Deep RL Foundations in 6 Lectures

Lecture 6: Model-based RL

Pieter Abbeel

Lecture Series

- Lecture 1: MDPs Foundations and Exact Solution Methods
- Lecture 2: Deep Q-Learning
- Lecture 3: Policy Gradients, Advantage Estimation
- Lecture 4: TRPO, PPO
- Lecture 5: DDPG, SAC
- ***Lecture 6: Model-based RL***

Outline for This Lecture

- ***<u>Model-based RL</u>***
- Robust Model-based RL:Model-Ensemble TRPO (ME-TRPO)
- Adaptive Model-based RL: Model-based Meta-Policy Optimization (MB-MPO)

“Algorithm”: Model-Based RL

- For iter = 1, 2, …
  - Collect data under current policy
  - Learn dynamics model from past data
  - Improve policy by using dynamics model
    (either by backprop-through-time through the learned model,
    or by using the learned model as a sim to run RL)

Why Model-Based RL?

- Anticipate data-efficiency
  - Get model out of data, which might allow for more significant policy updates than just a policy gradient
- Learning a model
  - Re-usable for other tasks [assuming general enough]

“Algorithm”: Model-Based RL

for iter = 1, 2, …

- Collect data under current policy
- Learn dynamics model from past data
- Improve policy by using dynamics model

Anticipated benefit?
– much better sample efficiency

So why not used all the time?
-- training instability → ME-TRPO
-- not achieving same asymptotic performance as model-free methods → MB-MPO

Outline for This Lecture

- Model-based RL
- ***<u>Robust Model-based RL:Model-Ensemble TRPO (ME-TRPO)</u>***
- Adaptive Model-based RL: Model-based Meta-Policy Optimization (MB-MPO)

Overfitting in Model-based RL

- Standard overfitting (in supervised learning)
  - Neural network performs well on training data, but poorly on test data
    - E.g. on prediction of s_next from (s, a)
- New overfitting challenge in Model-based RL
  - policy optimization tends to exploit regions where insufficient data is available to train the model, leading to catastrophic failures
  - = “model-bias” (Deisenroth & Rasmussen, 2011; Schneider, 1997; Atkeson & Santamaria, 1997)
  - Proposed fix: Model-Ensemble Trust Region Policy Optimization (ME-TRPO)

Model-Ensemble Trust-Region Policy Optimization

Algorithm 1 Vanilla Model-Based Deep Reinforcement Learning

Initialize a policy π_θ and a model $\hat{f}_\phi$.
Initialize an empty dataset D.
repeat
  Collect samples from the real environment f using π_θ and add them to D.
  Train the model $\hat{f}_\phi$ using D.
  repeat
    Collect fictitious samples from $\hat{f}_\phi$ using π_θ.
    Update the policy using BPTT on the fictitious samples.
    Estimate the performance $\hat{\eta}(\theta; \phi)$.
  until the performance stop improving.
until the policy performs well in real environment f.

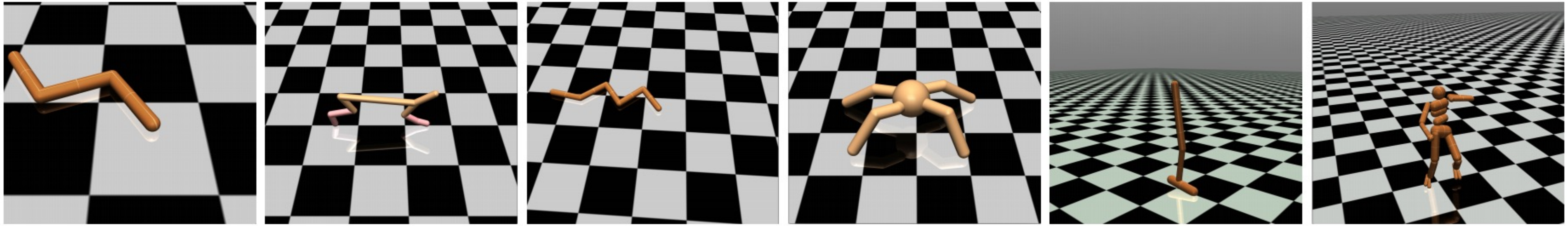
Algorithm 2 Model Ensemble Trust Region Policy Optimization (ME-TRPO)

Initialize a policy π_θ and all models $\hat{f}_{\phi_1}, \hat{f}_{\phi_2}, ..., \hat{f}_{\phi_K}$.
Initialize an empty dataset $\mathcal{D}$.
repeat
  Collect samples from the real system f using π_θ and add them to $\mathcal{D}$.
  Train all models using $\mathcal{D}$.
  repeat $\triangleright$ Optimize π_θ using all models.
    Collect fictitious samples from $\{\hat{f}_{\phi_i}\}_{i=1}^{K}$ using π_θ.
    Update the policy using TRPO on the fictitious samples.
    Estimate the performances $\hat{\eta}(\theta; \phi_i)$ for $i = 1, ..., K$.
  until the performances stop improving.
until the policy performs well in real environment f.

