Approximate information state based convergence analysis of recurrent Q-learning

Erfan Seyedsalehi
McGill University
seyederfan.seyedsalehi@mail.mcgill.ca

Nima Akbarzadeh
McGill University
nima.akbarzadeh@mail.mcgill.ca

Amit Sinha
McGill University
amit.sinha@mail.mcgill.ca

Aditya Mahajan
McGill University
aditya.mahajan@mcgill.ca

Abstract

In spite of the large literature on reinforcement learning (RL) algorithms for partially observable Markov decision processes (POMDPs), a complete theoretical understanding is still lacking. In a partially observable setting, the history of data available to the agent increases over time so most practical algorithms either truncate the history to a finite window or compress it using a recurrent neural network leading to an agent state that is non-Markovian. In this paper, it is shown that in spite of the lack of the Markov property, recurrent Q-learning (RQL) converges in the tabular setting. Moreover, it is shown that the quality of the converged limit depends on the quality of the representation which is quantified in terms of what is known as an approximate information state (AIS). Based on this characterization of the approximation error, a variant of RQL with AIS losses is presented. This variant performs better than a strong baseline for RQL that does not use AIS losses. It is demonstrated that there is a strong correlation between the performance of RQL over time and the loss associated with the AIS representation.

1 Introduction

In recent years, Reinforcement Learning (RL) has witnessed many successes such as achieving human-level performance in Go [Sil+16], learning to play Atari [Mni+13; Mni+15], as well as solving many control problems arising in engineering and robotics [Sch+15b; Sch+17; Haa+18; Tas+18]. These successes are achieved by algorithms with strong theoretical basis [SB18; BT96]. However, RL theory, for the most part, is limited to models with full state information.

In various applications such as finance, healthcare, and robotics, the agent does not observe the full state of the environment. Such partially observed systems are mathematically modeled as partially observable Markov decision processes (POMDPs). When the system model is known, POMDPs can be viewed as MDPs by considering the belief state (i.e., the posterior distribution of the partially observed environmental state) as an information state [Ast65]. Furthermore, there are various efficient algorithms to compute approximately optimal planning solutions [SPK13].

However, it is not possible to generalize these planning results to develop learning algorithms because constructing the belief state requires knowledge of the model. So, an agent operating in an unknown partially observed environment cannot construct a belief state based on its observation. Two approaches are commonly used in the literature to circumvent this conceptual difficulty (i) use a finite window of observations (rather than the full history) and (ii) use a recursively updateable (or recurrent) *agent state*. A key difficulty in analyzing these learning algorithms is that the state of the agent may evolve in a non-Markovian manner. Furthermore, for the recurrent algorithm, the representation mapping histories to agent states needs to be learnt in parallel, which is especially difficult in sparse reward environments. So, even though there is a rich and large literature on RL theory for POMDPs [see LM92; Lit94; SJJ94; JSJ94; McC96; WL95; LS98; PP02, and follow-up liteature], much of the literature either analyzes the case where the agent does not have a memory, or only provides empirical evidence but does not include a detailed convergence or approximation analysis.

In this paper, we investigate one of the most popular RL algorithms for POMDPs: Recurrent Q-learning (RQL), which uses a recurrent neural network (RNN) for approximating a history-based Q-function. RQL was initially proposed by [Sch91; HS92] with a substantial follow up literature [Bak02; Wie+07; Wie+10; DSH13]. There is growing empirical evidence suggesting that variants of RQL work well in practice [HS15; Kap+19; Foe+16; Ou+21; Mou+17; Sor+15]. However, a detailed theoretical understanding of the algorithm is lacking.

Review of theoretical papers analyzing RL for POMDPs There are a few recent papers which analyze closely related problems. A general framework of approximation for POMDPs based on the notion of approximate information state (AIS) is proposed in [Sub+22]. It is shown that a planning policy computed using an AIS is approximately optimal with bounded loss of optimality. Furthermore, an actor-critic algorithm which uses the AIS-approximation losses as an auxiliary loss is presented and it is demonstrated that the proposed algorithm has good empirical performance. Even though AIS may be viewed as a recurrent agent state, the analysis presented in [Sub+22] is for actor-critic algorithms and is not directly applicable to RQL.

Approximate planning and Q-learning for POMDPs with a finite window of observations is presented in [KY22a; KY22b], where the approximation error and convergence are quantified. The special case of just using the current observation has also been analyzed in [SJJ94]. Even though our analysis uses similar technical tools as [SJJ94; KY22a; KY22b], the analysis of these papers is for Q-learning with finite window of observations and is not directly applicable to RQL.

Regret guarantees for RL agents operating in non-Markovian environments and using an optimistic variant of Q-learning is presented in [DVZ22]. Even though the agent state in [DVZ22] is a recurrent state, the analysis of [DVZ22] is tuned for an optimistic variant of Q-learning and is not directly applicable to RQL.

The papers closest to our work are [MH18; CBD22] which establish convergence of Q-learning in a non-Markovian environment under the assumption that the state-observation-action process is stationary and ergodic[1] (an additional technical assumption of state uniformity is also imposed in [MH18]). Asymptotic rates of convergence are also characterized in [CBD22]. However, [MH18; CBD22] do not present explicit approximation bounds. In our analysis, we do not assume that the state-observation-action process is stationary and ergodic to provide approximation bounds.

Contributions Our main contributions are as follows. First, we show that in spite of the non-Markovian evolution of the agent state, RQL converges. As far as we are aware, this is the first result that establishes the convergence of RQL without making any assumptions on the stationarity of the agent state. Second, using ideas from approximate information state (AIS) [Sub+22], we quantify the quality of the converged limit of RQL in terms of error in representation. Third, we propose a variant of RQL called RQL-AIS which incorporates AIS losses. We implement RQL-AIS bssed on R2D2 [Kap+19], a popular RQL algorithm, by incorporating auxiliary AIS losses. On perform detailed numerical experiments on the Minigrid benchmark [CWP18], and find that RQL-AIS performs better than R2D2. We also empirically demonstrate that there is a strong correlation between the performance of RQL over time and the loss associated with the AIS representation.

2 Background

Partially observable Markov decision processes (POMDPs) A partially observable Markov decision process (POMDP) is a tuple $\langle \mathcal{S}, \mathcal{Y}, \mathcal{A}, P, O, r, \gamma \rangle$ where:

[1]This assumption implies that the initial distribution of (state, observation, action) is the same as the steady state distribution induced by the exploration policy.

- $\mathcal{S}$ denotes the state space, which is assumed to be finite. The state at time t is denoted by S_t.
- $\mathcal{Y}$ denotes the observation space, which is also assumed to be finite. The observation at time t is denoted by Y_t.
- $\mathcal{A}$ denotes the action space, which is also assumed to be finite. The action at time t is denoted by A_t.
- $P\colon \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ denote the state transition matrix. In particular, for any time t and any $s_{1:t+1} \in \mathcal{S}^{t+1}$, $y_{1:t} \in \mathcal{Y}^{t+1}$, and $a_{1:t} \in \mathcal{A}^t$, we have

$$\begin{aligned}&\mathbb{P}(S_{t+1} = s_{t+1} \mid S_{1:t} = s_{1:t}, Y_{1:t} = y_{1:t}, A_{1:t} = a_{1:t})\\ &\qquad = \mathbb{P}(S_{t+1} = s_{t+1} \mid S_t = s_t, A_t = a_t) =: P(s_{t+1} \mid s_t, a_t).\end{aligned}$$

- $O\colon \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{Y})$ denotes the observation probability matrix. In particular, for any time t any $s_{1:t} \in \mathcal{S}^t$, $y_{1:t} \in \mathcal{Y}^t$, and $a_{1:t-1} \in \mathcal{A}^{t-1}$, we have

$$\begin{aligned}&\mathbb{P}(Y_t = y_t \mid S_{1:t} = s_{1:t}, Y_{1:t-1} = y_{1:t-1}, A_{1:t-1} = a_{1:t-1})\\ &\qquad = \mathbb{P}(Y_t = y_t \mid S_t = s_t, A_{t-1} = a_{t-1}) =: O(y_t \mid s_t, a_{t-1}).\end{aligned}$$

- $r\colon \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the per-step reward function.
- $\gamma \in (0, 1)$ denotes the discount factor.

We use the following additional notation. $H_t = (Y_1, A_1, Y_2, A_2, \dots, Y_t)$ denotes the history of observations and actions until time t; $\mathcal{H}_t = \mathcal{Y}^t \times \mathcal{A}^{t-1}$ denotes the space of all histories until time t; $R_t = r(S_t, A_t)$ denotes the random reward received at time t.

A policy $\pi = (\pi_1, \pi_2, \dots)$ is a collection of history dependent randomized decision rules $\pi_t\colon \mathcal{H}_t \to \Delta(\mathcal{A})$ such that the action at time t is chosen according to $A_t \sim \pi_t(H_t)$. The performance of any policy π starting from history $h_t \in \mathcal{H}_t$ at time t is given by the value function $V^\pi_t(h_t)$ defined as

$$V^\pi_t(h_t) \coloneqq \mathbb{E}^\pi\Big[\sum_{\tau=t}^{\infty} \gamma^{\tau-t} r(S_\tau, A_\tau) \Bigm| H_t = h_t\Big]. \tag{1}$$

The corresponding action-value function or Q-function $Q^\pi_t(h_t, a_t)$ is defined as

$$Q^\pi_t(h_t, a_t) \coloneqq \mathbb{E}^\pi\big[r(S_t, A_t) + \gamma V^\pi_{t+1}(H_{t+1}) \bigm| H_t = h_t, A_t = a_t\big]. \tag{2}$$

A policy $\pi^\star$ is called *optimal* if for every other policy π, we have $V^{\pi^\star}_t(h_t) \geq V^\pi_t(h_t)$, for all $t \in \mathbb{Z}_{>0}$ and $h_t \in \mathcal{H}_t$. The value function and action-value function of optimal policies are denoted by $V^\star_t$ and $Q^\star_t$.

Integral Probability Metrics (IPMs) Integral probability metrics (IPMs) are a family of semi-metrics on probability measures defined in terms of a dual relationship [Mül97].

Definition 1. Let $(\mathcal{X}, \mathscr{G})$ be a measurable space and $\mathfrak{F}$ be a class of measurable real-valued functions on $(\mathcal{X}, \mathscr{G})$. The integral probability metric (IPM) between two probability distributions $\mu, \nu \in \mathscr{P}(\mathcal{X})$ with respect to the function class $\mathfrak{F}$ is defined as $d_{\mathfrak{F}}(\mu, \nu) \coloneqq \sup_{f \in \mathfrak{F}} \big| \int_{\mathcal{X}} f d\mu - \int_{\mathcal{X}} f d\nu \big|$.

A key property of IPMs is that for any function f (not necessarily in $\mathfrak{F}$), we have

$$\Big| \int_{\mathcal{X}} f d\mu - \int_{\mathcal{X}} f d\nu \Big| \leq \rho_{\mathfrak{F}}(f) \cdot d_{\mathfrak{F}}(\mu, \nu), \tag{3}$$

where $\rho_{\mathfrak{F}}(f) \coloneqq \inf\{\rho \in \mathbb{R}_{>0} : \rho^{-1} f \in \mathfrak{F}\}$ is called the Minkowski functional of f.

