1 Background

1. η denotes the expected sum of discounted rewards.
2. The goal of RL is to find:

$$\pi^* = \underset{\pi}{\operatorname{argmax}} \eta[\pi] = \underset{\pi}{\operatorname{argmax}} E_\pi \left[\sum_{t=0}^{\infty} \gamma^t r\left(s_t, a_t\right) \right]$$

3. The dynamics is denoted by $p\left(s'|s,a\right)$ and the learned dynamics function is denoted by $p_\theta\left(s'|s,a\right)$.

2 Monotonic Improvement with Model Bias

2.1 Monotonic Model-based Improvement

1. They want to construct a bound of the form $\eta[\pi] \geq \hat{\eta}[\pi] - C$.
2. This bound guarantees that as long as we improve by at least C under the model, we can guarantee improvement on the true MDP.
3. They argue that the true returns and model returns, C, can be expressed in terms of two error quantities of the model: generalization error due to sampling, and distribution shift due to the updated policies encountering states not seen during training.
4. To quantify the sampling error, they use standard PAC generalization bounds, which bound the difference in expected loss and empirical loss by a constant with high probability.
5. The generalization error is denoted by $\epsilon_m = \max_t E_{s \sim \pi_{D,t}} \left[D_{TV}\left(p\left(s', r|s,a\right) \| p_\theta\left(s', r|s,a\right)\right)\right]$.
6. The error is estimated in practice by measuring the validation loss of the model.
7. The distribution shift is denoted by the maximum total-variation distance: $\max_s D_{TV}\left(\pi || \pi_D\right) \leq \epsilon_\pi$.
8. By assuming that the expected TV-distance between 2 transition distributions and the policy are bounded, they proved that the true returns and the model returns are related as:

$$\eta[\pi] \geq \hat{\eta}[\pi] - \underbrace{\left[\frac{2\gamma r_{\max}\left(\epsilon_m + 2\epsilon_\pi\right)}{(1-\gamma)^2} + \frac{4 r_{\max}\epsilon_\pi}{(1-\gamma)} \right]}_{C(\epsilon_m, \epsilon_\pi)}$$

2.2 Interpolating model-based and model-free updates

1. They argue that the previous analysis relies on running full rollouts through the model, allowing model errors to compound.
2. They introduce the notion of a branched rollout, where the rollout is started from a state under the previous policy state distribution and run k step according to the current policy under the learned model p_θ.
3. Under this scheme, the returns can be bounded as followed:

$$\eta[\pi] \geq \eta^{\text{ branch }}[\pi] - 2r_{\max}\left[\frac{\gamma^{k+1}\epsilon_\pi}{(1-\gamma)^2} + \frac{\gamma^k + 2}{(1-\gamma)}\epsilon_\pi + \frac{k}{1-\gamma}\left(\epsilon_m + 2\epsilon_\pi\right) \right]$$

2.3 Model Generalization in Practice

1. The lower bound is maximized when $k = 0$, corresponding to not using the model at all.
2. One limitation of the analysis is pessimistic scaling of model error ϵ_m with respect to policy shift ϵ_π, as they do not make any assumptions about the generalization capacity or smoothness properties of the model.

3. To measure the model generalization in practice, they empirically measure how the model error under the new policies increases with policy change ϵ_π.
4. They train a model on the state distribution of a data-collecting policy π_D and then continue policy optimization while measuring the model's loss on all intermediate policies π during this optimization.
5. They shown that model error increases with the divergence between the current policy and the data-collecting policy.
6. However, the rate of this increase depends on the amount of data collected by the data-collecting policy . The decreasing dependence on policy shift shows that not only do models trained with more data perform better on their training distribution, but they also generalize better to nearby distributions.
7. They argue that the previous mathematical analysis assumed access to only model error ϵ_m on the distribution of the most recent data-collecting policy π_D and approximated the error on the current distribution as $\epsilon_m + 2\epsilon_\pi$.
8. They can instead approximate the model error as a linear function of the policy divergence $\hat{\epsilon}_{m'}(\epsilon_\pi) \approx \epsilon_m + \epsilon_\pi \frac{\mathrm{d}\epsilon_{m'}}{\mathrm{d}\epsilon_\pi}$.
9. They proceed to present a new bound

$$\eta[\pi] \geq \eta^{\text{ branch}}[\pi] - 2r_{\max}\left[\frac{\gamma^{k+1}\epsilon_\pi}{(1-\gamma)^2} + \frac{\gamma^k \epsilon_\pi}{(1-\gamma)} + \frac{k}{1-\gamma}(\epsilon_{m'})\right]$$

10. This bound suggests non-zero length rollout

$$k^* = \underset{k}{\operatorname{argmin}}\left[\frac{\gamma^{k+1}\epsilon_\pi}{(1-\gamma)^2} + \frac{\gamma^k \epsilon_\pi}{(1-\gamma)} + \frac{k}{1-\gamma}(\epsilon_{m'})\right] > 0$$

3 Model-Based Policy Optimization with Deep Reinforcement Learning

1. They use the ensemble approach in PETS.
2. To generate a prediction from the ensemble, they simply select a model uniformly at random, allowing for different transitions along a single model rollout to be sampled from different dynamic model.
3. They use SAC as the policy optimization algorithm.
4. They argue that the use of the model allows them to take many more policy gradient steps per environment sample than is typically stable in model-free algorithms.
5. Their approach alternates between optimizing the model, generate short rollout from the model, and training the policies with the short rollout.

4 Questions

1. For this sentence to be true: "Such a statement guarantees that, as long as we improve by at least C under the model, we can guarantee improvement on the true MDP.", do we have to make additional assumptions? For example, do we need to assume that $\eta[\pi] \geq \hat{\eta}[\pi]$.
2. What is π_D in section 4.1?
3. How does the mathematical analysis suggests picking the value for k. Is the value of k chosen based on the math or is it based on hyper-parameter search?
4. Is the buffer D_{model} cleared between policy optimization step?
5. What is the performance of the algorithm on Humanoid?
6. Section 6.2 has this sentence: "The ratio between the number of gradient updates and environment samples, G, is much higher in MBPO than in comparable model-free algorithms because the model-generated data reduces the risk of overfitting"". What is overfitting in RL?

7. What is the bottleneck in performance? Is it the accuracy of the branched rollout? A way to test is to use the ground truth transition function to do the branched rollout, and see what the performance of that is.

8. Did they try ensemble of deterministic NN instead of PETS? Mujoco's transition function is not stochastic.