is updated by a weighted MLE (Equation 4). The next we will explain how the weights are computed.

A. Computation of the Weighting

The weight $w^{[k]}$ for sample $\boldsymbol{\theta}^{[k]}$ can be estimated by an exponential transformation of the corresponding return $R^{[k]}$, i.e.,

$$w^{[k]} \propto \exp(R^{[k]}/\eta), \tag{1}$$

where η specifies the temperature of the exponential transformation, such as applied by the PI^2 algorithm [14], [3], PoWER [13] and REPS [4]. The next we will explain how different algorithms set the η.

[1]In the episode-based case, PoWER [6] and PI^2 [3] are equivalent.

[2]The goal is that, the new upper-level policy or new search distribution spans samples with higher returns than the old upper-level policy

[3]Each weight is a pseudo-probability for the corresponding sample

a) PoWER and PI^2: In the PI^2 and PoWER algorithms, the temperature parameter η is chosen by a heuristic. PI^2 chooses

$$\eta = \lambda(\max_k R^{[k]} - \min_k R^{[k]}),$$

where $R^{[k]}$ is the return of sample $\boldsymbol{\theta}^{[k]}$ and λ is typically set between 5 and 15. For PoWER, η is often hand tuned. While PoWER and PI^2 are actually equivalent if the same strategy for η is used[4], both algorithms are derived from very different principles.

b) REPS: REPS [16], [4] bounds the Kullback-Leibler divergence between the old policy $q(\boldsymbol{\theta})$ used for sampling and the newly estimated policy $\pi(\boldsymbol{\theta})$. The policy update can hence be formulated as constrained optimization problem where we want to maximize the expected return of the new policy under the KL constraint, i.e.,

$$\pi^* = \text{argmax}_\pi \int \pi(\boldsymbol{\theta}) R(\boldsymbol{\theta}) d\boldsymbol{\theta} \tag{2}$$

$$\text{s.t. KL}(\pi(\boldsymbol{\theta})||q(\boldsymbol{\theta})) \leq \epsilon, \qquad \int \pi(\boldsymbol{\theta}) d\boldsymbol{\theta} = 1$$

The main intuition behind this bound is that we can directly control the exploration-exploitation trade-off with the ϵ parameter. For a large ϵ (exploitation), the entropy/variance of the new upper level policy will shrink quickly such that, it will always choose the sample with highest return in our dataset while for a small ϵ (exploration), the new search policy and the old search policy would be almost identical. While this optimization problem can not be solved analytically as $R_{\boldsymbol{\theta}}$ is unknown, it can be solved for our samples $\{\boldsymbol{\theta}^{[k]}, R^{[k]}\}_{k=1...N}$. The solution for the sample based problem results in a weight $w^{[k]} \propto \exp(R^{[k]}/\eta)$ for each sample, where the temperature parameter η can be found by optimizing the dual function

$$g(\eta) = \eta\epsilon + \eta \log\left(\sum_{k=1}^{N} \frac{1}{N} \exp\left(\frac{R^{[k]}}{\eta}\right)\right) \tag{3}$$

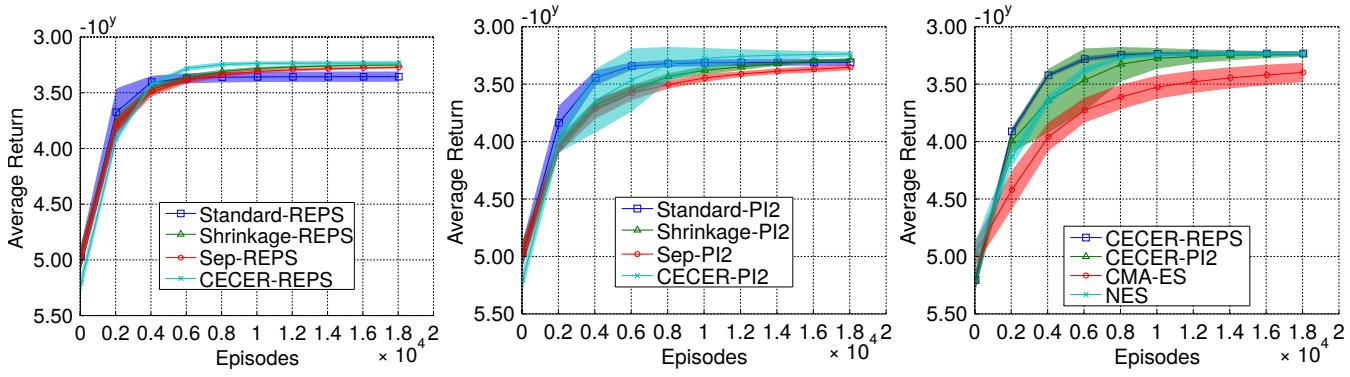
of the optimization problem. The optimal value for η can be obtained by minimizing the dual function $g(\eta)$ such that $\eta > 0$, see [4], [17]. The next, we will explain how the weightings can be used to update the upper-level policy.

