Iterative Model-Fitting and Local Controller Optimization - Towards a Better Understanding of Convergence Properties

Manuel Wüthrich [1] **Bernhard Schölkopf** [1]

Abstract

An intuitive strategy in model-based reinforcement learning (RL) is the following: We always execute the controller which is locally optimal with respect to the current model, while the model is updated continuously with the newly collected data. This strategy seems to be quite widely used, but to the best of our knowledge, a theoretical analysis does not exist yet. Herein we take first steps to correct this deficiency. We believe that such an analysis will help us understand when this strategy is applicable, and how the different components (e.g. the model fitting) have to be designed in order to guarantee convergence.

1. Introduction

In model based RL we run experiments on the real platform in order to build a dynamics model. This model is then used to find a good controller. Here we focus on the exploration problem, i.e. which experiments to run in order to quickly find a good controller. In Figure 1 we represent a strategy which makes intuitive sense and seems to be quite commonly used: We always execute the controller which is locally optimal with respect to the current model, while the model is updated continuously with the newly collected data, see e.g. (Deisenroth & Rasmussen, 2011; Wahlström et al., 2015; Deisenroth et al., 2013). Initially, the model may be arbitrarily wrong, but intuitively we would expect that it becomes more and more accurate and we ultimately find a controller which is locally optimal with respect to the true dynamics. However, this is merely an intuition and it is not clear under what conditions this procedure will actually converge to a local optimum of the true dynamics. Such a theoretical analysis of this strategy does not exist yet to the best of our knowledge, herein we take first steps to correct this deficiency. We believe that a better theoretical understanding of this type of algorithm will help us understand when it is applicable, and how the different components (e.g. the model fitting) have to be designed in order to guarantee convergence.

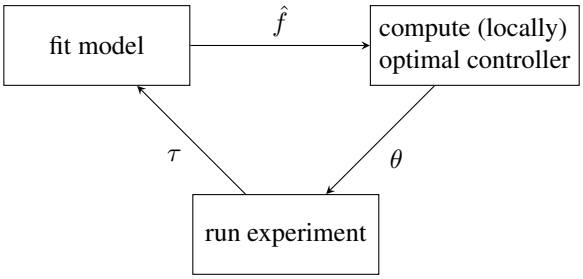

Figure 1. Flow of information: 1) We compute a controller θ which is locally optimal with respect to the current dynamics model $\hat{f}$, 2) we execute this controller on the real system and obtain the data $\tau = (x_{1:T}, u_{1:T})$ consisting of the state and control trajectory, 3) we fit the model to all the data seen thus far etc.

Herein we consider a simplified setting, an analysis of a more realistic situation is future work. We assume the true dynamics to be deterministic and fully observable. We assume that it is known to the agent that the true dynamics belong to a finite set of possible dynamics. We prove that in this setting this algorithm indeed converges to a local optimum of the true dynamics. This preliminary result is promising, and we believe that it can be extended to more realistic settings.

2. Related Work

There is a large body of literature investigating the exploration-exploitation problem for bandit settings, see e.g. (Lai & Robbins, 1985; Auer, 2002; Madani et al., 2004; Audibert & Bubeck, 2010). However, these algorithms are typically concerned with global optimization. Similarly, it seems that theoretical work considering the full RL problem typically assumes discrete systems with the goal of finding a globally optimal policy (e.g. (Kearns & Singh, 2002; Brafman & Tennenholtz, 2002; Jaksch et al., 2010)). Some results in the derivative-free convex optimization lit-

[1]Empirical Inference Department, Max Planck Institute for Intelligent Systems, Tübingen, Germany. Correspondence to: Manuel Wüthrich <manuel.wuthrich@google.com>.

Accepted at the FAIM workshop "Prediction and Generative Modeling in Reinforcement Learning", Stockholm, Sweden, 2018.

erature (e.g. (Zinkevich, 2003; Agarwal et al., 2010; Conn et al., 2009)) seem to be more relevant for the situation we are considering here. However, in their setting one can merely observe the cost at the evaluated point, while here we observe entire state-action trajectories, giving us richer information. Furthermore, here we are interested in a particular algorithm which does not seem to have been studied in that literature. Nevertheless, there may be interesting connections worth exploring in the future.

