ST455: Reinforcement Learning

Lecture 10: Offline Reinforcement Learning

Chengchun Shi

Lecture Outline

1. Introduction to Offline RL
2. The Pessimistic Principle
3. Model-based Offline Policy Optimization (MOPO)
4. An Overview of My Research

Lecture Outline

So Far, We Focused on Online RL Applications

(a) Video Games

(b) AlphaGo

This Lecture Considers Offline Settings

(a) Health Care

(b) Robotics

(c) Ridesharing

(d) Auto-driving

This Lecture Considers Offline Settings (Cont'd)

- What is offline RL?
  - RL with a pre-collected historical dataset
- Why offline RL?
  - Online interaction with the environment is **impractical**
  - Either because online data collection is **expensive** (e.g., robotics or healthcare); rely on historical data
  - Or **dangerous** (e.g., healthcare, ridesharing or auto-driving)

Online v.s. Offline RL

Online RL:

- Data are **adaptively** generated, i.e., able to select **any** action at each time
- Data are **cheap** to generate, i.e., able to simulate **numerous** observations
- Likely to **satisfy** MDP assumption (Markovianity & time-homogeneity)

Offline RL:

- Data are **pre-collected**, i.e., from an observational study
- Size of data is **limited**
- MDP assumption likely to be **violated** (Non-Markovianity or Non-stationarity)

Offline RL Challenges and Solutions

- Data are **pre-collected**
  - Learning relies entirely on the historical data
  - Not possible to improve exploration
  - For actions that are less-explored, difficult to accurately learn their values
  - **Solution**: the pessimistic principle (focus of this lecture)
- Size of data is **limited**
  - **Solution**: develop sample-efficient RL algorithms (to be discussed in Lecture 11)
- **Violation** of MDP assumption
  - Cannot directly apply existing state-of-the-art RL algorithms
  - **Solution**: statistical hypothesis testing for model selection (to be covered in this lecture)

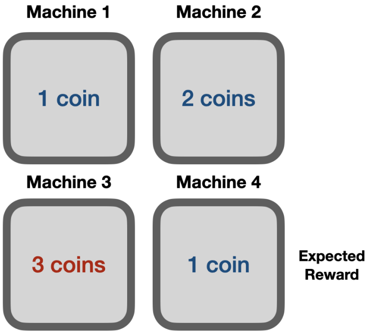
Lecture Outline

Recap: Multi-Armed Bandit Problem

- The **simplest** RL problem
- A casino with **multiple** slot machines
- Playing each machine yields an independent **reward**.
- Limited knowledge (unknown reward distribution for each machine) and resources (**time**)
- **Objective**: determine which machine to pick at each time to maximize the expected **cumulative rewards**

Offline Multi-Armed Bandit Problem

- k-armed bandit problem (k machines)
- $A_t \in \{1, \cdots, k\}$: arm (machine) pulled (experimented) at time t
- $R_t \in \mathbb{R}$: reward at time t
- $Q(a) = \mathbb{E}(R_t | A_t = a)$ expected reward for each arm a (**unknown**)
- **Objective**: Given $\{A_t, R_t\}_{0 \le t < T}$, identify the best arm



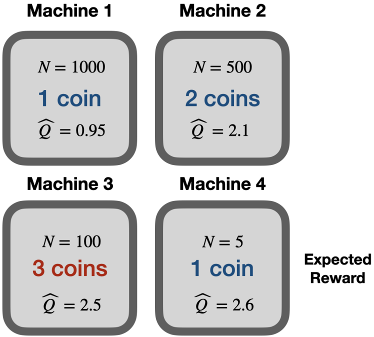
Greedy Action Selection

- **Action-value methods**:

$$\widehat{Q}(a) = N^{-1}(a) \sum_{t=0}^{T-1} R_t \mathbb{I}(A_t = a)$$

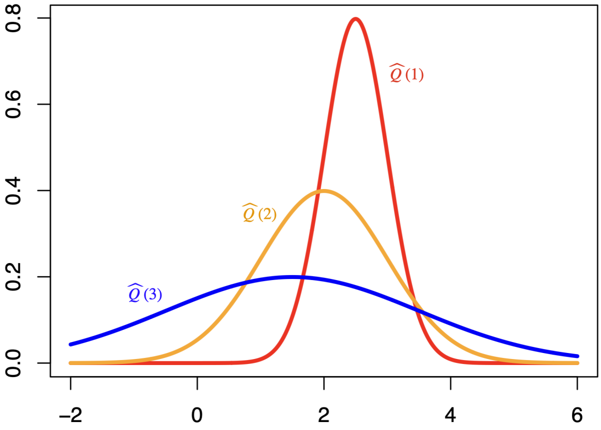
  where $N(a) = \sum_{t=0}^{T-1} \mathbb{I}(A_t = a)$ denotes the action counter

- **Greedy policy**: $\arg\max_a \widehat{Q}(a)$
- Less-explored action $\to$ $N(a)$ is small $\to$ inaccurate $\widehat{Q}(a)$ $\to$ **suboptimal** policy (see the plot on the right)



Recap: The Optimistic Principle

- Used in **online** settings to balance exploration-exploitation tradeoff
- The more **uncertain** we are about an action-value
- The more **important** it is to explore that action
- It could be the **best** action
- Likely to pick blue action
- Forms the basis for **upper confidence bound** (UCB)



Recap: Upper Confidence Bound

- Estimate an **upper confidence** $U_t(a)$ for each action value such that

$$Q(a) \leq \widehat{Q}_t(a) + U_t(a),$$

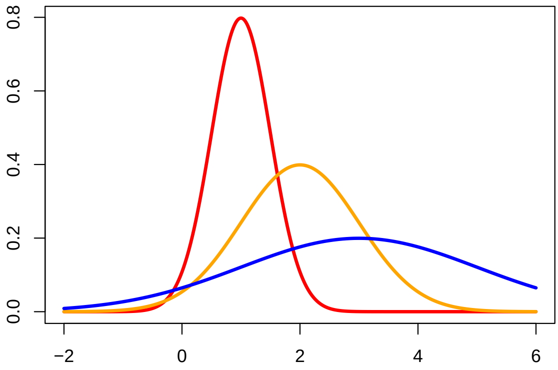
  with high probability.
- $U_t(a)$ quantifies the **uncertainty** and depends on $N_t(a)$ (number of times arm a has been selected up to time t)
  - Large $N_t(a) \rightarrow$ small $U_t(a)$;
  - Small $N_t(a) \rightarrow$ large $U_t(a)$.
- Select actions maximizing upper confidence bound

$$a^* = \arg\max_a [\widehat{Q}_t(a) + U_t(a)].$$

- Combines **exploration** ($U_t(a)$) and **exploitation** ($\widehat{Q}_t(a)$).