B. Weighted ML Policy Updates

In each iteration, after computing the weightings $w_{k=1...N}^{[k]}$, the new upper level policy is computed by using the samples and their weightings $\{\boldsymbol{\theta}^{[k]}, w^{[k]}\}_{k=1...N}$. PI^2 , PoWER and REPS directly use an unbiased weighted maximum likelihood estimate [1] for estimating $\boldsymbol{\mu}$ and the sample covariance $\boldsymbol{S}$ of a Gaussian upper level policy which is given by

$$\boldsymbol{\mu} = \sum_{i=1}^{N} w^{[k]} \boldsymbol{\theta}^{[k]}, \quad \boldsymbol{S} = \frac{\sum_{i=1}^{N} w^{[k]} (\boldsymbol{\theta}^{[k]} - \boldsymbol{\mu})^T (\boldsymbol{\theta}^{[k]} - \boldsymbol{\mu})}{1 - \sum_{i=1}^{N} \left(w^{[k]}\right)^2}. \tag{4}$$

[4]This is true at least for the episode-based version that neglects the time steps.



(a) REPS (b) PI^2 (c) Stochastic Search Algorithms

Fig. 1. The performance comparison for reaching task using a 5 link planar robot. The results show that CECER outperforms the other covariance update methods for the both REPS and PI^2. In (c) we see that, CECER-REPS has faster learning rate than the other algorithms.

Here, the sample covariance matrix of a p dimensional parameter space has $n = \frac{p+p^2}{2}$ free parameters to estimate. Typically, the number of samples used for the estimate is much smaller than this number of free parameters. In this case, it has been shown that the sample covariance matrix from Equation 4 is not a good estimate of the true covariance matrix [9] and biases the search distribution towards a specific region of the search space. Due to this effect, the search distribution uncontrollably looses its exploration/entropy along many dimensions of the parameter space and will therefore causes premature convergence. That is a highly unwanted effect in policy search. Alternatively, instead of estimating a full covariance matrix, we could estimate a diagonal covariance matrix which has fewer parameters to estimate and, hence, will not suffer so severely from over-fitting. However, using a diagonal covariance matrix neglects the correlations between the parameters, which might again lead to a slow learning progress [18], [8].

III. Covariances Regularization for ML based Policy Search

One way to achieve a more accurate covariance estimate is to use regularization techniques that combine the sample estimate of the covariance matrix with a target estimate of the covariance matrix [9]. Different target covariance estimate can be used such as the diagonal covariance matrix or even an identity matrix that is multiplied with some factor [9]. In policy search, we also have the possibility to use the old covariance matrix as target covariance estimate, as we know that the unweighted samples have been generated using it. There are different ways to determine the interpolation factor between the sample covariance and the target covariance estimate. We will discuss first a standard algorithm for determining this interpolation factor and subsequently present our new method based on a controlled reduction of the entropy of the resulting policy.