[Kurutach, Clavera, Duan, Tamar, Abbeel, ICLR 2018]

ME-TRPO Evaluation

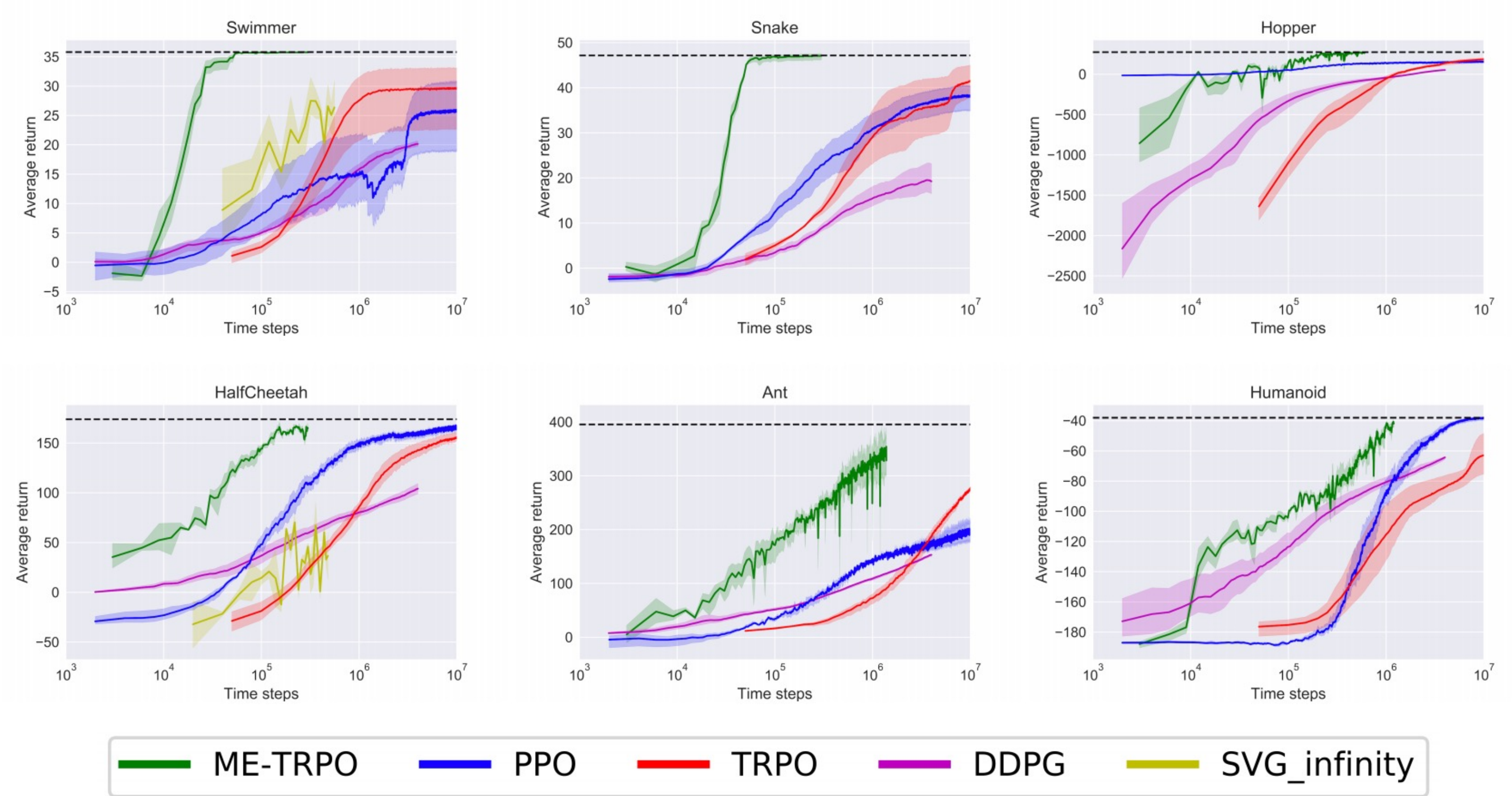
- Environments:



