Proceedings of Machine Learning Research vol 120:1–10, 2020 2nd Annual Conference on Learning for Dynamics and Control

Objective Mismatch in Model-based Reinforcement Learning

Nathan Lambert NOL@BERKELEY.EDU
University of California, Berkeley

Brandon Amos BDA@FB.COM
Omry Yadan OMRY@FB.COM
Roberto Calandra RCALANDRA@FB.COM
Facebook AI Research

Editors: A. Bayen, A. Jadbabaie, G. J. Pappas, P. Parrilo, B. Recht, C. Tomlin, M. Zeilinger

Abstract

Model-based reinforcement learning (MBRL) is a powerful framework for data-efficiently learning control of continuous tasks. Recent work in MBRL has mostly focused on using more advanced function approximators and planning schemes, with little development of the general framework. In this paper, we identify a fundamental issue of the standard MBRL framework – what we call *objective mismatch*. Objective mismatch arises when one objective is optimized in the hope that a second, often uncorrelated, metric will also be optimized. In the context of MBRL, we characterize the objective mismatch between training the forward dynamics model w.r.t. the likelihood of the one-step ahead prediction, and the overall goal of improving performance on a downstream control task. For example, this issue can emerge with the realization that dynamics models effective for a specific task do not necessarily need to be globally accurate, and vice versa globally accurate models might not be sufficiently accurate locally to obtain good control performance on a specific task. In our experiments, we study this objective mismatch issue and demonstrate that the likelihood of one-step ahead predictions is not always correlated with control performance. This observation highlights a critical limitation in the MBRL framework which will require further research to be fully understood and addressed. We propose an initial method to mitigate the mismatch issue by re-weighting dynamics model training. Building on it, we conclude with a discussion about other potential directions of research for addressing this issue.

1. Introduction

Model-based reinforcement learning (MBRL) is a popular approach for learning to control nonlinear systems that cannot be expressed analytically (Bertsekas, 1995; Sutton and Barto, 2018; Deisenroth and Rasmussen, 2011; Williams et al., 2017). MBRL techniques achieve the state of the art performance for continuous-control problems with access to a limited number of trials (Chua et al., 2018; Wang and Ba, 2019) and in controlling systems given only visual observations with no observations of the original system's state (Hafner et al., 2018; Zhang et al., 2019). MBRL approaches typically learn a *forward dynamics model* that predicts how the dynamical system will evolve when a set of control signals are applied. This model is classically fit with respect to the maximum likelihood of a set of trajectories collected on the real system, and then used as part of a control algorithm to be executed on the system (e.g., model-predictive control).

In this paper, we highlight a fundamental problem in the MBRL learning scheme: the *objective mismatch* issue. The learning of the forward dynamics model is decoupled from the subsequent

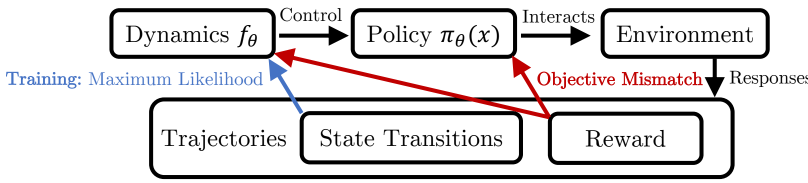


Figure 1: Objective mismatch in MBRL arises when a dynamics model is trained to maximize the likelihood but then used for control to maximize a reward signal not considered during training.

controller through the optimization of two different objective functions – prediction accuracy or loss of the single- or multi-step look-ahead prediction for the dynamics model, and task performance for the policy optimization. While the use of log-likelihood (LL) for system identification is an historically accepted objective, it results in optimizing an objective that does not necessarily correlate to controller performance. The contributions of this paper are to: 1) identify and formalize the problem of objective mismatch in MBRL; 2) examine the signs of and the effects of objective mismatch on simulated control tasks; 3) propose an initial mechanism to mitigate objective mismatch; 4) discuss the impact of objective mismatch and outline future directions to address this issue.