A. Combining Diagonal and Full Covariance Matrix Estimates by Covariance Shrinkage

In covariance shrinkage estimation [9], we shrink a high-dimensional estimated covariance $\boldsymbol{S}$ towards a lower-dimensional covariance $\boldsymbol{G}$ with fewer parameters (e.g. diagonal matrix) by a weighted average, i.e.,

$$\boldsymbol{\Sigma} = \lambda \boldsymbol{G} + (1-\lambda)\boldsymbol{S} \tag{5}$$

where $\lambda \in [0,1]$ is the shrinkage intensity. It has been shown In [9], that the combination of covariance estimators with high bias(e.g. diagonal covariance) and high variance (sample covariance) in Equation 5 gives us a regularized estimate that outperforms each of those two estimators in terms of estimation error. In the case of our policy update, the matrix $\boldsymbol{G}$ is a diagonal covariance matrix(with p parameters) and $\boldsymbol{S}$ is a sample covariance matrix (with $\frac{p+p^2}{2}$ parameters) estimated by the samples $\{\boldsymbol{\theta}^{[k]}, w^{[k]}\}_{k=1\ldots N}$. Intuitively, in this method, we want to shrink the overestimated correlations between parameters in matrix $\boldsymbol{S}$ towards zero to get a better conditioned covariance matrix. And the diagonal elements will stay unchanged. To do so, we parametrize our desired covariance matrix for the policy update in terms of variances and correlations, i.e.,

$$\Sigma_{ij} = \begin{cases} S_{ij} & \text{if } i = j, \\ R^*_{ij}\sqrt{S_{ii}S_{jj}} & \text{if } i \neq j, \end{cases} \tag{6}$$

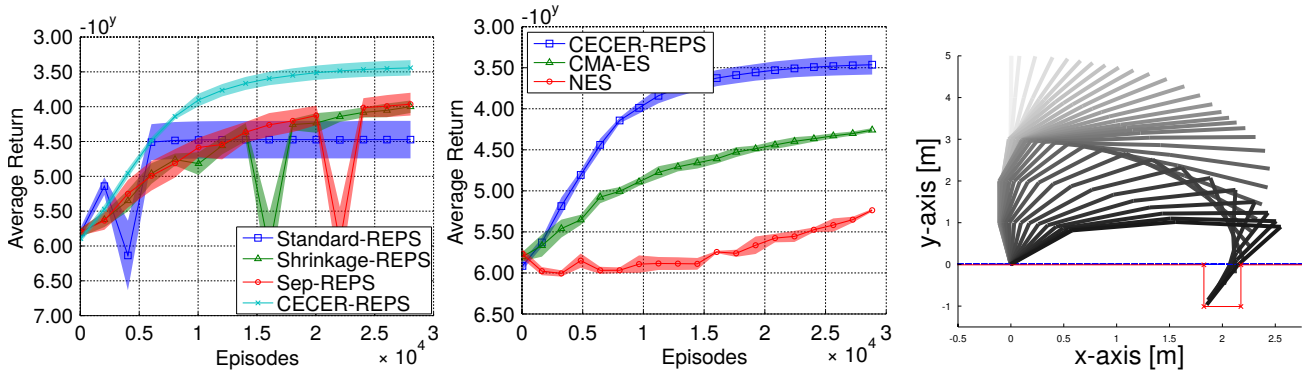
where R^*_{ij} is the element of the shrunk correlation matrix, i.e,

$$R^*_{ij} = \begin{cases} 1 & \text{if } i = j, \\ R_{ij}\min(1, \max(\sigma, 1-\lambda^*)) & \text{if } i \neq j. \end{cases} \tag{7}$$

where λ^* is the optimum shrinkage intensity. The min-max term in Equation 7 is used for limiting λ^* between 0 and $1-\sigma$. Typically, $\sigma = 0$ is used. Yet, we empirically found that policy search algorithms performed slightly better if we set σ to

$$\sigma = \min\left(\frac{\phi_{\text{eff}}}{p^2}, 1\right), \quad \phi_{\text{eff}} = \frac{1}{\sum_{k=1}^{N}(w^{[k]})^2}, \tag{8}$$

where ϕ_{eff} is the number of effective samples which is computed as in [10] and p is the number of dimensions of the parameter vector $\boldsymbol{\theta}$. The reason is that, covariance matrices that need to be estimated for our policy search methods are



(a) REPS (b) Stochastic Search Algorithms (c) Planar Reaching Task

Fig. 2. The performance comparison for high dimensional reaching task using a 20-link planar robot. The results show that CECER clearly outperforms the other covariance update methods for REPS policy update and CECER-REPS has better performance than NES and CMA-ES (c) The planar hole reaching task used for our comparisons. 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. 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 the via-point.

high dimensional considering the small amount of data that we want to use. As a consequence, matrix shrinkage algorithms will, in many cases, just decide to take the estimator with less variance (which is the diagonal covariance matrix) with a factor of 100%. With this rule we force the shrinkage algorithm to always take a small part from the full sample covariance matrix therefore the algorithm always exploits the correlations between parameters. Typically, σ has a very small value close to 0. Next we will explain how the value of λ^* is computed.

