Agent-state based policies in POMDPs: Beyond belief-state MDPs

Amit Sinha and Aditya Mahajan

***Abstract*— The traditional approach to POMDPs is to convert them into fully observed MDPs by considering a belief state as an information state. However, a belief-state based approach requires perfect knowledge of the system dynamics and is therefore not applicable in the learning setting where the system model is unknown. Various approaches to circumvent this limitation have been proposed in the literature. We present a unified treatment of some of these approaches by viewing them as models where the agent maintains a local recursively updateable "agent state" and chooses actions based on the agent state. We highlight the different classes of agent-state based policies and the various approaches that have been proposed in the literature to find good policies within each class. These include the designer's approach to find optimal non-stationary agent-state based policies, policy search approaches to find a locally optimal stationary agent-state based policies, and the approximate information state to find approximately optimal stationary agent-state based policies. We then present how ideas from the approximate information state approach have been used to improve Q-learning and actor-critic algorithms for learning in POMDPs.**

I. Introduction

Partially observable Markov decision processes (POMDPs) are a widely used model for the optimal control of dynamical systems with partial state observation. They have been extensively studied across various research communities including systems and control, operations research, and artificial intelligence.

A key conceptual challenge for POMDPs is that the data available at the agent—the history of observations and actions—is increasing with time. The standard approach is to compress this increasing data into a finite dimensional statistic known as the belief state, which is the posterior density of the unobserved state conditioned on the history of observations and actions and, therefore, may be viewed as a generalization of non-linear filtering of controlled processes. The belief state is sufficient for evaluating the per-step reward, can be updated recursively, and is strategy independent. Therefore, we can write a dynamic programming decomposition using the belief state as an information state [3], [14], [49], [76]. Furthermore, the value function of the corresponding belief-state MDP has certain qualitative properties (it is piecewise linear and convex), which can be leveraged for efficient computational algorithms. The earliest such algorithm was the "one-pass" algorithm by Smallwood and Sondik [76]. Various efficient refinements of this algorithm have been presented in the literature, including linear-support algorithm [20], witness algorithm [14], incremental pruning [13], [93], point-based methods [66], [77], [79], and others. See [38], [46] for a unified overview of the numerical methods.

There are two limitations of the belief-state based approach. First, implementing a belief-state based policy is computationally challenging. Keeping track of the belief state requires non-linear filtering, which can be approximated via particle filtering, but is still computationally heavy and is therefore difficult to implement in embedded hardware in robotics and other applications. Second, the belief state is model dependent. Therefore, it cannot be used in model-free reinforcement learning (RL).

An alternative approach, which is more amenable to the learning setting, is to consider what we call the ***agent state***. In particular, we relax the assumption that the agent can use the entire history of observations and actions (or a model-dependent compression of it like the belief state) to make a decision. Rather, we assume that the agent maintains a local state that does not depend on the model and makes a decision as a function of the agent state. Such an agent-state formulation has been proposed multiple times in the literature [29], [39], [54], [57], [78], [80]. Perhaps the simplest example of the agent state is an agent keeping track of a finite window of past observations and actions, which was first considered in [67], [88] and is commonly referred to as *frame stacking* in the RL literature [58]. It is argued in [29], [54] that in many of the popular implementations of RL algorithms, the agent state can be further decomposed into three parts: an algorithmic state which is data to be used by subsequent computations, a situational state, which is a summary of the agent's current situation, and an epistemic state, which is the summary of the agent's current knowledge of the environment. We do not pursue such a distinction here but, roughly speaking, our notion of agent state is similar to the situational state in [29], [54].

Notation

We use uppercase letters to denote random variables (e.g. S, A, etc.), lowercase letters to denote their realizations (e.g. s, a, etc.) and calligraphic letters to denote sets (e.g. $\mathcal{S}, \mathcal{A}$; etc.). Subscripts (e.g. S_t, A_t, etc.) denote variables at time t. $\Delta(\mathcal{S})$ denotes the space of probability measures on a set $\mathcal{S}$; $\mathbb{P}(\cdot)$ and $\mathbb{E}[\cdot]$ denote the probability of an event and the expectation of a random variable, respectively; and $\mathbb{1}$ denotes the indicator function.

The authors are with the Department of Electrical and Computer Engineering, McGill University, Montreal, QC, Canada. They are also affiliated with CIM, GERAD, Mila, and ILLS. Emails: amit.sinha@mail.mcgill.ca, aditya.mahajan@mcgill.ca

This research was supported in part by NSERC Alliance International Catalyst Grant ALLRP 580801-2022 and in part by a grant from Google's Institutional Research Program in collaboration with Mila.

Organization

The rest of the paper is organized as follows. In Sec. II we present the mathematical model of POMDPs, introduce the agent-state framework, and present sufficient conditions for the agent state to be an information state. In Sec. III we discuss the different classes of agent-state policies and present the three approaches for finding optimal policies among different policy classes. In Sec. IV we discuss various RL approaches taken for learning agent-state based policies. Finally, we present concluding remarks and discussions in Sec. V.

II. The POMDP model and agent-state based policies

A. System model

Consider a stochastic dynamical system with state $S_t \in \mathcal{S}$, input $A_t \in \mathcal{A}$, and output $Y_t \in \mathcal{Y}$. To simplify the discussion, we will assume that all sets are finite valued and ignore integrability and measurability issues, existence of suprema, etc. See the companion paper [26] for a nuanced discussion of these issues. The system operates in discrete time with the dynamics given as follows: for any time $t \in \mathbb{N}$, we have

$$\begin{aligned}\mathbb{P}(S_{t+1}, Y_{t+1} \mid S_{1:t}, Y_{1:t}, A_{1:t}) &= \mathbb{P}(S_{t+1}, Y_{t+1} \mid S_t, A_t) \\ &=: P(S_{t+1}, Y_{t+1} \mid S_t, A_t)\end{aligned}$$

where P is a probability transition matrix.[1] In addition, at each time the system yields a reward $R_t = r(S_t, A_t)$. We will assume that $R_t \in [0, R_{\max}]$.

There is an agent (also called a controller or a decision maker, depending on the research community) which observes the outputs of the system and chooses control actions as inputs to the system. In principle, this agent can be as sophisticated as we want and can, therefore, use the entire history of past observations and actions to choose its action, i.e.,[2]

$$A_t = \vec{\pi}_t(Y_{1:t}, A_{1:t-1}), \quad t \in \mathbb{N}$$

where $\vec{\pi}_t$ is called the control law or decision rule at time t. We are using the vector accent to highlight the fact that the policy is history dependent.

We assume that the system runs for an infinite horizon and use $\vec{\boldsymbol{\pi}} = (\vec{\pi}_1, \vec{\pi}_2, \dots)$ to denote the control policy (or simply the policy).[3] Let $\vec{\Pi}_{\text{ND}}$ denote the set of all history dependent (indicated by the vector accent) non-stationary (i.e., time-varying) and deterministic policies.

We assume that the initial state S_1 is distributed according to probability mass function ξ_1. Then, the performance of any policy $\vec{\boldsymbol{\pi}} \in \vec{\Pi}_{\text{ND}}$ is given by

$$J^{\vec{\boldsymbol{\pi}}} := \mathbb{E}^{\vec{\boldsymbol{\pi}}}\left[\sum_{t=1}^{\infty} \gamma^{t-1} R_t \;\middle|\; S_1 \sim \xi_1\right]$$

where $\gamma \in (0,1)$ is the discount factor. Let $\vec{J}^{\star}_{\text{ND}}$ (again, the vector accent highlights that we are optimizing over all history dependent policies) denote the optimal performance in $\vec{\Pi}_{\text{ND}}$, i.e.,

$$\vec{J}^{\star}_{\text{ND}} := \sup_{\vec{\boldsymbol{\pi}} \in \vec{\Pi}_{\text{ND}}} J^{\vec{\boldsymbol{\pi}}}.$$

B. Some remarks on the model

1) The system described above is referred to as a *partially observed* Markov decision process (POMDP) to highlight the fact that the agent sees *partial* observations of the state of the environment.
2) Since the per-step reward is uniformly bounded, $J^{\vec{\boldsymbol{\pi}}}$ is well defined. However, it is not immediately obvious that there exists an optimal policy $\vec{\boldsymbol{\pi}}^{\star}$ such that $\vec{J}^{\star}_{\text{ND}} = J^{\vec{\boldsymbol{\pi}}^{\star}}$.
3) There are several technical questions that need to be resolved carefully when the variables are continuous valued. We refer the reader to the companion paper [26] for a detailed discussion.
4) In the literature, it is often assumed that

$$\begin{aligned}&\mathbb{P}(S_{t+1}, Y_{t+1} \mid S_t, A_t) \\ &\quad = \mathbb{P}(S_{t+1} \mid S_t, A_t)\mathbb{P}(Y_{t+1} \mid S_{t+1}, A_t)\end{aligned}$$

or sometimes even

$$\begin{aligned}&\mathbb{P}(S_{t+1}, Y_{t+1} \mid S_t, A_t) \\ &\quad = \mathbb{P}(S_{t+1} \mid S_t, A_t)\mathbb{P}(Y_{t+1} \mid S_{t+1}).\end{aligned}$$

Such an assumption is not needed for the discussion presented in this paper.
5) In the discussion above, we have restricted attention to deterministic policies. In principle, we could have also considered non-stationary history dependent stochastic policies $\vec{\boldsymbol{\pi}} = (\vec{\pi}_1, \vec{\pi}_2, \dots)$, where $\vec{\pi}_t \colon \mathcal{H}_t \to \Delta(\mathcal{A})$. Let $\vec{\Pi}_{\text{NS}}$ denote the set of all non-stationary history dependent stochastic policies. Define

$$\vec{J}^{\star}_{\text{NS}} := \sup_{\vec{\boldsymbol{\pi}} \in \vec{\Pi}_{\text{NS}}} J^{\vec{\boldsymbol{\pi}}}.$$

By definition $\vec{\Pi}_{\text{ND}} \subseteq \vec{\Pi}_{\text{NS}}$; hence, $\vec{J}^{\star}_{\text{ND}} \le \vec{J}^{\star}_{\text{NS}}$. However, since the agent has perfect recall (i.e., remembers everything that it has seen and done in the past) it can be shown that ***there is no loss of optimality in***

[1] For the simplicity of notation, we are using a slightly informal notation. Terms such as $\mathbb{P}(S_{t+1}, Y_{t+1} \mid S_t, A_t)$ should either be viewed as the numerical value $\mathbb{P}(S_{t+1} = s_{t+1}, Y_{t+1} = y_{t+1} \mid S_t = s_t, A_t = a_t)$ for specific realizations $(s_{t+1}, y_{t+1}, s_t, a_t)$ of $(S_{t+1}, Y_{t+1}, S_t, A_t)$ or as probability mass function $\mathbb{P}(S_{t+1} = \cdot, Y_{t+1} = \cdot \mid S_t = s_t, A_t = a_t)$ or as a distribution-valued random variable $\mathbb{P}(S_{t+1} = \cdot, Y_{t+1} = \cdot \mid S_t, A_t)$. Typically, all of these interpretations are consistent. If a specific interpretation is needed, we will use a more elaborate notation as appropriate.

[2] Again, we are using a slightly informal notation. We can interpret the above either as $a_t = \vec{\pi}_t(y_{1:t}, a_{1:t-1})$ for special realizations $(y_{1:t}, a_{1:t})$ of $(Y_{1:t}, A_{1:t})$ or as an equality between random variables $A_t(\omega)$ and $\vec{\pi}_t(Y_{1:t}(\omega), A_{1:t-1}(\omega))$.

