Evolution Strategies for Direct Policy Search

Verena Heidrich-Meisner and Christian Igel

Institut für Neuroinformatik, Ruhr-Universität Bochum, Germany
{Verena.Heidrich-Meisner,Christian.Igel}@neuroinformatik.rub.de

Abstract. The covariance matrix adaptation evolution strategy (CMA-ES) is suggested for solving problems described by Markov decision processes. The algorithm is compared with a state-of-the-art policy gradient method and stochastic search on the double cart-pole balancing task using linear policies. The CMA-ES proves to be much more robust than the gradient-based approach in this scenario.

1 Introduction

Reinforcement learning (RL) aims at maximizing accumulated reward over time by improving a behavioral policy mapping states to actions. The learning is based on interaction with the environment, where perceived transitions between states and scalar reward signals, which may be sparse, noisy, and/or delayed, drive the adaptation [1–3]. Various evolutionary algorithms (EAs) have been successfully applied to RL problems (see, e.g., [4–7]) and performed well in comparison with alternative approaches (see [8, 9] for recent studies). Still, EAs are often met with scepticism from the RL community. The main argument is that a general purpose optimization technique such as an EA, even if slightly tailored to the learning problem, is not likely to compete with highly specialized methods developed solely for canonical RL scenarios. Strong empirical evidence for the power of evolutionary RL and convincing arguments why certain EAs are particularly well suited for certain RL problem classes are needed to dispel this concern, and this study is a further step in this direction.

Unfortunately, it is not easy to conduct a fair comparison of evolutionary and standard RL techniques. Of course, they can be applied to the same benchmark problems, as for example done in [8, 9]. However, usually the search spaces are different, which can introduce a strong bias in the comparison. Many RL algorithms are value function approaches, which learn a function that predicts the expected future reward given a state or an action in a particular state. The policy is then defined on top of this value function [1–3]. In contrast, EAs are typically used for direct policy search, that is, they directly optimize a mapping between states and actions. Thus, it is insightful to compare EAs with other methods searching directly in policy space such as policy gradient methods (PGMs), which are well established in the RL community.

We propose variable metric evolution strategies (ESs) for RL [10–13]. Evolution strategies are per se powerful direct search methods [14]. They usually

outperform gradient-based approaches in the presence of noise and on multimodal objective functions (especially if the local optima have only small basins of attraction). We argue that this makes them particularly well-suited for RL. In RL, noise arises from several sources. The state-transitions and the reward signals may be stochastic. Further, the state observations may be noisy. In addition, the initial state usually varies. This makes it necessary to approximate the quality of a behavioral policy based on a finite number of episodes (or roll-outs). That is, the quality of a policy is a random variable. Evolution strategies adapt the policy as well as parameters of their search strategy (such as the variable metric) based on ranking policies, which is much less error prone than estimating absolute performances or performance gradients [13]. But also for deterministic tasks, evolutionary RL can be advantageous. As we will illustrate in this paper for a non-noisy task, benchmarks problems typically used in RL can be multimodal and are therefore difficult for purely gradient-based methods.

In order to demonstrate the performance of ESs for RL, we compare the covariance matrix adaptation ES (CMA-ES, [15, 16]) with random search and a PGM, where we try to make the comparison as fair as possible. Here we consider the natural actor-critic (NAC, [17–19]) algorithm, which is an established, state-of-the-art method and our favorite PGM. The NAC is a powerful algorithm for fine-tuning policies and it is arguably one of the best developed and most elaborated PGMs. It is well-suited for comparison with the CMA-ES, because the two algorithms have some conceptual similarities as discussed in [12, 13]. In [12, 13] NAC and CMA-ES were compared on simple RL problems where the policies had only very few parameters. It is an open question how these results scale with problem dimensionality and difficulty. In this study, we therefore consider convergence speed and success rate on a more difficult variant of the pole balancing problem and take a look at the fitness landscape near optimal solutions. In contrast to EAs, PGMs need a differentiable structure on the space of candidate policies. Here, we consider simple linear policies, which are often used in combination with the NAC.

In the next section, the basic formalism of RL is introduced before we briefly describe the NAC, random weight guessing, and our approach of using the CMA-ES for RL. After that, we describe our experiments in Section 3. Then the results are presented and discussed.