2. Model-based Reinforcement Learning

We now outline the MBRL formulation used in the paper. At time t, we denote the state $s_t \in \mathbb{R}^{d_s}$, the actions $a_t \in \mathbb{R}^{d_a}$, and the reward $r(s_t, a_t)$. We say that the MBRL agent acts in an environment governed by a state transition distribution $p(s_{t+1}|s_t, a_t)$. We denote a parametric model f_θ to approximate this distribution with $p_\theta(s_{t+1}|s_t, a_t)$. MBRL follows the approach of an agent acting in its environment, learning a model of said environment, and then leveraging the model to act. While iterating over parametric control policies, the agent collects measurements of state, action, next-state and forms a dataset $\mathcal{D} = \{(s_n, a_n, s'_n)\}_{n=1}^N$. With the dynamics data $\mathcal{D}$, the agent learns the environment in the form of a neural network forward dynamics model, learning an approximate dynamics f_θ. This dynamics model is leveraged by a controller that takes in the current state s_t and returns an action sequence $a_{t:t+T}$ maximizing the expected reward $\mathbb{E}_{\pi_\theta(s_t)} \sum_{i=t}^{t+T} r(s_i, a_i)$, where T is the predictive horizon and $\pi_\theta(s_t)$ is the set of state transitions induced by the model p_θ.

In our paper, we primarily use as probabilistic neural networks designed to maximize the LL of the predicted parametric distribution p_θ, denoted as P, or ensembles of probabilistic networks denoted PE, and compare to deterministic networks minimizing the mean squared error (MSE), denoted D or DE. Unless otherwise stated we use the models as in PETS (Chua et al., 2018) with an expectation-based trajectory planner and a cross-entropy-method (CEM) optimizer.

3. Objective Mismatch and its Consequences

The Origin of Objective Mismatch: The Subtle Differences between MBRL and System Identification Many ideas and concepts in model-based RL are rooted in the field of optimal control and system identification (Sutton, 1991; Bertsekas, 1995; Zhou et al., 1996; Kirk, 2012; Bryson, 2018). In system identification (SI), the main idea is to use a two-step process where we first generate (optimal) elicitation trajectories τ to fit a dynamics model (typically analytical), and subsequently we apply this model to a specific task. This particular scheme has several assumptions:

1) the elicitation trajectories collected cover the entire state-action space; 2) the presence of virtually infinite amount of data; 3) the global and generalizable nature of the model resulting from the SI process. With these assumptions, the theme of system identification is effectively to collect a large amount of data covering the whole state-space to create a sufficiently accurate, global model that we can deploy on any desired task, and still obtain good performance. If these assumptions are true, using the closed-loop of MBRL should further improve performance over traditional open-loop SI (Hjalmarsson et al., 1996).

