Planning with Inaccurate Models

Sanjiban Choudhury
Elephant in the room:
Why can’t we just learn a model?
“Just pretend I’m not here...”
Model Based Reinforcement Learning

Learn Model → Plan with Learned Model

iLQR!
Why Model?
Models are *necessary*

Robots can’t just try out random actions in the world!
Models are *necessary*

We invested heavily in simulators for helicopters and self-driving to verify behaviors before deployment.
Models work in *theory*

Model-Based Reinforcement Learning with a Generative Model is Minimax Optimal

Alekh Agarwal  
Microsoft  
alekha@microsoft.com

Sham Kakade  
University of Washington  
sham@cs.washington.edu

Lin F. Yang  
University of California, Los Angeles  
linyang@ee.ucla.edu

April 7, 2020
Models work in *practice*

Hafner et al. 2023
Learning Models.
2560, 2.5 second trajectories sampled with cost-weighted average @ 60 Hz

Georgia Tech Auto Rally (Byron Boots lab)
Activity!
Think-Pair-Share

Think (30 sec): What features / architecture would you use to learn a model for rally car? What planner would you use?

Pair: Find a partner

[Diagram: Think, Learn Model, Plan with Learned Model]

Share (45 sec): Partners exchange ideas
Part 1: System Identification

Information Theoretic MPC for Model-Based Reinforcement Learning

Grady Williams, Nolan Wagener, Brian Goldfain, Paul Drews, James M. Rehg, Byron Boots, and Evangelos A. Theodorou

Learn Model

Plan with Learned Model

Collect data of rally car \((x_1, u_1, x_2, u_2, \ldots)\)

\[ x_{t+1} = F(x_t, u_t) = \begin{bmatrix} q_t + \dot{q}_t \Delta t \\ \dot{q}_t + f(x_t, u_t) \Delta t \end{bmatrix} \]

2 Layer MLP
Part 2: Planning

Information Theoretic MPC for Model-Based Reinforcement Learning

1. Sample and evaluate trajectories
2. Compute control update
3. Execute first control in sequence, receive state feedback
4. Repeat, using the un-executed portion of the previous control sequence to warm-start the trajectory

Cross Entropy like approach!
Question: How do you collect data for learning model?

2560, 2.5 second trajectories sampled with cost-weighted average @ 60 Hz
Another Example: Helicopter Aerobatics

A nose-in funnel!
(Super cool work by Pieter Abeel et al. https://people.eecs.berkeley.edu/~pabbeel/autonomous_helicopter.html)
Part 1: System Identification

Learn a linear model around reference

\[ \Delta x_{t+1} = A_t x_t + B_t u_t \]
Part 2: Planning

Learn Model

Plan with Learned Model

Use LQR with learnt models
How do we collect data to train our model?
Strategy

Train a model on state actions visited by the expert!
If I perfectly fit a model (i.e. training error zero), this should work, right?
World

\[ s' = M^*(s, a) \]

Experts picks action \( a \) to go to the goal
Model agrees with world, i.e. train error zero!
What if the model is optimistic?
Predicts a short cut to the goal by taking action $a'$
In reality the shortcut ends in death …
Training on Expert Data

(From Ross and Bagnell, 2012)
Strategy

Train a model on state actions visited by the expert!

Train a model on state actions visited by the learner!
Improve model where policy goes

Collect more data along current policy’s trajectory
Don’t we know an algorithm that does this?
DAGGER for Model-based RL!!

Roll-out current policy

New Transitions
State → Action → Next State

New Policy

Aggregate Dataset

All previous transitions

Fit Model

Planner

New Model
Model Based RL v2.0

If I perfectly fit a model (i.e. training error zero), this should work, right?
Model
\( s' = \hat{M}(s, a) \)

World
\( s' = M^*(s, a) \)
Model predictions vs. actual outcomes:

- Model: $s' = \hat{M}(s, a)$
- World: $s' = M^*(s, a)$

Diagram:

Model predicts it can't get to trophy, but can get to $1$. 
Model plans to get $1
Training error is zero!
But the model is just pessimistic!
Strategy

Train a model on state actions visited by the expert!

Train a model on state actions visited by the learner!

Train a model on state actions visited by both the expert and the learner!
Model Learning with Planner in Loop

(Cross & Bagnell, 2012)
Model learning on both expert and learner data works!

(From Ross & Bagnell, 2012)
How do we derive this strategy?
Theoretical Foundations for Model Based RL

Agnostic System Identification for Model-Based Reinforcement Learning

Stéphane Ross
Robotics Institute, Carnegie Mellon University, PA USA

J. Andrew Bagnell
Robotics Institute, Carnegie Mellon University, PA USA

STEFANEROSS@CMU.EDU

DBAGNEILL@RI.CMU.EDU
Lemma: Performance Difference via Planning in Model

\[ J_{M^*}(\pi^*) - J_{M^*}(\hat{\pi}) \]

\[ \leq \mathbb{E}_{s_0} \left[ V_{\hat{M}}(s_0) - V_{\pi^*}(s_0) \right] + TV_{\max} \mathbb{E}_{s,a \sim \pi^*} \| \hat{M}(s,a) - M^*(s,a) \| \]

Planning error

Model fit on expert states

\[ + TV_{\max} \mathbb{E}_{s,a \sim \hat{\pi}} \| \hat{M}(s,a) - M^*(s,a) \| \]

Model fit on policy states
The Challenge.
Planning is like finding a needle in an exponential haystack.
A Tree MDP
Planning is $\exp(T)$!
Planning is $\exp(T)!$
How much planning do we need when learning models?
Learnt model has hidden portals!
Model at iteration 0
Run planning for \( \exp(T) \)
Policy at iteration 0
Model at iteration 1
Run planning for $\exp(T)$
Policy at iteration 1

Plan for exp(T) to find policy!
Run planning for \( \exp(T) \)
Plan for $\exp(T)$ to find policy!
After many iterations ......
Exponential Complexity of Model Learning

Every iteration, planning is \( \exp(T) \) computation

Repeat for many iterations to eliminate all portals
Key Insight.
Be Lazy.

