Reinforcement learning for partially observed systems

Aditya Mahajan
McGill University

CESG Seminar, TAMU
14 Oct, 2022

email: aditya.mahajan@mcgill.ca
homepage: http://cim.mcgill.ca/~adityam
Recent successes of RL
Recent successes of RL

Alpha Go
Recent successes of RL

Arcade games
Recent successes of RL

Robotic grasping
Recent successes of RL
- Algorithms based on comprehensive theory

Robotic grasping

RL for partially observed systems–(Mahajan)
Recent successes of RL

- Algorithms based on comprehensive theory
- The theory is restricted almost exclusively to systems with perfect state observations.
Recent successes of RL
- Algorithms based on comprehensive theory
- The theory is restricted almost exclusively to systems with perfect state observations.

Many real-world applications are partially observed
- Healthcare
- Autonomous driving
- Finance (portfolio management)
- Retail and marketing
Recent successes of RL
- Algorithms based on comprehensive theory
- The theory is restricted almost exclusively to systems with **perfect state observations**.

Many real-world applications are **partially observed**
- Healthcare
- Autonomous driving
- Finance (portfolio management)
- Retail and marketing

How do we develop a theory for RL for partially observed systems?
Outline

Background

- Review of MDPs and RL
- Review of POMDPs
- Why is RL for POMDPs difficult?
Outline

Background
- Review of MDPs and RL
- Review of POMDPs
- Why is RL for POMDPs difficult?

Approximate Planning for POMDPs
- Preliminaries on information state
- Approximate information state
- Approximation bounds
Outline

**Background**
- Review of MDPs and RL
- Review of POMDPs
- Why is RL for POMDPs difficult?

**Approximate Planning for POMDPs**
- Preliminaries on information state
- Approximate information state
- Approximation bounds

**RL for POMDPs**
- From approximation bounds to RL
- Numerical experiments

RL for partially observed systems–(Mahajan)
Review: Markov decision processes (MDPs)

MDP: MARKOV DECISION PROCESS

Dynamics: \( \mathbb{P}(S_{t+1} | S_t, A_t) \)
Observations: \( S_t \)
Reward \( R_t = r(S_t, A_t) \).

Action: \( A_t \sim \pi_t(S_{1:t}, A_{1:t-1}) \).
\( \pi = (\pi_t)_{t \geq 1} \) is called a policy.

The objective is to choose a policy \( \pi \) to maximize:

\[
J(\pi) := \mathbb{E}^{\pi} \left[ \sum_{t=1}^{\infty} \gamma^{t-1} R_t \right]
\]
**MDP: MARKOV DECISION PROCESS**

- **Dynamics:** $\mathbb{P}(S_{t+1} | S_t, A_t)$
- **Observations:** $S_t$
- **Reward** $R_t = r(S_t, A_t)$
- **Action:** $A_t \sim \pi_t(S_1:t, A_{1:t-1})$

$\pi = (\pi_t)_{t \geq 1}$ is called a **policy**.

The objective is to choose a policy $\pi$ to maximize:

$$J(\pi) \equiv \mathbb{E}_\pi \left[ \sum_{t=1}^{\infty} \gamma^{t-1} R_t \right]$$

**Conceptual challenge**

- Brute force search has an exponential complexity in time horizon.
- How to efficiently search an optimal policy?
**Review: Markov decision processes (MDPs)**

**Key simplifying ideas**

**Principle of Irrelevant Information**

| Structure of optimal policy | There is no loss of optimality in choosing the action $A_t$ as a function of the current state $S_t$ |

---

**Review: Markov decision processes (MDPs)**

**Key simplifying ideas**

**Principle of Irrelevant Information**

| Structure of optimal policy | There is no loss of optimality in choosing the action $A_t$ as a function of the current state $S_t$ |


**Principle of Optimality**

| Dynamic Program | The optimal control policy is given a DP with state $S_t$: $V(s) = \max_{a \in A} \left\{ r(s, a) + \gamma \int V(s')P(ds'|s, a) \right\}$ |

Review: Reinforcement Learning (RL)

The (online) RL setting
- Dynamics and reward functions are unknown.
- Agent can interact with the environment and observe states and rewards.
- Design an algorithm that asymptotically identifies an optimal policy.

Dynamics and reward functions are unknown. Agent can interact with the environment and observe states and rewards. Design an algorithm that asymptotically identifies an optimal policy.
The (online) RL setting

- Dynamics and reward functions are unknown.
- Agent can interact with the environment and observe states and rewards.
- Design an algorithm that asymptotically identifies an optimal policy.