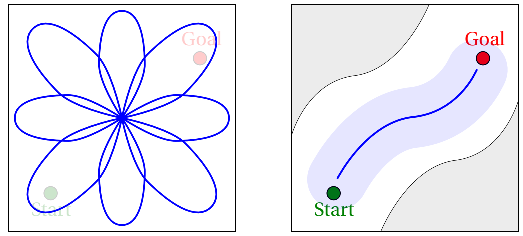


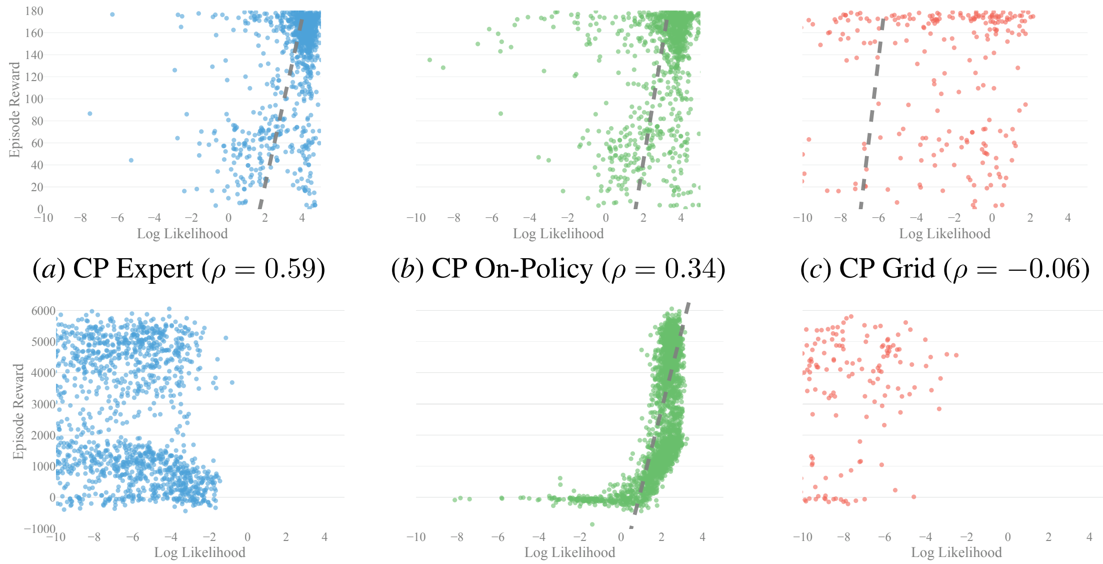
Figure 2: Sketches of state-action spaces. (*Left*) In system identification, the elicitation trajectories are designed off-line to cover the entire state-action space. (*Right*) In MBRL instead, the data collected during learning is often concentrated in trajectories towards the goal, with other parts of the state-action space being largely unexplored (grey area).

When adopting the idea of learning the dynamics model used in optimal control for MBRL, it is important to consider if these assumptions still hold. The assumption of virtually infinite data is visibly in tension with the explicit goal of MBRL which is to reduce the number of interactions with the environment by being "smart" about the sampling of new trajectories. In fact, in MBRL the offline data collection performed via elicitation trajectories is largely replaced by on-policy sampling to explicitly reduce the need to collect large amount of data (Chua et al., 2018). Moreover, in the MBRL setting the data will not usually cover the entire state-action space, since they are generated by optimizing one task. In conjunction with the use of non-parametric models, this results in learned models that are strongly biased towards capturing the distribution of the locally accurate, task-specific data. Nonetheless, this is not an immediate issue since the MBRL setting rarely tests for generalization capabilities of the learned dynamics. In practice, we can now see how the assumptions and goals of system identification are in contrast with the ones of MBRL. Understanding these differences and the downstream effects on algorithmic approach is crucial to design new families of MBRL algorithms.

Objective Mismatch During the MBRL process of iteratively learning a controller, the reward signal from the environment is diluted by the training of a forward dynamics model with a independent metric, as shown in Fig. 1. In our experiments, we highlight that the minimization of some network training cost does not hold a strong correlation to maximization of episode reward. As dynamic environments becoming increasingly complex in dimensionality, the assumptions of collected data distributions become weaker and over-fitting to different data poses an increased risk.

Formally, the problem of objective mismatch appears as two de-coupled optimization problems repeated over many cycles of learning, shown in Eq. (1a,b), which could be at the cost of minimizing the final reward. This loop becomes increasingly difficult to analyze as the dataset used for model training changes with each experimental trial – a step that is needed to include new data from previously unexplored states. In this paper we characterize the problems introduced by the interaction of these two optimization problems, but, for simplicity, we do not consider the interactions added by the changes in the dynamics-data distribution during the learning process. In addition, we discuss potential solutions, but do not make claims about the best way to do so, which is left for future work.

$$\textbf{Training:}\ \arg\max_{\theta} \sum_{i=1}^{N} \log p_\theta(s_i'|s_i, a_i), \quad \textbf{Control:}\ \arg\max_{a_{t:t+T}} \mathbb{E}_{\pi_\theta(s_t)} \sum_{i=t}^{t+T} r(s_i, a_i) \qquad \text{(1a,b)}$$



(a) CP Expert ($\rho = 0.59$) (b) CP On-Policy ($\rho = 0.34$) (c) CP Grid ($\rho = -0.06$)

(d) HC Expert ($\rho = 0.07$) (e) HC On-Policy ($\rho = 0.46$) (f) HC Sampled ($\rho = 0.19$)

Figure 3: The distribution of dynamics models ($M_{models} = 1000, 2400$ for cartpole, half cheetah) from our experiments plotting in the LL-Reward space on three datasets, with correlation coefficients ρ. Each reward point is the mean over 10 trials. There is a trend of high reward to increased LL that breaks down as the datasets contain more of the state-space than only expert trajectories.