Computing the Shrinkage Intensity: We can find the optimum λ^* efficiently in closed form using the method given in [9]. This results in an optimal lambda value of

$$\lambda^* = \frac{\sum_{i \neq j} \widehat{\text{Var}}(R_{ij})}{\sum_{i \neq j} R_{ij}^2}. \tag{9}$$

The term $\widehat{\text{Var}}(R_{ij})$ denotes the variance of the elements of the matrix $\boldsymbol{R}$ which can be estimated from the samples and their weightings $\{\boldsymbol{\theta}^{[k]}, w^{[k]}\}_{k=1\ldots N}$ by

$$\widehat{\text{Var}}(R_{ij}) = \frac{\sum_{k=1}^N (w^{[k]})^2}{(1 - \sum_{k=1}^N (w^{[k]})^2)^3} \sum_{k=1}^N w^{[k]} (C_{ij}^{[k]} - \overline{C}_{ij})^2, \tag{10}$$

$$C_{ij}^{[k]} = \frac{(\theta_i^{[k]} - \mu_i)(\theta_j^{[k]} - \mu_j)}{\sqrt{S_{ii} S_{jj}}}, \quad \overline{C}_{ij} = \sum_{k=1}^N w^{[k]} C_{ij}^{[k]},$$

where $\mu_i = \sum_{k=1}^N w^{[k]} \theta_i^{[k]}$ is the mean of ith element of the parameter vector $\boldsymbol{\theta}$. For more details how to compute $\widehat{\text{Var}}(R_{ij})$ from samples, we refer to the appendix of [9].

B. Covariance Estimation with Controlled the Entropy Reduction

While the covariance shrinkage can already improve the performance of weighted ML algorithms, it still did not lead to fully satisfying results. Yet, in policy search, we can use more information as in standard density estimation. First, we know a good prior upper level policy from which the unweighted samples have been generated from. Moreover, we know that the policy update should not reduce the entropy of the new upper level policy too quickly which leads to premature convergence. In our new algorithm, Covariance Estimation with Controlled Entropy Reduction (CECER), we combine the sample estimate of the covariance matrix $\boldsymbol{S}$ with the old covariance matrix $\boldsymbol{\Sigma}_q$ that has been used to generate the data, i.e.,

$$\boldsymbol{\Sigma} = \lambda \boldsymbol{\Sigma}_q + (1 - \lambda) \boldsymbol{S}.$$

The factor $\lambda \in [0, 1]$ is chosen in such a way that the entropy of the new upper level policy is reduced by a certain amount ΔH. The entropy of a Gaussian distribution only depends on its covariance $\boldsymbol{\Sigma}$ and is given by