[Kurutach, Clavera, Duan, Tamar, Abbeel, ICLR 2018]

ME-TRPO Evaluation

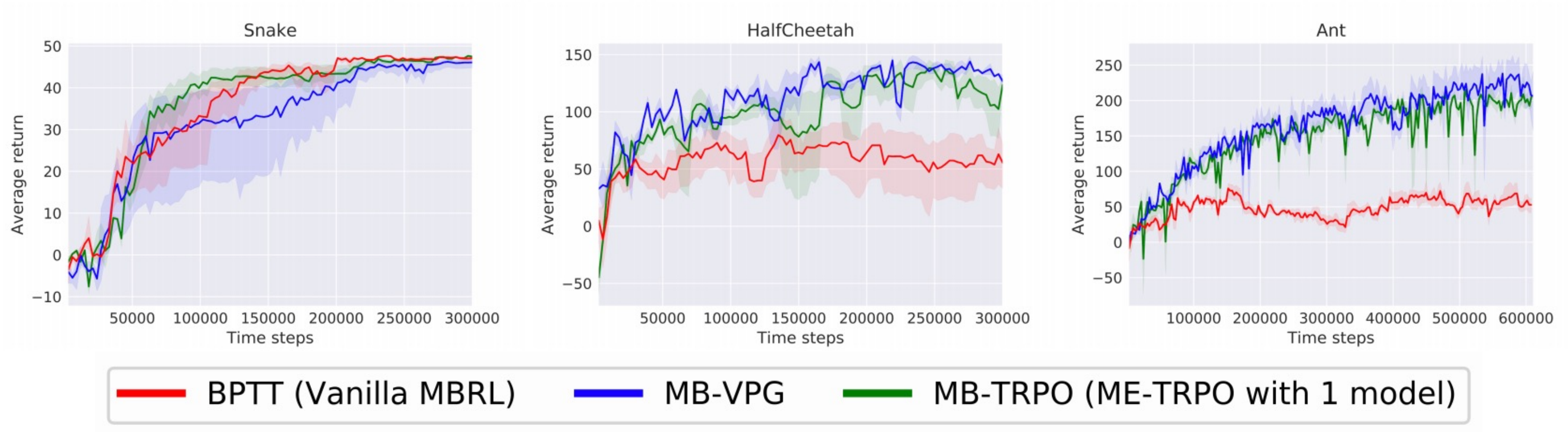
- Comparison with state of the art



[Kurutach, Clavera, Duan, Tamar, Abbeel, ICLR 2018]

ME-TRPO -- Ablation

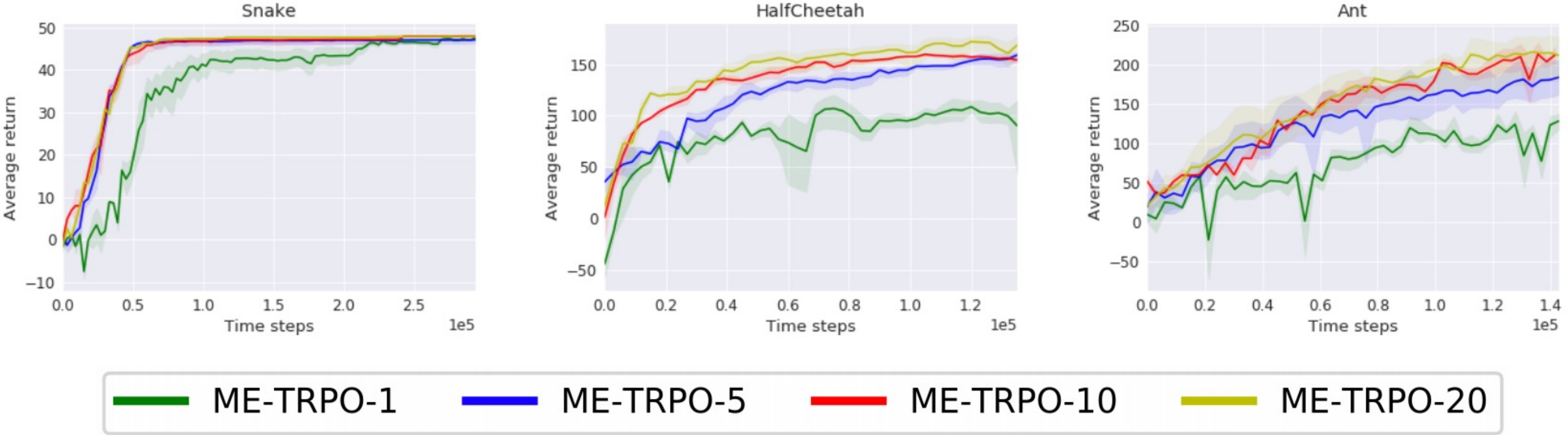
TRPO vs. BPTT in standard model-based RL