[3] We are making a deliberate choice of using bold $\vec{\boldsymbol{\pi}}$ to denote the policy to distinguish between control laws and control policies.

restricting attention to deterministic strategies,[4] i.e.,

$$\vec{J}^{\star}_{\text{ND}} = \vec{J}^{\star}_{\text{NS}}. \tag{1}$$

C. Agent-state based policies

We now describe the agent-state based approach, where it is assumed that instead of using the entire history of observations and actions to make a decision the agent maintains a local state (which we will refer to as the agent state and denote by $Z_t \in \mathcal{Z}$) and makes a decision as a function of the agent state. The agent starts with an initial state $Z_1 = \phi_0(Y_1)$ and recursively updates it as follows:[5]

$$Z_{t+1} = \phi(Z_t, Y_{t+1}, A_t), \quad \forall t \in \mathbb{N}. \tag{2}$$

We call ϕ_0 the *state-initialization function* and ϕ the *state-update function*. The agent chooses an action either using a deterministic control law $\pi_t \colon \mathcal{Z} \to \mathcal{A}$ as

$$A_t = \pi_t(Z_t)$$

or using a stochastic control law $\pi_t \colon \mathcal{Z} \to \Delta(\mathcal{A})$ as

$$A_t \sim \pi_t(Z_t).$$

We call such a policy as an ***agent-state based policy***.

A simple example of agent-state based policy is when the agent uses a finite window of past observations and actions, i.e., $Z_t = (Y_{t-n:t}, A_{t-n:t-1})$. Such a model is sometimes called frame-stacking in the RL literature [58], [88]. Another example is when the controller is a finite state automaton [36], [67], [69]. It is not necessary for the agent state to be finite. In fact, the belief-state representation is a special case of agent-state model where the agent state belongs to a $|\mathcal{S}|$-dimensional simplex. However, for the convenience of notation, we would assume that $\mathcal{Z}$ is finite. The discussion can be generalized to continuous valued $\mathcal{Z}$ under appropriate technical assumptions.

There are three fundamental questions when working with an agent state:

(Q1) When is there no loss of optimality in restricting attention to agent-state based policies?

(Q2) For a given agent-state update function ϕ, how do we find the optimal agent-state based policy?

(Q3) For a given agent-state space $\mathcal{Z}$, what is the optimal agent-state update function and agent-state based policy?

[4]To explain the high-level idea of the result, we borrow the terminology of pure, mixed, and behavioral strategies used in game theory. A pure strategy is what we call deterministic policy; a behavioral strategy is what we call stochastic policy. A mixed strategy is a probability distribution over pure strategies where the agent picks a pure strategy at the beginning of the game according to specified probability distribution and then follows the pure strategy throughout the game. With this terminology, the result follows from two facts. First, since we are in a single agent unconstrained expectation maximization setting, mixed and pure strategies have the same performance. Second, since we are in a perfect recall setting, mixed and behavioral strategies have the same performance due to Kuhn's theorem (see e.g., [4]).

[5]In principle, the update can be stochastic and be given by $Z_{t+1} = \phi(Z_t, Y_{t+1}, A_t, W_t)$ where $\{W_t\}_{t\geq 1}$ is a sequence of i.i.d. (independent and identically distributed) randomizing variables that are independent of all other primitive random variables.

The short answer to (Q1) is simple: ***if an agent state is an information state, then there is no loss of optimality in restricting attention to an agent-state policy.*** Of course, this answer only makes sense if we define an information state, which we do in the next section. The answers to (Q2) and (Q3) are more difficult and depend on what we mean by "optimal". We will discuss the different conceptual approaches that have been used in the literature in Sec. III.

D. Information state

We start with some notation. Let $H_t = (Y_{1:t}, A_{1:t-1})$ denote the history of observations and actions of the agent up to time t and let $\mathcal{H}_t$ denote the space of realization of all such histories. We can recursively unroll the agent-state update function (2) and define a sequence of functions $\vec{\boldsymbol{\sigma}} := (\vec{\sigma}_1, \vec{\sigma}_2, \dots)$, where $\vec{\sigma}_t \colon \mathcal{H}_t \to \mathcal{Z}$, such that $Z_t = \vec{\sigma}_t(H_t)$. In particular,

$$\vec{\sigma}_1(H_1) = \phi_0(Y_1), \;\; \vec{\sigma}_2(H_2) = \phi(\vec{\sigma}_1(H_1), Y_2, A_1), \;\; \text{etc.} \tag{3}$$

We call $\vec{\boldsymbol{\sigma}}$ to be the *history compression function* corresponding to the agent-state initialization and update functions (ϕ_0, ϕ).

The agent state is an information state if it satisfies the following two properties:

(P1) Sufficient for performance evaluation. There exists a function $r_{\text{IS}} \colon \mathcal{Z} \times \mathcal{A} \to \mathbb{R}$ such that for any t and H_t and A_t, we have

$$\mathbb{E}[R_t \mid H_t, A_t] = r_{\text{IS}}(\vec{\sigma}_t(H_t), A_t).$$

(P2) Sufficient for predicting itself. There exists a controlled transition probability matrix $P_{\text{IS}} \colon \mathcal{Z} \times \mathcal{A} \to \Delta(\mathcal{Z})$ such that for any t and H_t and A_t, we have

$$\mathbb{P}(Z_{t+1} \mid H_t, A_t) = P_{\text{IS}}(Z_{t+1} \mid \vec{\sigma}_t(H_t), A_t).$$

Recall that the agent state updates in a state-like manner (2). Thus,

$$\mathbb{P}(Z_{t+1} \mid H_t, A_t) = \sum_{y_{t+1} \in \mathcal{Y}} \mathbb{P}(y_{t+1} \mid H_t, A_t) \mathbb{1}_{\{Z_{t+1} = \phi(\vec{\sigma}_t(H_t), y_{t+1}, A_t)\}}$$

It was shown in [80] that the above relationship implies that a sufficient condition for (P2) is the following.

(P2b) Sufficient for predicting output. There exists a controlled transition probability matrix $P_{\text{IS}} \colon \mathcal{Z} \times \mathcal{A} \to \Delta(\mathcal{Y})$ such that for any t and H_t and A_t, we have

$$\mathbb{P}(Y_{t+1} \mid H_t, A_t) = P_{\text{IS}}(Y_{t+1} \mid \vec{\sigma}_t(H_t), A_t).$$

The belief-state satisfies properties (P1) and (P2) and is, therefore, an information state. For specific models such as the linear quadratic Guassian (LQG) control or certain classes of machine repair models, simpler and model-specific information states exist. We refer the reader to [80] for a detailed discussion of the history of information states and various other examples.

TABLE I: Abbreviations used for different dynamic programs

Abbreviation	Meaning
IS	Information-state based DP
DES	DP using designer's approach
PROD	DP based on product processes
AIS	Approximate information state based DP
ASQL	Agent-state based Q-learning

TABLE II: List of different policy classes

Policy class	Meaning
$\vec{\Pi}_{\text{ND}}$	history based non-stationary deterministic policies
$\vec{\Pi}_{\text{NS}}$	history based non-stationary stochastic policies
Π_{ND}	agent-state based non-stationary deterministic policies
Π_{NS}	agent-state based non-stationary stochastic policies
Π_{SD}	agent-state based stationary deterministic policies
Π_{SS}	agent-state based stationary stochastic policies

A key implication of an information state is that it always leads to a dynamic programming decomposition.[6] In particular, we have the following [80, Theorem 5 and 25].

Theorem 1: Information-state based DP

Suppose the agent state $\{Z_t\}_{t\ge1}$ is an information state, i.e., satisfies properties (P1) and (P2) with some $(r_{\text{IS}}, P_{\text{IS}})$. Define the following dynamic program[a]:

$$Q^\star_{\text{IS}}(z,a) = r_{\text{IS}}(z,a) + \gamma \sum_{z'\in\mathcal{Z}} P_{\text{IS}}(z'|z,a) V^\star_{\text{IS}}(z'), \quad \text{(4a)}$$

$$V^\star_{\text{IS}}(z) = \max_{a\in\mathcal{A}} Q^\star_{\text{IS}}(z,a). \quad \text{(4b)}$$

Let $\pi^\star_{\text{IS}}(z)$ denote any arg max of the right hand side of (4b). Then the policy $\vec{\boldsymbol{\pi}}^\star_{\text{IS}} = (\vec{\pi}^\star_{\text{IS},1}, \vec{\pi}^\star_{\text{IS},2}, \dots)$ given by

$$\vec{\pi}^\star_{\text{IS},t}(h_t) = \pi^\star_{\text{IS}}(\vec{\sigma}_t(h_t))$$

is optimal, i.e., $J^{\vec{\boldsymbol{\pi}}^\star_{\text{IS}}} = \vec{J}^\star_{\text{ND}}$.

[a] For simplicity, we state the result for finite $\mathcal{Z}$. Similar decomposition holds for continuous $\mathcal{Z}$.

The classical belief-state based dynamic programming decomposition may be considered as an instance of Theorem 1. For specific models which have a simpler information state such as LQG control, Theorem 1 provides a simpler dynamic program than the standard belief-state based dynamic program.

In spite of its generality, Theorem 1 is only applicable when the agent state satisfies properties (P1) and (P2), which is not always the case. A simple counterexample is when the agent uses a finite window of past observations with $Z_t = (Y_{t-n:t}, A_{t-n:t-1})$. In the companion paper [26], approximation results for this model are presented using ideas from filter stability. In the next section, we present other approaches that have been used in the literature for identifying optimal policies for models where properties (P1) and (P2) are not satisfied.

III. Optimal agent-state based policies

In this section, we consider the different policy classes (deterministic/stochastic and stationary/non-stationary) of agent-state based policies. We present examples to show that non-stationary policies may outperform stationary ones and that stochastic policies may outperform deterministic ones. We then present different schemes that have been used in the literature to identify optimal or sub-optimal policies within the different policy classes. In particular, the designer's approach for finding an optimal non-stationary agent-state based policy; policy evaluation and policy gradients for stationary agent-state based polices; and approximate information state approach for finding good stationary and deterministic agent-state based policies.

A. Policy classes

We start with a notation to denote (one-step) decision rules. Let

- $\mathcal{D}_{\text{D}} = [\mathcal{Z} \to \mathcal{A}]$ denote the family of all deterministic decision rules, and
- $\mathcal{D}_{\text{S}} = [\mathcal{Z} \to \Delta(\mathcal{A})]$ denote the family of all stochastic decision rules.

We define the following classes of agent-state based policies. Notation wise, we will use the absence of the vector accent to differentiate agent-state based policies from history-based policies. A summary of the different policy class used in this paper is shown in Table II.