Don’t compute optimal plan.

Just do better than expert.
The Virtues of Laziness in Model-based RL: A Unified Objective and Algorithms

Anirudh Vemula ¹  Yuda Song ²  Aarti Singh ²  J. Andrew Bagnell ¹ ²  Sanjiban Choudhury ³
How do we turn planning

\[ \text{Exp}(T) \rightarrow \text{Poly}(T) \]
How do we turn planning
\[ \text{Exp}(T) \rightarrow \text{Poly}(T) \]?

Restart from *expert states*
Policy Search via Dynamic Programming (PSDP)
(Bagnell, et al. 2003)

Iterate from T-1 and go back in time

At each time t, restart from expert state $s_t^*$

Solve for best policy $\pi_t$, given future policies $\pi_{t+1}, \pi_{t+2}, \cdots \pi_T$

$$\pi_t = \arg \max_{\pi} r(s_t^*, \pi(s_t^*)) + \mathbb{E}_{s_{t+1}} V^\pi_{t+1:T}(s_{t+1})$$
Policy Search via Dynamic Programming (PSDP)

Let's say we have expert states
Policy Search via Dynamic Programming (PSDP)

What is the best policy $\pi_{T-1}$?
Policy Search via Dynamic Programming (PSDP)

What is the best policy $\pi_{T-2}$, given $\pi_{T-1}$?
What is the best policy $\pi_{T-2}$, given $\pi_{T-1}$?
Policy Search via Dynamic Programming (PSDP)

What is the best policy $\pi_{T-3}$, given $\pi_{T-2}$, $\pi_{T-1}$?
Only took poly(T) steps!
PSDP is Lazy

Instead of searching all states to find the best policy

Just do better on states the expert visits
Is being lazy a good idea for model learning?
Model at iteration 0
Run lazy policy search poly(T)
Policy at iteration 0
Model at iteration 1
Run lazy policy search $\text{poly}(T)$
Policy at iteration 1
Run lazy policy search poly(T)
Policy at iteration 2

Converged!!!
Final Model + Policy

Note since the planner search the whole tree, it may not remove all the hidden portals
But can we prove that lazy is good for model learning?
A New Lemma!
Lemma: Performance Difference via Advantage in Model

\[ J_{M^*}(\pi^*) - J_{M^*}(\hat{\pi}) \]

\[ \leq \mathbb{E}_{s^* \sim \pi^*} [A_{\pi^*}(s^*, a^*)] + TV_{\max} \mathbb{E}_{s, a \sim \pi^*} | | \hat{M}(s, a) - M(s, a) | | \\
\text{Model fit on expert states} \\
\text{Advantage of expert in model} \\
+ TV_{\max} \mathbb{E}_{s, a \sim \pi} | | \hat{M}(s, a) - M(s, a) | | \\
\text{Model fit on policy states} \]
Lazy Model-based Policy Search (LAMPS)
LAMPS finds a better policy with fewer samples + fewer computation

SysID: Use planner (iLQR)

LAMPS: Use PSDP (LQR on expert traj)
LAMPS converges faster than both SysID and MBPO
LAMPS makes better use of Expert Data

10000 samples

50000 samples
Recap

Model Learning with Planner in Loop
(Ross & Bagnell, 2012)

Lazy Model-based Policy Search (LAMPS)

Planning $\exp(T)!$

Lazy $\text{poly}(T)!$
Another challenge.
Mismatched Objectives
Fitting model with L2 loss is mismatched with how good the resulting policy is.
True Dynamics
Learnt Model A

Gets everything right but 1
Learnt Model B

Gets everything wrong but 1
Which model has lower loss? Which one do we prefer?

Learnt Model A

Gets everything right but 1

Learnt Model B

Gets everything wrong but 1

Can we have change the loss for how we fit the model?
Our new lemma actually prescribes matching values!

\[ J_{M^*}(\hat{\pi}) - J_{M^*}(\pi) \]

\[ = \mathbb{E}_{s^* \sim \pi^*} [A_{\hat{\pi}}(s^*, a^*)] + T \mathbb{E}_{s, a \sim \hat{\pi}} [E_{s' \sim \hat{M}} V_{\hat{\pi}}(s') - E_{s'' \sim M^*} V_{\hat{\pi}}(s'')] \]

**Advantage of expert in model**

\[ + T \mathbb{E}_{s, a \sim \hat{\pi}} [E_{s' \sim \hat{M}} V_{\hat{\pi}}(s') - E_{s'' \sim M^*} V_{\hat{\pi}}(s'')] \]

**Value matching on expert states**

\[ + T \mathbb{E}_{s, a \sim \hat{\pi}} [E_{s' \sim \hat{M}} V_{\hat{\pi}}(s') - E_{s'' \sim M^*} V_{\hat{\pi}}(s'')] \]

**Value matching on learner states**
LAMPS with Moment Matching (LAMPS-MM)

Collect Expert Data → Fit Model → Lazy Planner

Value Loss

Rollout Policy
Challenge 1: Planning is computationally expensive

Solution 1: Be lazy, restart from expert states

New Lemma: Performance Difference via Advantage in Model

Solution 2: Match value loss

Challenge 2: Mismatched Objective