Some examples of IPMs are as follows: (i) **Total variation distance** where $\mathfrak{F} = \mathfrak{F}_{\text{TV}} \coloneqq \{f : \operatorname{span}(f) \leq 1\}$ (where $\operatorname{span}(f)$ is the span semi-norm of a function). For this case, $\rho_{\text{TV}}(f) = \operatorname{span}(f)$. (ii) **Wasserstein distance** where $\mathfrak{F} = \mathfrak{F}_{\text{Was}} \coloneqq \{f : \operatorname{Lip}(f) \leq 1\}$ (where $\mathcal{X}$ is a metric space and $\operatorname{Lip}(f)$ is the Lipschitz constant of the function f, computed with respect to the metric on $\mathcal{X}$). For this case, $\rho_{\text{Was}}(f) = \operatorname{Lip}(f)$. (iii) **Maximum mean discrepancy (MMD)** where $\mathfrak{F} = \mathfrak{F}_{\text{MMD}} \coloneqq \{f \in \mathcal{H} \colon \|f\|_{\mathcal{H}} \leq 1\}$ (where $\mathcal{H}$ is a reproducing kernel Hilbert space of real-valued functions on $\mathcal{X}$ and $\|f\|_{\mathcal{H}}$ is the Hilbert space norm of f). For this case, $\rho_{\text{MMD}}(f) = \|f\|_{\mathcal{H}}$.

Approximate information state (AIS) Approximate information state (AIS) is a self-predictive representation for POMDPs, first proposed in [Sub+22].

Definition 2. Given a function class $\mathfrak{F}$ and a measurable space $\mathcal{Z}$, an $(\varepsilon_t, \delta_t)_{t\geq 1}$ AIS-generator is a tuple $\langle \{\sigma_t\}_{t\geq 1}, \tilde{P}, \tilde{r}\rangle$ of history compression functions $\sigma_t \colon \mathcal{H}_t \to \mathcal{Z}$, transition approximator $\tilde{P} \colon \mathcal{Z} \times \mathcal{A} \to \Delta(\mathcal{Z})$, and reward approximator $\tilde{r} \colon \mathcal{Z} \times \mathcal{A} \to \mathbb{R}$ such that for all $h_t \in \mathcal{H}_t$ and $a_t \in \mathcal{A}$

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

$$d_{\mathfrak{F}}\big(\mathbb{P}\left(Z = \cdot \mid H_t = h_t, A_t = a_t\right), \tilde{P}(\cdot \mid \sigma_t\left(h_t\right), a_t)\big) \leq \delta_t.$$

Given an AIS generator, consider the following dynamic program:

$$\tilde{Q}(z,a) = \tilde{r}(z,a) + \gamma \int_{\mathcal{Z}} \tilde{P}(dz' \mid z, a) \max_{\tilde{a}\in\mathcal{A}} \tilde{Q}(z', \tilde{a}). \tag{4}$$

Let $\tilde{Q}^\star$ denote the unique fixed point of (4). Define $\tilde{V}^\star \colon \mathcal{Z} \to \mathbb{R}$ to be the value function corresponding to $\tilde{Q}^\star$ and $\tilde{\pi}^\star \colon \mathcal{Z} \to \mathcal{A}$ to be the greedy policy[2] with respect to $\tilde{Q}^\star$, i.e.,

$$\tilde{V}^\star(z) = \max_{a\in\mathcal{A}} \tilde{Q}^\star(z,a), \quad \text{and} \quad \tilde{\pi}^\star(z) = \arg\max_{a\in\mathcal{A}} \tilde{Q}^\star(z,a).$$

Then, the following result is a generalization of [Sub+22, Theorem 27].

Theorem 1. Let $\tilde{\pi} = (\tilde{\pi}_1, \tilde{\pi}_2, \dots)$ be a time-varying and history-dependent policy given by $\tilde{\pi}_t(h_t) = \tilde{\pi}^\star(\sigma_t(h_t))$. Then, for any time t and any history $h_t \in \mathcal{H}_t$ and action $a_t \in \mathcal{A}$, we have

- **Bounds on value approximation:**

$$\big|Q^\star_t(h_t, a_t) - \tilde{Q}^\star(\sigma_t(h_t), a_t)\big| \leq (1-\gamma)^{-1}\big[\bar{\varepsilon}_t + \gamma\bar{\delta}_t\rho_{\mathfrak{F}}(\tilde{V}^\star)\big], \tag{5}$$

$$\big|V^\star_t(h_t) - \tilde{V}^\star(\sigma_t(h_t))\big| \leq (1-\gamma)^{-1}\big[\bar{\varepsilon}_t + \gamma\bar{\delta}_t\rho_{\mathfrak{F}}(\tilde{V}^\star)\big], \tag{6}$$

  where $\bar{\varepsilon}_t = (1-\gamma)\sum_{\tau=t}^{\infty}\gamma^{\tau-t}\varepsilon_\tau$ and $\bar{\delta}_t = (1-\gamma)\sum_{\tau=t}^{\infty}\gamma^{\tau-t}\delta_\tau$.

- **Bounds on policy approximation:**

$$\big|V^\star_t(h_t) - V^{\tilde{\pi}}_t(h_t)\big| \leq 2(1-\gamma)^{-1}\big[\bar{\varepsilon}_t + \gamma\bar{\delta}_t\rho_{\mathfrak{F}}(\tilde{V}^\star)\big]. \tag{7}$$

Latent state models for MDPs were introduced in [Gel+19] and it was shown that similar to the AIS setting, minimizing the per-step reward and next latent state prediction can provide a bound on the quality of the learned representations. With AIS models, we focus on partially observable settings which are more general than MDPs.

Recurrent Neural Networks (RNN) Recurrent neural networks (RNNs) are neural networks with feedback connections that are used to process sequential data by keeping track of a state. At an abstract level, we may model an RNN with a hidden state[3] z_t to be a function of the past sequence of inputs $x_1, \dots, x_t$, which is updated recursively using a non-linear activation function: $z_t = f(z_{t-1}, x_t)$. Typically, $f(\cdot)$ is a parameterized family of functions, e.g., in vanilla RNN, $f(z_{t-1}, x_t) = \tanh(W_{zz}z_{t-1} + W_{zx}x_t + W_b)$, where (W_{zz}, W_{zx}, W_b) are parameters. In practice, one uses more sophisticated RNN architectures such as long short-term memory (LSTM) [HS97] or gated recurrent units (GRUs) [Cho+14], which mitigate the problem of vanishing gradients.

Recurrent Q-learning Recurrent Q-learning (RQL) is a variant of Q-learning algorithm for POMDPs, which uses an RNN to estimate the Q-function $Q_t(h_t, a_t)$ [Sch91; HS92]. In particular, an RNN with input (Y_t, A_{t-1}) is used to generate a hidden state $z_t \in \mathcal{Z}$ which is updated recursively as $z_t = f(z_{t-1}, y_t, a_{t-1})$, where $f(\cdot)$ is the update function of an RNN. We will sometimes write $z_t = \sigma_t(h_t)$ to highlight the fact that z_t is a function of the history h_t.

In RQL the learning agent uses an exploration policy π_{expl} to generate experience and updates an estimate of the Q-function using the following recursion:

$$\widehat{Q}_{t+1}(z_t, a_t) = \widehat{Q}_t(z_t, a_t) + \alpha_t(z_t, a_t)\big[R_t + \gamma \max_{\tilde{a}\in\mathcal{A}} \widehat{Q}_t(z_{t+1}, \tilde{a}) - \widehat{Q}_t(z_t, a_t)\big] \tag{8}$$

[2]To avoid ambiguity due to the non-uniqueness of the arg-max, we assume that there is a deterministic rule to break ties is pre-specified so that the arg-max is always unique.

[3]Normally, the hidden state of an RNN is denoted using h_t. However, we are using h_t to denote the history of a POMDP. So, we use z_t to denote the hidden state of an RNN.

where $\{\alpha_t(z_t, a_t)\}_{t\ge1}$ is the learning rate. Define $\hat{V}_t \colon \mathcal{Z} \to \mathbb{R}$ to be the value function corresponding to $\widehat{Q}_t$ and $\hat{\pi}_t \colon \mathcal{Z} \to \mathcal{A}$ to be the greedy policy w.r.t. $\widehat{Q}_t$, i.e.,

$$\hat{V}_t(z) = \max_{a\in\mathcal{A}} \widehat{Q}_t(z,a) \quad \text{and} \quad \hat{\pi}_t(z) = \arg\max_{a\in\mathcal{A}} \widehat{Q}_t(z,a).$$

3 Theoretical results

The key challenge in characterizing the convergence of RQL is that the agent state $\{Z_t\}_{t\ge1}$ is not a controlled Markov process. Therefore, the standard results on the convergence of Q-learning [JJS93] are not directly applicable. In Sec. 3.1, we show that it is possible to adapt the standard convergence arguments to show that RQL converges.

The quality of the converged solution depends on choice of the exploration policy as well as the representation. The dependence of the representation is not surprising. For example, it is clear that when the representation is bad (e.g., a representation that maps all histories to a single agent state), then RQL will converge to a limit which is far from optimal. So, it is important to quantify the degree of sub-optimality of the converged limit. We do so in Sec. 3.2.

3.1 Establishing the convergence of RQL

Lemma 1. Under any policy $\pi \colon \mathcal{Z} \to \Delta(\mathcal{A})$, the process $\{(S_t, Y_t, Z_t, A_t)\}_{t\ge1}$ is a Markov chain.

We impose the following assumptions:

(A1) The state space, action space, and the recurrent state space are finite.
(A2) The exploration policy $\pi_{\text{expl}} \colon \mathcal{Z} \to \Delta(\mathcal{A})$ is such that the Markov chain $\{(S_t, Y_t, Z_t, A_t)\}_{t\ge1}$ has a unique stationary distribution ξ. Moreover, for every (s, y, z, a), $\xi(s, y, z, a) > 0$.
(A3) The learning rate $\alpha_t(z, a)$ is given by $\alpha_t(z,a) = \mathbb{1}_{\{Z_t=z, A_t=a\}} \big/ \big(1 + \sum_{\tau=0}^{t} \mathbb{1}_{\{Z_\tau=z, A_\tau=a\}}\big)$.

We impose assumption **(A1)** to analyze the simplest version of the RQL. Assumption **(A2)** is a mild assumption on the exploration policy and is commonly assumed in several variations of Q-learning with function approximation [TV97]. Assumption **(A3)** is a common assumption on the step-size of stochastic approximation algorithms.

For the ease of notation, we continue to use ξ to denote marginal and conditional distributions w.r.t. ξ. For example, $\xi(y, z, a) = \sum_{s\in\mathcal{S}} \xi(s, y, z, a)$ and similar notation holds for other marginals. Similarly, $\xi(s|z) = \xi(s, z)/\xi(z)$ and similar notation holds for other conditional distributions.

Given a steady-state distribution ξ corresponding to the exploration policy, define a reward function $r_\xi \colon \mathcal{Z} \times \mathcal{A} \to \mathbb{R}$ and transition probability $P_\xi \colon \mathcal{Z} \times \mathcal{A} \to \Delta(\mathcal{Z})$ as follows:

$$r_\xi(z,a) = \sum_{s\in\mathcal{S}} r(s,a)\xi(s \mid z,a),$$
$$P_\xi(z' \mid z,a) = \sum_{s\in\mathcal{S}} \xi(s \mid z,a) \sum_{s'\in\mathcal{S}} P(s'|s,a) \sum_{y'\in\mathcal{Y}} O(y'|s',a)\mathbb{1}_{\{z'=f(z,y',a)\}}.$$

Furthermore, define $Q^\star_\xi$ to be the unique fixed point of the following fixed point equation:

$$Q^\star_\xi(z,a) = r_\xi(z,a) + \gamma \sum_{z'\in\mathcal{Z}} P_\xi(z' \mid z,a) \max_{\tilde{a}\in\mathcal{A}} Q^\star_\xi(z,\tilde{a}). \tag{9}$$

Define $V^\star_\xi \colon \mathcal{Z} \to \mathbb{R}$ be the value function corresponding to $Q^\star_\xi$ and $\pi^\star_\xi \colon \mathcal{Z} \to \mathcal{A}$ to be the greedy policy with respect to $Q^\star_\xi$, i.e.,

$$V^\star_\xi(z) = \max_{a\in\mathcal{A}} Q^\star_\xi(z,a), \quad \text{and} \quad \pi^\star_\xi(z) = \arg\max_{a\in\mathcal{A}} Q^\star_\xi(z,a).$$

Theorem 2. Under Assumptions **(A1)–(A3)**, the iterates $\{\widehat{Q}_t\}_{t\ge1}$ of (8) converge almost surely to $Q^\star_\xi$ given by (9). Therefore $\{\hat{\pi}_t\}_{t\ge1}$ converges to $\pi^\star_\xi$ (see footnote 2 for uniqueness of arg-max).