- Π_{ND} denotes the class of non-stationary agent-state based deterministic policies $\boldsymbol{\pi} = (\pi_1, \pi_2, \dots)$ where $\pi_t \in \mathcal{D}_{\text{D}}$. Let
$$J^\star_{\text{ND}} = \sup_{\boldsymbol{\pi}\in\Pi_{\text{ND}}} J^{\boldsymbol{\pi}}$$
denote the optimal performance within class Π_{ND}. A policy $\boldsymbol{\pi}^\star_{\text{ND}} \in \Pi_{\text{ND}}$ is called optimal in policy class Π_{ND} if $J^{\boldsymbol{\pi}^\star_{\text{ND}}} = J^\star_{\text{ND}}$.
- Π_{NS} denotes the class of non-stationary agent-state based stochastic policies $\boldsymbol{\pi} = (\pi_1, \pi_2, \dots)$ where $\pi_t \in \mathcal{D}_{\text{S}}$. Let
$$J^\star_{\text{NS}} = \sup_{\boldsymbol{\pi}\in\Pi_{\text{NS}}} J^{\boldsymbol{\pi}}$$
denote the optimal performance within class Π_{NS}. A policy $\boldsymbol{\pi}^\star_{\text{NS}} \in \Pi_{\text{NS}}$ is called optimal in policy class Π_{NS} if $J^{\boldsymbol{\pi}^\star_{\text{NS}}} = J^\star_{\text{NS}}$.
- Π_{SD} denotes the class of stationary agent-state based deterministic policies $\boldsymbol{\pi} = (\pi, \pi, \dots)$ where $\pi \in \mathcal{D}_{\text{D}}$. Let
$$J^\star_{\text{SD}} = \sup_{\boldsymbol{\pi}\in\Pi_{\text{SD}}} J^{\boldsymbol{\pi}}$$
denote the optimal performance within class Π_{SD}. A policy $\boldsymbol{\pi}^\star_{\text{SD}} \in \Pi_{\text{SD}}$ is called optimal in policy class Π_{SD} if $J^{\boldsymbol{\pi}^\star_{\text{SD}}} = J^\star_{\text{SD}}$.

[6] We present multiple dynamic programming decompositions and differentiate between them via subscripts. A summary of these subscripts is shown in Table I.

Fig. 1: The cells indicate the state of the environment. Cells with the same background color have the same observation. The cells with a thick red boundary correspond to elements of the set $\mathcal{D}_0 := \{n(n+1)/2 + 1 : n \in \mathbb{N}\}$, where the action 0 gives a reward of $+1$ and moves the state to the right, while the action 1 gives a reward of -1 and resets the state to 1. The cells with a thin black boundary correspond to elements of the set $\mathcal{D}_1 = \mathbb{N} \setminus \mathcal{D}_0$, where the action 1 gives the reward of $+1$ and moves the state to the right while the action 0 gives a reward of -1 and resets the state to 1.

- Π_{SS} denotes the class of stationary agent-state based stochastic policies $\boldsymbol{\pi} = (\pi, \pi, \dots)$ where $\pi \in \mathcal{D}_{\text{S}}$. Let

$$J^\star_{\text{SS}} = \sup_{\boldsymbol{\pi} \in \Pi_{\text{SS}}} J^{\boldsymbol{\pi}}$$

  denote the optimal performance within class Π_{SS}. A policy $\boldsymbol{\pi}^\star_{\text{SS}} \in \Pi_{\text{SS}}$ is called optimal in policy class Π_{SS} if $J^{\boldsymbol{\pi}^\star_{\text{SS}}} = J^\star_{\text{SS}}$.

Note that Π_{ND} and Π_{NS} agent-state based policies are different from $\vec{\Pi}_{\text{ND}}$ and $\vec{\Pi}_{\text{NS}}$, which are history based policies. Therefore, $J^\star_{\text{ND}}$ and $J^\star_{\text{NS}}$ are different from $\vec{J}^\star_{\text{ND}}$ and $\vec{J}^\star_{\text{NS}}$.

When the agent state is an information state and satisfies (P1) and (P2), Theorem 1 implies that all the performance functions $J^\star_{\text{ND}}$, $J^\star_{\text{NS}}$, $J^\star_{\text{SD}}$, $J^\star_{\text{SS}}$ are equal and are also equal to the performance of history-based policies $\vec{J}^\star_{\text{ND}}$ and $\vec{J}^\star_{\text{NS}}$. However, when the agent state is not an information state, then the agent state does not satisfy the controlled Markov property, i.e.,

$$\mathbb{P}(Z_{t+1} \mid Z_{1:t}, A_{1:t}) \neq \mathbb{P}(Z_{t+1} \mid Z_t, A_t)$$

or the property of being sufficient for reward evaluation, i.e.,

$$\mathbb{E}[r(S_t, A_t) \mid Z_{1:t}, A_{1:t}] \neq \mathbb{E}[r(S_t, A_t) \mid Z_t, A_t].$$

Moreover, the agent does not have perfect recall (i.e., at time t, the agent does not have access to all the information that was available to the information that was available to it in the past).

The absence of the information state properties imply that (F1) non-stationary policies may outperform stationary policies and (F2) stationary stochastic policies may outperform stationary deterministic policies. These facts were first reported in [73]. We illustrate them via examples borrowed from [74].

(F1) Non-stationary agent-state based policies may outperform stationary agent-state based policies: Consider the POMDP described in Fig. 1, where the system starts in state 1. Since the dynamics are deterministic, the agent can infer the current state from the history of past actions and can take the action to increment the current state and receive a per-step reward of $+1$. Thus, $\vec{J}^\star_{\text{ND}} = 1/(1-\gamma)$. Furthermore, since the system is deterministic, the optimal policy can be implemented via an open-loop policy given by $\{a^\star_t\}_{t \geq 1}$, where $a^\star_t = \mathbb{1}_{\{t \in \mathcal{D}_1\}}$, which can be implemented irrespective of the information structure. Thus, $J^\star_{\text{ND}} = \vec{J}^\star_{\text{ND}}$.

(a) Dynamics under action 0 (b) Dynamics under action 1

Fig. 2: A POMDP with $\mathcal{S} = \{0, 1, 2\}$, $\mathcal{A} = \{0, 1\}$ and $\mathcal{Y} = \{0\}$. The rewards functions are $r(\cdot, 0) = [-1, 0, 2]$ and $r(\cdot, 1) = -0.5$.

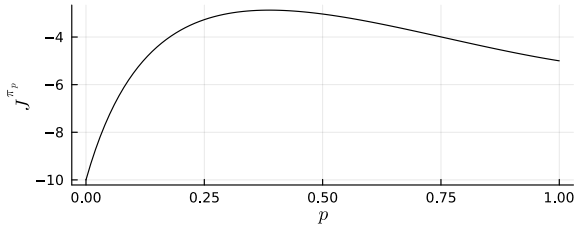

Fig. 3: Performance of stationary stochastic policies in the model of Fig. 2.

When $Z_t = Y_t$, Π_{SD} consists of four policies. A brute force evaluation shows that $J^\star_{\text{SD}} = (1 + \gamma - \gamma^2)/(1 - \gamma^3) < J^\star_{\text{ND}}$. See [75] for details. Thus, non-stationary agent-state policies may outperform stationary agent-state based policies.[7]

(F2) Stationary stochastic agent-state policies may outperform stationary deterministic agent-state based policies: Consider the POMDP shown in Fig. 2, where the system starts in state 0. The agent gets no observations, i.e., $Y_t \equiv 0$, and we consider agent-state policies with $Z_t = Y_t \equiv 0$. A policy $\boldsymbol{\pi} \in \Pi_{\text{SS}}$ is parameterized by a single parameter $p \in [0, 1]$, which indicates the probability of choosing action 1. We denote such a policy by $\boldsymbol{\pi}_p$. Note that if $p \in \{0, 1\}$, then $\boldsymbol{\pi}_p \in \Pi_{\text{SD}}$. Let (P_a, r_a) denote the probability transition matrix and reward function when $a \in \mathcal{A}$ is chosen and let $(P_p, r_p) = (1-p)(P_0, r_0) + p(P_1, r_1)$. Then, the performance of policy $\boldsymbol{\pi}_p$ is given by $J^{\boldsymbol{\pi}_p} = [(1 - \gamma P_p)^{-1} r_p]_0$. The performance for all $p \in [0, 1]$ for $\gamma = 0.9$ is shown in Fig. 3, which shows that the best performance is achieved by the stochastic policy $\boldsymbol{\pi}_p$ with $p \approx 0.39$.

In summary, ***the optimal agent-state based policy depends on the choice of the policy class***. The relationship between the different performance measures can be summarized as follows:

$$\begin{array}{ccccc} J^\star_{\text{SD}} & \longrightarrow & J^\star_{\text{ND}} & \longrightarrow & \vec{J}^\star_{\text{ND}} \\ \downarrow & & \updownarrow & & \updownarrow \\ J^\star_{\text{SS}} & \longrightarrow & J^\star_{\text{NS}} & \longrightarrow & \vec{J}^\star_{\text{NS}} \end{array} \tag{5}$$

where the arrows mean "less than or equal to". The solid

[7] The conclusion continue to hold if we compare with the performance on stationary stochastic policies. Although it is not possible to evaluate $J^\star_{\text{SS}}$ in closed form, a brute force Monte Carlo evaluation shows that $J^\star_{\text{SS}} \approx J^\star_{\text{SD}}$. Therefore, $J^\star_{\text{SS}} < J^\star_{\text{ND}}$.

arrows follow from set inclusion relationships (that is, one policy class is a subset of the other); the dashed arrow relationships need to established explicitly. The reason for the dashed arrow between $J^\star_{\text{NS}}$ and $J^\star_{\text{ND}}$ has already been presented in the discussion around (1). The reason for the dashed arrow between $J^\star_{\text{NS}}$ and $J^\star_{\text{ND}}$ will be presented later in (7).

Features (F1)–(F2) may appear to be surprising because they are not present in MDPs or POMDPs when using a belief state. In fact, they are absent when the agent state is an information state. However, when the agent state is not an information state, the system has what is called a *non-classical information structure*[8] and, therefore, the problem of finding the optimal agent-state based policy is a *decentralized* stochastic control problem (even though there is only one decision maker). This fact is well known in the decentralized control literature [55], [69] but perhaps not as well recognized in the POMDP literature.

The fact that the optimization problem at hand is a decentralized control problem means that we cannot directly use the standard results from Markov decision theory to find the optimal policy in a given policy class. In the remainder of this section, we summarize the approaches that have been taken in the literature to find the optimal or a good policy within a policy class.

B. The designer's approach to find the optimal non-stationary policy

Since the problem of finding the optimal non-stationary agent-state policy is a decentralized stochastic control policy, it is possible to use solution techniques from decentralized stochastic control to find the optimal policy in class Π_{ND} and Π_{NS}. One such approach is the designer's approach, which was proposed in [55] and can be viewed as a refinement of an earlier approach known as the standard form proposed in [89]. The designer's approach was presented in [55] for a two-agent decentralized control system where each agent followed an agent-state based policy. We present a simplified version of this approach, restricted to the single agent POMDP setting.

For any policy $\boldsymbol{\pi} = (\pi_1, \pi_2, \dots) \in \Pi_{\text{NS}}$, define $\xi^{\boldsymbol{\pi}}_t \in \Delta(\mathcal{S} \times \mathcal{Z})$ by

$$\xi^{\boldsymbol{\pi}}_t(s, z) = \mathbb{P}^{\boldsymbol{\pi}}(S_t = s, Z_t = z), \quad \forall s \in \mathcal{S}, z \in \mathcal{Z}.$$

Since $\Pi_{\text{SD}}, \Pi_{\text{SS}}, \Pi_{\text{ND}} \subseteq \Pi_{\text{NS}}$, the same definition holds for any policy in $\Pi_{\text{SD}}, \Pi_{\text{SS}}, \Pi_{\text{ND}}$ as well. The key idea of the designer's approach is to show that the process $\{\xi^{\boldsymbol{\pi}}\}_{t\ge1}$ is a controlled Markov process controlled by $\{\pi_t\}_{t\ge1}$.

We first start with some definitions.