The Pessimistic Principle

- In **offline** settings
- The less **uncertain** we are about an action-value
- The more **important** it is to use that action
- It could be the **best** action
- Likely to pick red action
- Yields the **lower confidence bound** (LCB) algorithm



Lower Confidence Bound

- Estimate an **lower confidence** $L(a)$ for each action value such that

$$Q(a) \geq \widehat{Q}(a) - L(a),$$

  with high probability.
- $L(a)$ quantifies the **uncertainty** and depends on $N(a)$ (number of times arm a has been selected in the historical data)
  - Large $N(a) \to$ small $L(a)$;
  - Small $N(a) \to$ large $L(a)$.
- Select actions maximizing lower confidence bound

$$a^* = \arg\max_a [\widehat{Q}(a) - L(a)].$$

Lower Confidence Bound (Cont'd)

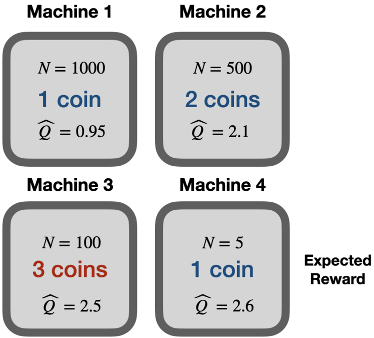
- Set $L(a) = \sqrt{c \log(T)/N(a)}$ for some positive constant c where T is the sample size of historical data
- According to **Hoeffding's inequality** (link), when rewards are bounded between 0 and 1, the event

$$|Q(a) - \widehat{Q}(a)| \leq L(a),$$

holds with probability at least $1 - 2T^{-2c}$ (converges to 1 as $T \to \infty$).

Lower Confidence Bound (Cont'd)

- $\widehat{Q}(4) > \widehat{Q}(3)$
- $T = 1605$. Set $c = 1$
- $L(3) = \sqrt{\log(T)/N(3)} = 0.272$
- $L(4) = \sqrt{\log(T)/N(4)} = 1.215$
- $\widehat{Q}(3) - L(3) > \widehat{Q}(4) - L(4)$
- $\widehat{Q}(3) - L(3) > \max(\widehat{Q}(1), \widehat{Q}(2))$
- Correctly identify optimal action



Algorithm

- **Input**: some positive constant c, offline data $\{(A_t, R_t)\}_{0 \le t < T}$.
- **Initialization**: $t = 0$, $\widehat{Q}(a) = 0$, $N(a) = 0$, for $a = 1, 2, \cdots, k$.
- **While** $t < T$:
  - **Update** N: $N(A_t) \leftarrow N(A_t) + 1$.
  - **Update** $\widehat{Q}$:

$$\widehat{Q}(A_t) \leftarrow \frac{N(A_t) - 1}{N(A_t)} \widehat{Q}(A_t) + \frac{1}{N(A_t)} R_t.$$

  - **Update** t: $t \leftarrow t + 1$.
- **LCB action selection**:

$$a^* \leftarrow \arg\max_a [\widehat{Q}(a) - \sqrt{c \log(T)/N(a)}].$$

Theory

Define the regret, as the difference between the expected reward under the **best arm** and that under the **selected arm**.

Theorem (Greedy Action Selection)

Regret of greedy action selection is upper bounded by $2\max_a |\widehat{Q}(a) - Q(a)|$, *whose value is bounded by* $2\sqrt{c\log(T)/\min_a N(a)}$ *(according to Hoeffding's inequality) with probability approaching* 1

- The upper bound depends on the estimation error of **each** Q-estimator
- The regret is small when **each** arm has sufficiently many observations
- However, it would yield a large regret when one arm is **less-explored**
- This reveals the **limitation** of greedy action selection
- Proof is simple (see Appendix)

Theory (Cont'd)

Theorem (LCB; see also Jin et al. [2021])

Regret of the LCB algorithm is upper bounded by $2\sqrt{c\log(T)/N(a^{opt})}$ *where* a^{opt} *denotes the best arm with probability approaching* 1

- The upper bound depends on the estimation error of best arm's Q-estimator **only**
- The regret is small when the **best** arm has sufficiently many observations
- This is much weaker than requiring **each** arm to have sufficiently many observations
- This reveals the **advantage** of LCB algorithm
- Proof given in the Appendix

Lecture Outline

1. Introduction to Offline RL
2. The Pessimistic Principle
3. **Model-based Offline Policy Optimization (MOPO)**
4. An Overview of My Research

Offline RL and Fitted Q-Iteration

- Offline data: $\{(S_t, A_t, R_t) : 0 \le t \le T\}$
- Fitted Q-Iteration can be naturally applied by repeating
  1. Compute $\widehat{Q}$ as the argmin of

$$\arg\min_{Q} \sum_{t} \left[R_t + \gamma \max_{a} \widetilde{Q}(S_{t+1}, a) - Q(S_t, A_t) \right]^2$$

  2. Set $\widetilde{Q} = \widehat{Q}$
- **Limitation**: for less-explored state-action pairs, their Q-values **cannot** be learned accurately
- **Solution**: the pessimistic principle

