Maximum Entropy Model Rollouts: Fast Model Based Policy Optimization without Compounding Errors

Chi Zhang [1] **Sanmukh Rao Kuppannagari** [2] **Viktor K Prasanna** [2]

Abstract

Model usage is the central challenge of model-based reinforcement learning. Although dynamics model based on deep neural networks provide good generalization for single step prediction, such ability is over exploited when it is used to predict long horizon trajectories due to compounding errors. In this work, we propose a Dyna-style model-based reinforcement learning algorithm, which we called Maximum Entropy Model Rollouts (MEMR). To eliminate the compounding errors, we only use our model to generate single-step rollouts. Furthermore, we propose to generate *diverse* model rollouts by non-uniform sampling of the environment states such that the entropy of the model rollouts is maximized. We mathematically derived the maximum entropy sampling criteria for one data case under Gaussian prior. To accomplish this criteria, we propose to utilize a prioritized experience replay. Our preliminary experiments in challenging locomotion benchmarks show that our approach achieves the same sample efficiency of the best model-based algorithms, matches the asymptotic performance of the best model-free algorithms, and significantly reduces the computation requirements of other model-based methods.

1. Introduction

Model-based reinforcement learning (MBRL) (Janner et al., 2019; Buckman et al., 2018; Xu et al., 2018; Chua et al., 2018) shows competitive performance compared with best model-free reinforcement learning (MFRL) algorithms (Schulman et al., 2017; 2015; Mnih et al., 2013; Haarnoja et al., 2018a;b) with significantly fewer environment samples on challenging robotics locomotion benchmarks (Todorov et al., 2012). A MFRL algorithm learns complex skills by maximizing a scalar reward designed by human engineering. However, to obtain promising performance a large number of environment interactions are needed which may take a long time in real-world applications. In such cases, MBRL is appealing due to its superior sample efficiency that relies on the generalization of a learned predictive dynamics model. However, the quality of the policy trained on imagined trajectories is often worse asymptotically than the best MFRL counterparts due to the imperfect models.

Recently, (Janner et al., 2019) proposed Model-based Policy Optimization (MBPO), including a theoretical framework that encourages short-horizon model usage based on an optimistic assumption of a bounded model generalization error given policy shift. Although empirical studies have shown support evidence, this property is hard to guarantee in the whole state distribution. Moreover, *uniform* sampling of the environment states to generate branched model rollouts degrades the *diversity* of the model dataset, especially when the policy shift is small, which makes the policy updates inefficient.

Our main contribution is a practical algorithm, which we called Maximum Entropy Model Rollouts (MEMR) based on the aforementioned insights. The differences between MEMR and MBPO are: 1) MEMR follows Dyna (Sutton, 1991) that only generates single-step model rollouts while MBPO encourages generating short-horizon model rollouts. The generalization ability of MEMR is strictly guaranteed by supervised machine learning theory, which can be empirically estimated by validation errors (Shalev-Shwartz & Ben-David, 2014). 2) MEMR utilizes a prioritized experience replay (Schaul et al., 2015) to generate *max-diversity* model rollouts for efficient policy updates. We validate this idea on challenging locomotion benchmarks (Todorov et al., 2012) and the experimental results show that MEMR matches the asymptotic performance and sample efficiency of MBPO (Janner et al., 2019) while significantly reducing the number of policy updates and model rollouts leading to faster learning speed.

[1]Department of Computer Science, University of Southern California, Los Angeles, CA [2]Department of Electrical and Computer Engineering, University of Southern California, Los Angeles, CA. Correspondence to: Chi Zhang <zhan527@usc.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning BIG Workshop, Vienna, Austria, PMLR 108, 2020.

2. Preliminaries

Reinforcement Learning algorithms aim to solve Markov Decision Process (MDP) with unknown dynamics. A Markov decision process (MDP) (Sutton & Barto, 2018) is defined as a tuple $< \mathcal{S}, \mathcal{A}, R, P, \mu >$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, $R(s, a, s') : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ defines the intermediate reward when the agent transits from state s to s' by taking action a, $P(s'|s, a) : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ defines the probability when the agent transits from state s to s' by taking action a, $\mu : \mathcal{S} \to [0, 1]$ defines the starting state distribution. The objective of reinforcement learning is to select policy $\pi : \mu \to P(A)$ such that

$$J(\pi) = \mathop{\mathbb{E}}_{\substack{s_0 \sim \mu, a_t \sim \pi(\cdot|s_t) \\ s_{t+1} \sim P(\cdot|s_t, a_t)}} [\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1})] \quad (1)$$

is maximized.