Proof outline. The main idea of the proof is inspired from [JJS93; KY22a]. To establish that $\widehat{Q}_t \to Q^\star_\xi$, a.s., we will show that $\Delta_t \coloneqq \widehat{Q}_t - Q^\star_\xi \to 0$, a.s. Define $\hat{V}_t(z) = \max_{a\in\mathcal{A}} \widehat{Q}_t(z,a)$. Combining (8) and (9), we get that

$$\Delta_{t+1}(z,a) = (1-\alpha_t(z,a))\Delta_t(z,a) + \alpha_t(z,a)\big[F^1_t(z,a) + F^2_t(z,a)\big], \tag{10}$$

where

$$F^1_t(z,a) = \gamma \hat{V}_t(Z_{t+1}) - \gamma V^\star_\xi(Z_{t+1}), \tag{11}$$

$$F^2_t(z,a) = R_t - r_\xi(z,a) + \gamma V^\star_\xi(Z_{t+1}) - \gamma \textstyle\sum_{z' \in \mathcal{Z}} P_\xi(z'|z,a) V^\star_\xi(z'). \tag{12}$$

Following [JJS93], we view (10) as a linear system with two inputs and do "state splitting" to write

$$\Delta_t(z,a) = W^1_t(z,a) + W^2_t(z,a) \tag{13}$$

where for $i \in \{1,2\}$, each "state component" $W^i_1(z,a)$ is initialized to 0 and evolves for $t \geq 1$ as

$$W^i_{t+1}(z,a) = (1 - \alpha_t(z,a)) W^i_t(z,a) + \alpha_t(z,a) F^i_t(z,a), \quad i \in \{1,2\}.$$

From (11), we have that

$$F^1_t(z,a) = \gamma\big[\hat{V}_t(Z_{t+1}) - V^\star_\xi(Z_{t+1})\big] \leq \gamma \|\hat{V}_t - V^\star_\xi\|_\infty \leq \gamma \|\widehat{Q}_t - Q^\star_\xi\|_\infty = \gamma \|\Delta_t\|_\infty. \tag{14}$$

Using assumptions **(A1)–(A3)**, we can show that $W^2_t(z,a) \to 0$, a.s., for all (z,a). See the supplementary material for proof. Therefore, there exists a set Ω_0 such that $\mathbb{P}(\Omega_0) = 1$ and for every $\omega \in \Omega_0$ and any $\epsilon > 0$, there exists a $T(\omega,\epsilon)$ such that for all $t > T(\omega,\epsilon)$, $|W^2_t(z,a)| < \epsilon$, a.s., for all (z,a). Now, pick C such that $\gamma(1 + 1/C) < 1$. For any $t > T(\omega,\epsilon)$ if $\|W^1_t(z,a)\|_\infty > C\epsilon$, then

$$F^1_t(z,a) \leq \gamma \|\Delta_t\|_\infty \leq \gamma \|W^1_t\|_\infty + \gamma\epsilon < \gamma\big(1 + \tfrac{1}{C}\big)\|W^1_t\|_\infty < \|W^1_t\|_\infty, \tag{15}$$

where the first inequality uses (13), the second uses the triangle inequality, and the others follow from the definition of C and ϵ. Consequently, for any $t > T(\omega,\epsilon)$ and $\|W^1_t(z,a)\|_\infty > C\epsilon$, we have that

$$W^1_{t+1}(z,a) = (1 - \alpha_t(z,a)) W^1_t(z,a) + \alpha_t(z,a) F^1_t(z,a) < \|W^1_t\|_\infty. \tag{16}$$

Hence, when $\|W^1_t\|_\infty > C\epsilon$, it decreases monotonically. So, there are two possibilities: either it gets below $C\epsilon$ or it never goes below $C\epsilon$. In the supplementary material, we show that the process cannot stay above $C\epsilon$ all the time. Hence, it must hit below $C\epsilon$ at some point. In the supplementary material, we show that once the process hits below $C\epsilon$, it stays there. Thus, we have shown that for all sufficiently large t, $\|W^1_t\|_\infty < C\epsilon$, a.s. Since ϵ is arbitrary, this implies that $W^1_t(z,a) \to 0$, a.s., for all (z,a). Combining this with the fact that $W^3_t(z,a) \to 0$, we get that $\Delta_t(z,a) = W^1_t(z,a) + W^2_t(z,a) \to 0$, a.s., for all (z,a). Let $\mathcal{F}_t = \sigma(W^1_{1:t}, F^1_{1:t}, \alpha_{1:t})$. Then, $\|\mathbb{E}[F^1_t(z,a) \mid \mathcal{F}_t]\|_\infty \leq \mathbb{E}[\|F^1_t(z,a)\|_\infty \mid \mathcal{F}_t] \leq \gamma\|\Delta_t\|_\infty$. Moreover, since the state space is finite, the per-step reward is bounded. Therefore, both $\hat{V}_t(Z_{t+1})$ and $V^\star_\xi(Z_{t+1})$ are bounded. Therefore, there exists a constant C such that $\text{var}(F^1_t(z,a) \mid \mathcal{F}_t) \leq C$. Thus, the iteration for $W^1_t(z,a)$ satisfies all conditions of [JJS93, Theorem 1], and by that result, $W^1_t(z,a) \to 0$, a.s., for all (z,a). For convenience, we restate [JJS93, Theorem 1] in the supplementary material. □

Remark 1. In the special case when $z_t = (y_{t-n:t}, a_{t-n:t-1})$ is the last n observations and actions (i.e., frame-stacking), the result of Theorem 2 recovers the result of [KY22a, Theorem 4.1, (i)].

Remark 2. In [CBD22, Theorem 1], it was shown that the results of Theorem 2 hold if **(A2)** and **(A3)** are replaced by the following:

(A2') Assumption **(A2)** holds and the initial distribution satisfies $\mathbb{P}(S_1 = s, Z_1 = z) = \xi(s,z)$.
(A3') The learning rate $\alpha_t(z,a)$ satisfies $\sum_{t\geq 1} \alpha_t(z,a) = \infty$ and $\sum_{t \geq 1} \alpha^2_t(z,a) < \infty$, a.s.

Note that **(A2')** is stronger than **(A2)** and implies that the process $\{(S_t, Y_t, Z_t, A_t)\}_{t\geq 1}$ is stationary and ergodic. In contrast, assumption **(A2)** does not impose any restriction on the initial state distribution and simply assumes that the induced Markov chain $\{(S_t, Y_t, Z_t, A_t)\}_{t \geq 1}$ has a steady-state distribution. However, the learning rate condition **(A3')** is weaker than **(A3)**.

Theorem 2 addresses the main challenge in convergence analysis of RQL: the non-Markovian dynamics. We have shown that RQL converges. So, the next question is: How good is the converged solution compared to the optimal? We address this question in the next section.

3.2 Characterizing the approximation error of the converged value

Our key observation is that for any $\mathfrak{F}$, (σ_t, P_ξ, r_ξ) is an $(\epsilon_t, \delta_t)_{t \geq 1}$ AIS-generator where

$$\varepsilon_t := \max_{h_t \in \mathcal{H}_t, a_t \in \mathcal{A}} \big|\mathbb{E}[r(S_t, A_t) \mid H_t = h_t, A_t = a_t] - r_\xi(\sigma_t(h_t), a_t)\big|,$$

$$\delta_t := \max_{h_t \in \mathcal{H}_t, a_t \in \mathcal{A}} d_{\mathfrak{F}}\big(\mathbb{P}(Z_{t+1} = \cdot \mid H_t = h_t, A_t = a_t), P_\xi(\cdot \mid \sigma_t(h_t), a_t)\big).$$

Therefore, an immediate implication of Theorem 1 is the following.

Theorem 3. Let $\tilde{\pi} = (\tilde{\pi}_1, \tilde{\pi}_2, \dots)$ be a time-varying and history-dependent policy given by $\tilde{\pi}_t(h_t) = \pi^\star_\xi(\sigma_t(h_t))$. Then, for any time t and any history $h_t \in \mathcal{H}_t$ and action $a_t \in \mathcal{A}$, we have:

- **Bounds on value approximation:**
$$\left|Q^\star_t(h_t, a_t) - Q^\star_\xi(\sigma_t(h_t), a_t)\right| \le (1-\gamma)^{-1}\left[\bar{\varepsilon}_t + \gamma\bar{\delta}_t\rho_{\mathfrak{F}}(V^\star_\xi)\right], \tag{17}$$
$$\left|V^\star_t(h_t) - V^\star_\xi(\sigma_t(h_t))\right| \le (1-\gamma)^{-1}\left[\bar{\varepsilon}_t + \gamma\bar{\delta}_t\rho_{\mathfrak{F}}(V^\star_\xi)\right], \tag{18}$$
where $\bar{\varepsilon}_t = (1-\gamma)\sum_{\tau=t}^{\infty}\gamma^{\tau-t}\varepsilon_\tau$ and $\bar{\delta}_t = (1-\gamma)\sum_{\tau=t}^{\infty}\gamma^{\tau-t}\delta_\tau$.

- **Bounds on policy approximation:**
$$\left|V^\star_t(h_t) - V^{\tilde{\pi}}_t(h_t)\right| \le 2(1-\gamma)^{-1}\left[\bar{\varepsilon}_t + \gamma\bar{\delta}_t\rho_{\mathfrak{F}}(V^\star_\xi)\right]. \tag{19}$$

Remark 3. We may upper bound $\bar{\varepsilon}_t$ by $\bar{\varepsilon}^\circ_t := \sup_{\tau \ge t}\varepsilon_\tau$ and $\bar{\delta}_t$ by $\bar{\delta}^\circ_t := \sup_{\tau \ge t}\delta_\tau$.

The bound of Theorem 3 is *instance dependent* because it depends on the value function $V^\star_\xi$. In the supplementary material, we illustrate *instance independent* bounds by upper bounding $\rho_{\mathfrak{F}}(V^\star_\xi)$ in terms of properties of the transition and reward functions.

4 RQL with AIS losses

The results of Theorem 3 suggest that the performance of RQL could be enhanced by improving the representation function f so as to minimize the approximation losses $\varepsilon = (\varepsilon_1, \varepsilon_2, \dots)$ and $\delta = (\delta_1, \delta_2, \dots)$. A similar idea was proposed in [Sub+22] to improve the performance of actor-critic algorithms. In this section, we verify that adding such *AIS losses* improves the performance of RQL.