Value based methods

Estimate the Q-function $Q(s, a) = r(s, a) + \gamma \int V(s') P(ds'|s, a)$ using temporal difference learning (i.e., stochastic approximation).

[Watkins and Dayan, 1992; Tsitsiklis, 1994]

Policy-based methods

Use parameterized policies $\pi_\theta$. Estimate $V_\theta V_\theta(s)$ using single trajectory gradient estimates (i.e., infinitesimal perturbation analysis).

[Sutton 2000, Marbach and Tsitsiklis 2001, Cao, 1985; Ho, 1987]
Why is learning difficult in partially observable environments?
**Review: Planning in partially observable environments**

**POMDP: PARTIALLY OBSERVABLE MARKOV DECISION PROCESS**

- **Dynamics:** $\mathbb{P}(S_{t+1} | S_t, A_t)$
- **Observations:** $\mathbb{P}(Y_t | S_t)$
- **Reward:** $R_t = r(S_t, A_t)$.

**Action:** $A_t \sim \pi_t(Y_{1:t}, A_{1:t-1})$.

$\pi = (\pi_t)_{t \geq 1}$ is called a policy.

The objective is to choose a policy $\pi$ to maximize:

$$J(\pi) := \mathbb{E}^\pi \left[ \sum_{t=1}^{\infty} \gamma^{t-1} R_t \right]$$

**RL for partially observed systems—(Mahajan)**
**Review: Planning in partially observable environments**

**POMDP: PARTIALLY OBSERVABLE MARKOV DECISION PROCESS**

- **Dynamics:** $\mathbb{P}(S_{t+1} \mid S_t, A_t)$
- **Observations:** $\mathbb{P}(Y_t \mid S_t)$
- **Reward:** $R_t = r(S_t, A_t)$.

**Action:** $A_t \sim \pi_t(Y_{1:t}, A_{1:t-1})$.

$\pi = (\pi_t)_{t \geq 1}$ is called a policy.

The objective is to choose a policy $\pi$ to maximize:

$$J(\pi) = \mathbb{E}_{\pi} \left[ \sum_{t=1}^{\infty} \gamma^{t-1} R_t \right]$$

**Conceptual challenge**

- Action is a function of the history of observations and actions.
- The history is increasing in time. So, the search complexity increases exponentially in time.
**Key simplifying idea**

Define **belief state** $B_t \in \Delta(S)$ as $B_t(s) = \mathbb{P}(S_t = s \mid Y_{1:t}, A_{1:t-1})$.

- Belief state updates in a state-like manner
  $$B_{t+1} = \text{function}(B_t, Y_{t+1}, A_t).$$

- Belief state is sufficient to evaluate rewards
  $$\mathbb{E}[R_t \mid Y_{1:t}, A_{1:t}] = \hat{r}(B_t, A_t).$$

Thus, $\{B_t\}_{t \geq 1}$ is a perfectly observed controlled Markov process.

---

**Review: Planning in partially observable environments**

**Key simplifying idea**

Define **belief state** $B_t \in \Delta(\mathcal{S})$ as $B_t(s) = \mathbb{P}(S_t = s \mid Y_{1:t}, A_{1:t-1})$.

- Belief state updates in a state-like manner
  $B_{t+1} = \text{function}(B_t, Y_{t+1}, A_t)$.

- Belief state is sufficient to evaluate rewards
  $\mathbb{E}[R_t \mid Y_{1:t}, A_{1:t}] = \hat{r}(B_t, A_t)$.

Thus, $\{B_t\}_{t \geq 1}$ is a **perfectly observed** controlled Markov process. Therefore:

**Structure of optimal policy**

There is no loss of optimality in choosing the action $A_t$ as a function of the belief state $B_t$.

**Dynamic Program**

The optimal control policy is given a DP with belief $B_t$ as state.

---

RL for partially observed systems–(Mahajan)
Implications of the POMDP modeling framework

**Implications for planning**
- Allows the use of the MDP machinery for partially observed systems.
- Various exact and approximate algorithms to efficiently solve the DP.
  - **Exact:** incremental pruning, witness algorithm, linear support algo
  - **Approximate:** QMDP, point based methods, SARSOP, DESPOT, . . .
### Implications of the POMDP modeling framework

<table>
<thead>
<tr>
<th>Implications for learning</th>
</tr>
</thead>
<tbody>
<tr>
<td>➤ The construction of the belief state depends on the system model.</td>
</tr>
<tr>
<td>➤ So, when the system model is unknown, we cannot construct the belief state and therefore cannot use standard RL algorithms.</td>
</tr>
</tbody>
</table>

---

*RL for partially observed systems–(Mahajan)*
Implications of the POMDP modeling framework

- Allows the use of the MDP machinery for partially observed systems.