2 Algorithms for Adapting Policy Parameters

Markov decision processes (MDP) are the basic formalism to describe RL problems. An MDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R} \rangle$ consists of the set of states $\mathcal{S}$, the possible actions $\mathcal{A}$, and for all $a \in \mathcal{A}$ and $s, s' \in \mathcal{S}$ the probabilities $\mathcal{P}^a_{s,s'}$ that action a taken in state s leads to state s' and the expected rewards $\mathcal{R}^a_{s,s'}$ received when going from state s to s' after performing action a. We consider agents interacting with the environment on a discrete time scale. The agent follows its actions according to a behavioral policy $\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, where $\pi(s, a)$ is the probability to choose action a in state s (for deterministic policies we write $\pi : \mathcal{S} \to \mathcal{A}$). The

goal of RL is to find a policy π such that some notion of expected future reward $\rho(\pi)$ is maximized. For example, for episodic tasks we can define $\rho(\pi) = \sum_{s,s' \in \mathcal{S}, a \in \mathcal{A}} d^\pi(s)\pi(s,a)\mathcal{P}^a_{s,s'}\mathcal{R}^a_{s,s'}$, where $d^\pi(s) = \sum_{t=0}^{\infty} \gamma^t \Pr\{s_t = s \,|\, s_0, \pi\}$ is the stationary state distribution, which we assume to exist, and s_t is the state in time step t and $\gamma \in]0,1]$ a discount parameter.

In the following, we briefly describe the RL algorithms compared in this study.

2.1 Natural Policy Gradient Ascent

In this section, we introduce the NAC algorithm according to [19]. Policy gradient methods operate on a predefined class of stochastic policies. They require a differentiable structure to ensure the existence of the gradient of the performance measure and ascent this gradient. Let the performance $\rho(\pi)$ of the current policy with parameters $\boldsymbol{\theta}$ be defined as above. Because in general neither d^π, $\mathcal{R}$, nor $\mathcal{P}$ are known, the performance gradient $\boldsymbol{\nabla}_{\boldsymbol{\theta}}\rho(\pi)$ with respect to the policy parameters $\boldsymbol{\theta}$ is estimated from interaction with the environment.

The policy gradient theorem [20] ensures that an unbiased estimate of the performance gradient can be determined from unbiased estimates of the state-action value function $Q^\pi(s,a) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} | \pi, s_0 = s, a_0 = a\right]$ (where $r_{t+1} \in \mathbb{R}$ is the reward received after the action in time step t) and the stationary distribution. For any MDP it holds

$$\boldsymbol{\nabla}_{\boldsymbol{\theta}}\rho(\pi) = \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} \boldsymbol{\nabla}_{\boldsymbol{\theta}}\pi(s,a) Q^\pi(s,a) \ . \tag{1}$$

This formulation contains explicitly the unknown value function, which has to be estimated. It can be replaced by a function approximator $f_{\boldsymbol{v}} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ (the *critic*) with real-valued parameter vector $\boldsymbol{v}$ satisfying the *convergence condition* $\sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} \pi(s,a) \left[Q^\pi(s,a) - f_{\boldsymbol{v}}(s,a)\right] \boldsymbol{\nabla}_{\boldsymbol{v}} f_{\boldsymbol{v}}(s,a) = 0$. This leads directly to the extension of the policy gradient theorem for function approximation. If $f_{\boldsymbol{v}}$ satisfies the convergence condition and is *compatible* with the policy parametrization in the sense that

$$f_{\boldsymbol{v}} = \boldsymbol{\nabla}_{\boldsymbol{\theta}} \ln(\pi(s,a))\boldsymbol{v} + \text{const} \ , \tag{2}$$

then the policy gradient theorem holds if $Q^\pi(s,a)$ in (1) is replaced by $f_{\boldsymbol{v}}(s,a)$ [20].