- Define a function $\phi_{\text{DES}} \colon \Delta(\mathcal{S} \times \mathcal{Z}) \times \mathcal{D}_{\text{S}} \to \Delta(\mathcal{S} \times \mathcal{Z})$ as follows: for any $\xi \in \Delta(\mathcal{S} \times \mathcal{Z})$, $\pi \in \mathcal{D}_{\text{S}}$ and for all $s', z' \in \mathcal{S} \times \mathcal{Z}$,

$$[\phi_{\text{DES}}(\xi, \pi)](s', z') = \sum_{\substack{(s,z,y',a)\in \\ \mathcal{S}\times\mathcal{Z}\times\mathcal{Y}\times\mathcal{A}}} \xi(s, z)\pi(a \mid z)P(y', s'|s, a)\mathbb{1}_{\{z'=\phi(z,y',a)\}}$$

- Define the function $r_{\text{DES}} \colon \Delta(\mathcal{S} \times \mathcal{Z}) \times \mathcal{D}_{\text{S}} \to \mathbb{R}$ as follows: for any $\xi \in \Delta(\mathcal{S} \times \mathcal{Z})$ and $\pi \in \mathcal{D}_{\text{S}}$:

$$r_{\text{DES}}(\xi, \pi) \coloneqq \sum_{\substack{(s,z,a)\in \\ \mathcal{S}\times\mathcal{Z}\times\mathcal{A}}} \xi(s, z)\pi(a \mid z)r(s, a).$$

Note that both ϕ_{DES} and r_{DES} are bilinear functions.[9]

Then a simple application of Bayes rule and the definition of expectation imply the following [55].

Proposition 1 *The process $\{\xi^{\pi}_t\}_{t\ge1}$ is a controlled Markov process controlled by $\{\pi_t\}_{t\ge1}$, i.e.,*

1) $\xi^{\boldsymbol{\pi}}_{t+1} = \phi_{\text{DES}}(\pi_t, \xi^{\boldsymbol{\pi}}_t)$;
2) $\mathbb{E}^{\boldsymbol{\pi}}[r(S, A)] = r_{\text{DES}}(\pi_t, \xi^{\boldsymbol{\pi}}_t)$.

The designer's approach is based on the following interpretation. Consider the system designer who wants to pick a policy $\boldsymbol{\pi} \in \Pi_{\text{NS}}$ (or Π_{ND}). From the point of view of such a designer, the system is a completely unobserved input-output system, where the system designer chooses the control input π_t, receives a per-step reward $r(S_t, A_t)$, but does not observe anything. Proposition 1 implies that $\xi^{\boldsymbol{\pi}}_t$ is an information state for this system. Therefore, the system designer can use dynamic programming to identify its "control actions". Consequently, we have the following result [55].

Theorem 2: DP using the designer's approach

Consider the following dynamic program: for all $\xi \in \Delta(\mathcal{S} \times \mathcal{Z})$,

$$V_{\text{DES}}(\xi) = \max_{\pi\in\mathcal{D}_{\text{S}}} \big\{ r_{\text{DES}}(\xi, \pi) + \gamma V_{\text{DES}}(\phi_{\text{DES}}(\xi, \pi)) \big\}. \quad (6)$$

Let $\psi_{\text{DES}}(\xi)$ denote any arg max of the right hand side of (6). Let $\xi_1(s_1, a_1) \coloneqq \mathbb{P}(S_1 = s_1, Z_1 = z_1)$ denote the initial distribution of the system and the agent state. Recursively define $\{\xi^\star_t\}_{t\ge1}$ and $\{\pi^\star_t\}_{t\ge1}$ as follows: $\xi^\star_1 = \xi_1$ and for $t \ge 1$:

$$\pi^\star_t = \psi_{\text{DES}}(\xi^\star_t) \quad \text{and} \quad \xi^\star_{t+1} = \phi_{\text{DES}}(\xi^\star_t, \pi^\star_t).$$

Then the policy $\boldsymbol{\pi}^\star = (\pi^\star_1, \pi^\star_2, \dots) \in \Pi_{\text{NS}}$ is the optimal policy in Π_{NS}.

Some remarks on the designer's approach

1) The designer's approach presented here is adapted from the presentation in [55] for decentralized control

[8] See [56] for a general discussion of non-classical information structures.

[9] Since $\Delta(\mathcal{S} \times \mathcal{Z})$ and $\mathcal{D}_{\text{S}}$ are bounded sets, linearity in the each component should be interpreted as follows: for any $\xi^1, \xi^2 \in \Delta(\mathcal{S} \times \mathcal{Z})$, $\pi \in \mathcal{D}_{\text{S}}$, and $\lambda \in [0, 1]$, we have

$$\phi_{\text{DES}}(\lambda\xi^1 + (1-\lambda)\xi^2, \pi) = \lambda\phi_{\text{DES}}(\xi^1, \pi) + (1-\lambda)\phi_{\text{DES}}(\xi^2, \pi)$$

with a similar interpretation for r_{DES} and the π component of both functions.

problems with two agents, where the map $\psi_{\text{DES}} \colon \Delta(\mathcal{S} \times \mathcal{Z}) \to \mathcal{D}_{\text{S}}$ was called a ***meta-policy***. The general idea of using the probability distribution over all pertinent variables as an information state goes back to [89]. The method was re-discovered in [28], where it was called occupation MDP. A similar idea was presented in [5] for MDPs and POMDPs with memoryless policies.

2) In the argument above, we maximized over $\pi \in \mathcal{D}_{\text{S}}$ in (6). If we instead maximize over $\pi \in \mathcal{D}_{\text{D}}$, we will obtain the optimal policy $\boldsymbol{\pi}^\star \in \Pi_{\text{ND}}$.
3) The probability distribution $\xi_t^{\boldsymbol{\pi}}$ may be viewed as the "belief" of the designer on the state sufficient for input-output mapping (i.e., (S_t, Z_t)) given the history of past observations and actions (i.e., $\pi_{1:t-1}$, since the designer does not observe anything). Therefore, as stated in [55] and [28], we can follow the standard argument for POMDPs [76] to argue that the value function V_{DES} is convex (it is piecewise linear and convex for finite horizon models).
4) Since V_{DES} is convex and ϕ_{DES} and r_{DES} are bilinear, it means that

$$Q_{\text{DES}}(\xi, \pi) \coloneqq r_{\text{DES}}(\xi, \pi) + \gamma V_{\text{DES}}(\phi_{\text{DES}}(\xi, \pi))$$

is convex in π. Therefore, for any fixed ξ, the maximum value of the convex function $Q_{\text{DES}}(\xi, \pi)$ over $\pi \in \mathcal{D}_{\text{S}}$ (which is a convex polyhedron) is achieved at a vertex of $\mathcal{D}_{\text{S}}$. Thus, the arg max in (6) is a deterministic decision rule $\pi \in \mathcal{D}_{\text{D}}$. Therefore,

$$\boxed{J^\star_{\text{ND}} = J^\star_{\text{NS}}.} \tag{7}$$

Consequently, ***there is no loss of optimality in restricting attention to non-stationary deterministic (rather than stochastic) agent-state based policies.***

5) As far as we are aware, the result in (7) is new. A similar result is claimed in [5, Proposition 1] for memoryless policies in POMDPs (i.e., $Z_t = Y_t$), but for a different definition of optimality.
6) Note that the dynamics of $\{\xi_t\}_{t \ge 1}$ in (6) are deterministic. Therefore, it is possible to use ideas from deterministic optimization to find the optimal *trajectory* $\{\xi_t^\star\}_{t \ge 1}$ and optimal *open-loop* "control actions" $\{\pi_t^\star\}_{t \ge 1}$.

C. Policy search methods to find a locally optimal policy in Π_{SS}

We can use standard results from policy search methods for MDPs to identify a stationary policy that is locally optimal (within the class of stationary policies). The high-level idea is based on the fact that the *product process* $\{(S_t, Z_t)\}_{t \ge 1}$ is a controlled Markov process with the controlled transition probability:

$$P_{\text{PROD}}(s', z' \mid s, z, a) \coloneqq \sum_{y' \in \mathcal{Y}} P(s', y' \mid s, a) \mathbb{1}_{\{z' = \phi(z, y', a)\}}.$$

Now, for any $\boldsymbol{\pi} = (\pi, \pi, \dots) \in \Pi_{\text{SS}}$, define:

$$P^{\boldsymbol{\pi}}_{\text{PROD}}(s', z' \mid s, z) \coloneqq \sum_{a \in \mathcal{A}} \pi(a|z) P_{\text{PROD}}(s', z' | s, z, a).$$

and

$$r^{\boldsymbol{\pi}}_{\text{PROD}}(s, z) \coloneqq \sum_{a \in \mathcal{A}} \pi(a|z) r(s, a).$$

Let $V^{\boldsymbol{\pi}}_{\text{PROD}} \colon \mathcal{S} \times \mathcal{Z} \to \mathbb{R}$ be the value function of the policy $\boldsymbol{\pi}$ in the (S, Z) space, i.e., it is the solution of the following policy evaluation formula:

$$V^{\boldsymbol{\pi}}_{\text{PROD}}(s, z) = r^{\boldsymbol{\pi}}_{\text{PROD}}(s, z) + \gamma \sum_{\substack{(s', z') \in \\ \mathcal{S} \times \mathcal{Z}}} P^{\boldsymbol{\pi}}_{\text{PROD}}(s', z' | s, z) V^{\boldsymbol{\pi}}_{\text{PROD}}(s', z'). \tag{8}$$

Let $Q^{\boldsymbol{\pi}}_{\text{PROD}} \colon \mathcal{S} \times \mathcal{Z} \times \mathcal{A} \to \mathbb{R}$ denote the action-value function of the policy $\boldsymbol{\pi}$, i.e., for any $(s, z, a) \in \mathcal{S} \times \mathcal{Z} \times \mathcal{A}$,

$$Q^{\boldsymbol{\pi}}_{\text{PROD}}(s, z, a) = r(s, a) + \gamma \sum_{\substack{(s', z') \in \\ \mathcal{S} \times \mathcal{Z}}} P_{\text{PROD}}(s', z' | s, z, a) V^{\boldsymbol{\pi}}_{\text{PROD}}(s', z').$$

Finally, let $d^{\boldsymbol{\pi}}_{\text{PROD}}$ denote the *unnormalized* occupancy measure of the policy $\boldsymbol{\pi}$, i.e.,

$$d^{\boldsymbol{\pi}}_{\text{PROD}}(s, z, a) = \sum_{t=1}^{\infty} \gamma^{t-1} \mathbb{P}^{\boldsymbol{\pi}}(S_t = s, Z_t = z, A_t = a).$$

Then, the performance of policy $\boldsymbol{\pi} \in \Pi_{\text{SS}}$ is given by

$$J^{\boldsymbol{\pi}} = \sum_{(s,z) \in \mathcal{S} \times \mathcal{Z}} \xi_1(s, z) V^{\boldsymbol{\pi}}_{\text{PROD}}(s, z),$$

where $\xi_1(s, z) \coloneqq \mathbb{P}(S_1 = s, Z_1 = z)$ denotes the initial distribution of the joint state of the environment and the agent. This shows that performance evaluation of any policy in Π_{SS} is straightforward.