$$H(\boldsymbol{\Sigma}) = 0.5(p + p \log(2\pi) + |\boldsymbol{\Sigma}|).$$

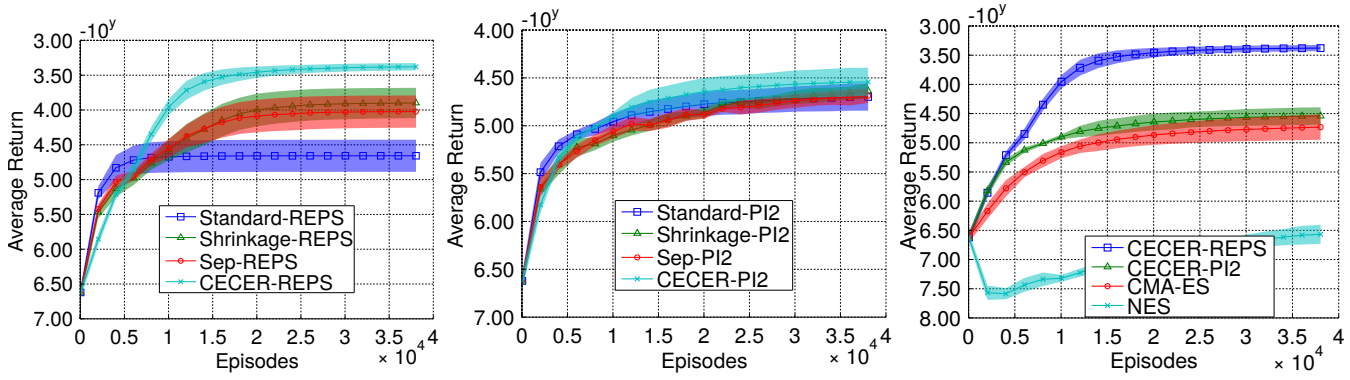
Where p is the dimension of the parameter space $\boldsymbol{\theta}$ and $|.|$ is the determinant operator. We choose λ such that we achieve a desired entropy reduction, i.e.,

$$H(\boldsymbol{\Sigma}_q) - H(\lambda \boldsymbol{\Sigma}_q + (1 - \lambda) \boldsymbol{S}) = \Delta H.$$

We scale $\Delta H = \alpha \phi_{\text{eff}}$ with the number of effective samples ϕ_{eff} that have been used to compute the sample covariance $\boldsymbol{S}$, i.e., if more samples are available for the sample estimate, the entropy reduction can be higher. A higher entropy reduction leads to a smaller λ value as we can rely more on our sample estimate. In order to find the correct λ value, we applied a simple exhaustive search and we could always find the λ with correct entropy reduction Algorithm. Algorithm 1 shows the Covariance Estimation with Controlled the Entropy Reduction (CECER) .

IV. Experiments

We use the full covariance, the diagonal covariance and the covariance shrinkage algorithm and compare it to the CECER algorithm. The comparisons are done for the policy



(a) REPS (b) PI^2 (c) Stochastic Search Algorithms

Fig. 3. The performance comparison for hole reaching task using a 5-link planar robot. The results show that CECER clearly outperforms the other covariance update methods for REPS and PI^2. Moreover CECER-REPS clearly outperforms the other stochastic search algorithms.

Algorithm 1 Covariance Estimation with Controlled the Entropy Reduction

Input : Data Set $\mathcal{D}\{\theta^{[k]}, w^{[k]}\}_{k=1\ldots N}$, the old covariance matrix $\boldsymbol{\Sigma}_q$ and the scaling factor α for entropy reduction

Compute the sample covariance $\boldsymbol{S}$:

$$\boldsymbol{\mu} = \sum_{i=1}^{N} w^{[k]}\boldsymbol{\theta}^{[k]}, \quad \boldsymbol{S} = \frac{\sum_{i=1}^{N} w^{[k]}(\boldsymbol{\theta}^{[k]} - \boldsymbol{\mu})^T(\boldsymbol{\theta}^{[k]} - \boldsymbol{\mu})}{1 - \sum_{i=1}^{N}\left(w^{[k]}\right)^2}.$$

Compute the number of effective samples ϕ_{eff} and the entropy reduction ΔH:

$$\phi_{\text{eff}} = \frac{1}{\sum_{k=1}^{N}(w^{[k]})^2}, \quad \Delta H = \alpha\phi_{\text{eff}}.$$

Choose the λ such that following equality is satisfied

$$H(\boldsymbol{\Sigma}_q) - H(\lambda\boldsymbol{\Sigma}_q + (1-\lambda)\boldsymbol{S}) = \Delta H.$$

Compute the new covariance matrix $\boldsymbol{\Sigma}$:

$$\boldsymbol{\Sigma} = \lambda\boldsymbol{\Sigma}_q + (1-\lambda)\boldsymbol{S}.$$