Stochastic policies π with parameters $\boldsymbol{\theta}$ are parametrized probability distributions. The Fisher information matrix $F(\boldsymbol{\theta})$ induces a metric in the space of probability distributions that is independent of the coordinate system [21]. The direction of steepest ascent in this metric space is given by $\tilde{\boldsymbol{\nabla}}_{\boldsymbol{\theta}}\rho(\pi) = F(\boldsymbol{\theta})^{-1}\boldsymbol{\nabla}_{\boldsymbol{\theta}}\rho(\pi)$, thus inducing "natural" gradient ascent in this direction. We have $F(\boldsymbol{\theta}) = \sum_{s \in \mathcal{S}} d^\pi(s) \sum_{a \in \mathcal{A}} \pi(s,a) \boldsymbol{\nabla}_{\boldsymbol{\theta}} \ln(\pi(s,a)) (\boldsymbol{\nabla}_{\boldsymbol{\theta}} \ln(\pi(s,a)))^{\mathrm{T}}$ using the definitions above. This implies $\boldsymbol{\nabla}_{\boldsymbol{\theta}}\rho(\pi) = F(\boldsymbol{\theta})\boldsymbol{v}$, which leads to the simple equality $\tilde{\boldsymbol{\nabla}}_{\boldsymbol{\theta}}\rho(\pi) = \boldsymbol{v}$.

Algorithm 1: episodic Natural Actor-Critic

initialize $\boldsymbol{\theta}$, $\boldsymbol{\Phi} = \mathbf{0}$, $\boldsymbol{R} = \mathbf{0}$, dimension n
for $k = 1, \ldots$ **do**
  // k counts number of policy updates
  for $e = 1, \ldots, e_{max}$ **do**
    // e counts number of episodes per policy update, $e_{\text{max}} \geq n + 1$
    for $t = 1, \ldots, T$ **do**
      // t counts number of time steps per episode
      begin
        observe state $\boldsymbol{s}_t$
        choose action a_t from $\boldsymbol{\pi}_{\boldsymbol{\theta}}$
        perform action a_t
        observe reward r_{t+1}
      end
      for $i = 1, \ldots, n$ **do**
        $\boldsymbol{\Phi}(e, i) \leftarrow \boldsymbol{\Phi}(e, i) + \gamma^t \frac{\partial}{\partial \theta_i} \ln \boldsymbol{\pi}_{\boldsymbol{\theta}}(s_t, a_t)$
      $\boldsymbol{R}(e) \leftarrow \boldsymbol{R}(e) + \gamma^t r_{t+1}$
    $\boldsymbol{\Phi}(e, n+1) \leftarrow 1$
  // update policy parameters:
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + (\boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{R}$

The function approximator $f_{\boldsymbol{v}}$ estimates the advantage function $A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)$, where $V^\pi(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} | \pi, s_0 = s\right]$ is the state value function. Inserting this in the Bellman equation for Q^π leads to

$$Q^\pi(s_t, a_t) = A^\pi(s_t, a_t) + V^\pi(s_t) = \sum_{s'} P^{a_t}_{s_t, s'} \left(\mathcal{R}^{a_t}_{s_t, s'} + \gamma V^\pi(s') \right) \ .$$

Now we sum over a sample path:

$$\sum_{t=0}^{T} \gamma^t A^\pi(s_t, a_t) = \sum_{t=0}^{T} \gamma^t r_{t+1} + \gamma^{T+1} V^\pi(s_{T+1}) - V(s_0) \ .$$

For an episodic task that is in its terminal state in time step T it holds that $V^\pi(s_{T+1}) = 0$, thus, after replacing A^π using (2), we get:

$$\sum_{t=0}^{T} \gamma^t (\boldsymbol{\nabla}_{\boldsymbol{\theta}} \ln \pi(s_t, a_t))^{\mathrm{T}} \boldsymbol{v} - V(s_0) = \sum_{t=0}^{T} \gamma^t r_{t+1} \ .$$

For fixed start states we have $V^\pi(s_0) = \rho(\pi)$, and we get a linear regression problem with $n + 1$ unknown variables $\boldsymbol{w} = [\boldsymbol{v}^{\mathrm{T}}, V^\pi(s_0)]^{\mathrm{T}}$ that can be solved after $n + 1$ observed episodes (where n is the dimension of $\boldsymbol{\theta}$ and $\boldsymbol{v}$):

$$\left[\sum_{t=0}^{T(e_i)} \left(\gamma^t \boldsymbol{\nabla}_{\boldsymbol{\theta}} \ln \pi(s_t^{e_i}, a_t^{e_i}) \right]^{\mathrm{T}}, -1 \right]^{\mathrm{T}} \boldsymbol{v} = \sum_{t=0}^{T(e_i)} \gamma^t r_{t+1}^{e_i} \ , \ i = 1, \ldots, n$$