Given the above performance evaluation formula, we can use policy-gradient based methods to find a good policy. In particular, suppose $\boldsymbol{\pi} = (\pi, \pi, \dots) \in \Pi_{\text{SS}}$ is a parameterized policy (e.g., softmax or mixture of Gaussians) with policy parameters θ. We will use $\boldsymbol{\pi}_\theta$ to denote the policy with parameters θ and π_θ to denote the decision rule with parameters θ. Then, by a straight forward modification of the policy gradient formula [12], [47], [48], [81], we get

$$\boxed{\nabla_\theta J^{\boldsymbol{\pi}_\theta} = \sum_{\substack{(s,z,a) \in \\ \mathcal{S} \times \mathcal{Z} \times \mathcal{A}}} d^{\boldsymbol{\pi}_\theta}_{\text{PROD}}(s, z, a) Q^{\boldsymbol{\pi}_\theta}_{\text{PROD}}(s, z, a) \nabla_\theta \log \pi_\theta(a|z).} \tag{9}$$

Consequently, we can use any policy gradient based search algorithm (such as actor-critic and its variants; see [86]) to find locally optimal policy parameters, i.e., policy parameters $\theta^\star$ such that

$$\nabla_\theta J^{\boldsymbol{\pi}_\theta} \Big|_{\theta = \theta^\star} = 0.$$

Then the policy $\boldsymbol{\pi}_{\theta^\star}$ is a locally optimal policy in class Π_{SS}.

Some remarks on the policy gradient approach

1) The policy evaluation formula (8) using the product state (S_t, Z_t) was initially presented in [67] and has been rediscovered in slightly different forms multiple times [15], [36], [37], [53].
2) There is a rich history of policy based algorithms in Systems and Control, Operations Research, and Artificial Intelligence. We refer the reader to [8], [11] for detailed overviews.
3) Several other approaches for policy search in Π_{NS} have been proposed in the literature [36], [46], [49], [61].
4) Note that the formula in (9) is applicable when the system dynamics are known (and therefore the $Q^{\boldsymbol{\pi}}_{\text{PROD}}$ can be computed via policy evaluation formula described above). In the RL setting, the environment state S_t is not observed, so the above formula cannot be used directly.
5) For RL, the idea of using policy gradient methods for POMDPs was first presented in [45] and variations have been proposed in [1], [6], [40], [49], [57], [65], [92]. However, these policy gradient formula proposed in these papers is for the RL setting where the agent state S_t is not observed; therefore the exact expressions were different. We present an alternative approach in Sec. IV.
6) As pointed out in [45], the agent state in a POMDP can be viewed as a *feature* of the entire history of observations and actions. With this viewpoint, the policy gradient formula (9) may be viewed as a special case of actor-critic algorithms with features [47], [48].
7) Another idea that has been used in the RL setting is asymmetric actor critic [7], [75], where it is assumed that the critic has access to the environment state (which is the case in simulation environments). In such settings, it is possible to learn $Q^{\boldsymbol{\pi}}(s, z, a)$ via temporal difference learning. See [7], [75] for a discussion.

D. The approximate information state approach to find a good policy in Π_{SD}

In this section, we present an alternative approach to finding good policies in Π_{SD} called the *approximate information state* (AIS), which was originally developed in [80]. We first start with the intuition and then explain the technical results.

Intuition: One way to obtain a policy is to posit *any* dynamics and rewards $(P_{\text{AIS}}, r_{\text{AIS}})$, where $P_{\text{AIS}} \colon \mathcal{Z} \times \mathcal{A} \to \Delta(\mathcal{S})$ and $r_{\text{AIS}} \colon \mathcal{Z} \times \mathcal{A} \to \mathbb{R}$, and obtain a policy $\vec{\boldsymbol{\pi}}_{\text{AIS}}$ by solving the following dynamic program:

$$Q_{\text{AIS}}(z, a) = r_{\text{AIS}}(z, a) + \gamma \sum_{z' \in \mathcal{Z}} P_{\text{AIS}}(z' | z, a) V_{\text{AIS}}(z'), \tag{10a}$$

$$V_{\text{AIS}}(z) = \max_{a \in \mathcal{A}} Q_{\text{AIS}}(z, a). \tag{10b}$$

Let $\pi_{\text{AIS}}(z)$ denote any arg max of the right hand side of (10b). Define the policy $\vec{\boldsymbol{\pi}}_{\text{AIS}} = (\vec{\pi}_{\text{AIS},1}, \vec{\pi}_{\text{AIS},2}, \dots)$ given by[10]

$$\vec{\pi}_{\text{AIS},t}(h_t) = \pi_{\text{AIS}}(\vec{\sigma}_t(h_t)) \tag{11}$$

As shown in Theorem 1, if the posited dynamics and rewards $(P_{\text{AIS}}, r_{\text{AIS}})$ satisfy properties (P1) and (P2), the policy $\vec{\boldsymbol{\pi}}_{\text{AIS}}$ is optimal. But what happens when properties (P1) and (P2) are not satisfied?

Let $(\boldsymbol{\varepsilon}, \boldsymbol{\delta})$ with $\boldsymbol{\varepsilon} = (\varepsilon_1, \varepsilon_2, \dots)$ and $\boldsymbol{\delta} = (\delta_1, \delta_2, \dots)$ with $\varepsilon_t, \delta_t \in \mathbb{R}_{\geq 0}$ be such that the following properties are satisfied:

(AP1) Approximately sufficient for performance evaluation. For any t, and H_t and A_t we have

$$\big| \mathbb{E}[R_t \mid H_t, A_t] - r_{\text{AIS}}(\vec{\sigma}_t(H_t), A_t) \big| \leq \varepsilon_t.$$

(AP2) Sufficient for predicting itself. For any t, and H_t and A_t we have

$$d_{\mathfrak{F}}(\mathbb{P}(Z_{t+1} \mid H_t, A_t), P_{\text{AIS}}(Z_{t+1} \mid \vec{\sigma}_t(H_t), A_t)) \leq \delta_t.$$

where $d_{\mathfrak{F}}$ is a metric on $\Delta(\mathcal{Z})$, which we will make precise later.

Then the tuple $(\vec{\boldsymbol{\sigma}}, P_{\text{AIS}}, r_{\text{AIS}})$ is called an *approximate information state* (AIS).

Engineering intuition suggests that the sub-optimality of $\vec{\boldsymbol{\pi}}_{\text{AIS}}$, i.e.,

$$\alpha := \vec{J}^{\star}_{\text{ND}} - J^{\vec{\boldsymbol{\pi}}_{\text{AIS}}}$$

should be a continuous function of $(\boldsymbol{\varepsilon}, \boldsymbol{\delta})$. The results of [80] formalize this intuition.

The choice of metric: To formalize the above intuition, we need to choose a metric $d_{\mathfrak{F}}$ on probability spaces. For our purposes, it is convenient to work with a class of metrics known as integral probability measures (IPMs) [60]. Let $\mathcal{V}$ denote the family of all bounded (measurable) functions from $\mathcal{Z}$ to $\mathbb{R}$ and $\mathcal{P}$ denote the set of all probability measures on $\mathcal{Z}$ with finite mean.[11]

Definition 1 Let $\mathfrak{F}$ be a convex and balanced[12] subset of $\mathcal{V}$. Then, the IPM distance (w.r.t. $\mathfrak{F}$) between two probability laws $\nu_1, \nu_2 \in \mathcal{P}$ is given by

$$d_{\mathfrak{F}}(\nu_1, \nu_2) = \sup_{f \in \mathfrak{F}} \left| \int f d\nu_1 - \int f d\nu_2 \right|.$$

Definition 2 In the setting of Definition 1, the Minkowski functional of any measurable function $f \in \mathcal{V}$ is defined as

$$\rho_{\mathfrak{F}}(f) = \inf \left\{ \rho \in \mathbb{R}_{>0} : \frac{f}{\rho} \in \mathfrak{F} \right\}.$$

Note that if for every positive ρ, $f/\rho \notin \mathfrak{F}$, then $\rho_{\mathfrak{F}}(f) = \infty$.

An immediate consequence of the above two definitions is that for any measurable function $f \in \mathcal{V}$,

$$\left| \int f d\nu_1 - \int f d\nu_2 \right| \leq \rho_{\mathfrak{F}}(f) d_{\mathfrak{F}}(\nu_1, \nu_2). \tag{12}$$

[10] Recall that $\vec{\boldsymbol{\sigma}} = (\vec{\sigma}_1, \vec{\sigma}_2, \dots)$ is the history compression function corresponding to the agent-state update function ϕ and is given by (3).

[11] When $\mathcal{Z}$ is finite $\mathcal{V} \equiv \mathbb{R}^{|\mathcal{Z}|}$ and $\mathcal{P}$ is the set of all probability mass functions on $\mathcal{Z}$.

[12] A subset $\mathfrak{F}$ of $\mathcal{V}$ is balanced if for all $f \in \mathfrak{F}$ and scalars a such that $|a| \leq 1$, $af \in \mathfrak{F}$.

Many of the commonly used metrics on probability spaces are IPMs. For example

- **Total variation distance**, denoted by d_{TV}, corresponds to $\mathfrak{F} = \mathfrak{F}_{\mathrm{TV}} := \{f \in \mathcal{V} : \frac{1}{2}\operatorname{span}(f) \le 1\}$, where $\operatorname{span}(f) = \sup(f) - \inf(f)$ [60], [84]. In this case $\rho_{\mathfrak{F}}(f) = \frac{1}{2}\operatorname{span}(f)$.
- **Wasserstein distance**. Suppose $(\mathcal{Z}, d_{\mathcal{Z}})$ is a metric space. Then the Wasserstein distance, denoted by d_{Was}, corresponds to $\mathfrak{F} = \mathfrak{F}_{\mathrm{Was}} := \{f \in \mathcal{V} : \operatorname{Lip}(f) \le 1\}$ where $\operatorname{Lip}(f)$ denotes the Lipschitz constant of a function f [83], [84]. In this case $\rho_{\mathfrak{F}}(f) = \operatorname{Lip}(f)$.

Other distances such as Kantorovich distance, bounded Lipschitz, maximum mean discrepancy, are also instances of IPM. See [60], [80] for details.

We now state the main result of [80].

Theorem 3: AIS based DP

Given a function class $\mathfrak{F}$, let $(\vec{\boldsymbol{\sigma}}, P_{\mathrm{AIS}}, r_{\mathrm{AIS}})$ be an $(\boldsymbol{\varepsilon}, \boldsymbol{\delta})$-AIS with respect to $d_{\mathfrak{F}}$. Consider the DP (10). Then, the policy $\vec{\boldsymbol{\pi}}_{\mathrm{AIS}}$ given by (11) satisfies

$$\vec{J}^{\star}_{\mathrm{ND}} - J^{\vec{\boldsymbol{\pi}}_{\mathrm{AIS}}} \le \frac{2}{1-\gamma}\big[\varepsilon + \gamma\delta\rho_{\mathfrak{F}}(V^{\star}_{\mathrm{AIS}})\big], \qquad (13)$$

where

$$\varepsilon = (1-\gamma)\sum_{t=1}^{\infty}\gamma^{t-1}\varepsilon_t, \quad \delta = (1-\gamma)\sum_{t=1}^{\infty}\gamma^{t-1}\delta_t.$$

Some remarks on AIS

1) We have presented the simplest form of the AIS result. Several variations including finite horizon models, multi-agent systems, and others are presented in [80].
2) The results of Theorem 3 may be viewed as a generalization of approximation bounds for MDPs. The earliest such result is [59]. Several variations of such results for MDPs have been derived in the literature, including model approximation in MDPs [2], state abstraction in MDPs [2], [31], [52]. As shown in [80], many of these results are special instances of Theorem 3.
3) When defining information state, it was mentioned that (P2b) is a sufficient condition for (P2). Along similar lines, condition (AP2) can be relaxed to approximately predicting the next observation. See [80] for details.
4) One option for solving belief-state based POMDPs is by discretizing the belief space. Approximation bounds for belief discretization were derived in [30] and it was shown in [80] that such bounds may be viewed as a special instance of Theorem 3.
5) A heuristic algorithm to jointly learn a good AIS representation and a agent-state based policy is proposed in [91].
6) In addition to the AIS approach presented here, there are two popular approaches to determine good policies in Π_{SD}: bisimulation metrics [16], [17] and predictive state representations [10], [35], [41], [50], [51], [68]. These approaches are based on different approximation philosophies and are conceptually different.
7) The results of Theorem 3 are also applicable for worst case design. In particular, let the *worst-case distance* $d_{\mathfrak{F}}(X_1, X_2)$ between two bounded random variables X_1 and X_2 be defined as the Hausdorff distance between their supports. It is then shown in [23]–[25] that under an additional smoothness assumption on the dynamics, one may define an AIS in terms of the the worst-case distance and generalize the results of Theorem 3 to provide worst case guarantees for agent-state based policies.