2.1. Prioritized Experience Replay

Prioritized experience replay (Schaul et al., 2015) is introduced to increase the learning efficiency of DQN (Mnih et al., 2013), where the probability of each transition is proportional to the absolute TD error (Watkins & Dayan, 1992). To avoid overfitting, stochastic prioritization is utilized and the bias is corrected via annealed importance sampling. In this work, we adopt the same idea with a custom prioritization criteria such that the joint entropy of the state and action in the model dataset is maximized.

2.2. Model-based Policy Optimization

Model-based policy optimization (MBPO) (Janner et al., 2019) achieves state-of-the-art sample efficiency and matches the asymptotic performance of MFRL approaches. MBPO optimizes a policy with soft actor-critic (SAC) (Haarnoja et al., 2018a) under the data distribution collected by unrolling the learned dynamics model using the current policy. However, the sample efficiency comes at the cost of 2.5x to 5x increased number of policy updates compared with SAC (Haarnoja et al., 2018a) and a large number of model rollouts, that significantly decreases the training speed. To mitigate this bottleneck, we analyze the model usage and model rollout distribution and propose insights on how to improve MBPO to obtain better computation efficiency.

Model usage. In MBPO, learned dynamics model is used to generate branched model rollouts with short horizons (Janner et al., 2019). Although (Janner et al., 2019) presented theoretical analysis to bound the policy performance trained using model generate rollouts, the over exploitation of model generalization can't be eliminated. In this work, *one of our core idea is that we only rely on learned model to generate one-step rollouts, in which case we interpret it as model-based exploration.* The nice property of this model usage is the natural bounded model generalization error, which can be estimated in practice by the validation dataset (Shalev-Shwartz & Ben-David, 2014).

Model rollout distribution. Uniform sampling of true states [1] to generate model rollouts is adopted in MBPO (Janner et al., 2019). This potentially generates large amount of similar data when the policy and the learned model changes slowly as training progresses. As result, the efficiency of the policy updates is deteriorated. In this work, *we propose to sample true states to generate single-step model rollouts such that the joint entropy of the state and action of the model dataset is maximized.* The intuition is to increase the "diversity" of the model dataset, from which the policy can benefit for efficient learning.

3. Maximum Entropy Model Rollouts

In this section, we unveil the technical details of our Maximum Entropy Model Rollouts (MEMR) for model based policy optimization. First, we propose the Maximum Entropy Sampling Theorem to help understand the choice of our prioritization criteria. Based on the theoretical analysis, we propose a practical implementation of this idea and discuss the challenges posed by runtime complexity along with their fixes.

3.1. Maximum Entropy Sampling Criteria

We begin by considering the following problem definition:

Problem 3.1 (Maximum Entropy Sampling)**.** *Let $\mathcal{D}_{env} = \{s_i\}_{i=1}^{N_{env}}$ be the collection of all the states in the environment dataset. Let $\mathcal{D}_{model} = \{(s, a)\}_{j=1}^{N_{model}}$ be the collection of all the state-action pairs in the model dataset*[2]*. Assume for each state in $\mathcal{D}_{env}$, we sample action $a_i \sim \pi_\phi(\cdot|s_i)$ using the current policy denoted as $\mathcal{D}_{sample} = \{(s_i, a_i)\}_{i=1}^{N_{env}}$. Assume we parameterize the policy distribution derived from the model dataset as a Gaussian distribution with diagonal covariance: $\pi_\psi(a_i|s_i) = \mathcal{N}(\mu_\psi(s_i), \Sigma_\psi(s_i))$. Let the joint entropy of the state-action in the model dataset be $H(S, A)$. Now we select (s_k, a_k) from D_{sample} and add it to the $\mathcal{D}_{model}$. Let the joint entropy of the new $\mathcal{D}'_{model} = \mathcal{D}_{model} \cup \{(s_k, a_k)\}$ be $H(S', A')$, the optimal sampling criteria problem is to choose index k such that $H(S', A') - H(S, A)$ is maximized.*

Theorem 3.1 (Maximum Entropy Sampling Theorem)**.** *Assume $N_{model} \gg 1$ such that the state distribution of $\mathcal{D}_{model}$*

[1] States encountered in real environment as opposed to imagined states that are generated by the model.