The superscripts indicate the episodes. In Algorithm 1 the likelihood information for a sufficient number of episodes is collected in a matrix $\boldsymbol{\Phi}$ and the return for each episode in $\boldsymbol{R}$. In every update step one inversion of the matrix $\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}$ is necessary [19].

2.2 Random Weight Guessing

We use simple random search (random weight guessing) as a baseline comparison [9]. In every iteration new policy parameters are drawn uniformly from an interval $[-\theta_{\max}, \theta_{\max}]^k$, where k is the number of policy parameters. This candidate solution is evaluated and is maintained if it outperforms the best solution so far and discarded otherwise.

2.3 Evolution Strategies

We promote using the CMA-ES for solving MDPs. The highly efficient use of information and the fast adaptation of step size and covariance matrix (which corresponds to learning the metric underlying the optimization problem) makes the CMA-ES one of the best direct search algorithms for real-valued optimization [14]. For a detailed description of the CMA-ES we refer to the articles by Hansen et al. [15, 16].

For the first time the CMA-ES was proposed for RL in [10]. It was found that the CMA-ES outperforms alternative evolutionary RL approaches on variants of the pole balancing benchmark in fully and partially observable environments. In a more recent study by [9], these results were compared to 8–12 (depending on the task) other RL algorithms including value-function and policy gradient approaches. On the four test problems where the CMA-ES was considered, it ranked first, second (twice), and third. In [11] the CMA-ES was applied to learn the behavior of a driver assistance system, where neural attractor dynamics were used to represent the policies. In [22] and [23] the CMA-ES was used for RL in robotics. The authors combined the CMA-ES with evolutionary topology optimization to evolve artificial neutral networks.

Recently, we performed a systematic comparison between the CMA-ES and policy gradient methods with variable metrics [12, 13]. The preliminary experiments indicate that the CMA-ES is much more robust regarding the choice of hyperparameters and initial policies.

3 Experiments

The experiments conducted in this paper extend our previous work described in [12], where we analyzed the cart pole balancing task, which is a well-known benchmark in RL. In this paper we study a more difficult variant, double-pole balancing, which has already been solved successfully with evolutionary methods, see [24, 10, 9].