In the adaptive control literature, a strategy very similar to the one we are considering here is known as self-tuning regulator, see e.g. (Åström, 1995). Convergence results have been obtained for some cases such as linear systems (e.g. Theorem 4.1 in (Åström, 1995)) and robotic manipulators with unknown inertias (which is linear in parameters, see e.g. (Slotine & Li, 1987)). These results are very interesting and may provide a good starting point, but they do not apply to the highly nonlinear models which are used in recent work on model-based RL.

In the model-based RL literature, the main focus seems to lie on the question of how to build a model from data, e.g. the top left box in Figure 1. There are many different approaches to this, some common ideas are to use simple local models (e.g. (Atkeson et al., 1997)), Gaussian processes (e.g. (Deisenroth et al., 2015; Eleftheriadis et al., 2017)) or neural networks (e.g. (Watter et al., 2015; Wahlström et al., 2015; Nagabandi et al., 2017)). However, it seems that the question of how to pick the experiments to execute is somewhat neglected. The typical strategy is to compute a (locally) optimal controller with respect to the model, collect some data (single or batch rollouts), then improve the model etc. However, it is not obvious that this approach will indeed converge to a (locally) optimal controller for the true dynamics. To the best of our knowledge, a theoretical analysis of this procedure does not exist yet, and hence it is not known under what circumstances it is applicable.

3. Problem Statement

A discrete-time, deterministic optimal control problem is defined by the tuple $(\breve{X}, \breve{U}, T, f, \text{cost}, x1)$ where

- $\breve{X} \subseteq \mathbb{R}^N$ is the set of states,
- $\breve{U} \subseteq \mathbb{R}^M$ is the set of controls,
- $T \in \mathbb{N}$ is the number of time steps,
- $f : \breve{X} \times \breve{U} \times \{1, ..., T\} \to \breve{X}$ is the dynamics function,
- $\text{cost} : (\breve{X} \times \breve{U})^T \to \mathbb{R}$ is the cost function,
- $x1 \in \breve{X}$ is the initial state.

Algorithm 1

input: $(f, H = (H_l)_{l \in \{1,...,L\}}, \text{loc\_opt}_\epsilon, \epsilon, K)$
initialize: $H^1 \leftarrow H$
for $k = 1$ **to** K **do**
  $\theta^k \leftarrow \text{loc\_opt}_\epsilon(H_1^k)$
  $\delta^k \leftarrow$ sample from ball(ϵ)
  $H^{k+1} \leftarrow$ hypotheses in H^k consistent with rollout(f, θ^k) and rollout$(f, \theta^k + \delta^k)$
end for
return θ^K

The goal is to find the optimal policy $\pi : \breve{X} \times \{1, ..., T\} \to \breve{U}$. We assume that the policy is parametrized by some real parameter vector θ. For convenience we define

$$\text{rollout}(f,\theta) := (x_{1:T}, u_{1:T}) \tag{1}$$
$$\text{with } x_1 = x1 \tag{2}$$
$$x_{t+1} = f(x_t, u_t, t) \quad \forall t \in \{1, .., T-1\} \tag{3}$$
$$u_t = \pi_\theta(x_t, t) \quad \forall t \in \{1, .., T\}. \tag{4}$$

Hence, the optimal parameters are the ones which minimize

$$\text{cost}(\text{rollout}(f, \theta)). \tag{5}$$

While global optimization of this objective is typically infeasible, very impressive results have been obtained using local optimizers (e.g. in robotic optimal control, see (Todorov et al., 2012)). Here we define an ϵ-local optimum as a point θ which has a lower cost than every other point within some ϵ-ball except for a set of points with measure zero, i.e.

$$\mathbb{P}(\text{cost}(\text{rollout}(f, \theta)) \leq \text{cost}(\text{rollout}(f, \theta+\delta))) = 1 \quad \text{with } \delta \sim \text{ball}(\epsilon). \tag{6}$$

We assume that we have access to a local optimizer, i.e. there is a function

$$\text{loc\_opt}_\epsilon(f) \tag{7}$$

which returns an ϵ-local optimal controller for any dynamics f. Such local optimizers are usually very efficient since they can use gradient information. However, here we do not know f, but we only know a set of possible dynamics $f \in H$ (assumed to have cardinality L). The question is whether we can still guarantee convergence to a local optimum of the unknown, true dynamics f.