Pessimistic Principle in RL

- In multi-armed bandit, we select action to maximize lower confidence bound

$$a^* = \arg\max_{a}[\widehat{Q}(a) - L(a)]$$

- In more general RL, we can adopt a similar principle by setting

$$\pi(a|s) = \begin{cases} 1, & \text{if } a = \arg\max \widehat{Q}(a,s) - L(a,s) \\ 0, & \text{otherwise} \end{cases}$$

  where the lower bound satisfies that with probability approaching 1,

$$Q^{\pi^{\text{opt}}}(a,s) \geq \widehat{Q}(a,s) - L(a,s), \qquad \forall a, s.$$

- Many offline algorithms [see e.g., Wu et al., 2019, Kumar et al., 2020, Levine et al., 2020] adopt similar ideas, but do not directly use the above formula

Model-based Offline Policy Optimisation (MOPO)

- As we discussed in Lecture 9, **model-based** method is preferred in offline settings
- Online RL algorithms are **not** applicable, as adaptive interaction is not feasible
- Model-based method
  - learns a model using the **offline** data
  - allows to **adaptively** generate data based on the model
  - applies **online** RL algorithms to simulated data for policy optimisation
  - embraces the power of online RL algorithms for offline policy optimisation
- MOPO [Yu et al., 2020] integrates model-based method with **pessimistic** principle

MOPO: Offline Model Learning

- Learn the conditional distribution of (S_{t+1}, R_t) given (A_t, S_t)
- Approximate the conditional distribution using Gaussian, i.e.,

$$(S_{t+1}, R_t)|(A_t, S_t) \sim N(\mu_\theta(A_t, S_t), \Sigma_\phi(A_t, S_t))$$

- Parametrize μ_θ and Σ_ϕ using e.g., neural networks
- Use bootstrap to learn N different models $\{\mathcal{M}_i\}_{i=1,\cdots,N}$

MOPO: The Pessimism Principle

- Penalize reward to incorporate pessimism
- Simulate reward r given the state-action pair (s, a) from model
- Define the **transformed** reward

$$\widetilde{r} = r - L(a, s),$$

  for some lower bound $L(a, s)$ that quantifies the **uncertainty** of model
- More uncertain $\longrightarrow$ smaller transformed reward
- Less uncertain $\longrightarrow$ larger transformed reward
- Apply online RL to transformed data (see next slide)

MOPO: Adaptive Data Simulation

1. Action simulation
   - For **value-based** method, sample actions using ε-greedy policy
   - For **policy-gradient** method, sample actions using the estimated policy
2. Reward and next-state simulation
   - Randomly pick a model $\mathcal{M}_i = N(\mu_{\theta_i}(A_t, S_t), \Sigma_{\phi_i}(A_t, S_t))$
   - Sample (S_{t+1}, R_t) from this Gaussian model
   - Compute transformed reward $\widetilde{R_t} = R_t - L(A_t, S_t)$
   - Use $(S_t, A_t, \widetilde{R_t}, S_{t+1})$ to update the policy/Q-function
3. Repeat the above two steps for data simulation and policy learning

MOPO: Pseudocode

Algorithm 2 MOPO instantiation with regularized probabilistic dynamics and ensemble uncertainty

Require: reward penalty coefficient λ rollout horizon h, rollout batch size b.

Train on batch data $\mathcal{D}_{\text{env}}$ an ensemble of N probabilistic dynamics $\{\widehat{T}^i(s', r \mid s, a) = \mathcal{N}(\mu^i(s,a), \Sigma^i(s,a))\}_{i=1}^N$.

Initialize policy π and empty replay buffer $\mathcal{D}_{\text{model}} \leftarrow \varnothing$.

for epoch $1, 2, \ldots$ **do** ▷ This for-loop is essentially one outer iteration of MBPO

  for $1, 2, \ldots, b$ (in parallel) **do**

    Sample state s_1 from $\mathcal{D}_{\text{env}}$ for the initialization of the rollout.

    for $j = 1, 2, \ldots, h$ **do**

      Sample an action $a_j \sim \pi(s_j)$.

      Randomly pick dynamics $\widehat{T}$ from $\{\widehat{T}^i\}_{i=1}^N$ and sample $s_{j+1}, r_j \sim \widehat{T}(s_j, a_j)$.

      Compute $\tilde{r}_j = r_j - \lambda \max_{i=1}^N \|\Sigma^i(s_j, a_j)\|_{\text{F}}$.

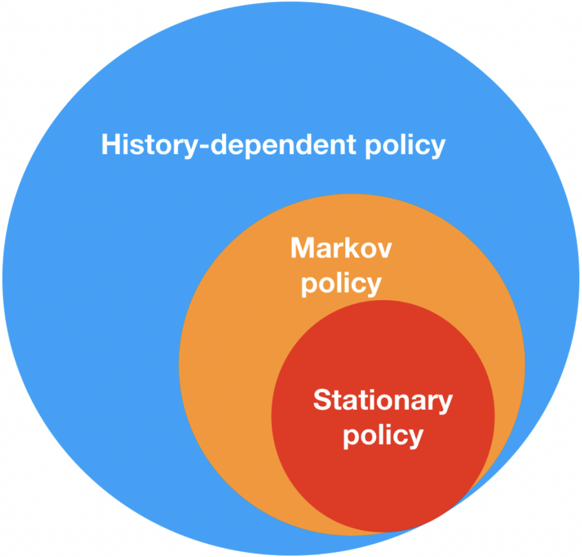
      Add sample $(s_j, a_j, \tilde{r}_j, s_{j+1})$ to $\mathcal{D}_{\text{model}}$.

  Drawing samples from $\mathcal{D}_{\text{env}} \cup \mathcal{D}_{\text{model}}$, use SAC to update π.