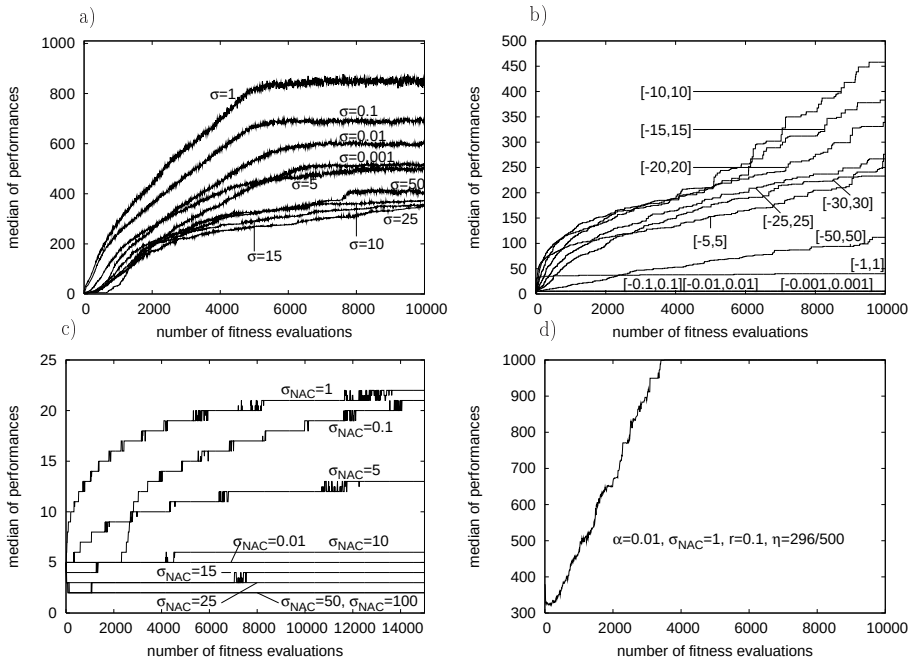

Fig. 1. Performance of CMA-ES, NAC, and random weight guessing on the double pole balancing task. The median over 500 independent trials is shown for the CMA in a) and for random weight guessing in b). For the NAC only exemplary the performance for learning rate $\alpha = 0.1$ is shown in c). The performance of the NAC with initial policy parameters drawn uniformly from a hypersphere with radius $r = 0.1$ centered at a global optimum at $(-4.19408, -13.2605, -1.54318, -36.91, 4.39037, 3.53484)$ and parameter values $\alpha = 0.01$ and $\sigma_{\text{NAC}} = 1$ is shown in d).

Double-pole balancing. Two poles of different length ($l_1 = 1$m for the first pole with mass $m_1 = 0.1$kg and $l_2 = 0.1$m for the second pole with mass $m_2 = 0.01$kg) are mounted side by side on the same 1-dimensional cart with mass $m_{\text{c}} = 1$kg and are to be balanced simultaneously. The equations of motion for two poles are given in [25].[1] This task is only solvable if the two poles differ in length. The state $\boldsymbol{s} = [x, \dot{x}, \zeta_1, \dot{\zeta}_1, \zeta_2, \dot{\zeta}_2]^{\text{T}}$ is given by the cart's distance to the center of

[1] For $i \in \{1, 2\}$ we have:

$$\ddot{x} = \frac{F - \mu_{\text{c}}\ \text{sgn}(\dot{x}) + \sum_{i=1}^{2} \tilde{F}_i}{m_{\text{c}} + \sum_{i=1}^{2} \tilde{m}_i}\ ,\ \ddot{\zeta}_i = -\frac{3}{8\, l_i}\left(\ddot{x}\, \cos\zeta_i + g\, \sin\zeta_i + \frac{2\mu_i\, \dot{\zeta}_i}{m_i\, l_i}\right)$$

$$\tilde{F}_i = 2 m_i\, l_i\, \dot{\zeta}_i^2\, \sin\zeta_i + \frac{3}{4}\, m_i\, \cos\zeta_i \left(\frac{2\mu_i\, \dot{\zeta}_i}{m_i\, l_i} + g \sin\zeta_i\right)\ ,\ \tilde{m}_i = m_i \left(1 - \frac{3}{4}\, \cos^2\, \zeta_i\right)$$

Here $g = 9.81\,\text{m/sec}^2$ is the acceleration due to gravity and $\mu_{\text{c}} = 5 \cdot 10^{-4}\,\text{Ns/m}$ the coefficient of friction of the cart, $\mu_1 = \mu_2 = 2 \cdot 10^{-6}\,\text{Nms}$ are the coefficients of friction for the first and second pole, respectively. The effective force from pole i on

the track $x \in [-2.4, 2.4]$ and velocity $\dot{x}$, the current angle ζ_1 of the longer pole and its angular velocity $\dot{\zeta}_1$ and the current angle ζ_2 of the second pole together with its angular velocity $\dot{\zeta}_2$. Actions are continuous forces $a = F$ applied to the cart parallel to the x-axis. The dynamical system is numerically solved using fourth-order Runge-Kutta integration with step size $\tau = 0.01\,\mathrm{s}$.