### Implications for learning

- The construction of the belief state depends on the system model.
- So, when the system model is unknown, we cannot construct the belief state and therefore cannot use standard RL algorithms.

**On the theoretical side:**

- Propose alternative methods: PSRs (predictive state representations), bisimulation metrics, . . .
- Good theoretical guarantees, but difficult to scale.
### Implications of the POMDP modeling framework

#### Allows the use of the MDP machinery for partially observed systems.

#### Implications for planning

- The construction of the belief state depends on the system model.
- So, when the system model is unknown, we cannot construct the belief state and therefore cannot use standard RL algorithms.

#### Implications for learning

- **On the theoretical side:**
  - Propose alternative methods: PSRs (predictive state representations), bisimulation metrics, ...  
  - Good theoretical guarantees, but difficult to scale.

- **On the practical side:**
  - Simply stack the previous $k$ observations and treat it as a “state”.
  - Instead of a CNN, use an RNN to model policy and action-value fn.
  - Can be made to work but lose theoretical guarantees and insights.
Our result: A theoretically grounded method for RL in partially observable models which has strong empirical performance for high-dimensional environments.
Outline

**Background**
- Review of MDPs and RL
- Review of POMDPs
- Why is RL for POMDPs difficult?

**Approximate Planning for POMDPs**
- Preliminaries on information state
- Approximate information state
- Approximation bounds

**RL for POMDPs**
- From approximation bounds to RL
- Numerical experiments

RL for partially observed systems–(Mahajan)
System model

- In RL, unobserved state space may not be known
- So, we work directly with input-output model
System model

Control input: $A_t$  →  Stochastic System  →  Output: $Y_t$

Stochastic input: $W_t$  →  Reward: $R_t$

$Y_t = f_t(A_{1:t}, W_{1:t})$,  
$R_t = r_t(A_{1:t}, W_{1:t})$.

In RL, unobserved state space may not be known
So, we work directly with input-output model
Stochastic System

Control input: $A_t$  
Stochastic input: $W_t$  
Output: $Y_t$  
Reward: $R_t$

$Y_t = f_t(A_{1:t}, W_{1:t})$,  
$R_t = r_t(A_{1:t}, W_{1:t})$.

$H_t = (Y_{1:t-1}, A_{1:t-1})$ denotes the history of all data available to the agent at time $t$.

Agent chooses an $A_t \sim \pi_t(H_t)$.

$\pi = (\pi_1, \pi_2, \ldots)$ denotes the control policy.

The objective is to choose a policy $\pi$ to maximize:

$$J(\pi) := \mathbb{E}^\pi \left[ \sum_{t=1}^{\infty} \gamma^{t-1} R_t \right]$$

In RL, unobserved state space may not be known.

So, we work directly with input-output model.
Key solution concept: **Information state**

Informally, an information state is a compression of information which is sufficient for performance evaluation and predicting itself.
Key solution concept: **Information state**

Informally, an information state is a compression of information which is sufficient for performance evaluation and predicting itself.

**Historical overview**

- **Old concept.** May be viewed as a generalization of the notion of state (Nerode, 1958).
- Informal definitions given in Kwakernaak (1965), Bohlin (1970), Davis and Varaiya (1972), Kumar and Varaiya (1986) but no formal analysis.
- Related to but different from concepts such as bisimulation, predictive state representations (PSR), and $\varepsilon$-machines.
Information state: Definition

Given a Banach space $\mathcal{Z}$, an INFORMATION STATE GENERATOR is a tuple of

- history compression functions $\{\sigma_t : \mathcal{H}_t \to \mathcal{Z}\}_{t \geq 1}$
- reward function $\hat{r} : \mathcal{Z} \times \mathcal{A} \to \mathbb{R}$
- transition kernel $\hat{P} : \mathcal{Z} \times \mathcal{A} \to \Delta(\mathcal{Z})$

which satisfies two properties:
Given a Banach space \( \mathcal{Z} \), an INFORMATION STATE GENERATOR is a tuple of
- history compression functions \( \{\sigma_t : \mathcal{H}_t \to \mathcal{Z}\}_{t \geq 1} \)
- reward function \( \hat{r} : \mathcal{Z} \times \mathcal{A} \to \mathbb{R} \)
- transition kernel \( \hat{P} : \mathcal{Z} \times \mathcal{A} \to \Delta(\mathcal{Z}) \)

which satisfies two properties:

(P1) The reward function \( \hat{r} \) is sufficient for performance evaluation:

\[
\mathbb{E}[R_t \mid H_t = h_t, A_t = a_t] = \hat{r}(\sigma_t(h_t), a_t).
\]
Information state: Definition

Given a Banach space $\mathcal{Z}$, an INFORMATION STATE GENERATOR is a tuple of

- history compression functions $\{\sigma_t : \mathcal{H}_t \to \mathcal{Z}\}_{t \geq 1}$
- reward function $\hat{r} : \mathcal{Z} \times \mathcal{A} \to \mathbb{R}$
- transition kernel $\hat{P} : \mathcal{Z} \times \mathcal{A} \to \Delta(\mathcal{Z})$

which satisfies two properties:

(P1) The reward function $\hat{r}$ is sufficient for performance evaluation:

$$\mathbb{E}[R_t \mid H_t = h_t, A_t = a_t] = \hat{r}(\sigma_t(h_t), a_t).$$

(P2) The transition kernel $\hat{P}$ is sufficient for predicting the info state:

$$\mathbb{P}(Z_{t+1} \in B \mid H_t = h_t, A_t = a_t) = \hat{P}(B \mid \sigma_t(h_t), a_t).$$
An information state *always* leads to a dynamic programming decomposition.
Let $\{Z_t\}_{t \geq 1}$ be any information state process. Let $\hat{V}$ be the fixed point of:

$$\hat{V}(z) = \max_{a \in \mathcal{A}} \left\{ \hat{r}(z, a) + \gamma \int \hat{V}(z_+) \hat{P}(dz_+|z, a) \right\}$$

Let $\pi^*(z)$ denote the arg max of the RHS. Then, the policy $\pi = (\pi_t)_{t \geq 1}$ given by $\pi_t = \pi^* \circ \sigma_t$ is optimal.
Examples of information state

- Markov decision processes (MDP)
  - Current state $S_t$ is an info state

- POMDP
  - Belief state is an info state
### Examples of information state

<table>
<thead>
<tr>
<th>Markov decision processes (MDP)</th>
<th>MDP with delayed observations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current state $S_t$ is an info state</td>
<td>$(S_{t-\delta+1}, A_{t-\delta+1:t-1})$ is an info state</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>POMDP</th>
<th>POMDP with delayed observations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Belief state is an info state</td>
<td>$(\mathbb{P}(S_{t-\delta}</td>
</tr>
</tbody>
</table>
Examples of information state

<table>
<thead>
<tr>
<th>Markov decision processes (MDP)</th>
<th>MDP with delayed observations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current state $S_t$ is an info state</td>
<td>$(S_{t-\delta+1}, A_{t-\delta+1:t-1})$ is an info state</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>POMDP</th>
<th>POMDP with delayed observations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Belief state is an info state</td>
<td>$(\mathbb{P}(S_{t-\delta}</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Linear Quadratic Gaussian (LQG)</th>
<th>Machine Maintenance</th>
</tr>
</thead>
<tbody>
<tr>
<td>The state estimate $\mathbb{E}[S_t</td>
<td>H_t]$ is an info state</td>
</tr>
</tbody>
</table>
Main idea

- Info state is defined in terms of two properties (P1) & (P2).
- An AIS is a process which satisfies these approximately.
Main idea

- Info state is defined in terms of two properties (P1) & (P2).
- An AIS is a process which satisfies these **approximately**
- Show that AIS always leads to approx. DP
- Recover (and improve upon) many existing results

And now to Approximate Information States . . .
An \((\varepsilon, \delta)\)-APPROXIMATE INFORMATION STATE (AIS) generator is a tuple \((\sigma_t, \hat{r}, \hat{P})\) which approximately satisfies (P1) and (P2):
An \((\varepsilon, \delta)\)-APPROXIMATE INFORMATION STATE (AIS) generator is a tuple \((\sigma_t, \hat{r}, \hat{P})\) which approximately satisfies (P1) and (P2):

(AP1) \(\hat{r}\) is sufficient for approximate performance evaluation:

\[
|\mathbb{E}[R_t \mid H_t = h_t, A_t = a_t] - \hat{r}(\sigma_t(h_t), a_t)| \leq \varepsilon
\]
Approximate Information state: Definition

An \((\varepsilon, \delta)\)-APPROXIMATE INFORMATION STATE (AIS) generator is a tuple \((\sigma_t, \hat{r}, \hat{P})\) which approximately satisfies (P1) and (P2):