Lecture Outline

1. Introduction to Offline RL
2. The Pessimistic Principle
3. Model-based Offline Policy Optimization (MOPO)
4. **An Overview of My Research**

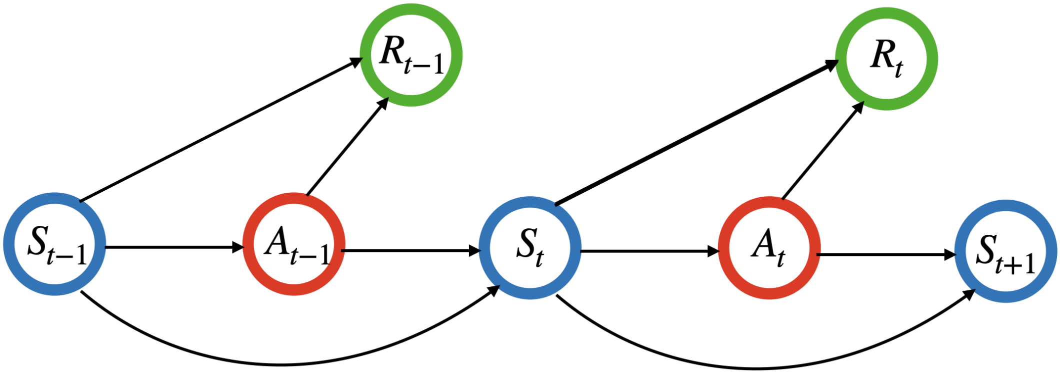
Recap: The Agent's Policy



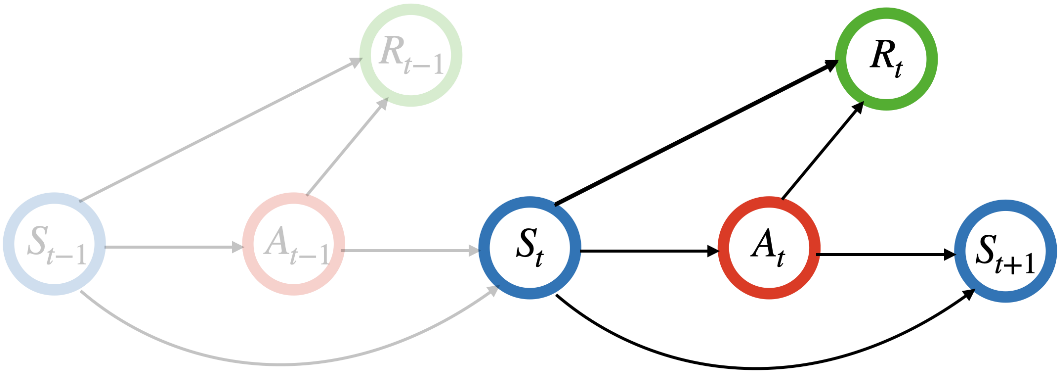
Recap: Foundations of RL

- **RL algorithms**: policy iteration, value iteration (Lecture 3), SARSA, Q-learning (Lecture 4), gradient-based methods, fitted Q-iteration (Lecture 5), deep Q-network (Lecture 7), REINFORCE, actor critic (Lecture 8), Dyna-Q (Lecture 9)
- **Foundations** of aforementioned algorithms:
  - **Markov decision process** [MDP, Puterman, 2014]: ensures the optimal policy is **stationary**, and is **not** history-dependent
  - **Markov assumption**: conditional on the present (e.g., S_t, A_t), the future (e.g., R_t, S_{t+1}) and the past data history are independent
  - **Time-homogeneity assumption**: The conditional distribution of (R_t, S_{t+1}) given $(S_t = s, A_t = a)$ is time-homogeneous

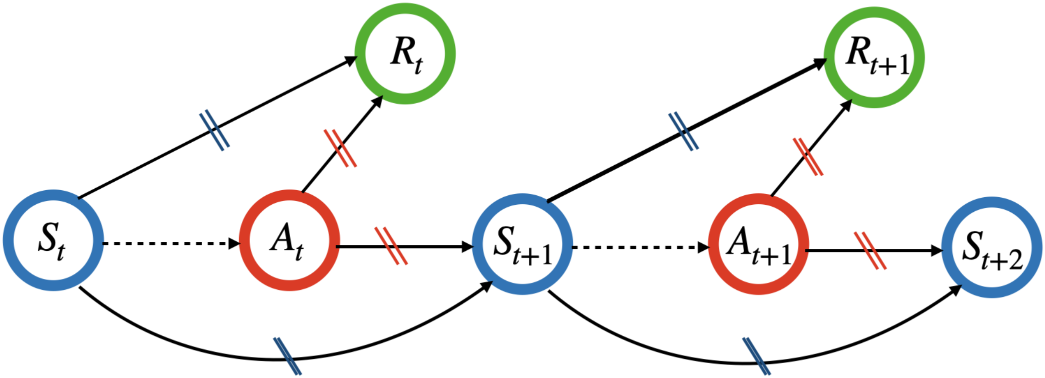
Recap: Markov Assumption



Recap: Markov Assumption



Recap: Time-Homogeneity Assumption



Violation of MDP Assumption

- Violation of Markov assumption
  - Statistical hypothesis testing for model selection: MDP, high-order MDP ($\boldsymbol{k}$th order for $\boldsymbol{k} \geq \mathbf{2}$), POMOP ($\infty$th order MDP)
- Violation of time-homogeneity assumption
  - Statistical hypothesis testing for selecting the "best data segment"