Experimental setup. The agent follows a deterministic policy $\pi_{\mathrm{deter}}(\boldsymbol{s}) = \boldsymbol{s}^{\mathrm{T}}\boldsymbol{\theta}$, with $\boldsymbol{\theta} \in \mathbb{R}^6$. The policy parameters $\boldsymbol{\theta}$ are initialized with zero.[2] For learning, the NAC uses the stochastic policy $\pi^{\mathrm{stoch}}_{\boldsymbol{\theta}}(\boldsymbol{s}, a) = \mathrm{N}(\pi^{\mathrm{deter}}_{\theta}(\boldsymbol{s}), \sigma_{\mathrm{NAC}})$, where the standard deviation σ_{NAC} is viewed as an additional adaptive seventh parameter of the method and is initialized independently. After every time step the agent receives a reward signal of $r_{t+1} = 1$. A time step corresponds to 0.02s simulation time. An episode ends after 1000 time steps (20s) or when either of the poles leaves the feasible region $[-36°, 36°]$ or the cart leaves the interval $[-2.4, 2.4]$. All episodes start in the same initial state $\boldsymbol{s}_0 = [0, 0, 1°, 0, 0, 0]^{\mathrm{T}}$. Since the task is episodic we use a discount factor of $\gamma = 1$ in the performance measure. The fitness function used by the CMA-ES is the accumulated reward observed over one episode (one episode is sufficient because the task is deterministic in our experiments) $\rho(\pi) = \sum_{t=1}^{T} r_t = T$. Thus the fitness of a policy is determined by the number of time steps T the poles are balanced without the cart leaving the feasible region. The same function is used for evaluation of the NAC and of random weight guessing.

We employ the CMA-ES with rank-μ covariance update [16], where all parameters are set to default values. The population sizes are $\mu = 3$ and $\lambda = 6$, accordingly. Candidate solutions outside the box $[-50, 50]^6$ are discarded and a new offspring is generated. We test different initial global step sizes $\sigma \in \{0.001, 0.1, 1, 1, 10, 15, 20, 25, 50\}$.

In the case of random weight guessing, we vary the interval lengths $\theta_{\max} \in \{0.001, 0.1, 1, 1, 10, 15, 20, 25, 30, 35, 50\}$. For the NAC, we test all combinations of $\alpha \in \{0.0001, 0.001, 0.01, 0.1, 0.3\}$ and $\sigma_{\mathrm{NAC}} \in \{0.1, 1, 5, 10, 15, 25, 50, 100\}$.

Each algorithm gets a budget of 10000 episodes per trial. A trial is stopped and regarded as successful when the poles are balanced for 1000 time steps.

4 Results

Selected results are shown in Fig. 1. Table 1 lists the success rates for the different hyperparameters. The episodic NAC never managed to balance the poles, except for $(\alpha = 0.001, \sigma_{NAC} = 100)$, and $(\alpha = 0.01, \sigma_{NAC} = 50)$, where it found a solution in 1 and 2 out of 500 trials, respectively.

While the double-pole balancing benchmark is not very difficult for evolutionary RL with neural network policies [10, 9], it is obviously a challenging

the cart is given by $\tilde{F}_i$ and its effective mass by $\tilde{m}_i$. The sign function sgn "inherits" the unit of measurement of its argument.

[2] We assume that the measurement units of the single components of $\boldsymbol{\theta}$ are chosen such that $\boldsymbol{s}^{\mathrm{T}}\boldsymbol{\theta}$ is a force.

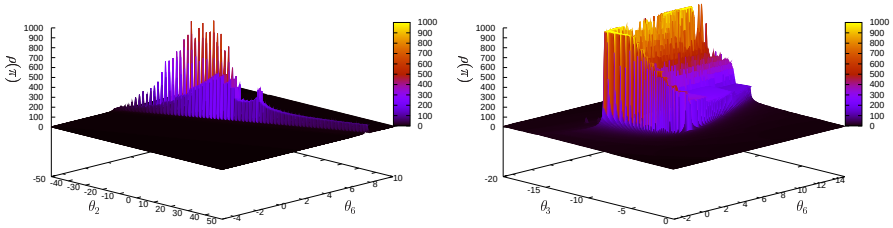

Fig. 2. 2-dimensional projection of the fitness landscape around a global optimum. On the left, the parameters θ_2 and θ_6 are varied while the other parameters are fixed ($\theta_1 = -4.19408$, $\theta_3 = -1.54318$, $\theta_4 = -36.91$, $\theta_5 = 4.39037$). On the right, θ_2 and θ_5 are varied ($\theta_1 = -4.19408$, $\theta_3 = -1.54318$, $\theta_4 = -36.91$, $\theta_6 = 3.53484$).