4.1 Adding AIS losses to RQL

The key idea is to model each component of the AIS generator as a parametric family of functions/distribution and then use stochastic gradient descent to update these parameters. Note that in RQL, we use a RNN to model the state update function f. Please keep in mind that we are relaxing the discreteness assumption for the recurrent hidden state space in this section. This allows us to use RNN architectures such as LSTMs [HS97]. As explained in [Sub+22, Proposition 8], in this case we can use an observation predictor $\tilde{P}^y : \mathcal{Z} \times \mathcal{A} \to \Delta(\mathcal{Y})$ as a proxy for the state predictor $\tilde{P}$ and replace δ_t by $\tilde{\delta}_t / \bar{\kappa}_{\mathfrak{F}}(f)$, where
$$\tilde{\delta}_t := \max_{h_t \in \mathcal{H}_t, a_t \in \mathcal{A}} d_{\mathfrak{F}}\big(\mathbb{P}(Y_{t+1} = \cdot \mid H_t = h_t, A_t = a_t), \tilde{P}^y(\cdot \mid \sigma_t(h_t), a_t)\big).$$
and $\bar{\kappa}_{\mathfrak{F}}(f) = \sup_{z \in \mathcal{Z}, a \in \mathcal{A}} \kappa_{\mathfrak{F}}(f(z, \cdot, a))$.

Let ψ denote the combined parameters of the AIS-generator. Then, we could choose the AIS loss function $\mathcal{L}_{\text{AIS}}$ as any monotonic function of $(\varepsilon_t, \delta_t)$ because reducing $(\varepsilon_t, \delta_t)$ reduces the upper bound of Theorem 3. As suggested in [Sub+22], we choose the AIS loss as
$$\mathcal{L}_{\text{AIS}}(\psi) = \frac{1}{T}\sum_{t=1}^{T}(\lambda\varepsilon^2 + (1-\lambda)\tilde{\delta}^2)$$
where λ is a hyper-parameter and T is the batch size. A detailed discussion of the choice of IPM is presented in [Sub+22]. We choose $d_{\mathfrak{F}}$ as ℓ_2-distance based MMD [see Sub+22, Proposition 32] for which case the AIS loss can be simplified as [Sub+22, Proposition 33]:
$$\mathcal{L}_{\text{AIS}}(\psi) = \frac{1}{T}\sum_{t=1}^{T}\Big[\lambda\big|R_t - \tilde{r}(Z_t, A_t)\big|^2 + (1-\lambda)(M^y_t - 2Y_t)^\intercal M^y_t\Big] + \text{const}(\psi)$$
where M^y_t is the mean of the distribution $\tilde{P}^y(\cdot \mid Z_t, A_t)$ and $\text{const}(\psi)$ are terms which do not depend on ψ and can therefore be ignored. Then, we update the parameters ψ of the AIS generator using stochastic gradient descent.

The updates of the representation using AIS losses is carried out in parallel with RQL. For our experiments, we choose R2D2 [Kap+19], which generalizes Double Q-learning (DQL) with replay buffers to RNNs [HGS15]. Note that as in [Sub+22], we do not backpropagate the Q-learning losses to the AIS generator. A block diagram showing the network architecture is shown in Figure 1. The complete implementation details are presented in the supplementary material.

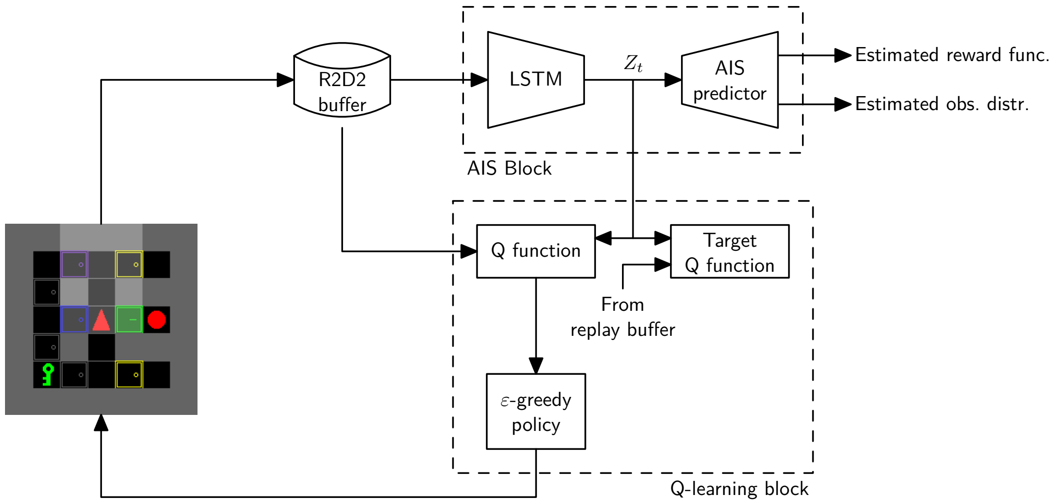


Figure 1: Network architecture of RQL with AIS losses

4.2 Empirical evaluation

In this section, we compare the performance of `RQL-AIS` (the algorithm described in the previous section) with a non-distributed[4] variant of R2D2 proposed in [Kap+19], which we label as `ND-R2D2`. The exact implementation details, including the choice of hyper-parameters are presented in the supplementary material. We want to emphasize that the two implementations are identical except that `RQL-AIS` includes the AIS loss block and updates the parameters ψ of the representation by backpropogating the AIS loss $\mathcal{L}_{\text{AIS}}(\psi)$ rather than backpropogating the Q-learning losses.

We evaluate the two algorithms on 22 environments from the MiniGrid benchmark [CWP18], which are partially observed MiniGrid environments with tasks with increasing complexity, described in detail in the supplementary material. In all environments, the layout at the start of each episode is randomly chosen, so the agent cannot solve the task by memorizing an exact sequence of actions. Similar to [Sub+22; HS18], before running the experiments, we train an autoencoder on a dataset of random agent observations to compress the observations of the agent into compact vectors. The weights of the autoencoder are kept frozen during the RL experiments.

Table 1: Comparison of RQL-AIS and ND-R2D2 on the MiniGrid benchmark

Environment	RQL-AIS	ND-R2D2
SimpleCrossingS9N1	0.789 ± 0.166	0.926 ± 0.044
SimpleCrossingS9N2	0.944 ± 0.007	0.757 ± 0.423
SimpleCrossingS9N3	0.934 ± 0.006	0.933 ± 0.019
SimpleCrossingS11N5	0.835 ± 0.072	0.285 ± 0.140
LavaCrossingS9N1	0.853 ± 0.054	0.955 ± 0.005
LavaCrossingS9N2	0.926 ± 0.014	0.934 ± 0.034
LavaCrossingS9N3	0.871 ± 0.063	0.920 ± 0.029
LavaCrossingS11N5	0.316 ± 0.145	0.310 ± 0.254
Unlock	0.956 ± 0.011	0.969 ± 0.002
UnlockPickup	0.517 ± 0.474	0.000 ± 0.000
DoorKey-5x5	0.926 ± 0.053	0.964 ± 0.001
DoorKey-6x6	0.954 ± 0.005	0.967 ± 0.005
DoorKey-8x8	0.942 ± 0.038	0.371 ± 0.508
KeyCorridorS3R1	0.937 ± 0.026	0.944 ± 0.012
KeyCorridorS3R2	0.885 ± 0.038	0.000 ± 0.000
KeyCorridorS3R3	0.155 ± 0.347	0.000 ± 0.000
ObstructedMaze-1Dl	0.916 ± 0.020	0.000 ± 0.000
ObstructedMaze-1Dlh	0.000 ± 0.000	0.000 ± 0.000
MultiRoom-N2-S4	0.790 ± 0.049	0.839 ± 0.010
MultiRoom-N4-S5	0.438 ± 0.400	0.000 ± 0.000
RedBlueDoors-6x6	0.865 ± 0.075	0.977 ± 0.011
RedBlueDoors-8x8	0.977 ± 0.009	0.962 ± 0.018

We train both `RQL-AIS` and `ND-R2D2` for $T = 4 \cdot 10^6$ environment steps for $N = 5$ seeds. During the data gathering phase, the agent behaves according to the epsilon-greedy approach [Mni+13] to allow for exploration of the environment with epsilon value exponentially decreasing over time. We run two set of studies: one with uniform sampling from the replay buffer and the other using prioritized

[4]In [Kap+19], the distributed setup is introduced to allow for parallel data collection and training in environments requiring very high number of interactions. This is not the case in our setup and both the RQL-AIS and R2D2 are implemented in a non-distributed manner.

experience replay (PER) [Sch+15a]. We report the results for uniform sampling here and the results for PER in the supplementary material.

The mean and standard deviation of the final performance for both algorithms is shown in Table 1. Figure 2 shows the training curves for 8 representative environments. Training curves for all environments are included in the supplementary material.

Discussion of the results For the simpler environments, both `RQL-AIS` and `ND-R2D2` learn to solve the task, with `ND-R2D2` performing slightly better. However, there are seven environments where `ND-R2D2` fails to learn. This is not surprising as the MiniGrid environments are sparse reward environments which are used as a benchmark for research on exploration methods in RL [Par+21; Mav+22; Zha+21; JGR21; Al-+18; Goy+19]. `ND-R2D2`, which does not include any exploration bonuses, fails to learn in some of the larger environments. What is surprising is that `RQL-AIS` is still able to learn in such environments without any exploration bonuses. We show in the supplementary material that adding prioritized experience replay boosts the performance of `RQL-AIS` in the harder environments.

What is the impact of AIS losses on learning? To understand the impact of AIS losses on learning, we compare the evolution of the MMD distance and the episodic return with the number of environment steps. The results are shown in Figure 3, where the environment steps are plotted on a log scale. In each environment, when the MMD loss is initially high, the episodic return does not improve considerably. However, when the MMD distance loss becomes smaller (≈ 0.5), the episodic return starts improving. This suggests that it takes some time for the agent to learn a good representation through the AIS losses, and once a good representation has been obtained (small MMD loss), RL losses through Q-learning are much more effective in training policies, thus allowing policies to improve much quicker, i.e., with fewer samples. This makes sense because optimizing the AIS losses effectively gives the same representation to several different possible histories (but identical from the perspective of performance), which then allows Q-learning updates to propagate through all these possible histories instead of just a single history trajectory. The ability to map more histories to fewer representations accurately improves with the quality of approximation of the AIS. Thus, AIS losses not only help in establishing theoretical performance bounds, but it is evident that they help in learning in empirical experiments.

5 Conclusion

In this work, we establish the convergence of recurrent Q-learning (RQL) in the tabular setting for POMDPs using a representation of the history called an approximate information state (AIS). We also establish upper bounds on the degree of sub-optimality of the converged solution. These bounds quantify the relationship between the quality of representation and the quality of the converged solution of RQL. Based on these bounds, a variant of RQL called RQL-AIS was proposed and it was observed that RQL-AIS performs better than the state-of-the-art baseline ND-R2D2 on the

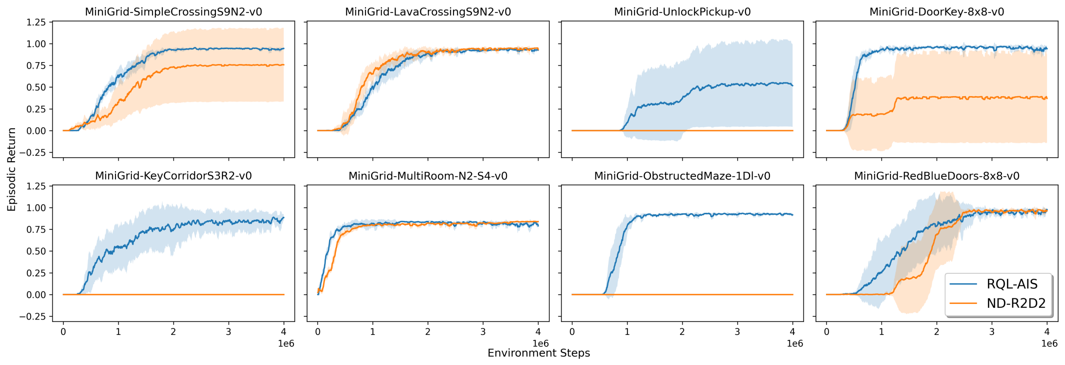


Figure 2: Results from 8 selected MiniGrid environments. RQL-AIS successfully solves all 8 while ND-R2D2 fails at solving 4 environments. This demonstrates the effectiveness of AIS state representations in empirical settings.