Markov and Non-Markov Models

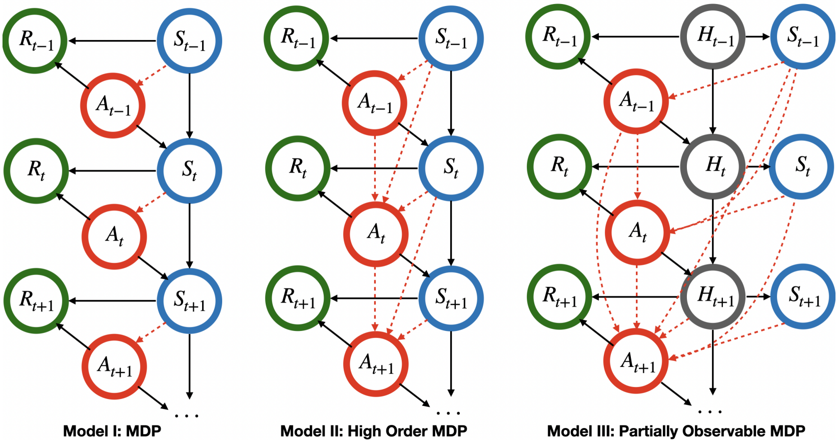
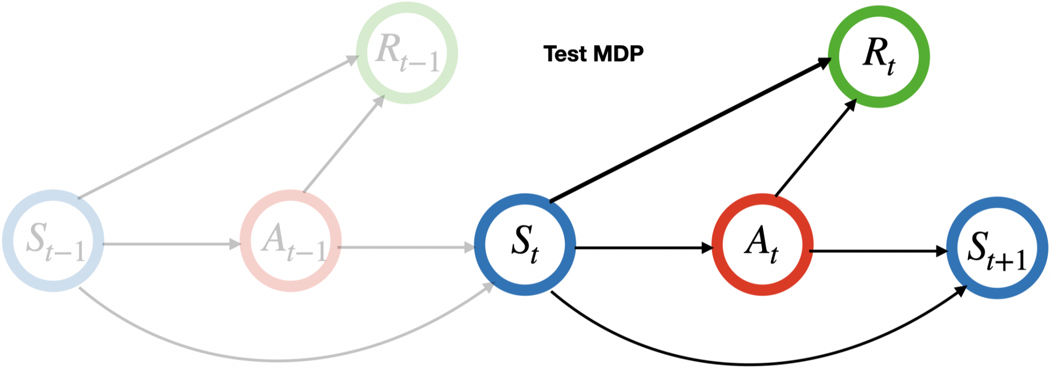


Figure: Causal diagrams for MDPs, HMDPs and POMDPs. The solid lines represent the causal relationships and the dashed lines indicate the information needed to implement the optimal policy. $\{\boldsymbol{H}_t\}_t$ denotes latent variables.

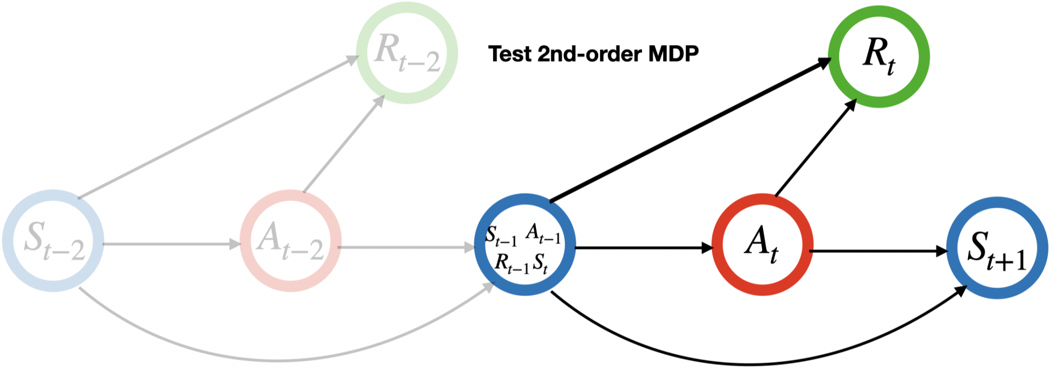
Test Markov Assumption [Shi et al., 2020]

- Develop a **forward-backward learning procedure** to test the Markov assumption (MA) in RL
  - Null hypothesis $\mathcal{H}_0$: MA holds (MDP)
  - Alternative hypothesis $\mathcal{H}_1$: MA is violated (high-order MDP, POMDP)
- Sequentially apply the test for **model selection**
  - Suppose the data follows a Kth order MDP
  - Sequentially test whether it is kth order for $k = 1, 2, \cdots$
  - by concatenating S_t with $\{(S_{t-j}, A_{t-j}, R_{t-j})\}$ for $1 \le j < k$
  - $\mathcal{H}_0$ holds when $k \ge K$ and $\mathcal{H}_1$ holds otherwise
  - Select the model when $\mathcal{H}_0$ is not rejected for the first time

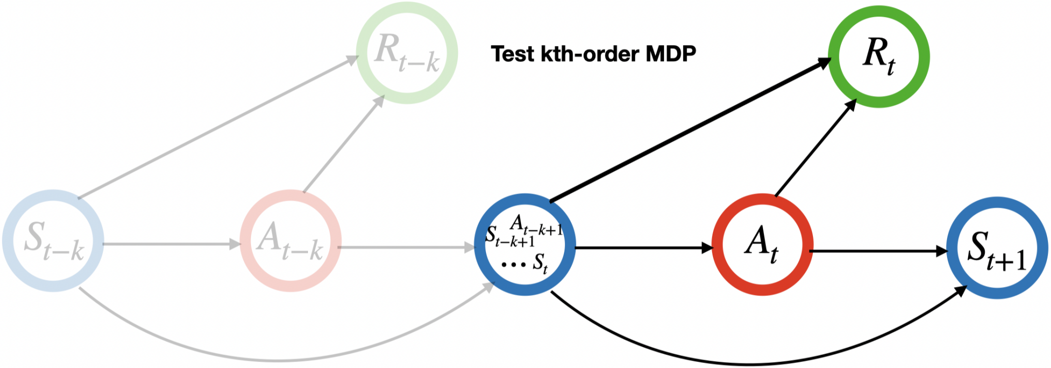
Test Markov Assumption (Cont'd)



Test Markov Assumption (Cont'd)



Test Markov Assumption (Cont'd)