(AP1) \(\hat{r}\) is sufficient for approximate performance evaluation:

\[
\left| \mathbb{E}[R_t \mid H_t = h_t, A_t = a_t] - \hat{r}(\sigma_t(h_t), a_t) \right| \leq \varepsilon
\]

(AP2) \(\hat{P}\) is sufficient for approximately predicting next AIS:

\[
d_{\mathcal{S}}(\mathbb{P}(Z_{t+1} = \cdot \mid H_t = h_t, A_t = a_t), \hat{P}(\cdot \mid \sigma_t(h_t), a_t)) \leq \delta
\]
An \((\varepsilon, \delta)\)-APPROXIMATE INFORMATION STATE (AIS) generator is a tuple \((\sigma_t, \hat{r}, \hat{P})\) which approximately satisfies (P1) and (P2):

(AP1) \(\hat{r}\) is sufficient for approximate performance evaluation:

\[
\left| \mathbb{E}[R_t \mid H_t = h_t, A_t = a_t] - \hat{r}(\sigma_t(h_t), a_t) \right| \leq \varepsilon
\]

(AP2) \(\hat{P}\) is sufficient for approximately predicting next AIS:

\[
d_{\mathcal{F}}(\mathbb{P}(Z_{t+1} = \cdot \mid H_t = h_t, A_t = a_t), \hat{P}(\cdot \mid \sigma_t(h_t), a_t)) \leq \delta
\]

Results depend on the choice of metric on probability spaces.
Let $V$ denote the optimal value and $\hat{V}$ denote the fixed point of the following equations:

$$\hat{V}(z) = \max_{a \in \mathcal{A}} \left\{ \hat{r}(z, a) + \gamma \int_{\mathcal{Z}} \hat{V}(z_+) \hat{P}(dz_+ | z, a) \right\}$$
AIS based approximation bounds

Let $V$ denote the optimal value and $\hat{V}$ denote the fixed point of the following equations:

$$\hat{V}(z) = \max_{a \in \mathcal{A}} \left\{ \hat{r}(z, a) + \gamma \int_{\mathcal{Z}} \hat{V}(z_+) \hat{P}(dz_+|z, a) \right\}$$

The value function $\hat{V}$ is approximately optimal, i.e.,

$$|V_t(h_t) - \hat{V}(\sigma_t(h_t))| \leq \alpha := \frac{\varepsilon + \gamma \rho_\pi(\hat{V}) \delta}{1 - \gamma}.$$
Let $V$ denote the optimal value and $\hat{V}$ denote the fixed point of the following equations:

$$
\hat{V}(z) = \max_{a \in \mathcal{A}} \left\{ \hat{r}(z, a) + \gamma \int_{\mathcal{X}} \hat{V}(z_{+}) \hat{P}(dz_{+}|z, a) \right\}
$$

The value function $\hat{V}$ is approximately optimal, i.e.,

$$
|V_t(h_t) - \hat{V}(\sigma_t(h_t))| \leq \alpha := \frac{\varepsilon + \gamma \rho \delta(\hat{V}) \delta}{1 - \gamma}.
$$
### AIS based approximation bounds

Let $V$ denote the optimal value and $\hat{V}$ denote the fixed point of the following equations:

$$\hat{V}(z) = \max_{a \in \mathcal{A}} \left\{ \hat{r}(z, a) + \gamma \int \hat{V}(z+) \hat{P}(dz+|z, a) \right\}$$

The value function $\hat{V}$ is approximately optimal, i.e.,

$$|V_t(h_t) - \hat{V}(\sigma_t(h_t))| \leq \alpha := \frac{\varepsilon + \gamma \rho_\delta(\hat{V})}{1 - \gamma}.$$  

Let $\hat{\pi}^*: \mathcal{Z} \rightarrow \Delta(\mathcal{A})$ be an optimal policy for $\hat{V}$. Then, the policy $\pi = (\pi_1, \pi_2, \ldots)$ where $\pi_t = \hat{\pi}^* \circ \sigma_t$ is approx. optimal:

$$V_t(h_t) - V_\pi^t(h_t) \leq 2\alpha.$$
Some remarks on AIS

- Two ways to interpret the results:
  - Given the information state space $\mathcal{Z}$, find the best compression $\sigma_t: \mathcal{H}_t \rightarrow \mathcal{Z}$
  - Given any compression function $\sigma_t: \mathcal{H}_t \rightarrow \mathcal{Z}$, find the approximation error.
Some remarks on AIS

- Two ways to interpret the results:
  - Given the information state space $\mathcal{Z}$, find the best compression $\sigma_t : \mathcal{H}_t \rightarrow \mathcal{Z}$.
  - Given any compression function $\sigma_t : \mathcal{H}_t \rightarrow \mathcal{Z}$, find the approximation error.