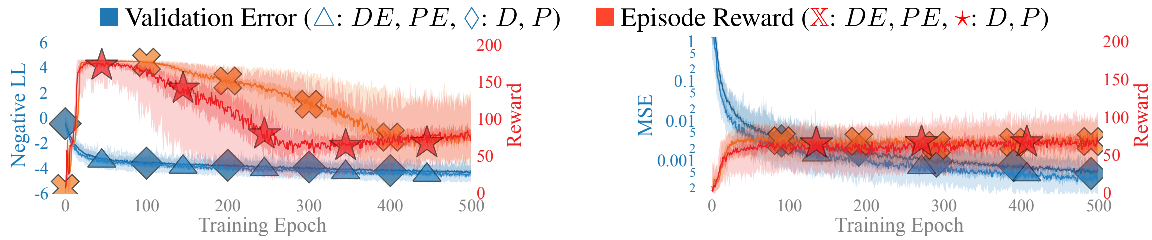
4. Identifying Objective Mismatch

We now experimentally study the issue of objective mismatch to answer the following: 1) Does the distribution of models obtained from running a MBRL algorithm show a strong correlation between LL and reward? 2) Are there signs of sub-optimality in the dynamics models training process that could be limiting performance? 3) What model differences are reflected in reward but not in LL?

Experimental Setting In our experiments, we use two popular RL benchmark tasks: the cartpole (CP) and half cheetah (HC). For more details on these tasks, model parameters, and control properties see Chua et al. (2018). We use a set of 3 different datasets to evaluate how assumptions in MBRL affect performance. We start with high-reward, expert datasets (cartpole $r > 179$, half cheetah $r > 10000$) to test if on-policy performance is linked to a minimal, optimal exploration. The two other baselines are datasets collected on-policy with the PETS algorithm and datasets of sampled tuples representative of the entire state space. The experiments validate over a) many re-trained models and b) many random seeds, to account for multiple sources of stochasticity in MBRL. Additional details and experiments can be found at: https://sites.google.com/view/mbrl-mismatch.

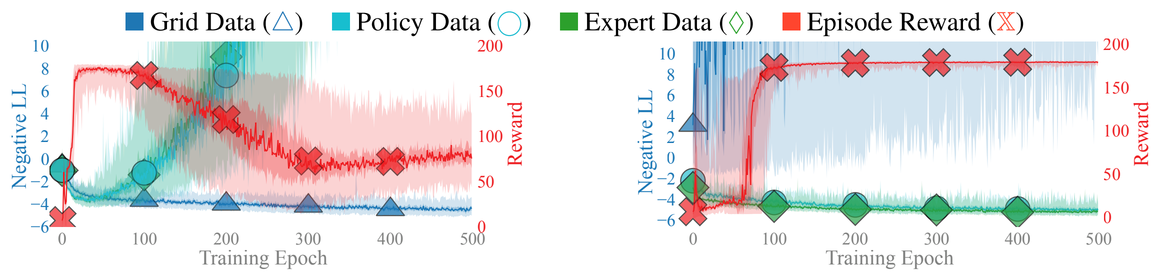
4.1. Exploration of Model Loss vs Episode Reward Space

The MBRL framework assumes a clear correlation between model accuracy and policy performance, which we challenge even in simple domains. We aggregated $M_{cp} = 1000$ cartpole models and $M_{hc} = 2400$ half cheetah models trained with PETS. The relationships between model accuracy and reward on data representing the full state-space (grid or sampled) show no clear trend in Fig. 3c,f.



(a) P, PE models. (b) D, DE models.

Figure 4: The reward when re-evaluating the controller at each dynamics model training epoch for different dynamics models, $M = 50$ per model type. Even for the simple cartpole environment, D, DE fail to achieve full performance, while P, PE reach higher performance but eventually over-fit to available data. The validation loss is still improving slowly at 500 epochs, not yet over-fitting.