[2] The tasks considered in this work are deterministic so we omit s' for simplicity.

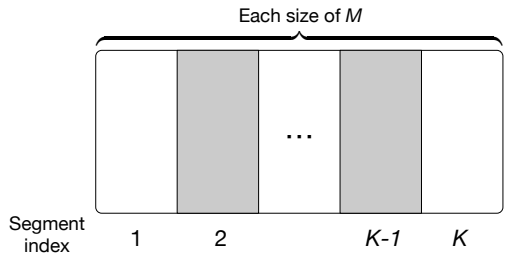

Figure 1. Segmented replay buffer for model generated rollouts. Each segment contains data sampled from the same environment state distribution.

and $\mathcal{D}'_{model}$ are identical, then

$$k = \arg\min_i \log(\sqrt{2\pi}\pi_\psi(a_i|s_i)\sigma(\pi_\psi(\cdot|s_i))) \tag{2}$$

where $\pi_\psi(a_i|s_i)$ is the probability of model data policy at (s_i, a_i), $\sigma(\pi_\psi(\cdot|s_i))$ is the standard deviation of the conditional distribution at s_i.

Proof. See Appendix A, Theorem A.2. □

3.2. Practical Implementation

Theorem 3.1 provides a mathematically justified criteria to select states from the environment dataset for rollout generation to maximize the "diversity" of the model dataset, yet it poses several practical challenges to implement: 1) It requires a full sweep of all the states in the environment dataset before each sampling, which is $O(N_{env})$. This is problematic because N_{env} grows linearly as training progresses. 2) Stochastic gradient descent assumes uniform sampling of the data distribution whereas prioritized sampling breaks this assumption and introduces bias. 3) Training the model data distribution to converge is expensive but crucial before evaluating the priority. A complete algorithm that handles the aforementioned practical challenges is presented in Algorithm 1.

Stochastic prioritization. Inspired by (Schaul et al., 2015), we only update the priority of the states that are just sampled to avoid an expensive full sweep before each sampling. An immediate consequence of this approach is that certain states with low priorities will not be sampled for a very long time. This potentially leads to overfitting. Following (Schaul et al., 2015), we use stochastic prioritization that interpolates between pure greedy and uniform sampling with the following probability of sampling state i:

$$P(i) = \frac{p_i^\alpha}{\sum_k p_k^\alpha} \tag{3}$$

where $p_i \geq 0$ is the priority of state and action i. The exponent α determines how much prioritization is used,

Algorithm 1 Maximum Entropy Model Rollouts for Model-Based Policy Optimization

Initialize environment dataset $\mathcal{D}_{\text{env}}$ and model dataset $\mathcal{D}_{\text{model}}$
Initialize SAC policy π_ϕ, predictive model p_θ and model derived policy distribution π_ψ
for $t = 1$: total_num_steps **do**
  if $t\%$model_update_freq $== 0$ **then**
    Train model p_θ on $\mathcal{D}_{\text{env}}$ via maximum likelihood
  end if
  Sample $a_t \sim \pi_\phi(\cdot|s_t)$; Execute a_t in the environment and observe s_{t+1}
  Compute priority p_t according to Equation 4; add (s_t, a_t, s_{t+1}, p_t) to $\mathcal{D}_{\text{env}}$
  for $j = 1 : M$ **do**
    Sample $s_j \sim P(j) = p_j^\alpha / \sum_i p_j^\alpha$ from $\mathcal{D}_{\text{env}}$
    Compute importance-sampling weight $w_j = (N \cdot P(j))^{-\beta} / \max_i w_i$
    Sample $a_j \sim \pi_\phi(\cdot|s_j)$; Perform one-step rollout using p_θ and obtain $\hat{s}'_j$.
  end for
  Add $\{(s_j, a_j, \hat{s}'_j, w_j)\}_{j=1}^M$ to the next segment in $\mathcal{D}_{\text{model}}$
  Update π_ψ on $\{(s_j, a_j)\}_{j=1}^M$ via maximum likelihood for D epochs
  Update the priority of s_j according to Equation 4 for all j
  for G iterations **do**
    Sample segment index k uniformly; Sample batch size B from segment k uniformly
    Update Q network as $\phi_Q \leftarrow \phi_Q - \lambda_\pi \frac{1}{B} \sum_{i=1}^B w_i \cdot \nabla_{\phi_Q} J_\pi(\phi_Q, i)$
    Update policy using $J_\pi(\phi) = \frac{1}{B} \sum_{i=1}^B [D_{KL}(\pi || \exp\{Q^\pi - V^\pi\})]$
  end for
end for