- Most of the existing literature on approximate DPs focuses on the first interpretation.
- The second interpretation allows us to develop AIS-based RL algorithms.
Some remarks on AIS

- Two ways to interpret the results:
  - Given the information state space $\mathcal{Z}$, find the best compression $\sigma_t : \mathcal{H}_t \rightarrow \mathcal{Z}$
  - Given any compression function $\sigma_t : \mathcal{H}_t \rightarrow \mathcal{Z}$, find the approximation error.

- Most of the existing literature on approximate DPs focuses on the first interpretation

- The second interpretation allows us to develop AIS-based RL algorithms

- Results depend on the choice of metric on probability spaces.
  - The bounds use what are known as integral probability metrics (IPM), which include many commonly used metrics:
    - Total variation
    - Wasserstein distance
    - Maximum mean discrepancy (MMD)
Examples of AIS
Example 1: Robustness to model mismatch in MDPs

What is the loss in performance if we choose a policy using the simulation model and use it in the real world?
**Example 1: Robustness to model mismatch in MDPs**

What is the loss in performance if we choose a policy using the simulation model and use it in the real world?

**Model mismatch as an AIS**

\[
\text{Identity, } \hat{P}, \hat{r} \text{ is an } (\epsilon, \delta)\text{-AIS with } \\
\epsilon = \sup_{s, a} |r(s, a) - \hat{r}(s, a)| \text{ and } \\
\delta_\delta = \sup_{s, a, \cdot} d_\delta(P(\cdot | s, a), \hat{P}(\cdot | s, a)).
\]
Example 1: Robustness to model mismatch in MDPs

Model mismatch as an AIS

$(\text{Identity}, \hat{P}, \hat{r})$ is an $(\varepsilon, \delta)$-AIS with $\varepsilon = \sup_{s,a} |r(s,a) - \hat{r}(s,a)|$ and $\delta_{\bar{\delta}} = \sup_{s,a} d_{\bar{\delta}}(P(\cdot|s,a), \hat{P}(\cdot|s,a))$.

$d_{\bar{\delta}}$ is total variation

$$V(s) - V^\pi(s) \leq \frac{2\varepsilon}{1 - \gamma} + \frac{\gamma \delta \text{span}(r)}{(1 - \gamma)^2}$$

Recover bounds of Müller (1997).
Example 1: Robustness to model mismatch in MDPs

Model mismatch as an AIS

(Identity, \(\hat{P}, \hat{r}\)) is an \((\varepsilon, \delta)\)-AIS with \(\varepsilon = \sup_{s, a} |r(s, a) - \hat{r}(s, a)|\) and \(\delta_{\hat{\gamma}} = \sup_{s, a} d_{\hat{\gamma}}(P(\cdot | s, a), \hat{P}(\cdot | s, a))\).

\(d_{\hat{\gamma}}\) is total variation

\[
V(s) - V^\pi(s) \leq \frac{2\varepsilon}{1 - \gamma} + \frac{\gamma \delta \text{span}(r)}{(1 - \gamma)^2}
\]

Recover bounds of Müller (1997).

\(d_{\hat{\gamma}}\) is Wasserstein distance

\[
V(s) - V^\pi(s) \leq \frac{2\varepsilon}{1 - \gamma} + \frac{2\gamma \delta L_r}{(1 - \gamma)(1 - \gamma L_p)}
\]


Müller, “How does the value function of a Markov decision process depend on the transition probabilities?” MOR 1997.

Example 2: Feature abstraction in MDPs

What is the loss in performance if we choose a policy using the abstract model and use it in the original model?

$$(\hat{P}, \hat{r})$$ is determined from $$(P, r)$$ using $$\varphi$$
What is the loss in performance if we choose a policy using the abstract model and use it in the original model?

Feature abstraction as AIS

\((\varphi, \hat{P}, \hat{r})\) is an \((\epsilon, \delta)\)-AIS with 
\[
\epsilon = \sup_{s,a} \left| r(s, a) - \hat{r}(\varphi(s), a) \right|
\]
and 
\[
\delta = \sup_{s,a} \text{d}_{\mathcal{S}}(P(\varphi^{-1}(\cdot)|s, a), \hat{P}(\cdot|\varphi(s), a)).
\]
Feature abstraction in MDPs

Example 2: Feature abstraction in MDPs

$$(\hat{P}, \hat{r})$$ is determined from $$(P, r)$$ using $$\phi$$

Feature abstraction as AIS

$$\phi(\hat{P}, \hat{r})$$ is an $$(\varepsilon, \delta)$$-AIS with $${\varepsilon = \sup_{s, a} |r(s, a) - \hat{r}(\phi(s), a)|}$$

and $${\delta_\mathcal{S} = \sup_{s, a} d_\mathcal{S}(P(\varphi^{-1}(\cdot)|s, a), \hat{P}(\cdot|\phi(s), a))}$$.