[Kurutach, Clavera, Duan, Tamar, Abbeel, ICLR 2018]

ME-TRPO -- Ablation

Number of learned dynamics models in the ensemble



[Kurutach, Clavera, Duan, Tamar, Abbeel, ICLR 2018]

Outline for This Lecture

- Model-based RL
- Robust Model-based RL:Model-Ensemble TRPO (ME-TRPO)
- ***<u>Adaptive Model-based RL: Model-based Meta-Policy Optimization (MB-MPO)</u>***

"Algorithm": Model-Based RL

for iter = 1, 2, …

- Collect data under current policy
- Learn dynamics model from past data
- Improve policy by using dynamics model

Anticipated benefit?
– much better sample efficiency

So why not used all the time?
-- training instability → ME-TRPO
-- not achieving same asymptotic performance as model-free methods → **MB-MPO**

Model-based RL Asymptotic Performance

- Because learned (ensemble of) model imperfect
  - Resulting policy good in simulation(s), but not optimal in real world
- Attempted Fix 1: learn better dynamics model
  - Such efforts have so far proven insufficient
- Attempted Fix 2: model-based RL via meta-policy optimization (MB-MPO)
  - Key idea:
    - Learn ensemble of models representative of generally how the real world works
    - Learn an ***adaptive policy*** that can quickly adapt to any of the learned models
    - Such adaptive policy can quickly adapt to how the real world works

Model-based via Meta-Policy Optimization MB-MPO

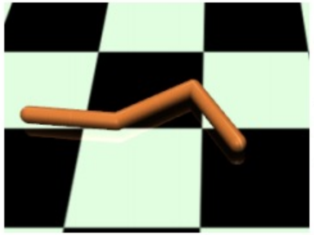
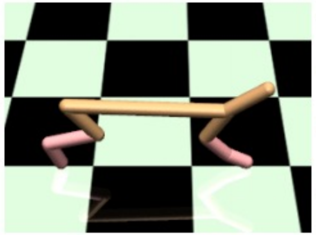
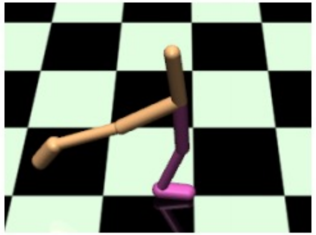
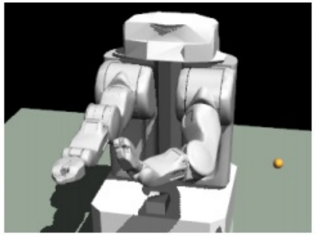
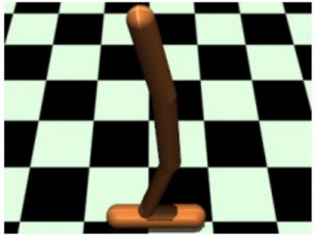
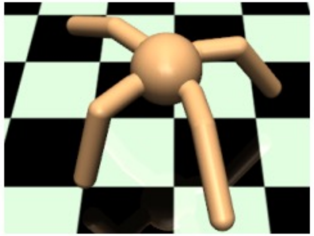
Algorithm 1 MB-MPO