with $\alpha = 0$ corresponding to the uniform case. According to Theorem 3.1, we compute p_i as

$$p_i = -\log(\sqrt{2\pi}\pi_\psi(a_i|s_i)\sigma(\pi_\psi(\cdot|s_i))) \tag{4}$$

Correcting the bias. Using prioritized sampling introduces bias when fitting the Q network of the SAC. Inspired by (Schaul et al., 2015), we apply weighted importance-sampling (IS) when calculating the loss of the Q network, where the weight for sample i is

$$w_i = \left(\frac{1}{N} \cdot \frac{1}{P(i)}\right)^\beta \tag{5}$$

Segmented replay buffer. According to Algorithm 1, we update the priority after sampling states from the environment dataset to perform model rollouts. Thus, the sampling

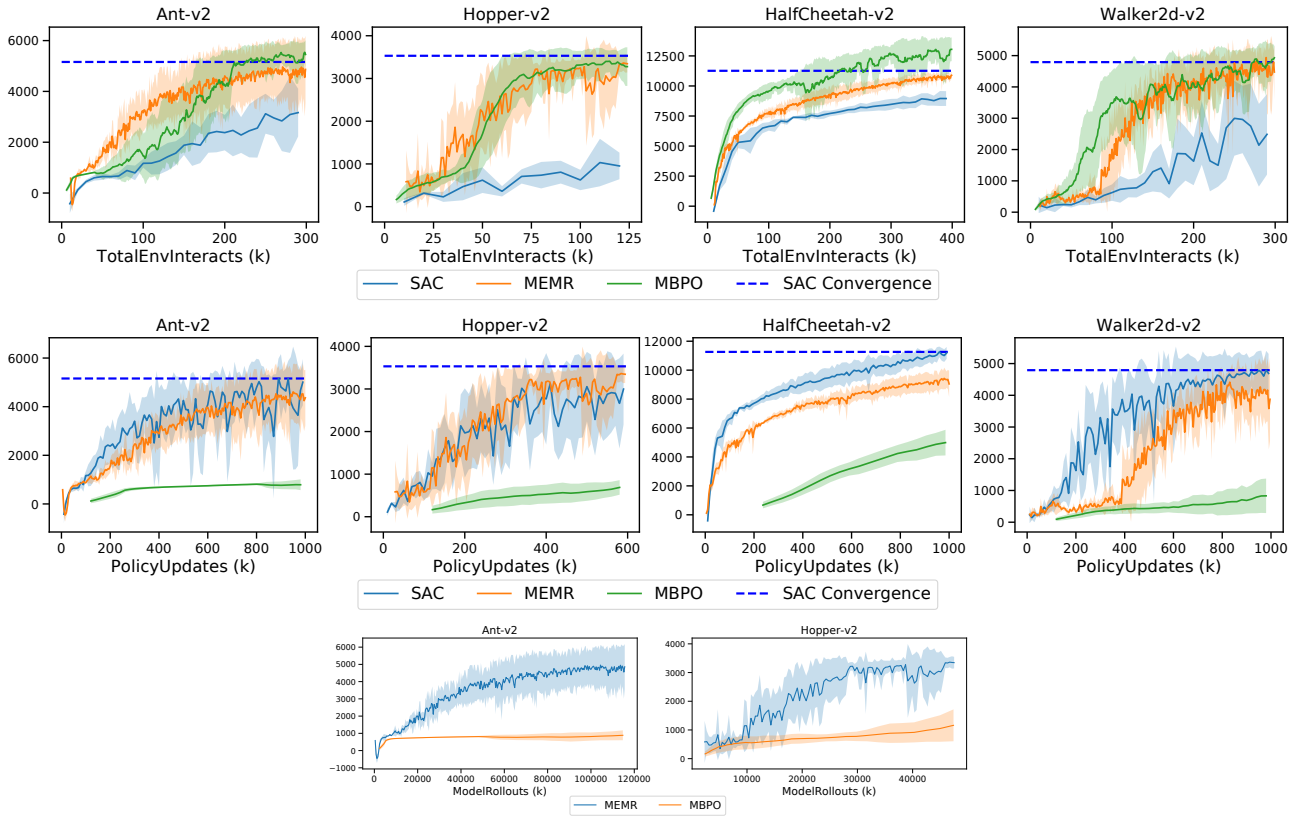

Figure 2. Training curves of MEMR and two baselines. Solid curves depict the mean of five trials and shaded regions correspond to standard deviation among trials. The first row depicts the performance vs. the total number of environment interactions. We observe that MEMR matches the performance of state-of-the-art model-based and model-free algorithms. The second row shows the performance vs. the number of policy updates and we observe that MEMR converges as fast as SAC in terms of the number of updates. The third row shows that MEMR generates only a fraction of model rollouts compared to MBPO, which indicates far less training time.

distribution of every M model rollout generation is different. This leads to incorrect importance weights if we randomly sample a batch from the model dataset that contains data generated from different distributions to perform policy updates. To fix it, we introduce segmented replay buffer that group every M rollouts in the same segment. During sampling for policy updates, we randomly sample a segment index, then sample a batch from that segment.