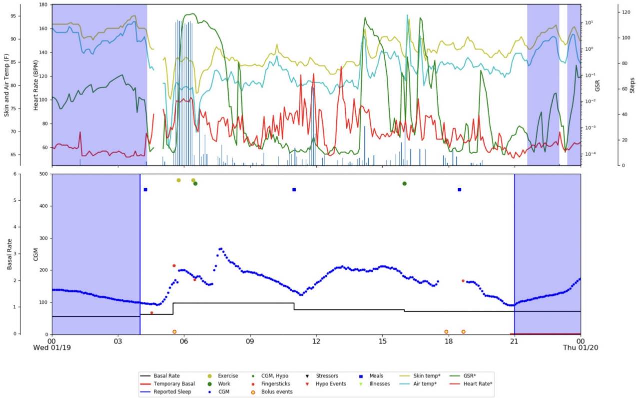


Test Markov Assumption (Cont'd)

- Uncritical to **online** domains:
  - Try different models online and see which model yields the best reward
- Critical to **offline** domains:
  - $\boldsymbol{K}$ remains unknown without prior knowledge
  - Cannot adaptively generate data
  - For **under-fitted** models ($\boldsymbol{k} < \boldsymbol{K}$), any stationary policy is not optimal
  - For **over-fitted** models ($\boldsymbol{k} > \boldsymbol{K}$), the estimated policy might be very noisy due to the inclusion of many irrelevant lagged variables

Diabetes

- Management of **Type-I diabetes**
- **Subject**: Patients with diabetes.
- **Objective**: Develop treatment policy to determine whether patients need to inject insulin at each time to improve their health
- S_t: Patient's **glucose levels**, **food intake**, **exercise intensity**
- A_t: **Insulin doses** injected
- R_t: **Index of Glycemic Control** (function of patient's glucose level)

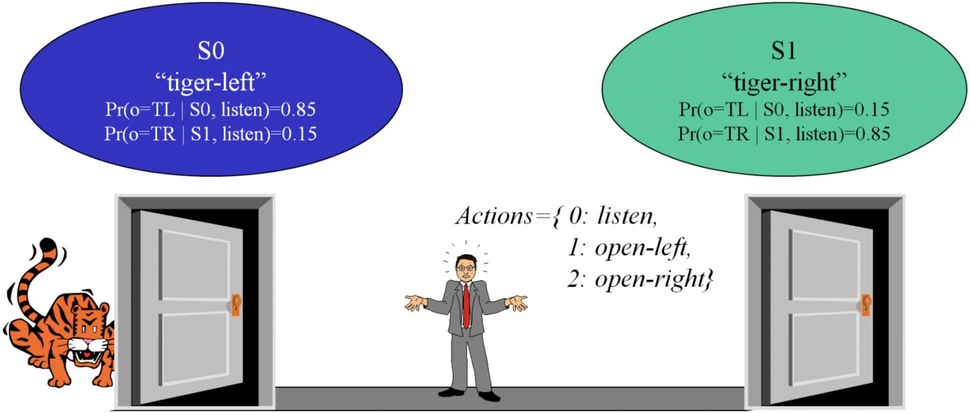


Diabetes (Cont'd)

- **Analysis I**:
  - sequentially apply our test to determine the order of MDP
  - conclude it is a **fourth-order** MDP
- **Analysis II**:
  - split the data into training/testing samples
  - policy optimization based on **fitted-Q iteration**, by assuming it is a k-th order MDP for $k = 1, \cdots, 10$
  - policy evaluation based on **fitted-Q evaluation** (to be covered in Lecture 11)
  - use **random forest** to model the Q-function
  - repeat the above procedure to compute the average value of policies computed under each MDP model assumption

order	1	2	3	4	5	6	7	8	9	10
value	-90.8	-57.5	-63.8	**-52.6**	-56.2	-60.1	-63.7	-54.9	-65.1	-59.6

Tiger



Reward Function

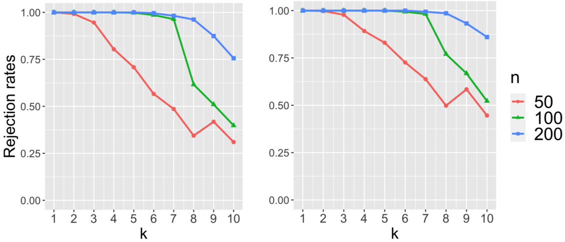
- *Penalty for wrong opening: -100*
- *Reward for correct opening: +10*
- *Cost for listening action: -1*

Observations

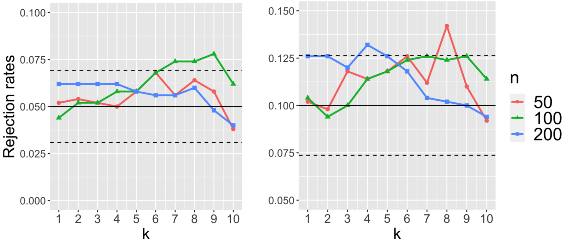
- *to hear the tiger on the left (TL)*
- *to hear the tiger on the right(TR)*

Tiger (Cont’d)

- Under the alternative hypothesis (MA is violated). $\alpha = (0.05, 0.1)$ from left to right.



- Under the null hypothesis (MA holds). $\alpha = (0.05, 0.1)$ from left to right.