$d_\mathcal{S}$ is total variation

$$V(s) - V^\pi(s) \leq \frac{2\varepsilon}{1 - \gamma} + \frac{\gamma \delta_\mathcal{S} \text{span}(r)}{(1 - \gamma)^2}$$

Improve bounds of Abel et al. (2016)

RL for partially observed systems–(Mahajan)
Example 2: Feature abstraction in MDPs

$$\hat{P}, \hat{r}$$ is determined from $(P, r)$ using $\varphi$

**Feature abstraction as AIS**

$(\varphi, \hat{P}, \hat{r})$ is an $(\varepsilon, \delta)$-AIS with $\varepsilon = \sup_{s,a} |r(s, a) - \hat{r}(\varphi(s), a)|$

and $\delta_{\hat{S}} = \sup_{s,a} d_{\hat{S}}(P(\varphi^{-1}(\cdot)|s, a), \hat{P}(\cdot|\varphi(s), a))$.

$d_{\hat{S}}$ is total variation

$$V(s) - V^\pi(s) \leq \frac{2\varepsilon}{1-\gamma} + \frac{\gamma \delta_{\hat{S}} \text{span}(r)}{(1-\gamma)^2}$$

**Improve** bounds of Abel et al. (2016)

$d_{\hat{S}}$ is Wasserstein distance

$$V(s) - V^\pi(s) \leq \frac{2\varepsilon}{1-\gamma} + \frac{2\gamma \delta_{\hat{S}} \|\hat{V}\|_{\text{Lip}}}{(1-\gamma)^2}$$

Recover bounds of Gelada et al. (2019).

Example 3: Belief approximation in POMDPs

What is the loss in performance if we choose a policy using the approximate beliefs and use it in the original model?
Example 3: Belief approximation in POMDPs

What is the loss in performance if we choose a policy using the approximate beliefs and use it in the original model?

Belief approximation in POMDPs

- Quantized cells of radius $\varepsilon$ (in terms of total variation) are $(\varepsilon \| r \|_\infty, 3\varepsilon)$–AIS.
Example 3: Belief approximation in POMDPs

Belief approximation in POMDPs

Quantized cells of radius $\varepsilon$ (in terms of total variation) are $(\varepsilon \|r\|_\infty, 3\varepsilon)$-AIS.

$$V(s) - V^\pi(s) \leq \frac{2\varepsilon \|r\|_\infty}{1 - \gamma} + \frac{6\gamma \varepsilon \|r\|_\infty}{(1 - \gamma)^2}$$

**Improve** bounds of Francois Lavet et al. (2019) by a factor of $1/(1 - \gamma)$.

Thus, the notion of AIS unifies many of the approximation results in the literature, both for MDPs and POMDPs.
Outline

Background
- Review of MDPs and RL
- Review of POMDPs
- Why is RL for POMDPs difficult?

Approximate Planning for POMDPs
- Preliminaries on information state
- Approximate information state
- Approximation bounds

RL for POMDPs
- From approximation bounds to RL
- Numerical experiments
Main idea

- AIS is defined in terms of two losses $\varepsilon$ and $\delta$.
- Minimizing $\varepsilon$ and $\delta$ will minimize the AIS approximation loss.
Main idea

- AIS is defined in terms of two losses $\varepsilon$ and $\delta$.
- Minimizing $\varepsilon$ and $\delta$ will minimize the AIS approximation loss.
- Use $\lambda \varepsilon^2 + (1 - \lambda) \delta^2$ as surrogate loss for the AIS generator.
- ... and combine it with standard actor–critic algorithm using multi-timescale stochastic approximation.
Reinforcement learning setup

AIS Generator

- Use LSTM for $\sigma_t: \mathcal{H}_t \rightarrow \mathcal{Z}$ and a NN for functions $\hat{r}$ and $\hat{P}$.
- Use $\lambda(\tilde{R}_t - R_t)^2 + (1 - \lambda)d_\mathcal{G}(\mu_t, \nu_t)^2$ as surrogate loss.
- We show that $\nabla d_\mathcal{G}(\mu_t, \nu_t)^2$ can be computed efficiently for Wasserstein distance and MMD.
Reinforcement learning setup

AIS Generator

- Use LSTM for $\sigma_t: \mathcal{H}_t \rightarrow \mathcal{Z}$ and a NN for functions $\hat{r}$ and $\hat{P}$.
- Use $\lambda(\tilde{R}_t - R_t)^2 + (1 - \lambda)d_\delta(\mu_t, \nu_t)^2$ as surrogate loss.
- We show that $\nabla d_\delta(\mu_t, \nu_t)^2$ can be computed efficiently for Wasserstein distance and MMD.

