Model-based Direct Policy Search

(Extended Abstract)

Jan Hendrik Metzen
Robotics Group, University Bremen
Robert-Hooke-Str. 5, D-28359 Bremen, Germany
jhm@informatik.uni-bremen.de

Frank Kirchner
Robotics Group, University Bremen
Robert-Hooke-Str. 5, D-28359 Bremen, Germany
frank.kirchner@informatik.uni-bremen.de

ABSTRACT

Scaling Reinforcement Learning (RL) to real-world problems with continuous state and action spaces remains a challenge. This is partly due to the reason that the optimal value function can become quite complex in continuous domains. In this paper, we propose to avoid learning the optimal value function at all but to use direct policy search methods in combination with model-based RL instead.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning; I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search

General Terms

Algorithms

Keywords

Direct policy search, model-based learning, reinforcement learning

1. INTRODUCTION

The use of a certain function approximation technique for representing the (state-action) value function (VF) of a policy in a continuous markov decision process (MDP) limits the class of representable policies. If the function approximator cannot represent the VF of an optimal (or any close-to-optimal) policy, the performance of any RL algorithm that is using this function approximator is deteriorated. Since sample-complexity (the number of interactions with the environment an agent requires to learn a sufficient policy) is usually a major concern in RL, a function approximator with broad generalization (and thus low resolution) is desirable. Thus, there is a conflict between sample efficiency and representability of the value function.

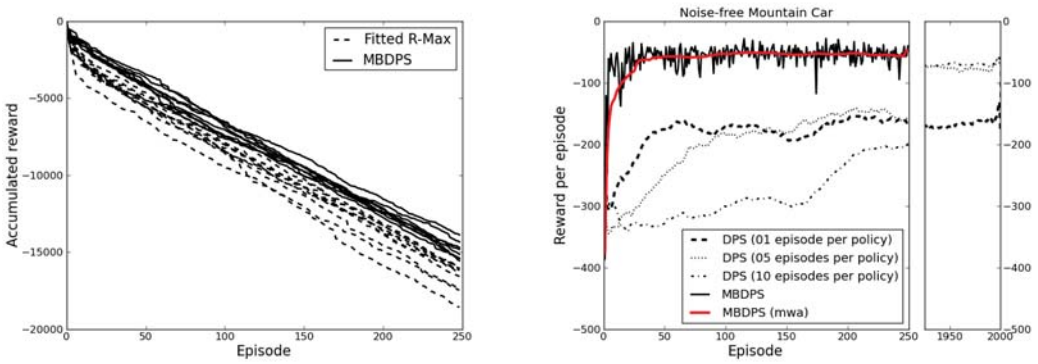
In contrast to VF-based learning, so-called direct policy search (DPS) algorithms, do not learn a policy's value function at all but search directly for a close-to-optimal policy. Typically, DPS algorithms optimize the parameters of a predefined class of policies. Thus, DPS algorithms can only find a close-to-optimal policy if such a policy is contained in the given policy class. One main difference is that VF-based learning is affected by the complexity of a policy's value function while DPS algorithms are affected by the complexity of the policy itself. Thus, if close-to-optimal policies can be represented more easily than their value functions, DPS might have an intrinsic advantage over VF-based methods with regard to sample complexity. A second difference between VF-based learning methods like temporal difference (TD) learning and DPS is that typically the former learns based on single state transitions and rewards while the later learns based solely on the accumulated reward a policy achieves in one (or several) episodes. Thus, TD learning makes more efficient use of the information obtained within one episode than DPS does. This reduces the sample-efficiency of DPS methods typically.

The hypothesis investigated in this paper is that this disadvantage of DPS can be alleviated by combining DPS with model-based learning. In general, model-based methods learn a model of the state transition probability function $\mathcal{P} : S \times A \times S \rightarrow [0, 1]$ and of the reward function $\mathcal{R} : S \times A \rightarrow \mathbb{R}$ of a MDP. We call a DPS method model-based if the performance estimates used during DPS are based on "simulated experience" sampled from a learned model and not on real experience. The sample-efficiency of such a method depends primarily on the model-learning algorithm since only this component learns based on real experience. Thus, the intuition for the stated hypothesis is that whether DPS makes efficient use of the sampled experience is less important in a model-based setting since simulated experience is "cheap".