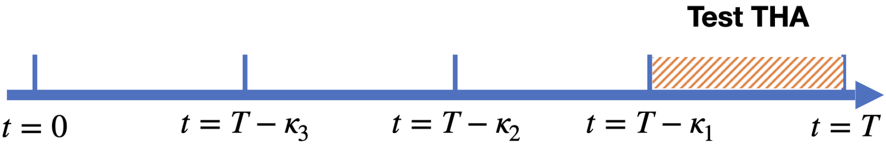


Test Time-Homogeneity [Li et al., 2022]

- Under **time-inhomogeneity**, using all data is not reasonable
- Natural to use **more recent observations** for policy optimisation
- **Challenging** to select the **best data "segment"**
  - Including too many past observations yields a suboptimal policy
  - Using only a few recent observations results in a very noisy policy
- Develop a **test procedure** for the time-homogeneity assumption (THA) in RL
  - Null hypothesis $\boldsymbol{\mathcal{H}_0}$: THA holds (MDP)
  - Alternative hypothesis $\boldsymbol{\mathcal{H}_1}$: THA is violated (Time-Varying MDP)
- Sequentially apply the test for selecting the **best data "segment"**

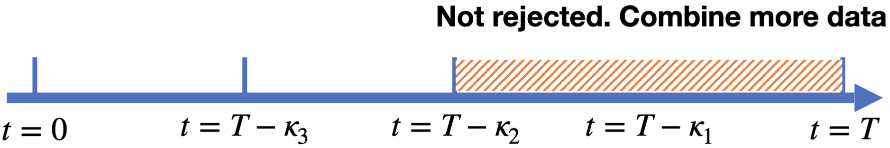
Test Time-Homogeneity (Cont'd)

- Sequentially apply the test for selecting the **best data "segment"**
  - Sequentially test whether THA holds on the data interval $[T - \kappa, T]$ for $\kappa_1 < \kappa_2 < \kappa_3 < \cdots$
  - Suppose THA is first rejected at some $\kappa = \kappa_{j_0}$
  - Use the data subset within the interval $[T - \kappa_{j_0-1}, T]$ for policy optimisation



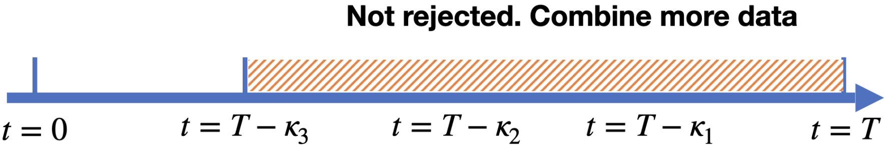
Test Time-Homogeneity (Cont'd)

- Sequentially apply the test for selecting the **best data "segment"**
  - Sequentially test whether THA holds on the data interval $[T - \kappa, T]$ for $\kappa_1 < \kappa_2 < \kappa_3 < \cdots$
  - Suppose THA is first rejected at some $\kappa = \kappa_{j_0}$
  - Use the data subset within the interval $[T - \kappa_{j_0 - 1}, T]$ for policy optimisation



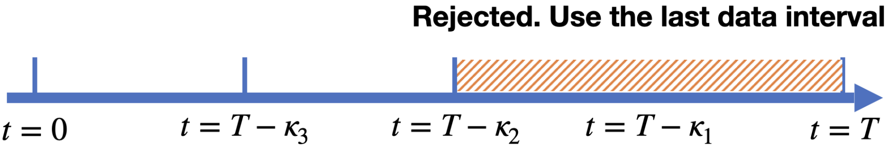
Test Time-Homogeneity (Cont'd)

- Sequentially apply the test for selecting the **best data "segment"**
  - Sequentially test whether THA holds on the data interval $[\boldsymbol{T} - \boldsymbol{\kappa}, \boldsymbol{T}]$ for $\boldsymbol{\kappa_1} < \boldsymbol{\kappa_2} < \boldsymbol{\kappa_3} < \cdots$
  - Suppose THA is first rejected at some $\boldsymbol{\kappa} = \boldsymbol{\kappa_{j_0}}$
  - Use the data subset within the interval $[\boldsymbol{T} - \boldsymbol{\kappa_{j_0 - 1}}, \boldsymbol{T}]$ for policy optimisation



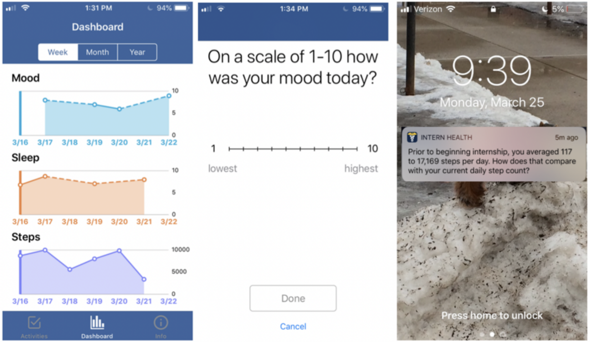
Test Time-Homogeneity (Cont'd)

- Sequentially apply the test for selecting the **best data "segment"**
  - Sequentially test whether THA holds on the data interval $[T - \kappa, T]$ for $\kappa_1 < \kappa_2 < \kappa_3 < \cdots$
  - Suppose THA is first rejected at some $\kappa = \kappa_{j_0}$
  - Use the data subset within the interval $[T - \kappa_{j_0 - 1}, T]$ for policy optimisation



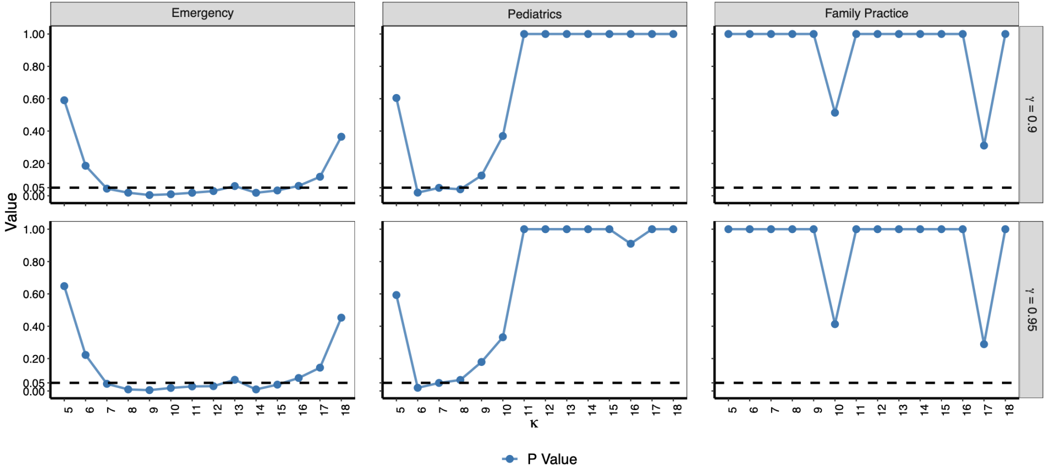
Intern Health Study

- **Subject**: First-year medical interns
- **Objective**: Develop treatment policy to determine whether to send certain text messages to interns to improve their health
- S_t: Interns' **mood scores**, **sleep hours** and **step counts**
- A_t: Send **text notifications** or not
- R_t: **Step counts**