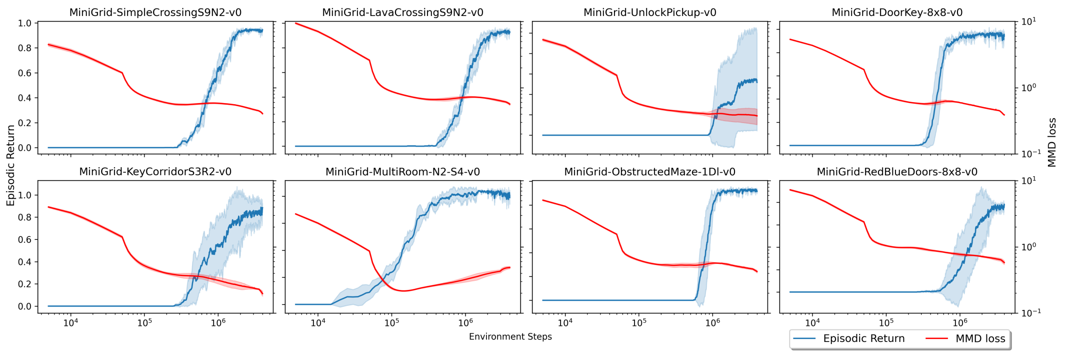


Figure 3: Episodic Return (left-axis) and MMD loss (right-axis, on log scale) plots for the RQL-AIS method on 8 selected MiniGrid environments. There is a close correlation between the drop in MMD loss value and improvement in episodic return. Both Environment Steps and MMD loss are in the logarithmic scale.

MiniGrid benchmark. A detailed comparison of the time evolution of AIS losses and performance strongly suggests that the improvement in performance is correlated to the decrease in AIS losses. In conclusion, the results of this paper show that AIS is a useful theoretical tool for analyzing the performance of RQL in POMDPs and also an effective block for algorithm design for POMDPs.

Our theoretical analysis is restricted to the tabular setting. Interesting future work includes generalizing the analysis to more general models, including using function approximation for approximating the Q-function. Another interesting avenue is formally showing convergence of an algorithm which learns the AIS and Q-function in parallel with two time-scale methods.

Acknowledgements

This work was supported in part by Natural Science and Engineering Council of Canada through Discovery Grant RGPIN-2021-03511 and Alliance International Catalyst Grant ALLRP 571054-21. The numerical experiments were enabled in part by compute resources provided by the Digital Research Alliance of Canada.

References

[Al-+18] Maruan Al-Shedivat et al. *On the Complexity of Exploration in Goal-Driven Navigation*. 2018. arXiv: 1811.06889 [cs.LG].

[Ast65] Karl J. Aström. "Optimal control of Markov processes with incomplete state information". In: *Journal of mathematical analysis and applications* 10.1 (1965), pp. 174–205.

[Bak02] Bram Bakker. "Reinforcement learning with long short-term memory". In: *Neural Information Processing Systems (NIPS)*. 2002.

[BT96] D.P. Bertsekas and J.N. Tsitsiklis. *Neuro-dynamic Programming*. Anthropological Field Studies. Athena Scientific, 1996. ISBN: 9781886529106.

[CBD22] Siddharth Chandak, Vivek S Borkar, and Parth Dodhia. "Reinforcement Learning in Non-Markovian Environments". In: *arXiv preprint arXiv:2211.01595* (2022).

[Cho+14] Kyunghyun Cho et al. "Learning phrase representations using RNN encoder-decoder for statistical machine translation". In: *Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2014, pp. 1724–1734.

[CUH15] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. *Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)*. 2015. DOI: 10.48550/ARXIV.1511.07289. URL: https://arxiv.org/abs/1511.07289.

[CWP18] Maxime Chevalier-Boisvert, Lucas Willems, and Suman Pal. *Minimalistic Gridworld Environment for Gymnasium*. https://github.com/Farama-Foundation/MiniGrid. 2018.

[DSH13] Mayank Daswani, Peter Sunehag, and Marcus Hutter. "Q-learning for history-based reinforcement learning". In: *Asian Conference on Machine Learning*. PMLR. 2013, pp. 213–228.

[DVZ22] Shi Dong, Benjamin Van Roy, and Zhengyuan Zhou. "Simple Agent, Complex Environment: Efficient Reinforcement Learning with Agent States". In: *Journal of Machine Learning Research* 23 (2022), pp. 1–54.

[Foe+16] Jakob Foerster et al. "Learning to Communicate to Solve Riddles with Deep Distributed Recurrent Q-Networks". In: (Feb. 2016).

[Fuk75] Kunihiko Fukushima. "Cognitron: A self-organizing multilayered neural network". In: *Biological Cybernetics* 20.3-4 (1975), pp. 121–136. DOI: 10.1007/bf00342633.

[Gel+19] Carles Gelada et al. "DeepMDP: Learning Continuous Latent Space Models for Representation Learning". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, Sept. 2019, pp. 2170–2179. URL: https://proceedings.mlr.press/v97/gelada19a.html.

[Goy+19] Anirudh Goyal et al. "Transfer and Exploration via the Information Bottleneck". In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=rJg8yhAqKm.

[Haa+18] Tuomas Haarnoja et al. "Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor". In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, July 2018, pp. 1861–1870. URL: https://proceedings.mlr.press/v80/haarnoja18b.html.

[HGS15] Hado van Hasselt, Arthur Guez, and David Silver. *Deep Reinforcement Learning with Double Q-learning*. 2015. DOI: 10.48550/ARXIV.1509.06461. URL: https://arxiv.org/abs/1509.06461.

[HS15] Matthew Hausknecht and Peter Stone. "Deep Recurrent Q-Learning for Partially Observable MDPs". In: *AAAI Fall Symposium*. 2015. URL: https://arxiv.org/abs/1507.06527.

[HS18] David Ha and Jürgen Schmidhuber. "Recurrent World Models Facilitate Policy Evolution". In: *Advances in Neural Information Processing Systems*. 2018. URL: https://proceedings.neurips.cc/paper/2018/file/2de5d16682c3c35007e4e92982f1a2ba-Paper.pdf.

[HS92] S. A. Harp and C. Sammut. "Recurrent reinforcement learning: A hybrid approach". In: *International Joint Conference on Neural Networks (IJCNN)*. Vol. 2. IEEE. 1992, pp. 523–528.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-term Memory". In: *Neural computation* 9 (Dec. 1997), pp. 1735–80. DOI: 10.1162/neco.1997.9.8.1735.

[JGR21] Minqi Jiang, Edward Grefenstette, and Tim Rocktäschel. *Prioritized Level Replay*. 2021. URL: https://openreview.net/forum?id=NfZ6g2OmXEk.

[JJS93] Tommi Jaakkola, Michael Jordan, and Satinder Singh. "On the convergence of stochastic iterative dynamic programming algorithms". In: *Neural computation* 6 (1993).

[JSJ94] Tommi Jaakkola, Satinder Singh, and Michael Jordan. "Reinforcement learning algorithm for partially observable Markov decision problems". In: *Advances in neural information processing systems* 7 (1994).

[Kap+19] Steven Kapturowski et al. "Recurrent Experience Replay in Distributed Reinforcement Learning". In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=r1lyTjAqYX.

[KB14] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2014. DOI: 10.48550/ARXIV.1412.6980. URL: https://arxiv.org/abs/1412.6980.

[Kra91] Mark A. Kramer. "Nonlinear principal component analysis using autoassociative neural networks". In: *Aiche Journal* 37 (1991), pp. 233–243.

[KT01] Vijay Konda and John Tsitsiklis. "Actor-Critic Algorithms". In: *Society for Industrial and Applied Mathematics* 42 (Apr. 2001).

[KY22a] Ali Devran Kara and Serdar Yüksel. “Convergence of Finite Memory Q-Learning for POMDPs and Near Optimality of Learned Policies Under Filter Stability”. In: *Mathematics of Operations Research* (2022).

[KY22b] Ali Devran Kara and Serdar Yüksel. “Near optimality of finite memory feedback policies in partially observed Markov decision processes”. In: *The Journal of Machine Learning Research* 23.1 (2022), pp. 437–482.

[Lit94] Michael L Littman. “Memoryless policies: Theoretical limitations and practical results”. In: *International conference on simulation of adaptive behavior*. Vol. 3. 1994, p. 238.

[LM92] Long-Ji Lin and Tom M Mitchell. *Memory approaches to reinforcement learning in non-Markovian domains*. Tech. rep. Carnegie Mellon University, USA, 1992.

[LS98] John Loch and Satinder P Singh. “Using Eligibility Traces to Find the Best Memoryless Policy in Partially Observable Markov Decision Processes.” In: *ICML*. 1998, pp. 323–331.

[Mav+22] Augustine Mavor-Parker et al. “How to Stay Curious while avoiding Noisy TVs using Aleatoric Uncertainty Estimation”. In: *International Conference on Machine Learning*. Vol. 162. PMLR, July 2022, pp. 15220–15240. URL: https://proceedings.mlr.press/v162/mavor-parker22a.html.

[McC96] Andrew Kachites McCallum. *Reinforcement learning with selective perception and hidden state*. University of Rochester, 1996.

[MH18] Sultan Javed Majeed and Marcus Hutter. “On Q-learning Convergence for Non-Markov Decision Processes”. In: *International Joint Conference on Artificial Intelligence*. Stockholm, Sweden, July 2018, pp. 2546–2552. ISBN: 978-0-9992411-2-7. DOI: 10.24963/ijcai.2018/353.

[Mni+13] Volodymyr Mnih et al. *Playing Atari with Deep Reinforcement Learning*. 2013. DOI: 10.48550/ARXIV.1312.5602. URL: https://arxiv.org/abs/1312.5602.

[Mni+15] Volodymyr Mnih et al. “Human-level control through deep reinforcement learning”. In: *Nature* 518.7540 (Feb. 2015), pp. 529–533. DOI: 10.1038/nature14236.

[Mou+17] Sajad Mousavi et al. *Learning to predict where to look in interactive environments using deep recurrent Q-learning*. 2017. arXiv: 1612.05753 [cs.CV].

[Mül97] Alfred Müller. “Integral probability metrics and their generating classes of functions”. In: *Advances in Applied Probability* 29.2 (1997), pp. 429–443.

[Ou+21] Jiajun Ou et al. “Autonomous quadrotor obstacle avoidance based on dueling double deep recurrent Q-learning with monocular vision”. In: *Neurocomputing* 441 (2021), pp. 300–310. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2021.02.017.

[Par+21] Simone Parisi et al. “Interesting Object, Curious Agent: Learning Task-Agnostic Exploration”. In: *Advances in Neural Information Processing Systems*. Ed. by A. Beygelzimer et al. 2021. URL: https://openreview.net/forum?id=knKJgksd7kA.

[PP02] Theodore Perkins and Mark Pendrith. *On the Existence of Fixed Points for Q-Learning and Sarsa in Partially Observable Domains.* Jan. 2002, pp. 490–497.

[SB18] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT Press, 2018.

[Sch+15a] Tom Schaul et al. *Prioritized Experience Replay*. 2015. DOI: 10.48550/ARXIV.1511.05952. URL: https://arxiv.org/abs/1511.05952.