Value approximator

- Use a NN to approx. action-value function $Q: \mathcal{Z} \times \mathcal{A} \rightarrow \mathbb{R}$.
- Update the parameters to minimize temporal difference loss.
Reinforcement learning setup

### AIS Generator
- Use LSTM for $\sigma_t: \mathcal{H}_t \rightarrow \mathcal{Z}$ and a NN for functions $\hat{r}$ and $\hat{P}$.
- Use $\lambda(\tilde{R}_t - R_t)^2 + (1 - \lambda)d_{\tilde{d}}(\mu_t, \nu_t)^2$ as surrogate loss.
- We show that $\nabla d_{\tilde{d}}(\mu_t, \nu_t)^2$ can be computed efficiently for Wasserstein distance and MMD.

### Policy approximator
- Use a NN to approx. policy $\pi: \mathcal{Z} \rightarrow \Delta(\mathcal{A})$.
- Use policy gradient theorem to efficiently compute $\nabla J(\pi)$.

### Value approximator
- Use a NN to approx. action-value function $Q: \mathcal{Z} \times \mathcal{A} \rightarrow \mathbb{R}$.
- Update the parameters to minimize temporal difference loss.
Reinforcement learning setup

- Use LSTM for $\sigma_t : \mathcal{H}_t \rightarrow \mathcal{Z}$ and a NN for functions $\hat{r}$ and $\hat{P}$.
- Use $\lambda(\tilde{R}_t - R_t)^2 + (1-\lambda)d\mathcal{F}(\mu_t, \nu_t)^2$ as surrogate loss.
- We show that $\nabla d\mathcal{F}(\mu_t, \nu_t)^2$ can be computed efficiently for Wasserstein distance and MMD.

Value approximator
- Use a NN to approx. action-value function $Q: \mathcal{Z} \times \mathcal{A} \rightarrow \mathbb{R}$.
- Update the parameters to minimize temporal difference loss.

Policy approximator
- Use a NN to approx. policy $\pi: \mathcal{Z} \rightarrow \Delta(\mathcal{A})$.
- Use policy gradient theorem to efficiently compute $\nabla J(\pi)$.

Convergence Guarantees
- Use multi-timescale stochastic approximation to simultaneously learn AIS generator, action-value function, and policy.
- Under appropriate technical assumptions, converges to the stationary point corresponding to the choice of function approximators.
Numerical Experiments
MiniGrid Environments

Simple Crossing  Lava Crossing  Key Corridor

Features
- Partially observable 2D grids. Agent has a view of a $7 \times 7$ field in front of it. Observations are obstructed by walls.
- Multiple entities (agents, walls, lava, boxes, doors, and keys)
- Multiple actions (Move Forward, Turn Left, Turn Right, Open Door/Box, . . . )
MiniGrid Environments

Algorithms

- AIS + MMD
  - AIS with MMD as IPM

- AIS + KL
  - AIS with KL as upper bound of Wasserstein distance

- PPO + LSTM
  - Baseline proposed in paper introducing minigrid envs
Simple Crossing

Simple Crossing S9N3

Simple Crossing S11N5

RL for partially observed systems–(Mahajan)
Key Corridor

RL for partially observed systems—(Mahajan)
Obstructed Maze

RL for partially observed systems–(Mahajan)
Summary

A conceptually clean framework for approximate DP and online RL in partially observed systems
Summary

A conceptually clean framework for approximate DP and online RL in partially observed systems

Approximation results generalize to

- observation compression
- action quantization
- lifelong learning
- multi-agent teams
Summary

A conceptually clean framework for approximate DP and online RL in partially observed systems

Approximation results generalize to
- observation compression
- action quantization
- lifelong learning
- multi-agent teams

Ongoing work
- Other RL settings such as model based RL, offline RL, inverse RL.
- A building block for multi-agent RL.
- Approximations in dynamic games
- . . .
Thank you

Funding: NSERC, DND

grip: JMLR, Feb 2022

code: https://github.com/info-structures/ais