IV. Reinforcement learning approaches to learn a policy in Π_{SD} or Π_{SS}

As highlighted in the introduction, one of the motivations for considering agent-state based policies is that they are easier to use in the RL setting because the agent-state update does not depend on the system model. In this section, we review the different approaches for RL in POMDPs with agent state.

Before stating the results, we start with a remark that is not sufficiently highlighted in the literature. In the standard POMDP model, it is assumed that the actions are a function of the history of past observations and actions. In the learning setup, actions are also sometimes allowed to be a function of the history of rewards. Therefore, in the literature on RL for POMDPs, it is always implicitly assumed that the observations include the rewards as well. If that is not the case, we need to consider an agent-state update rule of the form $Z_{t+1} = \phi(Z_t, Y_{t+1}, A_t, R_{t+1})$, but that does not fundamentally change the nature of the results.

A. Agent-state based Q-learning (ASQL)

One of the simplest RL algorithms is Q-learning [85], where the agent learns the Q-function by bootstrapping. For MDPs, it is well understood that the Q-learning iterates converge to the optimal Q-function, which is the solution of the dynamic programming equation.

Agent-state based Q-learning (ASQL) uses the same idea for POMDPs with agent state. In the simplest setting, the agent acts in the environment according to a behavior policy $\mu\colon \mathcal{Z} \to \Delta(\mathcal{A})$ and generates the experience $(Y_1, Z_1, A_1, R_1, Y_2, Z_2, A_2, R_2, \dots)$. The agent uses the experience to recursively update Q-function $\{Q_t\}_{t\ge1}$, $Q_t\colon \mathcal{Z}\times\mathcal{A} \to \mathbb{R}$ as follows:

$$\begin{aligned} Q_{t+1}(z,a) &= Q_t(z,a) \\ &\quad + \alpha_t(z,a)\Big[R_t + \gamma \max_{a'\in\mathcal{A}} Q_t(Z_{t+1}, a') - Q_t(z,a)\Big] \end{aligned} \qquad (14)$$

where $R_t = r(S_t, A_t)$ is the reward at time t and $\{\alpha_t\}_{t\ge1}$ are the learning rates which are chosen such that $\alpha_t(z,a) = 0$ when $(z,a) \ne (Z_t, A_t)$.

As mentioned earlier, the agent state does not satisfy the Markov property. So the standard proof of Q-learning for MDPs [9], [40], [85] is not directly applicable. Nonetheless,

it is shown in [74] that ASQL converges under the following assumptions:

(A1) The behavioral policy μ is such that the Markov chain[13] $\{(S_t, Y_t, Z_t, A_t)\}_{t\geq 1}$ is irreducible and aperiodic and, therefore, has a limiting distribution ζ^μ. Moreover, under the behavioral policy, all (z, a) are visited infinitely often, i.e., $\sum_{(s,y)} \zeta^\mu(s, y, z, a) > 0$.

(A2) The learning rates $\{\alpha_t\}_{t\geq 1}$ satisfy the standard Robbins–Monro conditions, i.e., for all (z, a) we have that

$$\sum_{t\geq 1} \alpha_t(z,a) = \infty \quad \text{and} \quad \sum_{t\geq 1} \alpha_t(z,a)^2 < \infty, a.s.$$

For the ease of notation, we will continue to use ζ^μ to denote the marginal and conditional distributions w.r.t. ζ^μ. In particular, for marginals we use $\zeta^\mu(y, z, a)$ to denote $\sum_{s\in\mathcal{S}} \zeta^\mu(s, y, z, a)$ and so on; for conditionals, we use $\zeta^\mu(s|z, a)$ to denote $\zeta^\mu(s, z, a)/\zeta^\mu(z, a)$ and so on.

We now state the main result of [74].[14]

Theorem 4: Convergence of ASQL

Under assumptions (A1) and (A2), the iterates $\{Q_t\}_{t\geq 1}$ of ASQL (14) converge almost surely to a limit Q^μ_{ASQL}, which is the fixed point of the following DP:

$$Q^\mu_{\text{ASQL}}(z,a) = r^\mu_{\text{ASQL}}(z,a) + \gamma \sum_{z'\in\mathcal{Z}} P^\mu_{\text{ASQL}}(z'|z,a) \max_{a'\in\mathcal{A}} Q^\mu_{\text{ASQL}}(z',a') \quad (15)$$

where

$$r^\mu_{\text{ASQL}}(z,a) = \sum_{s\in\mathcal{S}} \zeta^\mu(s|z,a) r(s,a),$$

$$P^\mu_{\text{ASQL}}(z'|z,a) = \sum_{(s,y')\in\mathcal{S}\times\mathcal{Y}} \mathbb{1}_{\{z'=\phi(z,y',a)\}} P(y'|s,a)\zeta^\mu(s|z,a).$$

Let $\pi^\mu_{\text{ASQL}} \in \Pi_{\text{SD}}$ denote the policy

$$\pi^\mu_{\text{ASQL}}(z) := \arg\max_{a\in\mathcal{A}} Q^\mu_{\text{ASQL}}(z,a) \quad (16)$$

Theorem 4 implies that the greedy policy corresponding to $\{Q_t\}_{t\geq 1}$ computed via (14) converges to π^μ_{ASQL}. Define $\vec{\boldsymbol{\pi}}^\mu_{\text{ASQL}} = (\vec{\pi}^\mu_{\text{ASQL},1}, \vec{\pi}^\mu_{\text{ASQL},2}, \dots)$ to be the history dependent policy given by

$$\vec{\pi}^\mu_{\text{ASQL},t}(h_t) = \pi^\mu_{\text{ASQL}}(\sigma_t(h_t)). \quad (17)$$

A natural follow-up question is: How well does $\vec{\boldsymbol{\pi}}^\mu_{\text{ASQL}}$ perform? The answer to this question is surprisingly easy because we can view $(\vec{\boldsymbol{\sigma}}, P^\mu_{\text{ASQL}}, r^\mu_{\text{ASQL}})$ as an AIS. Therefore, Theorem 3 provides an immediate bound on $\vec{J}^\star_{\text{ND}} - J^{\vec{\boldsymbol{\pi}}^\mu_{\text{ASQL}}}$ as follows:

Corollary 1 *We have that*

$$\vec{J}^\star_{\text{ND}} - J^{\vec{\boldsymbol{\pi}}^\mu_{\text{ASQL}}} \leq \frac{2}{1-\gamma}\big[\varepsilon + \gamma\delta\rho_{\mathfrak{F}}(V^\mu_{\text{ASQL}})\big]$$

where $V^\mu_{\text{ASQL}}(z) := \max_{a\in\mathcal{A}} Q^\mu_{\text{ASQL}}(z,a)$ and ε and δ are defined in a manner similar to Sec. III-D

The results of Theorem 4 and Corollary 1 suggest that the performance of ASQL could be improved by choosing the state update function ϕ so as to minimize the approximation losses ε and δ. In particular, consider an implementation of Q-learning where a recurrent neural network (RNN) such as LSTM (long short term memory) or GRU (gated recurrent unit) is used as a history compression function $\vec{\boldsymbol{\sigma}}$. Add an "AIS-block" which learns a generative model for $(P_{\text{ASQL}}, r_{\text{ASQL}})$ by using the *AIS loss* $\lambda\varepsilon^2 + (1-\lambda)\delta^2$ as an auxiliary loss. This algorithm is called RQL-AIS in [71]. Note that we can generate RQL-AIS version of any implementation of recurrent Q-learning for POMDPs.

In [71], a version of RQL-AIS using R2D2 [42] as a base implementation is compared with R2D2 on the minigrid benchmark [21], [22]. The results of [71] show that adding an AIS block significantly improve the performance of Q-learning in the larger and more complicated minigrid environments.

Remarks on ASQL

1) A key feature of Theorem 4 is that the limit Q^μ_{ASQL} depends on the behavioral policy μ, unlike in MDPs where the limit is policy-independent (as long as all state-action pairs are visited infinitely often). This dependence arises because the agent state is not an information state and thus is not policy-independent. See [90] for a discussion on policy independence of information states.
2) The idea of adding *AIS losses* as an auxiliary loss for ASQL was first proposed in [80] for actor-critic algorithms. It is argued in [62] that in ASQL it is not useful to learn the reward function r^μ_{ASQL} since the Q-function is already being learnt. Rather one should simply use δ^2 as an auxiliary loss.
3) Learning a dynamics model to minimize δ^2 is effectively the same as self-supervised representation learning, which has been used as a heuristic in different forms in MDPs and POMDPs in the literature. See [62] for a detailed discussion.
4) The convergence of the simplest version of ASQL was established in [73] for $Z_t = Y_t$ under the assumption that the actions are chosen i.i.d. (and do not depend on Z_t). In [64] it was established that Q^μ_{ASQL} is the fixed point of (14), but convergence of $\{Q_t\}_{t\geq 1}$ to Q^μ_{ASQL} was not established. The convergence of ASQL when the agent state is a finite window of past observations and actions was established in [44]. The convergence result for a general agent state presented in Theorem 4 follows essentially the same proof argument.
5) ASQL is closely related to Q-learning for MDPs with state aggregation or quantization [43], [72], [82]. In particular, consider an MDP with large or continuous state space, where we partition the state space into bins, and treat each bin as an aggregated state.

[13] As argued in Sec. III-C, under any policy $\mu \in \Pi_{\text{SS}}$ the product process $\{(S_t, Z_t)\}_{t\geq 1}$ is a Markov chain. This directly implies that the process $\{(S_t, Y_t, Z_t, A_t)\}_{t\geq 1}$ is a Markov chain as well.

[14] The result proved in [74] is a generalization of Theorem 4 to periodic agent-state based policies. Taking a period of 1 implies Theorem 4.

This model is equivalent to a PODMP where the observation equals the index of the bin corresponding to the current state. Thus, Q-learning in MDPs with state aggregation is equivalent to ASQL with agent state equal to current observation. Therefore, there are considerable similarities between the convergence analysis and approximation bounds of these two setups.

6) ASQL may also be viewed as a MDP with non-Markovian state [18], [27]. As such there are considerable similarities between the convergence analysis and approximation bounds for these two setups.
7) The regret of an optimistic variant of ASQL is presented in [29], though the exact setting there is slightly different. See [29] for details.

B. Agent-state based actor critic (ASAC)

ASQL always learns a deterministic policy (i.e., a policy belonging to Π_{SD}). As highlighted by (5) and fact (F2), stochastic policies can perform better than deterministic policies. One approach to learn stochastic policies is to use the actor-critic class of algorithms.