Require: Inner and outer step size α, β

Initialize the policy $\pi_{\boldsymbol{\theta}}$, the models $\hat{f}_{\boldsymbol{\phi}_1}, \hat{f}_{\boldsymbol{\phi}_2}, \ldots, \hat{f}_{\boldsymbol{\phi}_K}$ and $\mathcal{D} \leftarrow \emptyset$
repeat
  Sample trajectories from the real environment with the adapted policies $\pi_{\boldsymbol{\theta}'_1}, \ldots, \pi_{\boldsymbol{\theta}'_K}$. Add them to $\mathcal{D}$.
  Train all models using $\mathcal{D}$.
  for all models $\hat{f}_{\boldsymbol{\phi}_k}$ **do**
    Sample imaginary trajectories $\mathcal{T}_k$ from $\hat{f}_{\boldsymbol{\phi}_k}$ using $\pi_{\boldsymbol{\theta}}$
    Compute adapted parameters $\boldsymbol{\theta}'_k = \boldsymbol{\theta} + \alpha \, \nabla_{\boldsymbol{\theta}} J_k(\boldsymbol{\theta})$ using trajectories $\mathcal{T}_k$
    Sample imaginary trajectories $\mathcal{T}'_k$ from $\hat{f}_{\boldsymbol{\phi}_k}$ using the adapted policy $\pi_{\boldsymbol{\theta}'_k}$
  end for
  Update $\boldsymbol{\theta} \rightarrow \boldsymbol{\theta} - \beta \, \frac{1}{K} \sum_k \nabla_{\boldsymbol{\theta}} J_k(\boldsymbol{\theta}'_k)$ using the trajectories $\mathcal{T}'_k$
until the policy performs well in the real environment
return Optimal pre-update parameters $\boldsymbol{\theta}^*$

[Clavera*, Rothfuss*, Schulman, Fujita, Asfour, Abbeel, CoRL 2018]

MB-MPO Evaluation

[Clavera*, Rothfuss*, Schulman, Fujita, Asfour, Abbeel, CoRL 2018]

MB-MPO Evaluation

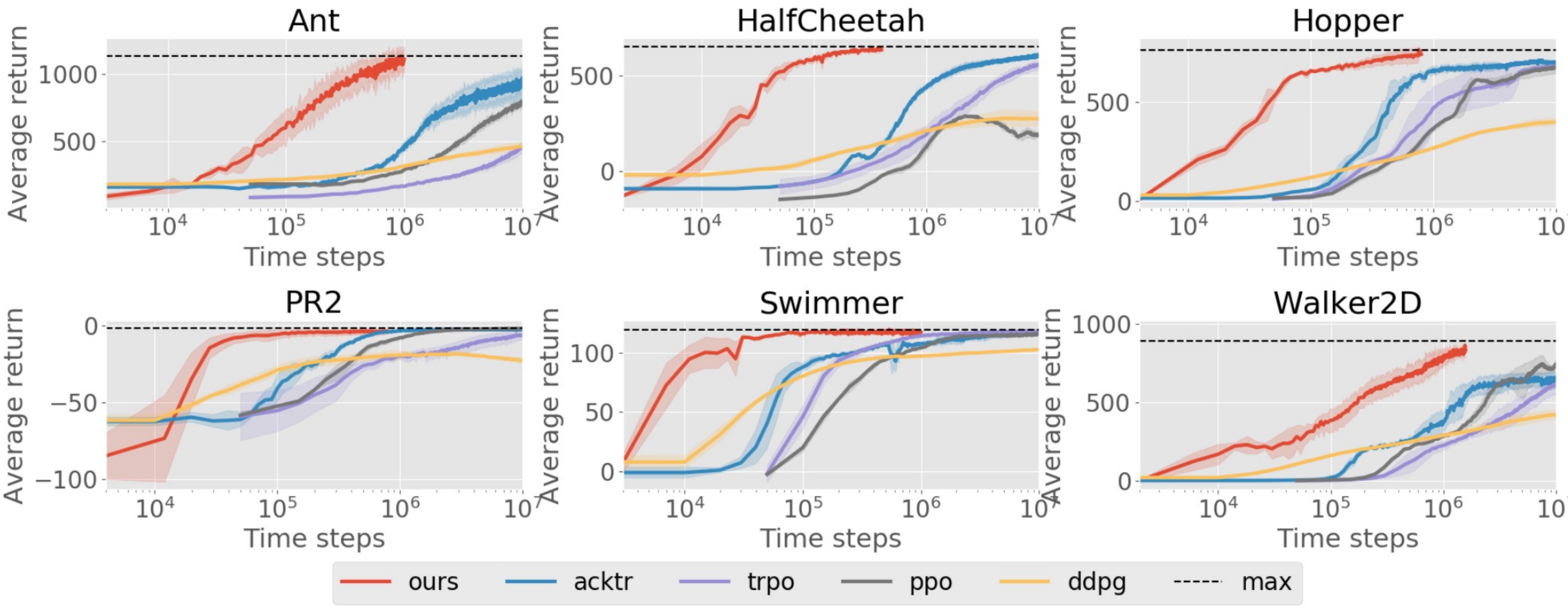
[Clavera*, Rothfuss*, Schulman, Fujita, Asfour, Abbeel, CoRL 2018]