(a) **Trained:** grid **Tested:** expert, on-policy (b) **Trained:** on-policy **Tested:** expert, grid

Figure 5: The effect of the dataset choice on model (P) training and accuracy in different regions of the state-space, $N = 50$ per model type. (*Left*) when training on the complete dataset, the model begins over-fitting to the on-policy data even before the performance drops in the controller. (*Right*) A model trained only on policy data does not accurately model the entire state-space. The validation loss is still improving slowly at 500 epochs in both scenarios.

The distribution of rewards versus LL shown in Fig. 3a-c shows substantial variance and points of disagreement overshadowing a visual correlation of increased reward and LL. This bi-model distribution on the half cheetah expert dataset, shown in Fig. 3d, relates to a unrecoverable failure mode in early half cheetah trials. The contrast between Fig. 3e and Fig. 3d,f shows a considerable per-dataset variation in the state-action transitions. The grid and sampled datasets, Fig. 3c,f, suffer from decreased likelihood because they do not overlap greatly with on-policy data from PETS.

If the assumptions behind MBRL were fully valid, the plots should show a perfect correlation between LL and reward. Instead these results confirm that there exists an objective mismatch which manifests as a decreased correlation between validation loss and episode reward. Hence, there is no guarantee that increasing the model accuracy (i.e., the LL) will also improve the control performance.

4.2. Model Loss vs Episode Reward During Training

This section explores how model training impacts performance at the per-epoch level. These experiments shed light onto the impact of the strong model assumptions outlined in Sec. 3. As a dynamics model is trained, there are two key inflection points - the first is the training epoch where episode reward is maximized, and the second is when error on the validation set is optimized. These

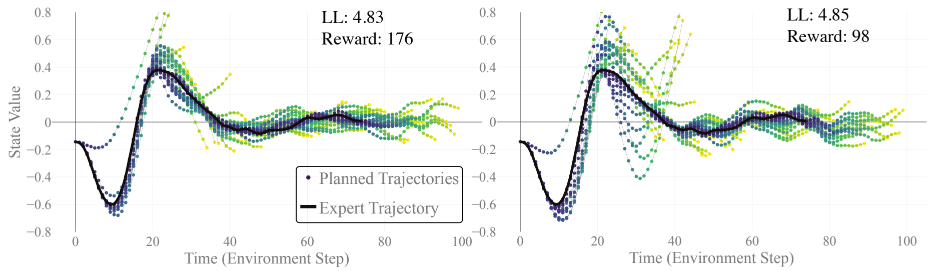


Figure 6: Example of planned trajectories along the expert trajectory for (*left*) a learned model and (*right*) the adversarially generated model trained to lower the reward. The planned control sequences are qualitatively similar except for the peak at $t = 25$. There, the adversarial attack applies a small nudge to the dynamics model parameters that significantly influences the control outcome with minimal change in terms of LL.

experiments highlight the disconnect between three practices in MBRL a) the assumption that the on-policy dynamics data can express large portions of the state-space, b) the idea that simple neural networks can satisfactorily capture complex dynamics, c) and the practice that model training is a simple optimization problem disconnected from reward. Note that in the figures of this section we use Negative Log-Likelihood (NLL) instead of LL, to reduce visual clutter.

For the grid cartpole dataset, Fig. 4 shows that the reward is maximized at a drastically different time than when validation loss is minimized for P, PE models. Fig. 5 highlights how the trained models are able to represent other datasets than they are trained on (with additional validation errors). Fig. 5b shows that on-policy data will not lead to a complete dynamics understanding because the grid validation data rapidly diverges. When training on grid data, the fact that the on-policy data diverges in Fig. 5a before the reward decreases is encouraging as objective mismatch may be preventable in simple tasks. Similar experiments on half cheetah are omitted because models for this environment are trained incrementally on aggregated data rather then fully on each dataset (Chua et al. (2018)).

4.3. Decoupling Model Loss from Controller Performance

We now study how differences in dynamics models – *even if they have similar LLs* – are reflected in control policies to show that a accurate dynamics model does not guarantee performance.