An agent-state based actor critic (ASAC) algorithm was proposed in [80]. We present it with a slightly different perspective here. Consider any policy $\pi \in \Pi_{\text{SS}}$. Motivated by the result of Theorem 4, we can run a temporal difference learning algorithm to learn a Q-function corresponding to π as follows:

$$Q^\pi_{t+1}(z,a) = Q^\pi_t(z,a) + \alpha_t(z,a)\Big[R_t + \gamma Q^\pi_t(Z_{t+1}, A_{t+1}) - Q^\pi_t(z,a)\Big] \quad (18)$$

where $R_t = r(S_t, A_t)$ as before. Theorem 4 implies that under conditions similar to (A1) and (A2), the iteration (18) converges almost surely to a limit Q^π_{ASAC}, which is the fixed point of the following DP:

$$Q^\pi_{\text{ASAC}}(z,a) = r^\pi_{\text{ASAC}}(z,a) + \gamma \sum_{(z',a')\in\mathcal{Z}\times\mathcal{A}} \pi(a'|z') P^\pi_{\text{ASAC}}(z'|z,a) Q^\pi_{\text{ASAC}}(z',a') \quad (19)$$

where

$$r^\pi_{\text{ASAC}}(z,a) = \sum_{s\in\mathcal{S}} \zeta^\pi(s,a) r(s,a),$$

$$P^\pi_{\text{ASAC}}(z'|z,a) = \sum_{(s,y')\in\mathcal{S}\times\mathcal{Y}} \mathbb{1}_{\{z'=\phi(z,y',a)\}} P(y'|s,a)\zeta^\pi(s|z,a).$$

Define $J^\pi_{\text{ASAC}} = \sum_{(z,a)\in\mathcal{Z}\times\mathcal{A}} \xi_1(z,a) Q^\pi_{\text{ASAC}}(z,a)$. Moreover, define $\vec{\boldsymbol{\pi}} = (\vec{\pi}_1, \vec{\pi}_2, \dots)$ to be the history dependent policy given by

$$\vec{\pi}_t(h_t) = \pi(\vec{\sigma}_t(h_t)).$$

Then, by an argument similar to Theorem 3 we have that (see [80])

$$\big|J^{\vec{\boldsymbol{\pi}}} - J^\pi_{\text{ASAC}}\big| \le \frac{1}{1-\gamma}\big[\varepsilon + \gamma\delta\rho_{\mathfrak{F}}(V^\pi_{\text{ASAC}})\big]$$

where $V^\pi_{\text{ASAC}}(z) = \max_{a\in\mathcal{A}} Q^\pi_{\text{ASAC}}(z,a)$ and ε and δ are defined in a manner similar to Sec. III-D.

The above discussion suggests the following algorithm. Consider an implementation of an actor-critic algorithm where an RNN such as LSTM or GRU is used as a history compression function $\vec{\boldsymbol{\sigma}}$. Similar to RQL-AIS, add an "AIS block" which learns a generative model for $(P_{\text{ASAC}}, r_{\text{ASAC}})$ by using $\lambda\varepsilon^2 + (1-\lambda)\delta^2$ as an auxiliary loss. This algorithm is called PORL-AIS in [80]. Note that we can generate a PORL-AIS version of any implementation of recurrent actor-critic for POMDPs.

In [80], a version of PORL-AIS is compared with LSTM-based actor critic on the minigrid benchmark [21], [22]. The results of [80] show that adding an AIS block significantly improves the performance of actor-critic algorithms in the larger and more complicated minigrid environments.

Remarks of ASAC

1) It is also possible to add AIS losses to actor only algorithms such as G(PO)MDP [8]. See [80] for details.
2) We have argued that the TD iterations for $\{Q_t\}_{t\ge1}$ converge to a limit. Therefore, for a parameterized policy π_θ, the gradient computed via the policy gradient formula

$$\frac{1}{T}\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(A_t|Z_t) Q^{\pi_\theta}_t(Z_t, A_t)$$

asymptotically converges to

$$\sum_{(z,a)\in\mathcal{Z}\times\mathcal{A}} \zeta^\pi(z,a) \log \pi_\theta(a|z) Q^\pi_{\text{ASQL}}(z,a)$$

which is an unbiased estimate of $\nabla_\theta J^{\pi_\theta}_{\text{ASAC}}$ rather than $\nabla_\theta J^{\vec{\boldsymbol{\pi}}_\theta}$. Therefore, PORL-AIS does not have the local convergence guarantees of actor critic for MDPs (but neither do other actor-critic algorithms for POMDPs!).
3) AIS error bounds for asymmetric actor critic [7] are developed in [75] but the numerical experiments presented there suggest that adding an AIS block to asymmetric actor critic algorithms does not provide any measurable performance improvement over the vanilla asymmetric actor critic algorithm.

V. Conclusion

In this tutorial, we have summarized some of the approaches used for planning and learning agent-state based policies in POMDPs. Our survey is not exhaustive. There are other alternative approaches which we have not discussed (e.g., predictive state representations [10], [35], [41], [51], bisimulation [16], [17], world models [32]–[34], [70], [87], decision transformers [19], [63]). Nonetheless, it is our hope that this paper will present the reader with a unified perspective on a subset of the literature. We conclude by emphasizing that this is a rapidly developing research area and many of the fundamental questions are still open.

Acknowledgment

The authors are grateful to Demosthenis Teneketzis (University of Michigan), Jayakumar Subramanian (Adobe Inc.),

and Matthieu Geist (Cohere) for discussions and collaborations which helped the authors develop the viewpoint presented in this tutorial. They also thank Charalambos Charalambous (University of Cyprus), Vijay Subramanian (University of Michigan), and Serdar Yüksel (Queen's University), and Tianwei Ni (University of Montreal) for helpful feedback..

REFERENCES

[1] D. Aberdeen and J. Baxter, "Scaling internal-state policy-gradient methods for POMDPs," in *International Conference on Machine Learning*. San Francisco, CA, USA: Morgan Kauffman, 2002, pp. 3–10.

[2] K. Asadi, D. Misra, and M. Littman, "Lipschitz continuity in model-based reinforcement learning," in *International Conference on Machine Learning*, vol. 80. PMLR, Jul. 2018, pp. 264–273.

[3] K. Åström, "Optimal control of Markov processes with incomplete state information," *Journal of Mathematical Analysis and Applications*, vol. 10, no. 1, pp. 174–205, Feb. 1965.

[4] R. J. Aumann, *Mixed and behavior strategies in infinite extensive games*. Princeton University Princeton, 1961.

[5] J. A. Bagnell, S. Kakade, A. Y. Ng, and J. Schneider, "Policy search by dynamic programming," in *Conference on Neural Information Processing Systems*. Cambridge, MA, USA: MIT Press, 2003, p. 831–838.

[6] L. Baird and A. Moore, "Gradient descent for general reinforcement learning," in *Conference on Neural Information Processing Systems*. Cambridge, MA, USA: MIT Press, 1998, p. 968–974.

[7] A. Baisero and C. Amato, "Unbiased asymmetric reinforcement learning under partial observability," in *International Conference on Autonomous Agents and Multiagent Systems*. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2022, p. 44–52.

[8] J. Baxter and P. L. Bartlett, "Infinite-horizon policy-gradient estimation," *Journal of Artificial Intelligence Research*, vol. 15, pp. 319–350, 2001.

[9] D. Bertsekas and J. N. Tsitsiklis, *Neuro-dynamic programming*. Athena Scientific, 1996.

[10] B. Boots, S. M. Siddiqi, and G. J. Gordon, "Closing the learning-planning loop with predictive state representations," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 954–966, 2011.

[11] X.-R. Cao, "From perturbation analysis to Markov decision processes and reinforcement learning," *Discrete Event Dynamic Systems*, vol. 13, no. 1/2, pp. 9–39, 2003.

[12] ——, *Stochastic Learning and Optimization*. Springer, 2007.

[13] A. Cassandra, M. L. Littman, and N. L. Zhang, "Incremental pruning: A simple, fast, exact method for partially observable Markov decision processes," in *Uncertainty in Artificial Intelligence*, 1997.

[14] A. R. Cassandra, L. P. Kaelbling, and M. L. Littman, "Acting optimally in partially observable stochastic domains," in *AAAI Conference on Artificial Intelligence*, vol. 94, 1994, pp. 1023–1028.

[15] A. R. Cassandra, "Exact and approximate algorithms for partially observable markov decision processes," Ph.D. dissertation, Brown University, 1998.

[16] P. S. Castro, T. Kastner, P. Panangaden, and M. Rowland, "Mico: Improved representations via sampling-based state similarity for Markov decision processes," *Advances in Neural Information Processing Systems*, vol. 34, pp. 30 113–30 126, 2021.

[17] P. S. Castro, P. Panangaden, and D. Precup, "Equivalence relations in fully and partially observable Markov decision processes," in *International Joint Conference on Artificial Intelligence*, 2009, pp. 1653–1658.

[18] S. Chandak, P. Shah, V. S. Borkar, and P. Dodhia, "Reinforcement learning in non-Markovian environments," *Systems & Control Letters*, vol. 185, p. 105751, 2024.

[19] L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, and I. Mordatch, "Decision transformer: Reinforcement learning via sequence modeling," in *Advances in Neural Information Processing Systems*, vol. 34. Curran Associates, Inc., 2021, pp. 15 084–15 097.

[20] H.-T. Cheng, "Algorithms for partially observable Markov decision processes," Ph.D. dissertation, University of British Columbia, Vancouver, BC, 1988.

[21] M. Chevalier-Boisvert, B. Dai, M. Towers, R. Perez-Vicente, L. Willems, S. Lahlou, S. Pal, P. S. Castro, and J. Terry, "Minigrid & Miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks," in *Advances in Neural Information Processing Systems*, vol. 36. Curran Associates, Inc., 2023, pp. 73 383–73 394.

[22] M. Chevalier-Boisvert, L. Willems, and S. Pal, "Minimalistic gridworld environment for gymnasium," https://github.com/Farama-Foundation/MiniGrid, 2018.

[23] A. Dave, I. Faros, N. Venkatesh, and A. A. Malikopoulos, "Worst-case control and learning using partial observations over an infinite time horizon," in *IEEE Conference on Decision and Control*. IEEE, 2023, pp. 6020–6025.

[24] A. Dave, N. Venkatesh, and A. A. Malikopoulos, "Approximate information states for worst-case control of uncertain systems," in *Conference on Decision and Control*. IEEE, 2022, pp. 4945–4950.

[25] ——, "Approximate information states for worst-case control and learning in uncertain systems," 2024, arXiv preprint, arXiv:2301.05089.

[26] A. Devran Kera and S. Yüksel, "Partially observed optimal stochastic control: Regularity, existence, approximations, and learning," in *submitted to IEEE Conference on Decision and Control*. IEEE, Dec. 2024.

[27] ——, "Q-learning for stochastic control under general information structures and non-markovian environments," *Transactions on Machine Learning Research*, 2024.

[28] J. S. Dibangoye, C. Amato, O. Buffet, and F. Charpillet, "Optimally solving Dec-POMDPs as continuous-state MDPs," *Journal of Artificial Intelligence Research*, vol. 55, pp. 443–497, Feb. 2016.

[29] S. Dong, B. V. Roy, and Z. Zhou, "Simple agent, complex environment: Efficient reinforcement learning with agent states," *Journal of Machine Learning Research*, vol. 23, no. 255, pp. 1–54, 2022.