Training model derived policy distribution. Fitting π_ψ using $\mathcal{D}_{\text{model}}$ via maximum likelihood to converge is costly since the size of $\mathcal{D}_{\text{model}}$ is large and this operation must be performed every time we generate model rollouts. Since the data in model buffer is swapped rapidly, we treat it as an online learning procedure and only perform several gradient updates on the newly stored data.

4. Experiments

Our experimental evaluation aims to study the following questions: How well does MEMR perform on RL benchmarks, compared to state-of-the-art model-based and model-free algorithms in terms of sample efficiency, asymptotic performance and computation efficiency?

We evaluate MEMR on Mujoco benchmarks (Todorov et al., 2012). We compare our method with the state-of-the-art model-based method, MBPO (Janner et al., 2019). As shown in Figure 2, MEMR matches the asymptotic performance of MBPO whereas MEMR only uses $1/4$ policy updates and a fraction of model rollouts. It indicates that MEMR is more efficient in terms of model rollouts data used for policy updates. It also indicates orders of training speedup. Compared with the state-of-the-art model-free method, SAC (Haarnoja et al., 2018a), MEMR matches the asymptotic performance and the data efficiency.

Acknowledgements

This work has been sponsored by the U.S. Army Research Office (ARO) under award number W911NF1910362 and the U.S. National Science Foundation (NSF) under award number 1911229.

References

Buckman, J., Hafner, D., Tucker, G., Brevdo, E., and Lee, H. Sample-efficient reinforcement learning with stochastic ensemble value expansion. *CoRR*, abs/1807.01675, 2018. URL http://arxiv.org/abs/1807.01675.

Chua, K., Calandra, R., McAllister, R., and Levine, S. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *CoRR*, abs/1805.12114, 2018. URL http://arxiv.org/abs/1805.12114.

de Boer, P.-T., Kroese, D. P., Mannor, S., and Rubinstein, R. Y. A tutorial on the cross-entropy method. *ANNALS OF OPERATIONS RESEARCH*, 134, 2004.

Deisenroth, M. P. and Rasmussen, C. E. Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pp. 465–472, Madison, WI, USA, 2011. Omnipress. ISBN 9781450306195.