problem using linear policies. Although the CMA-ES is usually much better than random weight guessing [13], this difference is not so clear in this study. The performance of both methods depends on the choice of the hyperparameter, the initial global step size and the bounds of the search interval, respectively. For $\theta_{\max} \in \{5, 10, 15\}$ random weight guessing succeeds significantly more frequently (χ^2-test, $p < .05$) to balance the poles compared to the CMA-ES with the *worst* hyperparameter choice $\sigma = 20$. For all other values of $\theta_{\max}$ (except $\theta_{\max} \in \{5, 20\}$) the purely random search performed significantly worse (χ^2-test, $p < .05$) than the CMA-ES regardless of the choice of intial global step size σ. The CMA-ES is always significantly better than the episodic NAC. The performance of the CMA-ES is comparatively independent of the choice of the initial step size. Random weight guessing requires that the search intervals fit the problem. If the boundaries are too large, it will not sample a good solution, if they are too small, there might be no good solutions in the parameter ranges at all.

The bad performance of the episodic NAC is striking. To understand the results, we visualized the objective function landscape around a global optimum found by the CMA-ES, see Fig. 2. In these projections, the objective function is clearly multi-modal and contains plateaus of equal (bad) quality. Thus, in these two-dimensional plots the landscapes are almost worst case scenarios for purely gradient-based methods. The objective function is also difficult for ESs,

Table 1: Success rates for CMA-ES and random weight guessing.

Success rate η of CMA-ES for different values of the initial global step size σ.										
σ	0.001	0.01	0.1	1	5	10	15	20	25	50
η	$\frac{151}{500}$	$\frac{172}{500}$	$\frac{201}{500}$	$\frac{245}{500}$	$\frac{154}{500}$	$\frac{144}{500}$	$\frac{145}{500}$	$\frac{141}{500}$	$\frac{161}{500}$	$\frac{161}{500}$

Success rate η of stochastic search for different intervals $[-\theta_{\max}, \theta_{\max}]$.												
$\theta_{\max}$	0.001	0.01	0.1	1	5	10	15	20	25	30	35	50
η	$\frac{0}{500}$	$\frac{0}{500}$	$\frac{0}{500}$	$\frac{0}{500}$	$\frac{171}{500}$	$\frac{182}{500}$	$\frac{191}{500}$	$\frac{159}{500}$	$\frac{113}{500}$	$\frac{85}{500}$	$\frac{77}{500}$	$\frac{20}{500}$

but at least there is some structure in the fitness landscape they can exploit and they are much less likely to get stuck in local optima with small basins of attraction. However, when initializing the policy parameters close to the difficult global optimum shown in Fig. 2 the NAC works very efficiently, see Fig. 1 d).

5 Conclusion

Evolutionary reinforcement learning (RL) using the covariance matrix adaptation evolution strategy (CMA-ES) resembles policy gradient methods, in particular the episodic natural actor-critic (NAC) algorithm. Both strategies search directly in the space of policies, are variable metric methods, and rely on normally distributed variations for exploration. Of course, the frequency and the level at which the variations are applied vary. We claim that in practice the CMA-ES is much more robust w.r.t. the choice of hyperparameters, policy initialization, and especially noise, while, given appropriate hyperparameters, the NAC can outperform the CMA-ES in terms of learning speed if initialized close to a desired policy. This is supported by the experiments on the double-pole balancing benchmark in this study, which turns out to be surprisingly difficult when linear policies are considered. Because of plateaus and undesired local optima in the objective function landscape, the CMA-ES is superior compared to approaches purely based on estimated performance gradients. However, even the CMA-ES has difficulties on this landscape as shown by the comparison with random search.

In future work we will extend the experiments to different, higher dimensional benchmark tasks and to other direct policy search methods.

Acknowledgment. The authors acknowledge support from the German Federal Ministry of Education and Research within the Bernstein group "The grounding of higher brain function in dynamic neural fields".