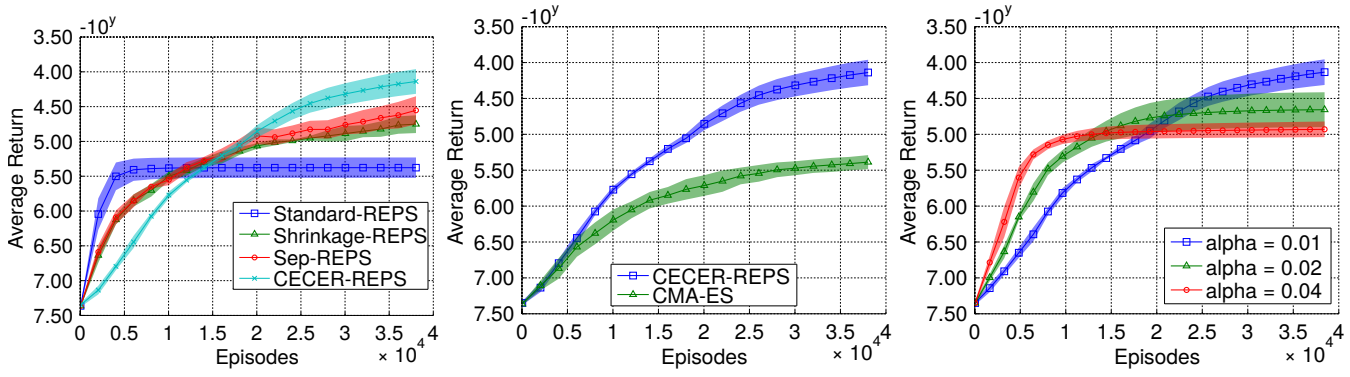
updates of REPS [4] and PI^2 [3][5]. Similar to Sep-CMA-ES [8], we call the algorithms with the diagonal matrix estimate, Sep-REPS and Sep-PI^2. We call the algorithms with shrinkage update, shrinkage-REPS and shrinkage-PI^2 respectively. CECER-REPS and CECER-PI^2 use CECER for policy update. We also compare these algorithms to other state of the art methods in stochastic search such as CMA-ES [10] and NES [2]. For our comparisons, we used a multi-link planar robot with DMPs [5] as underlying lower level control policy. Each link had a length of 1m. We used 5 basis functions per degree of freedom for the DMPs. We use a 5-link planar robot that has to reach a given point in task space. We call this task *reaching task*. The resulting lower level policy has 25 parameters, but we also test the algorithms in high-dimensional parameter spaces by scaling up the robot to 20 links (100 parameters). This task has a relatively smooth reward function and is therefore easy to learn. We make the task more difficult by introducing hard obstacles. We use the same planar robot to reach in a given hole on the ground, see Figure 2(c). Whenever the robot touches the ground with one of its links, a large penalty is added to the reward. Due to this discontinuity in the objective function, the task is much harder to learn. We call this task *hole reaching task*. For the hole reaching task, we used a 5-link and 15-link version of the robot, resulting in 30 parameters and 90 parameters lower-level policies to optimise. We compared the REPS and PI^2 algorithms with different policy updates individually and compared the best variant against CMA-ES and NES. In each iteration, we generated 40 new samples. For REPS and PI^2 we always keep the last $L = 400$ samples, while for NES and CMA-ES 40 current samples are kept[6]. We show the average as well as the variance of the results over 10 trials for each experiment.

A. 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 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 $\boldsymbol{\theta}$ for a 5-link robot with 5 basis function for each degree of freedom had 25 dimensions. The results in Figure 1 show that CECER outperforms the other covariance estimation methods where CECER-REPS reach the average reward -1714 and shrinkage-REPS achieve an average reward of -2000. CECER-REPS has a better learning rate compare to the other methods. Yet, all the algorithms perform good in this task due to simplicity of the task. We also evaluated the same task with a 20-link planar robot, resulting in a 100 dimensional parameter space. The results in Figure 2 show

[5]The full covariance matrix update is the standard policy update method for episode version of REPS and PI^2

[6]NES and CMA-ES algorithms typically only use the new samples and discard the old samples. We also tried keeping old samples which didn't lead to a better performance.