4. Algorithm

In Algorithm 1 we describe the simple method which we analyze in the following. It starts with some arbitrary hypothesis from the set of possible dynamics H and executes a locally optimal controller with respect to that hypothesis as well as the same controller perturbed with some small

random perturbation. Then all dynamics hypotheses which are not consistent with these rollouts are removed from H, and one of the remaining hypothesis is adopted etc. We have the following result for this procedure:

Lemma 4.1. *If we follow Algorithm 1, then the probability of the returned controller θ_K being an ϵ-local optimum of the true dynamics f converges to 1 as $K \to \infty$.*

Proof. Each hypothesis H_l has a certain probability of being revealed as false when executing it's associated controller

$$\theta_l = \text{loc\_opt}_\epsilon(H_l) \tag{8}$$

on the real system (with and without perturbation $\delta \sim \text{ball}(\epsilon)$)

$$\begin{aligned} p_l := \mathbb{P}(&\text{rollout}(f, \theta_l) \neq \text{rollout}(H_l, \theta_l) \vee \\ &\text{rollout}(f, \theta_l + \delta) \neq \text{rollout}(H_l, \theta_l + \delta)). \end{aligned}$$

Let us call a hypothesis H_l ϵ-consistent if it has $p_l = 0$, i.e. it looks exactly like the true dynamics when executing it's associated controller or a slightly perturbed version of it. Additionally, let

$$p_{min} := \min\{p_l : l \in \{1, ..., L\}, p_l > 0\} \tag{9}$$

be the smallest probability of all hypotheses which are not ϵ-consistent.

Probability of not finding an ϵ-consistent hypothesis

Let us now bound the probability of ending up with a hypothesis which is not ϵ-consistent

$$\mathbb{P}(H_1^K \text{ not } \epsilon\text{-consistent}). \tag{10}$$

Let us note that once the algorithm has found an ϵ-consistent hypothesis, it will stick with it. Furthermore, we assumed that the true dynamics function is among the hypothesis set, hence if the algorithm discards $L - 1$ hypotheses, the remaining one is necessarily the true one (and hence obviously ϵ-consistent). Since the algorithm goes through H in order, it will take it the longest to find an ϵ-consistent solution if the only such hypothesis is the very last one, i.e. H_L. Hence we have

$$\mathbb{P}(H_1^K \text{ not } \epsilon\text{-consistent}) \leq \tag{11}$$

$$\mathbb{P}(\text{discarding at most } L-2 \text{ hypotheses} \tag{12}$$

$$\text{given } H_1, ..., H_{L-1} \text{ not } \epsilon\text{-consistent}). \tag{13}$$

At each round, when the considered hypothesis is not ϵ-consistent, the algorithm discards it with probability at least p_{min}. Hence,

$$\mathbb{P}(H_1^K \text{ not } \epsilon\text{-consistent}) \leq \text{CDF}_{\text{Binomial}}(L-2, K, p_{min}) \tag{14}$$

$$\leq \exp\left(-2\frac{(Kp_{min} - L)^2}{K}\right) \tag{15}$$

where we have used a well-known tail-bound for the cumulative distribution function of the binomial distribution. Hence, we clearly have

$$\lim_{K\to\infty} \mathbb{P}(H_1^K \text{ not } \epsilon\text{-consistent}) = 0. \tag{16}$$

An ϵ-consistent hypothesis yields a locally optimal controller

Suppose we have an ϵ-consistent hypothesis H_l with corresponding locally optimal controller θ_l. We need to show now that θ_l is not just locally optimal with respect to H_l, but also the true dynamics f. This is easy to see, because for θ_l together with some random perturbation $\delta \sim \text{ball}(\epsilon)$ have the following properties:

- ϵ-consistency of H_l implies that
  - $\text{rollout}(H_l, \theta_l) = \text{rollout}(f, \theta_l)$,
  - with probability 1 $\text{rollout}(H_l, \theta_l{+}\delta) = \text{rollout}(f, \theta_l + \delta)$,
- and θ_l being a locally optimal controller of H_l means that
  - with probability 1 $\text{rollout}(H_l, \theta_l{+}\delta)$ will have a larger cost than $\text{rollout}(H_l, \theta_l)$.