References

1. Sutton, R., Barto, A.: Reinforcement Learning: An Introduction. MIT Press (1998)
2. Bertsekas, D., Tsitsiklis, J.: Neuro-Dynamic Programming. Athena Scientific (1996)
3. Heidrich-Meisner, V., Lauer, M., Igel, C., Riedmiller, M.: Reinforcement learning in a Nutshell. In: 15th European Symposium on Artificial Neural Networks (ESANN 2007), Evere, Belgien: d-side publications (2007) 277–288
4. Whitley, D., Dominic, S., Das, R., Anderson, C.W.: Genetic reinforcement learning for neurocontrol problems. Machine Learning **13**(2–3) (1993) 259–284
5. Moriarty, D., Schultz, A., Grefenstette, J.: Evolutionary Algorithms for Reinforcement Learning. Journal of Artificial Intelligence Research **11** (1999) 199–229
6. Chellapilla, K., Fogel, D.: Evolution, neural networks, games, and intelligence. IEEE Proc. **87**(9) (1999) 1471–1496
7. Stanley, K.O., Miikkulainen, R.: Evolving neural networks through augmenting topologies. Evolutionary Computation **10**(2) (2002) 99–127

8. Whiteson, S., Stone, P.: Evolutionary function approximation for reinforcement learning. Journal of Machine Learning Research **7** (2006) 877–917
9. Gomez, F., Schmidhuber, J., Miikkulainen, R.: Efficient non-linear control through neuroevolution. In: Proc. European Conference on Machine Learning (ECML 2006). Volume 4212 of LNCS., Springer-Verlag (2006) 654–662
10. Igel, C.: Neuroevolution for reinforcement learning using evolution strategies. In: Congress on Evolutionary Computation (CEC 2003). Volume 4., IEEE Press (2003) 2588–2595
11. Pellecchia, A., Igel, C., Edelbrunner, J., Schöner, G.: Making driver modeling attractive. IEEE Intelligent Systems **20**(2) (2005) 8–12
12. Heidrich-Meisner, V., Igel, C.: Similarities and differences between policy gradient methods and evolution strategies. In: 16th European Symposium on Artificial Neural Networks (ESANN), Evere, Belgien: d-side publications (2008) 149–154
13. Heidrich-Meisner, V., Igel, C.: Variable metric reinforcement learning methods applied to the noisy mountain car problem. In: European Workshop on Reinforcement Learning. (2008) Accepted.
14. Beyer, H.G.: Evolution strategies. Scholarpedia **2**(8) (2007) 1965
15. Hansen, N., Ostermeier, A.: Completely derandomized self-adaptation in evolution strategies. Evolutionary Computation **9**(2) (2001) 159–195
16. Hansen, N., Müller, S., Koumoutsakos, P.: Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). Evolutionary Computation **11**(1) (2003) 1–18
17. Peters, J., Vijayakumar, S., Schaal, S.: Reinforcement learning for humanoid robotics. In: Proc. 3rd IEEE-RAS Int'l Conf. on Humanoid Robots. (2003) 29–30
18. Riedmiller, M., Peters, J., Schaal, S.: Evaluation of policy gradient methods and variants on the cart-pole benchmark. In: Proc. 2007 IEEE Internatinal Symposium on Approximate Dynamic Programming and Reinforcement Learning (ADPRL 2007). (2007) 254–261
19. Peters, J., Schaal, S.: Natural actor-critic. Neurocomputing **71**(7-9) (2008) 1180–1190
20. Sutton, R., McAllester, D., Singh, S., Mansour, Y.: Policy gradient methods for reinforcement learning with function approximation. In: Advances in Neural Information Processing Systems. Volume 12. (2000) 1057–1063
21. Amari, S., Nagaoka, H.: Methods of Information Geometry. Number 191 in Translations of Mathematical Monographs. American Mathematical Society and Oxford University Press (2000)
22. Siebel, N.T., Sommer, G.: Evolutionary reinforcement learning of artificial neural networks. International Journal of Hybrid Intelligent Systems **4**(3) (2007) 171–183
23. Kassahun, Y., Sommer, G.: Efficient reinforcement learning through evolutionary acquisition of neural topologies. In: 13th European Symposium on Artificial Neural Networks, d-side (2005) 259–266
24. Gomez, F., Miikkulainen, R.: Solving non-Markovian control tasks with neuroevolution. Proceedings of the 16th International Joint Conference on Artificial Intelligence (1999) 1356–1361
25. Wieland, A.: Evolving neural network controllers for unstable systems. Neural Networks, 1991., IJCNN-91-Seattle International Joint Conference on **2** (1991)