Adversarial attack on model performance We performed an adversarial attack (Szegedy et al., 2013) on a deep dynamics model so that it attains a high likelihood but low reward. Specifically, we fine-tune the deep dynamics model's last layer with a zeroth-order optimizer, CMA-ES, (the cumulative reward is non-differentiable) to lower reward with a large penalty if the validation likelihood drops. As a starting point for this experiment we sampled a P dynamics model from the last trial of a PETS run on cartpole. This model achieves reward of 176 and has a LL of 4.827 on it's on-policy validation dataset. Using CMA-ES, we reduced the on-policy reward of the model to 98, on 5 trials, while slightly improving the LL; the CMA-ES convergence is shown in Fig. 7 and the difference between the two models is visualized in Fig. 6. Fine tuning of all model parameters would be more likely to find sub-optimal performing controllers because the output layer

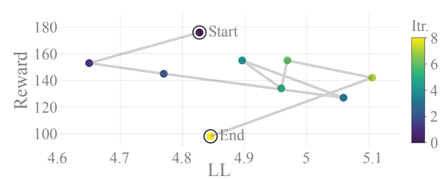


Figure 7: Convergence of the CMA-ES population's best member.

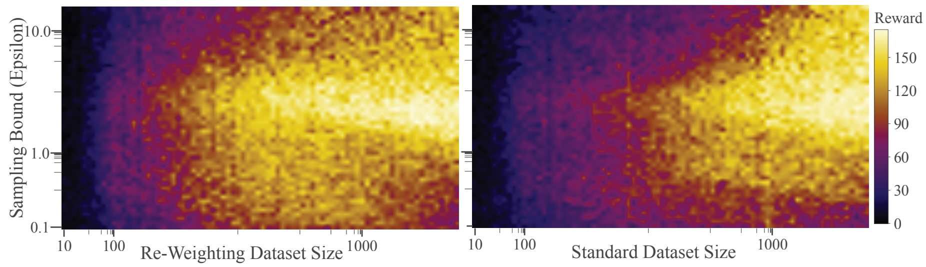


Figure 8: Mean reward of PETS trials ($N_{trials} = 100$), with (*left*) and without (*right*) model re-weighting, on a log-grid of dynamics model training sets with number of points S $\in [10, 2500]$ and sampling optimal-distance bounds $\epsilon \in [.28, 15.66]$. The re-weighting improves performance for smaller dataset sizes, but suffers from increased variance in larger set sizes. The performance of PETS declines when the dynamics model is trained on points too near to the optimal trajectory because the model lacks robustness when running online with the stochastic MPC.

consists of about 1% of the total parameters. This experiment shows that the model parameters that achieve a low model loss inhabit a broader space than the subset that also achieves high reward.

5. Addressing Objective Mismatch During Model Training

Tweaking dynamics model training can partially mitigate the problem of objective mismatch. Taking inspiration from imitation learning, we propose that the learning capacity of the model would be most useful when accurately modeling the dynamics along trajectories that are relevant for the task at hand, while maintaining knowledge of nearby transitions for robustness under a stochastic controller. Intuitively, it is more important to model accurately the dynamics along the optimal trajectory, rather than modeling part of the state-action space that might never be visited to solve the task. For this reason, we now propose a model loss aimed at alleviating this issue.

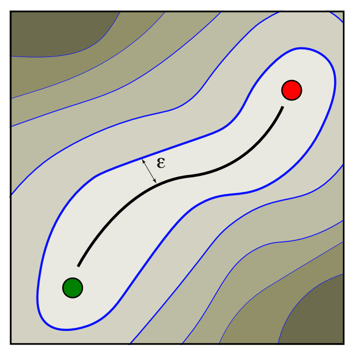


Figure 9: We propose to re-weight the loss of the dynamics model w.r.t. the distance ϵ from the optimal trajectory.