2. MODEL-BASED DIRECT POLICY SEARCH

One popular method for model-based RL in continuous domains is the VF-based algorithm *Fitted R-Max* that was proposed by Jong and Stone [3] and is based on the *R-Max* algorithm by Brafman and Tennenholtz [1]. While R-Max utilizes optimistic reward for state-action pairs that have not been tested a given number of times, Fitted R-Max uses optimistic reward for areas of the state-action space with a low density of samples. Learning the (generative) model itself is based on a method which is essentially k-Nearest-Neighbor (kNN) where the instances are (gaussian) weighted based on their distance to the query point. Based on this model, fitted value-iteration is used to derive a value function such that the corresponding greedy policy acts optimally with regard to the (optimistic) model. The main idea of this paper is to keep the model learning algorithm and the exploration

Cite as: Model-based Direct Policy Search (Extended Abstract), Jan Hendrik Metzen, Frank Kirchner, *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, van der Hoek, Kaminka, Lespérance, Luck and Sen (eds.), May, 10–14, 2010, Toronto, Canada, pp. 1589-1590



(a) (b)

Figure 1: Comparison of the performance of MBDPS with Fitted R-Max (a) and with DPS (b) in the noise-free mountain car domain. The figures show the accumulated reward (a) resp. the reward per episode (b) averaged over 8 independent runs. The DPS curves were smoothed using a moving window average of length 50 in order to compensate for fluctuations of the performance of the individual policies within a generation. "MBDPS (mwa)" shows the same moving window average for the MBDPS data.

mechanism (R-Max) unchanged but to change the planning algorithm that derives a policy from a learned model: The fitted value-iteration planner used in Fitted R-Max to learn a value function is replaced by a direct policy search planner that learns directly a policy. The hypothesis is that this increases the sample-efficience in domains for which representing a close-to-optimal value function is significantly more complex than representing a close-to-optimal policy.

The proposed method, called Model-based Direct Policy Search (MBDPS) alternates between sampling trajectories $(s_0, \pi(s_0), r_0, s_1, \pi(s_1), \ldots, r_n, s_n)$ from the model to obtain an estimate $\hat{R}(\pi) = \sum_{i=0}^{n} r_i$ of the accumulated reward a policy π would obtain in the actual environment, improving the policy based on these samples, and updating the estimate of the model based on new transitions (s, a, r, s') observed in the actual environment. In order to enforce exploration, the model from which trajectories are sampled is modified according to the R-Max principle such that actions that haven't been tried often in the proximity of the current state yield higher reward than predicted by the maximum likelihood model.

3. RESULTS

In this section, we present results in the mountain car domain [4]. The DPS planner learns a deterministic, linear policy with additional bias input and uses the CMA-ES [2] algorithm for black box optimization of the policy's parameters. Figure 1a shows a comparison of MBDPS and Fitted R-Max. MBDPS reaches the goal significantly faster during the first 10 episodes ($p < 0.003$) and obtains a larger accumulated reward after 250 episodes ($p < 4 * 10^{-4}$). Since the same mechanism for exploration control is used in both algorithms (namely R-Max), it is unlikely that the improved performance of MBDPS is caused by an increased level of "exploitatory behaviour". We suspect that the reason for the worse performance of Fitted R-Max during the first episodes is that it has to represent an approximation of the optimal value function explicitly. In Fitted R-Max, the value function is represented using a kNN-based generalization of the values of the already visited states. During the first episodes, there are only a few states the agent has visited and thus the value function can only be represented very coarsely. In contrast, the MBDPS agent is not affected by this since it does not need to represent a value function.

Figure 1b shows a comparison of MBDPS and model-free DPS for different number of episodes per policy evaluation. While MBDPS was stopped after 250 episodes since no further progress was observed, the DPS agents were evaluated for 2000 episodes in order to see the final level of performance they are able to reach. As can be seen, MBDPS learns faster (i.e. with less actual experience) than all variants of DPS and achieves significantly more accumulated reward during the first 250 episodes ($p < 4 * 10^{-4}$). At the same time, learning based on simulated instead of actual experience does not deteriorate the long-term performance of MBDPS.

4. CONCLUSION

We have shown that model-based direct policy search can learn more sample-efficient than other state of the art model-based RL methods like Fitted R-Max. Furthermore, MBDPS converges to policies that are not worse regarding the accumulated reward than those learned by model-free DPS.

5. REFERENCES

[1] R. I. Brafman and M. Tennenholtz. R-MAX - a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3:213–231, 2001.

[2] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9:159–195, 2001.

[3] N. K. Jong and P. Stone. Model-Based exploration in continuous state spaces. In *Abstraction, Reformulation, and Approximation*, pages 258–272. 2007.

[4] R. Sutton and A. Barto. *Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning)*. The MIT Press, 1998.