[Sch+15b] John Schulman et al. “Trust Region Policy Optimization”. In: *International Conference on Machine Learning*. Vol. 37. Lille, France: PMLR, July 2015, pp. 1889–1897. URL: https://proceedings.mlr.press/v37/schulman15.html.

[Sch+17] John Schulman et al. *Proximal Policy Optimization Algorithms*. 2017. URL: https://arxiv.org/abs/1707.06347.

[Sch91] Jürgen Schmidhuber. “Reinforcement learning through recurrent connectionist models”. In: *International Conference on Neural Networks*. IEEE. 1991, pp. 511–518.

[Sil+16] David Silver et al. “Mastering the game of Go with deep neural networks and tree search”. In: *Nature* 529.7587 (Jan. 2016), pp. 484–489. ISSN: 1476-4687. DOI: 10.1038/nature16961.

[SJJ94] Satinder P. Singh, Tommi Jaakkola, and Michael I. Jordan. "Learning Without State-Estimation in Partially Observable Markovian Decision Processes". In: *Machine Learning Proceedings 1994*. Elsevier, 1994, pp. 284–292. DOI: 10.1016/B978-1-55860-335-6.50042-8.

[Sor+15] Ivan Sorokin et al. *Deep Attention Recurrent Q-Network*. 2015. arXiv: 1512.01693 [cs.LG].

[SPK13] Guy Shani, Joelle Pineau, and Robert Kaplow. "A survey of point-based POMDP solvers". In: *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)* 27 (2013), pp. 1–51.

[Sub+22] Jayakumar Subramanian et al. "Approximate Information State for Approximate Planning and Reinforcement Learning in Partially Observed Systems". In: *Journal of Machine Learning Research* 23.12 (2022), pp. 1–83. URL: http://jmlr.org/papers/v23/20-1165.html.

[Tas+18] Yuval Tassa et al. *DeepMind Control Suite*. 2018. DOI: 10.48550/ARXIV.1801.00690. URL: https://arxiv.org/abs/1801.00690.

[TV97] J.N. Tsitsiklis and B. Van Roy. "An analysis of temporal-difference learning with function approximation". In: *IEEE Transactions on Automatic Control* 42.5 (1997), pp. 674–690. DOI: 10.1109/9.580874.

[Wie+07] Daan Wierstra et al. "Solving deep memory POMDPs with recurrent policy gradients". In: *International Conference on Artificial Neural Networks (ICANN)*. 2007.

[Wie+10] Daan Wierstra et al. "Recurrent policy gradients". In: *Logic Journal of the IGPL* 18.5 (2010), pp. 620–634.

[WL95] Steven D Whitehead and Long-Ji Lin. "Reinforcement learning of non-Markov decision processes". In: *Artificial Intelligence* 73.1-2 (1995), pp. 271–306.

[Zha+21] Tianjun Zhang et al. *BeBold: Exploration Beyond the Boundary of Explored Regions*. 2021. URL: https://openreview.net/forum?id=_ptUyYP19mP.

A Proof of claims in Theorem 1

The high-level idea of the proof is the same as that [Sub+22] [Theorem 27] and relies on the following two observations:

1. **Approximating the infinite horizon system by a finite horizon system.** First note that the finiteness of the state and action spaces implies that the per-step reward is uniformly bounded. Define $r_{\text{MIN}} = \min_{s \in \mathcal{S}, a \in \mathcal{A}} r(s,a)$ and $r_{\text{MAX}} = \max_{s \in \mathcal{S}, a \in \mathcal{A}} r(s,a)$. Then, $R_t \in [r_{\text{MIN}}, r_{\text{MAX}}]$.

   Now consider a finite horizon system that runs for horizon T. Let $V^\star_{t,T}$ and $V^{\tilde{\pi}}_{t,T}$ denote the optimal finite-horizon value function and the finite-horizon value function corresponding to policy $\tilde{\pi}$ (defined in Theorem 1), respectively. Moreover, let $Q^\star_{t,T}$ denote the optimal finite-horizon Q-function. More specifically, for any $t \le T$, and any history $h_t \in \mathcal{H}_t$ and action $a_t \in \mathcal{A}$, we have

$$V^\star_{t,T}(h_t) \coloneqq \sup_\pi \mathbb{E}^\pi \left[\sum_{\tau=t}^{T-1} \gamma^{\tau-t} r(S_\tau, A_\tau) \,\middle|\, H_t = h_t \right],$$

$$V^{\tilde{\pi}}_{t,T}(h_t) \coloneqq \mathbb{E}^{\tilde{\pi}} \left[\sum_{\tau=t}^{T-1} \gamma^{\tau-t} r(S_\tau, A_\tau) \,\middle|\, H_t = h_t \right],$$

$$Q^\star_{t,T}(h_t, a_t) \coloneqq \sup_\pi \mathbb{E}^\pi \left[\sum_{\tau=t}^{T-1} \gamma^{\tau-t} r(S_\tau, A_\tau) \,\middle|\, H_t = h_t, A_t = a_t \right].$$

   Then, the boundedness of the per-step reward implies the following:

   Proposition 1. *For any T, any $t \le T$, and any history $h_t \in \mathcal{H}_t$ and action $a_t \in \mathcal{A}$, we have the following:*

$$Q^\star_{t,T}(h_t, a_t) + \frac{\gamma^{T-t}}{1-\gamma} r_{\text{MIN}} \le Q^\star_t(h_t, a_t) \le Q^\star_{t,T}(h_t, a_t) + \frac{\gamma^{T-t}}{1-\gamma} r_{\text{MAX}}, \tag{20}$$

$$V^\star_{t,T}(h_t) + \frac{\gamma^{T-t}}{1-\gamma} r_{\text{MIN}} \le \quad V^\star_t(h_t) \le V^\star_{t,T}(h_t) + \frac{\gamma^{T-t}}{1-\gamma} r_{\text{MAX}}, \tag{21}$$

$$V^{\tilde{\pi}}_{t,T}(h_t) + \frac{\gamma^{T-t}}{1-\gamma} r_{\text{MIN}} \le \quad V^{\tilde{\pi}}_t(h_t) \le V^{\tilde{\pi}}_{t,T}(h_t) + \frac{\gamma^{T-t}}{1-\gamma} r_{\text{MAX}}. \tag{22}$$

2. **Approximating the finite-horizon system with the approximate AIS model.** Given the AIS reward $\tilde{r}$ and dynamics $\tilde{P}$, define a Bellman operator $\tilde{\mathcal{B}}_V \colon \mathcal{Z} \to \mathcal{Z}$ as follows. For any $\tilde{V} \colon \mathcal{Z} \to \mathbb{R}$,

$$\tilde{\mathcal{B}}_V \tilde{V}(z) = \max_{a \in \mathcal{A}} \left\{ \tilde{r}(z,a) + \gamma \int_{\mathcal{Z}} \tilde{V}(z') \tilde{P}(dz' \mid z, a) \right\},$$

   and define a Bellman operator $\tilde{\mathcal{B}}_Q \colon \mathcal{Z} \times \mathcal{A} \to \mathcal{Z} \times \mathcal{A}$ as follows. For any $\tilde{Q} \colon \mathcal{Z} \times \mathcal{A} \to \mathbb{R}$,

$$\begin{aligned} \tilde{V}(z) &= \max_{a \in \mathcal{A}} \tilde{Q}(z,a) \\ \tilde{\mathcal{B}}_Q \tilde{Q}(z,a) &= \tilde{r}(z,a) + \gamma \int_{\mathcal{Z}} \max_{a' \in \mathcal{A}} \tilde{Q}(z', a') \tilde{P}(dz' \mid z, a) \\ &= \tilde{r}(z,a) + \gamma \int_{\mathcal{Z}} \tilde{V}(z') \tilde{P}(dz' \mid z, a). \end{aligned}$$

   Thus the two Bellman operators $\tilde{\mathcal{B}}_V$ and $\tilde{\mathcal{B}}_Q$ are related as follows

$$\tilde{\mathcal{B}}_V \tilde{V}(z) = \max_{a \in \mathcal{A}} \tilde{\mathcal{B}}_Q \tilde{Q}(z,a).$$

   Now, define initial value functions $\tilde{V}^0(z) = 0$ and $\tilde{Q}^0(z,a) = 0$ for all z and a. Recursively define $\tilde{V}^{n+1} = \tilde{\mathcal{B}}_V \tilde{V}^n$ and $\tilde{Q}^{n+1} = \tilde{\mathcal{B}}_Q \tilde{Q}^n$. Then, we have the following.

   Proposition 2. *Arbitrarily fix the function space $\mathfrak{F}$. For any T and $t \le T$, define*

$$\eta_{t,T} = \sum_{\tau=t}^{T-1} \gamma^{\tau-t} \varepsilon_\tau + \sum_{\tau=t+1}^{T-1} \gamma^{\tau-t} \rho_{\mathfrak{F}}(\tilde{V}^{T-\tau}) \delta_{\tau-1}.$$

Then, we have the following bounds:

$$|Q^\star_{t,T}(h_t,a_t) - \tilde{Q}^{T-t}(\sigma_t(h_t),a_t)| \le \eta_{t,T}, \tag{23}$$
$$|V^\star_{t,T}(h_t) - \tilde{V}^{T-t}(\sigma_t(h_t))| \le \eta_{t,T}, \tag{24}$$
$$|V^{\tilde{\pi}}_{t,T}(h_t) - \tilde{V}^{T-t}(\sigma_t(h_t))| \le \eta_{t,T}. \tag{25}$$

A.1 Proof of Proposition 1

Consider the optimal value functions in the original model for the infinite horizon case, i.e, $V^\star_t(h_t)$ and $Q^\star_t(h_t,a_t)$.

$$\begin{aligned}
V^\star_t(h_t) &= \sup_\pi \mathbb{E}\left[\sum_{\tau=t}^{\infty} \gamma^{\tau-t} R_\tau \,\middle|\, H_t = h_t\right] \\
&\ge \sup_\pi \mathbb{E}\left[\sum_{\tau=t}^{T-1} \gamma^{\tau-t} R_\tau + \sum_{\tau=T}^{\infty} \gamma^{\tau-t} r_{\text{MIN}} \,\middle|\, H_t = h_t\right] \\
&= \sup_\pi \mathbb{E}\left[\sum_{\tau=t}^{T-1} R_\tau \,\middle|\, H_t = h_t\right] + \frac{\gamma^{T-t}}{1-\gamma} r_{\text{MIN}} \\
&= V^\star_{t,T}(h_t) + \frac{\gamma^{T-t}}{1-\gamma} r_{\text{MIN}}.
\end{aligned}$$

Similarly, it can be shown by reversing the inequality and using r_{MAX} instead of r_{MIN} that

$$V^\star_t(h_t) \le V^\star_{t,T}(h_t) + \frac{\gamma^{T-t}}{1-\gamma} r_{\text{MAX}}.$$

Thus, we have the result

$$V^\star_{t,T}(h_t) + \frac{\gamma^{T-t}}{1-\gamma} r_{\text{MIN}} \le V^\star_t(h_t) \le V^\star_{t,T}(h_t) + \frac{\gamma^{T-t}}{1-\gamma} r_{\text{MAX}}.$$