Hence with probability 1 $\text{rollout}(f, \theta_l + \delta)$ will have a larger cost than $\text{rollout}(f, \theta_l)$ which is precisely our definition of a locally optimal controller. □

5. Discussion

The exploration strategy we considered herein is to always act locally optimally with respect to the model, as it is being built incrementally from incoming data. While strategies along these lines are quite common in model-based RL, it seems that there is not much theoretical work on this topic yet. In this article, we have shown that this strategy will indeed converge to a local optimum of the true dynamics in the considered, simple setting. We believe that it is possible to extend this result to more realistic settings, which might give us important insights into the applicability and optimal design choices of this class of model-based RL algorithms. An interesting question for future work is whether we can extend this result to stochastic systems and hypothesis classes which do not necessarily contain the true dynamics function.

References

Agarwal, A., Dekel, O., and Xiao, L. Optimal Algorithms for Online Convex Optimization with Multi-Point Bandit Feedback. *COLT*, 2010.

Åström, K. J. *Adaptive Control*. Addison-Wesley, 1995.

Atkeson, C. G., Moore, A. W., and Schaal, S. Locally Weighted Learning. In Aha, D. W. (ed.), *Lazy Learning*, pp. 11–73. Springer Netherlands, Dordrecht, 1997.

Audibert, J.-Y. and Bubeck, S. Best arm identification in multi-armed bandits. In *COLT-23th Conference on Learning Theory-2010*, pp. 13–p, 2010.

Auer, P. Finite-time Analysis of the Multiarmed Bandit Problem. *Machine learning*, 47:235–256, 2002.

Brafman, R. I. and Tennenholtz, M. R-MAX - A General Polynomial Time Algorithm for Near-Optimal Reinforcement Learning. *Journal of machine learning research: JMLR*, 3(Oct):213–231, 2002.

Conn, A. R., Scheinberg, K., and Vicente, L. N. *Introduction to Derivative-Free Optimization*. SIAM, April 2009.

Deisenroth, M. and Rasmussen, C. E. PILCO: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, pp. 465–472, 2011.

Deisenroth, M. P., Neumann, G., and Peters, J. A Survey on Policy Search for Robotics. *Foundations and Trends® in Robotics*, 2(1–2):1–142, 2013.

Deisenroth, M. P., Fox, D., and Rasmussen, C. E. Gaussian Processes for Data-Efficient Learning in Robotics and Control. *IEEE transactions on pattern analysis and machine intelligence*, 37(2):408–423, February 2015.

Eleftheriadis, S., Nicholson, T., Deisenroth, M., and Hensman, J. Identification of Gaussian Process State Space Models. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30*, pp. 5309–5319. Curran Associates, Inc., 2017.

Jaksch, T., Ortner, R., and Auer, P. Near-optimal Regret Bounds for Reinforcement Learning. *Journal of machine learning research: JMLR*, 11:1563–1600, 2010.

Kearns, M. and Singh, S. Near-Optimal Reinforcement Learning in Polynomial Time. *Machine learning*, 49(2): 209–232, November 2002.

Lai, T. L. and Robbins, H. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, 6(1): 4–22, March 1985.

Madani, O., Lizotte, D. J., and Greiner, R. Active Model Selection. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, UAI '04, pp. 357–365, Arlington, Virginia, United States, 2004. AUAI Press.

Nagabandi, A., Yang, G., Asmar, T., Kahn, G., Levine, S., and Fearing, R. S. Neural Network Dynamics Models for Control of Under-actuated Legged Millirobots. November 2017.

Slotine, J.-J. E. and Li, W. On the Adaptive Control of Robot Manipulators. *The International journal of robotics research*, 6(3):49–59, September 1987.

Todorov, E., Erez, T., and Tassa, Y. MuJoCo: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, October 2012.

Wahlström, N., Schön, T. B., and Deisenroth, M. P. From Pixels to Torques: Policy Learning with Deep Dynamical Models. February 2015.

Watter, M., Springenberg, J. T., Boedecker, J., and Riedmiller, M. Embed to Control: A Locally Linear Latent Dynamics Model for Control from Raw Images. June 2015.

Zinkevich, M. Online convex programming and generalized infinitesimal gradient ascent. *Proceedings of the 20th International Conference on*, 2003.