[30] V. Francois-Lavet, G. Rabusseau, J. Pineau, D. Ernst, and R. Fonteneau, "On overfitting and asymptotic bias in batch reinforcement learning with partial observability," *Journal of Artificial Intelligence Research*, vol. 65, pp. 1–30, 2019.

[31] C. Gelada, S. Kumar, J. Buckman, O. Nachum, and M. G. Bellemare, "DeepMDP: Learning continuous latent space models for representation learning," in *International Conference on Machine Learning*, 2019, pp. 2170–2179.

[32] D. Ha and J. Schmidhuber, "Recurrent world models facilitate policy evolution," in *Advances in Neural Information Processing Systems*, vol. 31. Curran Associates, Inc., 2018.

[33] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, "Dream to control: Learning behaviors by latent imagination," in *International Conference on Learning Representations*, 2020.

[34] D. Hafner, T. P. Lillicrap, M. Norouzi, and J. Ba, "Mastering atari with discrete world models," in *International Conference on Learning Representations*, 2021.

[35] W. Hamilton, M. M. Fard, and J. Pineau, "Efficient learning and planning with compressed predictive states," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3395–3439, 2014.

[36] E. A. Hansen, "Solving POMDPs by searching in policy space," in *Uncertainty in Artificial Intelligence*, San Francisco, CA, USA, 1998, p. 211–219.

[37] M. Hauskrecht, "Planning and control in stochastic domains with imperfect information," Ph.D. dissertation, Massachusetts Institute of Technology, 1997.

[38] ——, "Value-function approximations for partially observable markov decision processes," *Journal of artificial intelligence research*, vol. 13, pp. 33–94, 2000.

[39] M. Hutter, *Universal artificial intelligence: Sequential decisions based on algorithmic probability*. Springer Science & Business Media, 2005.

[40] T. Jaakkola, S. Singh, and M. Jordan, "Reinforcement learning algorithm for partially observable Markov decision problems," in *Advances in Neural Information Processing Systems*, vol. 7. MIT Press, 1994, pp. 345–352.

[41] N. Jiang, A. Kulesza, and S. P. Singh, "Improving predictive state representations via gradient descent." in *AAAI Conference on Artificial Intelligence*, 2016, pp. 1709–1715.

[42] S. Kapturowski, G. Ostrovski, W. Dabney, J. Quan, and R. Munos, "Recurrent experience replay in distributed reinforcement learning," in *International Conference on Learning Representations*, 2019.

[43] A. D. Kara, N. Saldi, and S. Yüksel, "Q-learning for MDPs with general spaces: Convergence and near optimality via quantization under weak continuity," *Journal of Machine Learning Research*, 2023.

[44] A. D. Kara and S. Yüksel, "Convergence of finite memory Q learning for POMDPs and near optimality of learned policies under filter stability," *Mathematics of Operations Research*, Nov. 2022.

[45] H. Kimura, K. Miyazaki, and S. Kobayashi, "Reinforcement learning in POMDPs with function approximation," in *International Conference on Machine Learning*, 1997, pp. 152–160.

[46] M. J. Kochenderfer, T. A. Wheeler, and K. H. Wray, *Algorithms for decision making*. MIT press, 2022.

[47] V. Konda and J. Tsitsiklis, "Actor-critic algorithms," in *Advances in neural information processing systems*, vol. 12, 1999.

[48] V. R. Konda and J. N. Tsitsiklis, "On actor-critic algorithms," *SIAM Journal on Control and Optimization*, vol. 42, no. 4, pp. 1143–1166, 2003.

[49] V. Krishnamurthy, *Partially observed Markov decision processes*. Cambridge university press, 2016.

[50] A. Kulesza, N. Jiang, and S. Singh, "Low-rank spectral learning with weighted loss functions," in *Artificial Intelligence and Statistics*, 2015, pp. 517–525.

[51] A. Kulesza, N. Jiang, and S. P. Singh, "Spectral learning of predictive state representations with insufficient statistics." in *AAAI Conference on Artificial Intelligence*, 2015, pp. 2715–2721.

[52] L. Li, T. J. Walsh, and M. L. Littman, "Towards a unified theory of state abstraction for MDPs," in *ISAIM*, 2006.

[53] M. L. Littman, "Algorithms for sequential decision-making," Ph.D. dissertation, Brown University, 1996.

[54] X. Lu, B. Van Roy, V. Dwaracherla, M. Ibrahimi, I. Osband, Z. Wen *et al.*, "Reinforcement learning, bit by bit," *Foundations and Trends in Machine Learning*, vol. 16, no. 6, pp. 733–865, 2023.

[55] A. Mahajan, "Sequential decomposition of sequential dynamic teams: applications to real-time communication and networked control systems," Ph.D. dissertation, University of Michigan, Ann Arbor, MI, 2008.

[56] A. Mahajan, N. C. Martins, M. C. Rotkowitz, and S. Yuksel, "Information structures in optimal decentralized control," in *IEEE Conference on Decision and Control*. IEEE, Dec. 2012.

[57] N. Meuleau, L. Peshkin, K.-E. Kim, and L. P. Kaelbling, "Learning finite-state controllers for partially observable environments," in *Conference on Uncertainty in Artificial Intelligence*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999, p. 427–436.

[58] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[59] A. Müller, "How does the value function of a Markov decision process depend on the transition probabilities?" *Math. Oper. Res.*, vol. 22, no. 4, pp. 872–885, 1997.

[60] ——, "Integral probability metrics and their generating classes of functions," *Advances in Applied Probability*, vol. 29, no. 2, pp. 429–443, 1997.

[61] A. Y. Ng and M. I. Jordan, "PEGASUS: A policy search method for large mdps and pomdps," in *Conference on Uncertainty in Artificial Intelligence*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000, p. 406–415.

[62] T. Ni, B. Eysenbach, E. SeyedSalehi, M. Ma, C. Gehring, A. Mahajan, and P.-L. Bacon, "Bridging state and history representations: Understanding self-predictive RL," in *International Conference on Learning Representations*, 2024.

[63] T. Ni, M. Ma, B. Eysenbach, and P.-L. Bacon, "When do transformers shine in RL? decoupling memory from credit assignment," in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[64] T. J. Perkins and M. D. Pendrith, "On the existence of fixed points for q-learning and sarsa in partially observable domains," in *International Conference on Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2002, p. 490–497.

[65] L. Peshkin, N. Meuleau, and L. P. Kaelbling, "Learning policies with external memory," in *International Conference on Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999, p. 307–314.

[66] J. Pineau, G. Gordon, S. Thrun *et al.*, "Point-based value iteration: An anytime algorithm for POMDPs," in *International Joint Conference on Artificial Intelligence*, vol. 3, 2003, pp. 1025–1032.

[67] L. K. Platzman, "Finite memory estimation and control of finite probabilistic systems." Ph.D. dissertation, Massachusetts Institute of Technology, 1977.

[68] M. Rosencrantz, G. Gordon, and S. Thrun, "Learning low dimensional predictive representations," in *International Conference on Machine Learning*, 2004.

[69] J. Sandell, Nils R., "Control of finite-state, finite-memory stochastic systems," Ph.D. dissertation, Massachussets Institute of Technology, Cambridge, MA, 1974.

[70] J. Schmidhuber, *Making the world differentiable: on using self supervised fully recurrent neural networks for dynamic reinforcement learning and planning in non-stationary environments*. Inst. für Informatik, 1990, vol. 126.

[71] E. SeyedSalehi, N. Akbarzadeh, A. Sinha, and A. Mahajan, "Approximate information state based convergence analysis of recurrent Q-learning," in *European conference on reinforcement learning*, 2023. [Online]. Available: https://arxiv.org/abs/2306.05991

[72] S. Singh, T. Jaakkola, and M. Jordan, "Reinforcement learning with soft state aggregation," in *Advances in Neural Information Processing Systems*, vol. 7. MIT Press, 1994, pp. 361–368.

[73] S. P. Singh, T. Jaakkola, and M. I. Jordan, "Learning without state-estimation in partially observable markovian decision processes," in *Machine Learning*. Elsevier, 1994, pp. 284–292.

[74] A. Sinha, M. Geist, and A. Mahajan, "Title withheld due to double blind review restrictions," in *under review*, 2024.

[75] A. Sinha and A. Mahajan, "Asymmetric actor-critic with approximate information state," in *IEEE Conference on Decision and Control*, 2023, pp. 7810–7816.

[76] R. D. Smallwood and E. J. Sondik, "The optimal control of partially observable Markov processes over a finite horizon," *Operations Research*, vol. 21, no. 5, pp. 1071–1088, Oct. 1973.

[77] T. Smith and R. Simmons, "Heuristic search value iteration for POMDPs," in *Conference on Uncertainty in Artificial Intelligence*, Arlington, Virginia, USA, 2004, p. 520–527.

[78] E. J. Sondik, "The optimal control of partially observable markov processes over the infinite horizon: Discounted costs," *Operations research*, vol. 26, no. 2, pp. 282–304, 1978.

[79] M. T. Spaan and N. Vlassis, "Perseus: Randomized point-based value iteration for POMDPs," *Journal of Artificial Intelligence Research*, vol. 24, pp. 195–220, 2005.

[80] J. Subramanian, A. Sinha, R. Seraj, and A. Mahajan, "Approximate information state for approximate planning and reinforcement learning in partially observed systems," *Journal of Machine Learning Research*, vol. 23, no. 12, pp. 1–83, 2022.

[81] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Advances in Neural Information Processing Systems*, vol. 12, 1999.

[82] J. N. Tsitsiklis and B. Van Roy, "Feature-based methods for large scale dynamic programming," *Machine Learning*, vol. 22, no. 1, pp. 59–94, 1996.

[83] L. N. Vaserstein, "Markov processes over denumerable products of spaces, describing large systems of automata," *Problemy Peredachi Informatsii*, vol. 5, no. 3, pp. 64–72, 1969.

[84] C. Villani *et al.*, *Optimal transport: old and new*. Springer, 2008, vol. 338.

[85] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, pp. 279–292, 1992.

[86] L. Weng, "Policy gradient algorithms," 2018. [Online]. Available: https://lilianweng.github.io/posts/2018-04-08-policy-gradient/

[87] P. Werbos, "Beyond regression: New tools for prediction and analysis in the behavioral sciences," *PhD thesis, Committee on Applied Mathematics, Harvard University, Cambridge, MA*, 1974.

[88] C. C. White III and W. T. Scherer, "Finite-memory suboptimal design for partially observed markov decision processes," *Operations Research*, vol. 42, no. 3, pp. 439–455, 1994.

[89] H. S. Witsenhausen, "A standard form for sequential stochastic control," *Mathematical Systems Theory*, vol. 7, no. 1, pp. 5–11, Mar. 1973.

[90] ——, "On policy independence of conditional expectations," *Information and Control*, vol. 28, no. 1, pp. 65–75, 1975.

[91] L. Yang, K. Zhang, A. Amice, Y. Li, and R. Tedrake, "Discrete approximate information states in partially observable environments," in *American Control Conference*. IEEE, 2022, pp. 1406–1413.

[92] H. Yu, "Approximate solution methods for partially observable Markov and semi-Markov decision processes," Ph.D. dissertation, Massachusetts Institute of Technology, 2006.

[93] N. Zhang and W. Liu, "Planning in stochastic domains: Problem characteristics and approximation," Hong Kong Univeristy of Science and Technology, Tech. Rep. HKUST-CS96-31, 1996.