MB-MPO Evaluation

[Clavera*, Rothfuss*, Schulman, Fujita, Asfour, Abbeel, CoRL 2018]

MB-MPO Evaluation

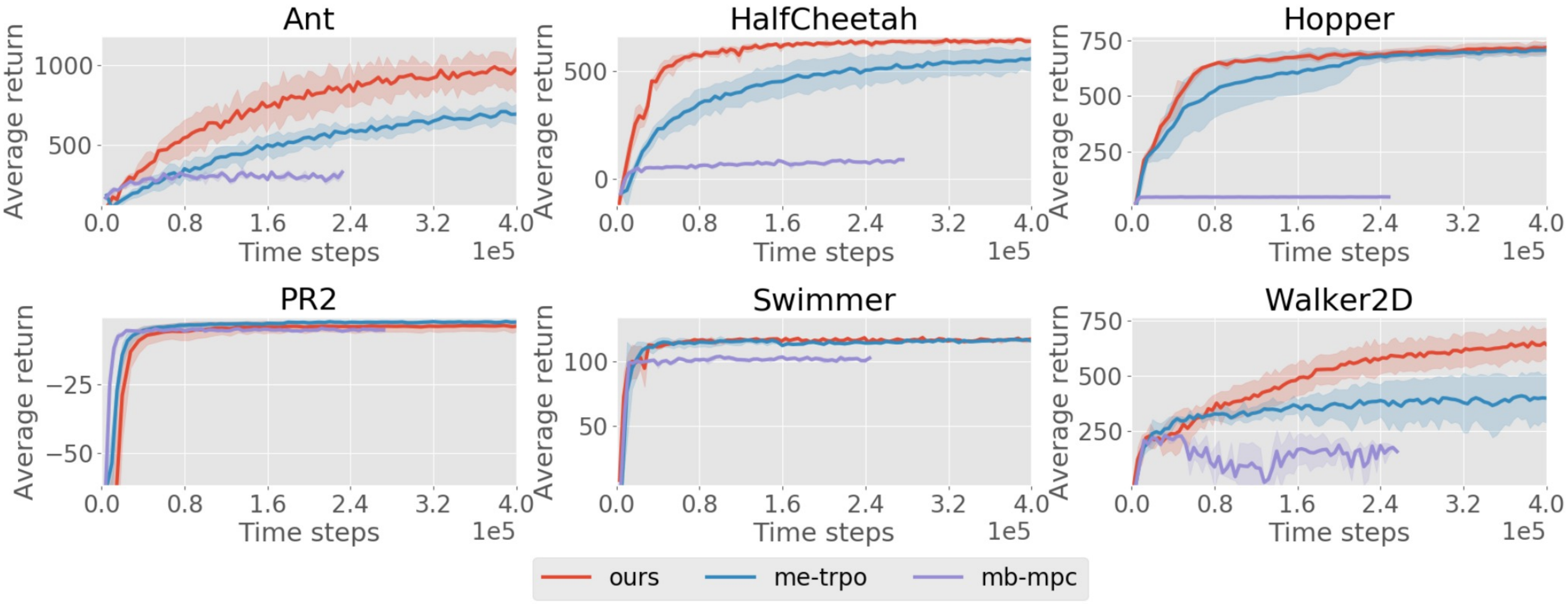
- Comparison with state of the art model-free



[Clavera*, Rothfuss*, Schulman, Fujita, Asfour, Abbeel, CoRL 2018]

MB-MPO Evaluation

- Comparison with state of the art model-based



[Clavera*, Rothfuss*, Schulman, Fujita, Asfour, Abbeel, CoRL 2018]

Summary of This Lecture

- Model-based RL
- Robust Model-based RL:Model-Ensemble TRPO (ME-TRPO)
- Adaptive Model-based RL: Model-based Meta-Policy Optimization (MB-MPO)

Summary of Lecture Series

- Lecture 1: MDPs Foundations and Exact Solution Methods
- Lecture 2: Deep Q-Learning
- Lecture 3: Policy Gradients, Advantage Estimation
- Lecture 4: TRPO, PPO
- Lecture 5: DDPG, SAC
- Lecture 6: Model-based RL