(i) App Dashboard (ii) Mood EMA (iii) Notifications

Intern Health Study (Cont'd)



Intern Health Study (Cont'd)

Number of Change Points	Specialty	Method	$\gamma = 0.9$	$\gamma = 0.95$
≥ 1	Emergency	Proposed	8237.16	8295.99
		Overall	8108.13	8127.55
		Behavior	7823.75	7777.32
		Random	8114.78	8080.27
≥ 2	Pediatrics	Proposed	7883.08	7848.57
		Overall	7925.44	7960.12
		Behavior	7730.98	7721.29
		Random	7807.52	7815.30
0	Family Practice	Proposed	8062.50	7983.69
		Overall	8062.50	7983.69
		Behavior	7967.67	7957.24
		Random	7983.52	7969.31

TABLE 3

Mean value estimates using decision tree in anaylsis of IHS. Values are normalised by multiplying $1-\gamma$. *All values are evaluated over 10 splits of data.*

- Mean value is the weekly average step counts per day
- The proposed method improves mean value by 50 – 150 steps, compared to the behavior policy

Summary

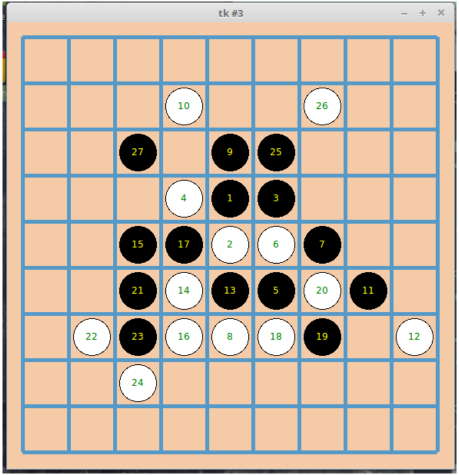
- Offline RL v.s. online RL
- The pessimistic principle
- Lower confidence bound

- Model-based offline policy optimisation

- Statistical hypothesis testing

Seminar Exercise

- Solutions to HW9 (Deadline: Wed 12 pm)
- Implementation of AlphaZero on Gomoku



References I

Ying Jin, Zhuoran Yang, and Zhaoran Wang. Is pessimism provably efficient for offline rl? In *International Conference on Machine Learning*, pages 5084–5096. PMLR, 2021.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191, 2020.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Mengbing Li, Chengchun Shi, Zhenke Wu, and Piotr Fryzlewicz. Reinforcement learning in possibly nonstationary environments. *arXiv preprint arXiv:2203.01707*, 2022.

Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

References II

Chengchun Shi, Runzhe Wan, Rui Song, Wenbin Lu, and Ling Leng. Does the markov decision process fit the data: Testing for the markov property in sequential decision making. *arXiv preprint arXiv:2002.01751*, 2020.

Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, 33:14129–14142, 2020.

Questions

Appendix: Proof of Regret

Consider the regret of greedy action selection first. Let a^* denote the action selected by the greedy policy. By definition, the regret is given by $Q(a^{opt}) - Q(a^*)$. Notice that

$$\begin{aligned} Q(a^{opt}) - Q(a^*) &= Q(a^{opt}) - \widehat{Q}(a^{opt}) + \widehat{Q}(a^{opt}) - \widehat{Q}(a^*) + \widehat{Q}(a^*) - Q(a^*) \\ &\leq Q(a^{opt}) - \widehat{Q}(a^{opt}) + \widehat{Q}(a^*) - Q(a^*), \end{aligned}$$

as a^* maximizes $\arg\max_a \widehat{Q}(a)$ by definition.

It is immediate to see that the right-hand-side is upper bounded by $2 \max_a |\widehat{Q}(a) - Q(a)|$. The proof is thus completed.

Appendix: Proof of Regret (Cont'd)

Next, consider the regret of the LCB algorithm. Let $\boldsymbol{a}^*$ denote the action selected by the LCB algorithm. By definition of $\boldsymbol{L}(\boldsymbol{a}^*)$, we have with probability approaching $\mathbf{1}$ that

$$\boldsymbol{Q}(\boldsymbol{a}^{opt}) - \boldsymbol{Q}(\boldsymbol{a}^*) \leq \boldsymbol{Q}(\boldsymbol{a}^{opt}) - \widehat{\boldsymbol{Q}}(\boldsymbol{a}^*) + \boldsymbol{L}(\boldsymbol{a}^*).$$

According to the LCB algorithm, $\widehat{\boldsymbol{Q}}(\boldsymbol{a}^*) - \boldsymbol{L}(\boldsymbol{a}^*) \geq \widehat{\boldsymbol{Q}}(\boldsymbol{a}^{opt}) - \boldsymbol{L}(\boldsymbol{a}^{opt})$. It follows that the right-hand-side is upper bounded by

$$\boldsymbol{Q}(\boldsymbol{a}^{opt}) - \widehat{\boldsymbol{Q}}(\boldsymbol{a}^{opt}) + \boldsymbol{L}(\boldsymbol{a}^{opt}),$$

which is further bounded by $\mathbf{2}\boldsymbol{L}(\boldsymbol{a}^{opt})$, by definition. The proof is completed by directly applying Hoeffding's inequality.