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
Model Based Reinforcement Learning

Learn Model \rightarrow Plan with Learned Model

Learning a model is EASY $\Rightarrow$ Just supervised learning!!
(Super cool work by Pieter Abeel et al. https://people.eecs.berkeley.edu/~pabbeel/autonomous_helicopter.html)
Activity!
Think (30 sec): How will you model the dynamics of a helicopter? What planner would you use to control it?

Pair: Find a partner

Learn Model

Plan with Learned Model

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

Learn Model

Plan with Learned Model

Predict velocities in “body” frame

\[ \begin{align*}
\dot{u} &= vr - wq + A_x u + g_x + w_u, \\
\dot{v} &= wp - ur + A_y v + g_y + D_0 + w_v, \\
\dot{w} &= uq - vp + A_z w + g_z + C_4 u_4 + D_4 + w_w, \\
\dot{p} &= qr(I_{yy} - I_{zz})/I_{xx} + B_x p + C_1 u_1 + D_1 + w_p, \\
\dot{q} &= pr(I_{zz} - I_{xx})/I_{yy} + B_y q + C_2 u_2 + D_2 + w_q, \\
\dot{r} &= pq(I_{xx} - I_{yy})/I_{zz} + B_z r + C_3 u_3 + D_3 + w_r.
\end{align*} \]
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!
Vanilla Model Based RL

Collect Expert Data → Fit Model

Test Execution → Planner
Problem: Train Test Mismatch!

![Diagram showing the problem of train test mismatch.]

Training Distribution $\neq$ Test Trajectory

Data gathering policy
Super retro video of distribution mismatch!
Okay but what if my TRAIN error is ZERO!
Experts picks action $a$ to go to the goal

World

$s' = M^* (s, a)$
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 ...
Is there a theorem that tells me how error in my model affects error in my performance?
The Simulation Lemma
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 Policy

New Model

Fit Model

Aggregate Dataset

New Transitions
State → Action → Next State

All previous transitions
DAGGER seems to work!
Are we done?
Model
\[ s' = \hat{M}(s, a) \]

World
\[ s' = M^*(s, a) \]
Model predicts it can’t get to trophy, but can get to $1
Model predicts it can’t get to trophy, but can get to $1
Model: $s' = \hat{M}(s, a)$

World: $s' = M^*(s, a)$

Training error is zero!
Model \( s' = \hat{M}(s, a) \)

World \( s' = M^*(s, a) \)

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!
How do we derive this strategy?
What makes a model good?

\[ J_{M^*}(\hat{\pi}) - J_{M^*}(\pi^*) \]
The Double Simulation Lemma
Insight

- Make a model good only where the expert visits
- Make a model good where the learner visits
- Make a model good in both places