Given a element of a state space (s_i, a_i), we quantify the distance of any two tuples, $d_{i,j}$. With this distance, we re-weight the loss, $l(y)$, of points further from the optimal policy to be lower, so that points in the optimal trajectory get a weight $\omega(y) = 1$, and points at the edge of the grid dataset used in Sec. 4 get a weight $\omega(y) = 0$. Using the expert dataset discussed in Sec. 4 as a distance baseline, we generated $25e6$ tuples of (s, a, s') by uniformly sampling across the state-action space of cartpole. We sorted this data by taking the minimum orthogonal distance, d^*, from each of the points to the 200 elements in the expert trajectory. To create different datasets that range from near-optimal to near-global, we vary the distance bound ϵ, and number of training points, S. This simple form of re-weighting the neural network loss, shown in Eq. (2a,b,c), demonstrated an improvement in sample efficiency to learn the cartpole task, as seen in Fig. 8. Unfortunately, this approach is impractical when the optimal trajectory is not known in advance. However, future work could develop an iterative method to jointly estimate and re-weight samples in an online training method to address objective mismatch.

$$\textbf{Weighting}\ \ \omega(y) = ce^{-d^*(y)} \qquad \textbf{Standard}\ \ l(\hat{y}, y) \qquad \textbf{Re-weight}\ \ l(\hat{y}, y) \cdot \omega(y) \qquad \text{(2a,b,c)}$$

6. Discussion, Related Work, and Future Work

Objective mismatch impacts the performance of MBRL – our experiments have gone deeper into this fragility. Beyond the re-weighting of the LL presented in Sec. 5, here we summarize and discuss other relevant works in the community.

Learning the dynamics model to optimize the task performance Most relevant are research directions on controllers that directly connect the reward signal back to the controller. In theory, this exactly solves the model mismatch problem, but in practice the current approaches have proven difficult to scale to complex systems. One way to do this is by designing systems that are fully differentiable and backpropagating the task reward through the dynamics. This has been investigated with differentiable MPC (Amos et al., 2018) and Path Integral control (Okada et al., 2017), Universal Planning Networks (Srinivas et al., 2018) propose a differentiable planner that unrolls gradient descent steps over the action space of a planning network. Bansal et al. (2017) use a zero-order optimizer to maximize the controller's performance without having to compute gradients explicitly.

Add heuristics to the dynamics model structure or training process to make control easier If it is infeasible or intractable to shape the dynamics of a controller, an alternative is to add heuristics to the training process of the dynamics model. These heuristics can manifest in the form of learning a latent space that is locally linear, *e.g.*, in Embed to Control and related methods (Watter et al., 2015), by enforcing that the model makes long-horizon predictions (Ke et al., 2019), ignoring uncontrollable parts of the state space (Ghosh et al., 2018), detecting and correcting when a predictive model steps off the manifold of reasonable states (Talvitie, 2017), adding reward signal prediction on top of the latent space Gelada et al. (2019), or adding noise when training transitions Mankowitz et al. (2019). Farahmand et al. (2017); Farahmand (2018) also attempts to re-frame the transitions to incorporate a notion of the downstream decision or reward. Finally, Singh et al. (2019) proposes stabilizability constraints to regularize the model and improve the control performance. None of these paper formalize or explore the underlying mismatch issue in detail.

Continuing Experiments Our experiments represent an initial exploration into the challenges of objective mismatch in MBRL. Sec. 4.2 is limited to cartpole due to computational challenges of training with large dynamics datasets and Sec. 4.3 could be strengthened by defining quantitative comparisons in controller performance. Additionally, these effects should be quantified in other MBRL algorithms such as MBPO (Janner et al., 2019) and POPLIN (Wang and Ba, 2019).

7. Conclusion

This paper identifies, formalizes and analyzes the issue of objective mismatch in MBRL. This fundamental disconnect between the likelihood of the dynamics model, and the overall task reward emerges from incorrect assumptions at the origins of MBRL. Experimental results highlight the negative effects that objective mismatch has on the performance of a current state-of-the-art MBRL algorithm. In providing a first insight on the issue of objective mismatch in MBRL, we hope future work will deeply examine this issue to overcome it with a new generation of MBRL algorithms.

Acknowledgments

We thank Rowan McAllister and Kurtland Chua for useful discussions.

References

Brandon Amos, Ivan Jimenez, Jacob Sacks, Byron Boots, and J Zico Kolter. Differentiable MPC for end-to-end planning and control. In *Neural Information Processing Systems*, pages 8289–8300, 2018.