We can obtain a similar bound for $Q^\star_t$. In particular, we have

$$\begin{aligned}
Q^\star_t(h_t,a_t) &= \sup_\pi \mathbb{E}\left[r(h_t,a_t) + \sum_{\tau=t+1}^{\infty} \gamma^{\tau-t} R_\tau \,\middle|\, H_t = h_t, A_t = a_t\right] \\
&\ge \sup_\pi \mathbb{E}\left[r(h_t,a_t) + \sum_{\tau=t+1}^{T-1} \gamma^{\tau-t} R_\tau + \sum_{\tau=T}^{\infty} \gamma^{\tau-t} r_{\text{MIN}} \,\middle|\, H_t = h_t, A_t = a_t\right] \\
&= \sup_\pi \mathbb{E}\left[r(h_t,a_t) + \sum_{\tau=t+1}^{T-1} R_\tau \,\middle|\, H_t = h_t, A_t = a_t\right] + \frac{\gamma^{T-t}}{1-\gamma} r_{\text{MIN}} \\
&= Q^\star_{t,T}(h_t,a_t) + \frac{\gamma^{T-t}}{1-\gamma} r_{\text{MIN}}.
\end{aligned}$$

Similarly, it can be shown by reversing the inequality and using r_{MAX} instead of r_{MIN} that

$$Q^\star_t(h_t,a_t) \le Q^\star_{t,T}(h_t) + \frac{\gamma^{T-t}}{1-\gamma} r_{\text{MAX}}.$$

Thus, we have

$$Q^\star_{t,T}(h_t,a_t) + \frac{\gamma^{T-t}}{1-\gamma} r_{\text{MIN}} \le Q^\star_t(h_t,a_t) \le Q^\star_{t,T}(h_t,a_t) + \frac{\gamma^{T-t}}{1-\gamma} r_{\text{MAX}}.$$

Next, consider the value function in the original model for the infinite horizon case $V_t^{\tilde{\pi}}(h_t)$.

$$\begin{aligned}
V_t^{\tilde{\pi}}(h_t) &= \mathbb{E}^{\tilde{\pi}}\left[\sum_{\tau=t}^{\infty}\gamma^{\tau-t}R_\tau \mid H_t = h_t\right] \\
&\geq \mathbb{E}^{\tilde{\pi}}\left[\sum_{\tau=t}^{T-1}\gamma^{\tau-t}R_\tau + \sum_{\tau=T}^{\infty}\gamma^{\tau-t}r_{\text{MIN}} \mid H_t = h_t\right] \\
&= \mathbb{E}^{\tilde{\pi}}\left[\sum_{\tau=t}^{T-1}R_\tau \mid H_t = h_t\right] + \frac{\gamma^{T-t}}{1-\gamma}r_{\text{MIN}} \\
&= V_{t,T}^{\tilde{\pi}}(h_t) + \frac{\gamma^{T-t}}{1-\gamma}r_{\text{MIN}}.
\end{aligned}$$

Similarly, it can be shown by reversing the inequality and using r_{MAX} instead of r_{MIN} that

$$V_t^{\tilde{\pi}}(h_t) \leq V_{t,T}^{\tilde{\pi}}(h_t) + \frac{\gamma^{T-t}}{1-\gamma}r_{\text{MAX}}.$$

Thus, we have the result

$$V_{t,T}^{\tilde{\pi}}(h_t) + \frac{\gamma^{T-t}}{1-\gamma}r_{\text{MIN}} \leq V_t^{\tilde{\pi}}(h_t) \leq V_{t,T}^{\tilde{\pi}}(h_t) + \frac{\gamma^{T-t}}{1-\gamma}r_{\text{MAX}}.$$

A.2 Proof of Proposition 2

First, we look at the proof for (23) and (24). A dynamic program can be written for the optimal value functions for $t \in \{1, \cdots, T-1\}$ as $\{V_{t,T}^{\star} \colon \mathcal{H}_t \to \mathbb{R}\}$ and $\{Q_{t,T}^{\star} \colon \mathcal{H}_t \times \mathcal{A}_t \to \mathbb{R}\}$ with $V_{T,T}^{\star}(h_T) = 0$ and $Q_{T,T}^{\star}(h_T, a_T) = 0$ such that:

$$\begin{aligned}
V_{t,T}^{\star}(h_t) &\coloneqq \max_{a_t\in\mathcal{A}} \mathbb{E}\left[R_t + \gamma V_{t+1,T}^{\star}(H_{t+1}) \mid H_t = h_t, A_t = a_t\right], \\
Q_{t,T}^{\star}(h_t, a_t) &\coloneqq \mathbb{E}\left[R_t + \gamma V_{t+1,T}^{\star}(H_{t+1}) \mid H_t = h_t, A_t = a_t\right].
\end{aligned}$$

We prove the result by backward induction. Consider the bounds $|Q_{T,T}^{\star}(h_T, a_T) - \tilde{Q}^0(\sigma_T(h_T), a_T)| \leq \eta_{T,T} = 0$ and $|V_{T,T}^{\star}(h_T, a_T) - \tilde{V}^0(\sigma_T(h_T), a_T)| \leq \eta_{T,T} = 0$ which hold by definition. Thus, the bound holds for $t = T$. This forms the basis of induction. Now, assume that the bound holds for $t+1$ and consider the system at time t. We have

$$\begin{aligned}
&|Q_{t,T}^{\star}(h_t, a_t) - \tilde{Q}^{T-t}(\sigma_t(h_t), a_t)| \\
&\quad\overset{(a)}{\leq} |\mathbb{E}[R_t \mid H_t = h_t, A_t = a] - \tilde{r}(\sigma_t(h_t), a)| \\
&\qquad + \gamma\mathbb{E}[|V_{t+1,T}^{\star}(H_{t+1}) - \tilde{V}^{T-t-1}(\sigma_{t+1}(H_{t+1}))| \mid H_t = h_t, A_t = a] \\
&\qquad + \gamma\left|\mathbb{E}[\tilde{V}^{T-t-1}(\sigma_{t+1}(H_{t+1})) \mid H_t = h_t, A_t = a] - \int_{\mathcal{Z}}\tilde{V}^{T-t-1}(z_{t+1})\tilde{P}(dz_{t+1} \mid \sigma_t(h_t), a)\right| \\
&\quad\overset{(b)}{\leq} \varepsilon_t + \gamma\eta_{t+1,T} + \gamma\rho_{\mathfrak{F}}(\tilde{V}^{T-t-1})\delta_t = \eta_{t,T}
\end{aligned}$$

where (a) follows from the triangle inequality, the definition of $Q_{t,T}^{\star}(h_t, a_t)$, the definition of $\tilde{Q}^{T-t}(\sigma_t(h_t), a_t)$; (b) follows from the induction hypothesis and the definitions of ε_t and δ_t. We also have,

$$\begin{aligned}
|V_{t,T}^{\star}(h_t) - \tilde{V}^{T-t}(\sigma_t(h_t))| &\overset{(c)}{\leq} \max_{a_t\in\mathcal{A}}|Q_{t,T}^{\star}(h_t, a_t) - \tilde{Q}^{T-t}(\sigma_t(h_t), a_t)| \\
&\leq \eta_{t,T} = \sum_{\tau=t}^{T-1}\gamma^{\tau-t}\varepsilon_\tau + \sum_{\tau=t+1}^{T-1}\gamma^{\tau-t}\rho_{\mathfrak{F}}(\tilde{V}^{T-\tau})\delta_{\tau-1}
\end{aligned}$$

where (c) follows from the inequalities $\max f(x) \le \max |f(x) - g(x)| + \max g(x)$. Thus, the result holds for t. Then, by the principle of induction, it holds for all $t \le T$. This completes the proof for (23) and (24).

Next, we look at the proof for (25). Consider value functions for $t \in \{1, \cdots, T-1\}$ as $\{V^{\tilde{\pi}}_{t,T} \colon \mathcal{H}_t \to \mathbb{R}\}$ with $V^{\tilde{\pi}}_{T,T}(h_T) = 0$ that are recursively defined such that:

$$V^{\tilde{\pi}}_{t,T}(h_t) \coloneqq \mathbb{E}^{\tilde{\pi}}\left[R_t + \gamma V^{\tilde{\pi}}_{t+1,T}(H_{t+1}) \mid H_t = h_t\right].$$

We prove this result by backward induction as well. Consider the bound $|V^{\tilde{\pi}}_{T,T}(h_T) - \tilde{V}^0(\sigma_T(h_T))| \le \eta_{T,T} = 0$ which holds by definition. Thus, the bound holds for $t = T$. This forms the basis of induction. Now assume that the bound holds for $t+1$ and consider the system at time t. We have

$$\begin{aligned}
&|V^{\tilde{\pi}}_{t,T}(h_t) - \tilde{V}^{T-t}(\sigma_t(h_t))| \\
&\quad\overset{(a)}{\le} \max_{a\in\mathcal{A}} |\mathbb{E}[R_t \mid H_t = h_t, A_t = a] - \tilde{r}(\sigma_t(h_t), a)| \\
&\qquad + \gamma \max_{a\in\mathcal{A}} \mathbb{E}[|V^{\tilde{\pi}}_{t+1,T}(H_{t+1}) - \tilde{V}^{T-t-1}(\sigma_{t+1}(H_{t+1}))| \mid H_t = h_t, A_t = a] \\
&\qquad + \gamma \max_{a\in\mathcal{A}} \left| \mathbb{E}[\tilde{V}^{T-t-1}(\sigma_{t+1}(H_{t+1})) \mid H_t = h_t, A_t = a] - \int_{\mathcal{Z}} \tilde{V}^{T-t-1}(z_{t+1}) \tilde{P}(dz_{t+1} \mid \sigma_t(h_t), a) \right| \\
&\quad\overset{(b)}{\le} \varepsilon_t + \gamma \eta_{t+1,T} + \gamma \rho_{\mathfrak{F}}(\tilde{V}^{T-t-1}) \delta_t = \eta_{t,T}
\end{aligned}$$

where (a) follows from the triangle inequality, the definition of $V^{\tilde{\pi}}_{t,T}(h_t)$, the definition of $\tilde{V}^{T-t}(\sigma_t(h_t))$, the fact that $\sum_{a\in\mathcal{A}} \tilde{\pi}(a \mid z) = 1$, and upper-bounding over all actions; (b) follows from the induction hypothesis and the definitions of ε_t and δ_t.

Thus, this gives us

$$|V^{\tilde{\pi}}_{t,T}(h_t) - \tilde{V}^{T-t}(\sigma_t(h_t))| \le \eta_{t,T} = \sum_{\tau=t}^{T-1} \gamma^{\tau-t} \varepsilon_\tau + \sum_{\tau=t+1}^{T-1} \gamma^{\tau-t} \rho_{\mathfrak{F}}(\tilde{V}^{T-\tau}) \delta_{\tau-1}.$$

Thus, the result holds for t. Then, by the principle of induction, it holds for all $t \le T$. This completes the proof for (25).

