- March 12, 2021 - Learning and Planning Techniques

  - We view two angles of AI:

    ① Planning: Given model, reason with model to select actions (DP)

    ② Learning: Analyze data using RL

      ↳ Model-Based RL combines both methods

        ↳ Use data to generate a model, then act in the realm of planning

          ↳ You can either use DP or sample from model and use RL. Both considered planning

  - Model-based RL: How RL is used in practice

    - Collect data and build approximate model of the real world.
    - Run simulations of real world on computer

      ↳ Let robot run on that simulator and use RL

      ↳ Even though we are using reinforcement learning algorithms, as soon as you use a model, you are doing RL

    - Advantages: Can efficiently learn model with Supervised learning
    - Disadvantage: Model is not always accurate

    - Simulators let you Sample States and returns

  - Model learning: Estimate model from Sequences of experiences

    - Tabular Model If State Space and action spaces are not huge

↳ Empirical transition properties of $(s, a) \rightarrow s'$

↳ Reward is average reward from (s, a)

↳ Data comes in form $\langle S_t, A_t, R_{t+1}, S_{t+1} \rangle$. Throw data into database, and sample randomly

- In real world, RL is done by building a model than sampling model
  - ↳ Models are complex, suffer from curse of dimensionality
  - ↳ Once you get model, you can sample and run RL
  - ↳ Sampling-based methods are more efficient than DP in real world

- To overcome curse of modeling and curse of dimensionality:
  1. Do good function approx
  2. Sample

- Assumption is slippage of models.
  - ↳ Problems come from limited data and function approx using RL
  - ↳ Two sources of slippage
  - ↳ It is important to remember that model performance can be inaccurate

    ∴ Two approaches:

    1. Don't build model, go directly to RL
    2. Reason explicitly about model uncertainty.
       - ↳ Do sensitivity analysis.
       - ↳ Perturb model assumptions and parameters and see how model predictions change

- **Dyna:** Do planning using RL from simulated experiences and also do direct RL on real data.

  - ↳ Simultaneously do model-free and model RL
  - ↳ RL algo pulls from real and simulated experiences

  * In practice, this is how to go about RL

    - ↳ You have to balance how much value / policy estimates comes from planning and how much comes from learning
    - ↳ Also, how often to update model

- **Forward Search:** Monte Carlo Tree Search

  - ↳ Build a model (planning) and leverage RL algorithms
  - ↳ Model = Search tree
    - ↳ When you enter state, create tree with root at state.
      - ↳ Consider all actions, then all next states, then all actions ...
    - ↳ Getting optimal action is recursive algorithm after growing tree many layers down
    - ↳ Build an MDP using a model
  - ↳ Each time you build a new tree from a new state, you can reuse knowledge from previous tree
  - ↳ In practice, you search tree using sampling
  - ↳ MC tree search algorithm does MC, collecting full experiences

• Origins of MC Tree Search Algorithm : Adaptive Multistage Sampling Algorithm

↳ Monte Carlo Tree Search (MCTS) popularized by alphaGo

① Selection : Take actions

② Expansion : Grow tree from leaf node

③ Simulation : For certain nodes, grow entire path to the end of the tree.

↳ Depending on simulation trees, update previous nodes

↳ ④ Back Propagation

Selection involves picking a child node with most confidence

↳ Affected by Explore - Exploit Dilemma

↳ For game trees, this uses Upper Confidence Bound algorithm from bandits, specifically Upper Confidence Bounds for Trees (UCT's)

↳ MCTS, UCT algorithms come from paper :

Adaptive Multistage Sampling Algorithm

for finite horizon MDP

• Adaptive Multistage Sampling (AMS) Algorithm :

• Take an MDP with finite time steps

• Assume large state space, small action space

• Assume we have a reward function and a sampling function

• Can we do better than backward induction when S is large and A is small ?

- AMS is based on fixed allocation of action selection for each state, in each time step
  - ↳ Based on explore vs. Exploit UCT formula
  - ↳ Leads you to estimate optimal action-value function
    - ↳ Then use argmax of $\hat{Q}^*$ to estimate optimal policy

- Three Probs in exam

RL → ① Programming (he will give us 80% of code. We fill it in)

Financial → ② Financial Modelling (No code, but how model can be modeled as an MDP)

Calculus using Bellman to closed form Solution → ③ Analytical problem - Closed-form Analytical using Bellman