S. Bansal, Roberto Calandra, T. Xiao, S. Levine, and C. J. Tomlin. Goal-driven dynamics learning via Bayesian optimization. In *IEEE Conference on Decision and Control (CDC)*, pages 5168–5173, 2017. doi: 10.1109/CDC.2017.8264425.

Dimitri P Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena scientific Belmont, MA, 1995.

Arthur Earl Bryson. *Applied optimal control: optimization, estimation and control*. Routledge, 2018.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *Neural Information Processing Systems*, pages 4754–4765, 2018.

Marc P. Deisenroth and Carl E. Rasmussen. PILCO: A Model-Based and Data-Efficient Approach to Policy Search. In *International Conference on Machine Learning (ICML)*, pages 465–472, 2011.

Amir-massoud Farahmand. Iterative value-aware model learning. In *Advances in Neural Information Processing Systems*, pages 9072–9083, 2018.

Amir-massoud Farahmand, Andre Barreto, and Daniel Nikovski. Value-aware loss function for model-based reinforcement learning. In *Artificial Intelligence and Statistics*, pages 1486–1494, 2017.

Carles Gelada, Saurabh Kumar, Jacob Buckman, Ofir Nachum, and Marc G Bellemare. Deepmdp: Learning continuous latent space models for representation learning. *arXiv preprint arXiv:1906.02736*, 2019.

Dibya Ghosh, Abhishek Gupta, and Sergey Levine. Learning actionable representations with goal-conditioned policies. *arXiv preprint arXiv:1811.07819*, 2018.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. *arXiv preprint arXiv:1811.04551*, 2018.

Håkan Hjalmarsson, Michel Gevers, and Franky De Bruyne. For model-based control design, closed-loop identification gives better performance. *Automatica*, 32(12):1659–1673, 1996.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *arXiv preprint arXiv:1906.08253*, 2019.

Nan Rosemary Ke, Amanpreet Singh, Ahmed Touati, Anirudh Goyal, Yoshua Bengio, Devi Parikh, and Dhruv Batra. Learning dynamics model in reinforcement learning by incorporating the long term future. *arXiv preprint arXiv:1903.01599*, 2019.

Donald E Kirk. *Optimal control theory: an introduction*. Courier Corporation, 2012.

Daniel J Mankowitz, Nir Levine, Rae Jeong, Abbas Abdolmaleki, Jost Tobias Springenberg, Timothy Mann, Todd Hester, and Martin Riedmiller. Robust reinforcement learning for continuous control with model misspecification. *arXiv preprint arXiv:1906.07516*, 2019.

Masashi Okada, Luca Rigazio, and Takenobu Aoshima. Path integral networks: End-to-end differentiable optimal control. *arXiv preprint arXiv:1706.09597*, 2017.

Sumeet Singh, Spencer M. Richards, Vikas Sindhwani, Jean-Jacques E. Slotine, and Marco Pavone. Learning stabilizable nonlinear dynamics with contraction-based regularization, 2019.

Aravind Srinivas, Allan Jabri, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Universal planning networks. *arXiv preprint arXiv:1804.00645*, 2018.

Richard S Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *ACM Sigart Bulletin*, 2(4):160–163, 1991. doi: 10.1145/122344.122377.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Erik Talvitie. Self-correcting models for model-based reinforcement learning. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Tingwu Wang and Jimmy Ba. Exploring model-based planning with policy networks. *arXiv preprint arXiv:1906.08649*, 2019.

Manuel Watter, Jost Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In *Neural information processing systems*, pages 2746–2754, 2015.

Grady Williams, Nolan Wagener, Brian Goldfain, Paul Drews, James M Rehg, Byron Boots, and Evangelos A Theodorou. Information theoretic MPC for model-based reinforcement learning. In *International Conference on Robotics and Automation (ICRA)*, pages 1714–1721, 2017.

Marvin Zhang, Sharad Vikram, Laura Smith, Pieter Abbeel, Matthew J. Johnson, and Sergey Levine. Solar: Deep structured latent representations for model-based reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2019.

Kemin Zhou, John Comstock Doyle, Keith Glover, et al. *Robust and optimal control*, volume 40. Prentice hall New Jersey, 1996.