Feinberg, V., Wan, A., Stoica, I., Jordan, M. I., Gonzalez, J. E., and Levine, S. Model-based value estimation for efficient model-free reinforcement learning. *CoRR*, abs/1803.00101, 2018. URL http://arxiv.org/abs/1803.00101.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *CoRR*, abs/1801.01290, 2018a. URL http://arxiv.org/abs/1801.01290.

Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., Kumar, V., Zhu, H., Gupta, A., Abbeel, P., and Levine, S. Soft actor-critic algorithms and applications. *CoRR*, abs/1812.05905, 2018b. URL http://arxiv.org/abs/1812.05905.

Janner, M., Fu, J., Zhang, M., and Levine, S. When to trust your model: Model-based policy optimization. *CoRR*, abs/1906.08253, 2019. URL http://arxiv.org/abs/1906.08253.

Kaiser, L., Babaeizadeh, M., Milos, P., Osinski, B., Campbell, R. H., Czechowski, K., Erhan, D., Finn, C., Kozakowski, P., Levine, S., Sepassi, R., Tucker, G., and Michalewski, H. Model-based reinforcement learning for atari. *CoRR*, abs/1903.00374, 2019. URL http://arxiv.org/abs/1903.00374.

Kakade, S. A natural policy gradient. In *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, NIPS'01, pp. 1531–1538, Cambridge, MA, USA, 2001. MIT Press.

Kurutach, T., Clavera, I., Duan, Y., Tamar, A., and Abbeel, P. Model-ensemble trust-region policy optimization. *CoRR*, abs/1802.10592, 2018. URL http://arxiv.org/abs/1802.10592.

Levine, S. and Koltun, V. Guided policy search. In Dasgupta, S. and McAllester, D. (eds.), *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pp. 1–9, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR. URL http://proceedings.mlr.press/v28/levine13.html.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. A. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013. URL http://arxiv.org/abs/1312.5602.

Nagabandi, A., Kahn, G., Fearing, R. S., and Levine, S. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. *CoRR*, abs/1708.02596, 2017. URL http://arxiv.org/abs/1708.02596.

Rao, A. A survey of numerical methods for optimal control. *Advances in the Astronautical Sciences*, 135, 01 2010.

Schaul, T., Quan, J., Antonoglou, I., and Silver, D. Prioritized experience replay, 2015. URL http://arxiv.org/abs/1511.05952. cite arxiv:1511.05952Comment: Published at ICLR 2016.

Schulman, J., Levine, S., Moritz, P., Jordan, M. I., and Abbeel, P. Trust region policy optimization. *CoRR*, abs/1502.05477, 2015. URL http://arxiv.org/abs/1502.05477.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL http://arxiv.org/abs/1707.06347.

Shalev-Shwartz, S. and Ben-David, S. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, USA, 2014. ISBN 1107057132.

Sutton, R. S. Dyna, an integrated architecture for learning, planning, and reacting. *SIGART Bull.*, 2(4): 160–163, July 1991. ISSN 0163-5719. doi: 10.1145/122344.122377. URL https://doi.org/10.1145/122344.122377.

Sutton, R. S. and Barto, A. G. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA, 2018. ISBN 0262039249.

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, 2012.

Wang, T. and Ba, J. Exploring model-based planning with policy networks. *CoRR*, abs/1906.08649, 2019. URL http://arxiv.org/abs/1906.08649.

Watkins, C. J. C. H. and Dayan, P. Q-learning. *Machine Learning*, 8(3):279–292, 1992. doi: 10.1007/BF00992698. URL https://doi.org/10.1007/BF00992698.

Williams, G., Aldrich, A., and Theodorou, E. A. Model predictive path integral control using covariance variable importance sampling. *CoRR*, abs/1509.01149, 2015. URL http://arxiv.org/abs/1509.01149.

Xu, H., Li, Y., Tian, Y., Darrell, T., and Ma, T. Algorithmic framework for model-based reinforcement learning with theoretical guarantees. *CoRR*, abs/1807.03858, 2018. URL http://arxiv.org/abs/1807.03858.