(a) REPS (b) Stochastic Search Algorithms (c) Evaluation of α

Fig. 4. The performance comparison for high dimensional hole reaching task using a 15-link planar robot. The results show that CECER clearly outperforms the other covariance update methods for REPS policy update and CECER-REPS outperforms the CMA-ES (c) It shows the performance of the CECER-REPS for three different entropy reduction scale factor α. The bigger α results in more entropy reduction of the covariance matrix.

that CECER and CECER-REPS clearly outperform the other methods.

B. Planar Hole Reaching Task

For completing the hole reaching task the robot end effector has to reach the bottom of a hole (35 cm wide and 1m deep) centred point $[2, 0]$ without any collision with the ground or the hole wall. The reward was given by a quadratic cost term for the desired final point, quadratic costs for high accelerations and quadratic costs for collisions with the environment. Note that this objective function is discontinuous due to the quadratic costs for collisions . The goal attractor for reaching the final state in this task is unknown and 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. The setup, including the learned policy is shown in Figure 2(c). The results in Figure 3 show that CECER has the best performance with significant difference. Covariance shrinkage performed the second best among all covariance estimation methods. We also see that CECER-REPS considerably outperforms the other methods. We also evaluated the same task with a 15-link planar robot, resulting in a 90 dimensional parameter space. The results in Figure 4 show that CECER and CECER-REPS clearly outperform the other methods with significant difference in performance. Using this task, we also evaluate the performance of CECER-REPS for three different α (Figure 4(c)). It turns out with bigger α the search distribution shrinks faster, resulting in premature convergence. For large α values, the algorithm will only use the full sample covariance matrix, and thus, perform like the standard REPS algorithm with full covariance estimation.

V. Conclusion

In this paper, we compared different methods for estimating the covariance matrix of a Gaussian policy for weighted ML based policy search methods. Weighted ML estimate of covariance matrices is an unreliable estimator with a high variance. The use of WMLE leads to over-fitted covariance estimates, and, hence the variance/entropy of the policy decreases too quickly, which may cause premature convergence. We proposed a new algorithm called Covariance Estimation with Controlled the Entropy Reduction. We showed that using the CECER, we could control the entropy reduction of the policy and get a better covariance matrix approximation, which results in an significant improved performance of the policy search algorithm.

Acknowledgment

The first author is supported by FCT under grant SFRH/BD/81155/2011. This project has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No #645582 (RoMaNS).

References

[1] M. P. Deisenroth, G. Neumann, and J. Peters. A survey on policy search for robotics. 2013.

[2] 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.

[3] F. Stulp and O. Sigaud. Path integral policy improvement with covariance matrix adaptation. In *International Conference on Machine Learning (ICML)*, 2012.

[4] 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.

[5] A. Ijspeert and S. Schaal. Learning Attractor Landscapes for Learning Motor Primitives. In *Advances in Neural Information Processing Systems 15(NIPS)*. 2003.

[6] J. Kober and J. Peters. Policy Search for Motor Primitives in Robotics. *Machine Learning*, pages 1–33, 2010.

[7] H. Karshenas, R. Santana, C. Bielza, and P. Larraaga. Regularized continuous estimation of distribution algorithms. *Applied Soft Computing*, 2013.

[8] R. Ros and N. Hansen. A simple modification in CMA-ES achieving linear time and space complexity. In *Parallel Problem Solving from Nature*, pages 296–305, 2008.

[9] J. Schfer and K. Strimmer. A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statistical applications in genetics and molecular biology*, 2005.

[10] 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.

[11] D. Wierstra, T. Schaul, Glasmachers T., Y. Sun, J. Peters, and J. Schmidhuber. Natural evolution strategies. 2014.

[12] 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.
[13] J. Kober and J. Peters. Policy search for motor primitives in robotics. In *Neural Information Processing Systems (NIPS)*, 2009.
[14] E. Theodorou, J. Buchli, and S. Schaal. A generalized path integral control approach to reinforcement learning. *The Journal of Machine Learning Research*, 2010.
[15] J. Peters and S. Schaal. Natural Actor-Critic. *Neurocomputation*, 71(7-9):1180–1190, 2008.
[16] 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.
[17] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.
[18] C. Daniel, G. Neumann, and J. Peters. Hierarchical Relative Entropy Policy Search. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2012.