A.3 Bounds on value approximation

Combining Proposition 1: (20), (21) with Proposition 2: (23), (24) gives us

$$\begin{aligned}
\tilde{V}^{T-t}(\sigma_t(h_t)) - \eta_{t,T} + \frac{\gamma^{T-t}}{1-\gamma} r_{\text{MIN}} &\le V^\star_t(h_t) \le \tilde{V}^{T-t}(\sigma_t(h_t)) + \eta_{t,T} + \frac{\gamma^{T-t}}{1-\gamma} r_{\text{MAX}}, \\
\tilde{Q}^{T-t}(\sigma_t(h_t), a_t) - \eta_{t,T} + \frac{\gamma^{T-t}}{1-\gamma} r_{\text{MIN}} &\le Q^\star_t(h_t, a_t) \le \tilde{Q}^{T-t}(\sigma_t(h_t), a_t) + \eta_{t,T} + \frac{\gamma^{T-t}}{1-\gamma} r_{\text{MAX}}.
\end{aligned}$$

Note that due to discounting both $\tilde{\mathcal{B}}_V$ and $\tilde{\mathcal{B}}_Q$ are contractions. Therefore, from the Banach fixed point theorem, we know that $\lim_{T\to\infty} \tilde{V}^{T-t} = \tilde{V}^\star$ and $\lim_{T\to\infty} \tilde{Q}^{T-t} = \tilde{Q}^\star$. Furhermore, by the continuity of $\rho_{\mathfrak{F}}(\cdot)$, we have $\lim_{T\to\infty} \rho_{\mathfrak{F}}(\tilde{V}^{T-t}) = \rho_{\mathfrak{F}}(\tilde{V}^\star)$. Thus, taking the limit $T \to \infty$ in the above equations gives us

$$\begin{aligned}
\tilde{V}^\star(\sigma_t(h_t)) - \eta^\star_t \le V^\star_t(h_t) \le \tilde{V}^\star(\sigma_t(h_t)) + \eta^\star_t, \\
\tilde{Q}^\star(\sigma_t(h_t), a_t) - \eta^\star_t \le Q^\star_t(h_t, a_t) \le \tilde{Q}^\star(\sigma_t(h_t), a_t) + \eta^\star_t,
\end{aligned}$$

where, $\eta^\star_t = \sum_{\tau=t}^\infty \gamma^{\tau-t} \varepsilon_\tau + \gamma \rho_{\mathfrak{F}}(\tilde{V}^\star) \sum_{\tau=t}^\infty \gamma^{\tau-t} \delta_\tau$.

This shows that

$$|V^\star_t(h_t) - \tilde{V}^\star(\sigma_t(h_t))| \le \eta^\star_t, \tag{26}$$

$$|Q^\star_t(h_t, a_t) - \tilde{Q}^\star(\sigma_t(h_t), a_t)| \le \eta^\star_t. \tag{27}$$

This establishes the bounds in the main theorem on value approximation.

A.4 Bounds on policy approximation

Combining Proposition 1: (22) with Proposition 2: (25) gives us

$$\tilde{V}^{T-t}(\sigma_t(h_t)) - \eta_{t,T} + \frac{\gamma^{T-t}}{1-\gamma} r_{\text{MIN}} \le V_t^{\tilde{\pi}}(h_t) \le \tilde{V}^{T-t}(\sigma_t(h_t)) + \eta_{t,T} + \frac{\gamma^{T-t}}{1-\gamma} r_{\text{MAX}},$$

Note that due to discounting, $\tilde{\mathcal{B}}_V$ is a contraction. Therefore, from the Banach fixed point theorem, we know that $\lim_{T\to\infty} \tilde{V}^{T-t} = \tilde{V}^{\tilde{\pi}^\star}$. Furthermore, by continuity of $\rho_{\mathfrak{F}}(\cdot)$, we have $\lim_{T\to\infty} \rho_{\mathfrak{F}}(\tilde{V}^{T-t}) = \rho_{\mathfrak{F}}(\tilde{V}^{\tilde{\pi}^\star})$. Thus, taking the limit $T \to \infty$ in the above equation gives us

$$\tilde{V}^{\tilde{\pi}^\star}(\sigma_t(h_t)) - \eta_t^{\tilde{\pi}^\star} \le V_t^{\tilde{\pi}}(h_t) \le \tilde{V}^{\tilde{\pi}^\star}(\sigma_t(h_t)) + \eta_t^{\tilde{\pi}^\star},$$

where, $\eta_t^{\tilde{\pi}^\star} = \sum_{\tau=t}^{\infty} \gamma^{\tau-t}\varepsilon_\tau + \gamma\rho_{\mathfrak{F}}(\tilde{V}^{\tilde{\pi}^\star}) \sum_{\tau=t}^{\infty} \gamma^{\tau-t}\delta_\tau$.

This shows that

$$|V_t^{\tilde{\pi}}(h_t) - \tilde{V}^{\tilde{\pi}^\star}(\sigma_t(h_t))| \le \eta_t^{\tilde{\pi}^\star}. \tag{28}$$

Note that $\tilde{V}^{\tilde{\pi}^\star}(\sigma_t(h_t)) = \tilde{V}^\star(\sigma_t(h_t))$. Finally, we have the bound on policy approximation:

$$\begin{aligned} |V_t^\star(h_t) - V_t^{\tilde{\pi}}(h_t)| &\overset{(a)}{\le} |V_t^\star(h_t) - \tilde{V}^\star(\sigma_t(h_t))| + |V_t^{\tilde{\pi}}(h_t) - \tilde{V}^{\tilde{\pi}^\star}(\sigma_t(h_t))| \\ &\overset{(b)}{\le} 2\sum_{\tau=t}^{\infty} \gamma^{\tau-t}\varepsilon_\tau + 2\gamma\rho_{\mathfrak{F}}(\tilde{V}^\star)\sum_{\tau=t}^{\infty}\gamma^{\tau-t}\delta_\tau, \end{aligned}$$

where (a) follows from the triangle inequality and (b) follows from (26) and (28). This establishes the bound in the main theorem on policy approximation.

B Instance independent bounds

In this section, we illustrate *instance independent* bounds by upper bounding $\rho_{\mathfrak{F}}(V_\xi^\star)$ in terms of properties of the transition and reward functions. These *instance independent* bounds have been established in [Sub+22].

When the IPM considered is the total variation distance, i.e., $\mathfrak{F} = \mathfrak{F}_{\text{TV}} \coloneqq \{f : \text{span}(f) \le 1\}$ (where $\text{span}(f)$ is the span semi-norm of a function), we have

$$\rho_{\mathfrak{F}}(V_\xi^\star) = \text{span}(V_\xi^\star) \le \frac{\text{span}(r)}{(1-\gamma)}.$$

Next, when the IPM considered is the Wasserstein distance, i.e., $\mathfrak{F} = \mathfrak{F}_{\text{Was}} \coloneqq \{f : \text{Lip}(f) \le 1\}$ (where $\mathcal{X}$ is a metric space and $\text{Lip}(f)$ is the Lipschitz constant of the function f, computed with respect to the metric on $\mathcal{X}$); the Lipschitz constants L_r of the reward function and L_P of the state transition matrix are finite; and $\gamma L_P < 1$; then we have

$$\rho_{\mathfrak{F}}(V_\xi^\star) = \text{Lip}(V_\xi^\star) \le \frac{L_r}{(1-\gamma L_P)}.$$

C Proof of claims in Theorem 2

C.1 $W_t^2(z,a)$ converges to 0 almost surely.

The definition of the learning rate $\{\alpha_t\}_{t\ge1}$ implies that

$$W_{t+1}^2(z,a) = \lim_{t\to\infty} \frac{\frac{1}{t}\sum_{k=0}^{t-1} F_k^2(z,a)\mathbb{1}_{\{Z_k=z,A_k=a\}}}{\frac{1}{t}\sum_{k=0}^{t-1}\mathbb{1}_{\{Z_k=z,A_k=a\}}} \tag{29}$$

Recall that according to Strong Law of Large Numbers for for Markov chains, given a aperiodic and irreducible Markov chain $\{X_t\}_{t\geq 1}$, $X_t \in \mathcal{X}$, with stationary distribution ρ and any (measurable) function h, the long run average is equal to the steady state value, i.e.,

$$\lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} h(X_t) = \sum_{x\in\mathcal{X}} h(x)\rho(x), a.s.. \tag{30}$$

Let $\xi(s,y,z,z_+,a) = \xi(s,y,z,a)P_\xi(z_+ \mid z,a)$. Then, the numerator of (29) may be simplified as

$$\sum_{(s,y,z,z_+,a)} \left[r(s,a) - r_\xi(z,a) + \gamma V^\star_\xi(z_+) - \gamma \sum_{z'} P_\xi(z'|z,a)V^\star_\xi(z') \right] \xi(s,y,z,z_+,a). \tag{31}$$

By definition of $r_\xi(z,a)$, we have

$$\sum_{(s,z,a)} [r(s,a) - r_\xi(z,a)]\xi(s,z,a) = \sum_{z,a}\Big[\sum_s r(s,a)\xi(s|z,a) - r_\xi(z,a)\Big] = 0$$

Similarly, by definition of $\xi(s,y,z,z_+,a)$, we have

$$\sum_{s,y,z,z_+,a} \big[V^\star_\xi(z_+) - \sum_{z'} P_\xi(z'|z,a)V^\star_\xi(z')\big]\xi(s,y,z,a)P_\xi(z_+ \mid z,a) = 0$$

Substituting the above two equations in (31), we get that the numerator of (29) is zero. Hence, $W_t^2(z,a) \to 0$, a.s., for all (z,a).

C.2 $W_t^1(z,a)$ cannot remain above $C\epsilon$ forever

We first state [JJS93, Lemma 3].[5]

Lemma 2. Let $\mathcal{X}$ be a finite set. Consider a vector valued stochastic process $\{X_t\}_{t\geq 1}$, where

$$X_{t+1}(x) = (1-\alpha(x))X_t(x) + \beta_t(x)\|X_t\|.$$

where $\sum_{t\geq 1}\alpha_t(x) = \infty$, $\sum_{t\geq 1}\alpha_t^2(x) < \infty$, $\sum_{t\geq 1}\beta_t(x) = \infty$, $\sum_{t\geq 1}\beta_t^2(x) < \infty$, almost surely, and $\mathbb{E}[\beta_t(x)] \leq \mathbb{E}[\alpha_t(x)]$. Then, $X_t(x)$ converges to zero almost surely for all $x \in \mathcal{X}$.

Assume that for any $t > T(\omega,\epsilon)$, we have $\|W_t^1\|_\infty > C\epsilon$. Recall that we have shown in (15) that if $\|W_t^1\|_\infty > C\epsilon$ then $F_t^1(z,a) \leq \|W_t^1\|_\infty$. Define $\beta_t(z,a) = \alpha_t(z,a)F_t^1(z,a)/\|W_t^1\|_\infty$. Then, the recursion for W_t^1 can be written as

$$W_{t+1}^1(z,a) = (1-\alpha_t(z,a))W_t^1(z,a) + \beta_t(z,a)\|W_t^1\|_\infty$$

which is an instance of the linear iteration described in Lemma 2. Therefore, by Lemma 2, $W_t^1(z,a)$ converges to zero almost surely. But this contradicts the assumption that $\|W_t\|_\infty > C\epsilon$. Hence, the assumption cannot be true.

C.3 If $\|W_t^1\|_\infty$ hits below $C\epsilon$ then it stays there

Recall that we have shown in (14) that $F_t^1(z,a) \leq \gamma\|\Delta_t\|_\infty$. Now suppose that for some $t > T(\epsilon,\omega)$, $\|W_t^1\|_\infty < C\epsilon$. Then,

$$F_t^1(z,a) \leq \gamma(\|W_t^1\|_\infty + \|W_t^2\|_\infty) \leq \gamma(C\epsilon + \epsilon) \leq \gamma(1+C)\epsilon \leq C\epsilon$$

where we have used the fact that $\gamma(1+\frac{1}{C}) < 1$ in the last inequality. Thus,

$$|W_{t+1}^1(z,a) \leq (1-\alpha_t(z,a))|W_t^1(z,a)| + \alpha_t(z,a)|F_t^1(z,a)| \leq (1-\alpha_t(z,a))C\epsilon + \alpha_t(z,a)C\epsilon = C\epsilon.$$

Hence $\|W_{t+1}\|_\infty \leq C\epsilon$.

[5] In [JJS93], the result is stated with using the iteration $X_{t+1}(x) = (1-\alpha(x))X_t(x) + \gamma\beta_t(x)\|X_t\|$. where $\gamma \in (0,1)$. But γ plays no role in the proof argument of [JJS93, Lemma 3] and may be omitted.

D Recurrent Q-learning with AIS

The training phase of RQL-AIS consists of sampling batches of sequential data from the R2D2 replay buffer and training: 1) the AIS generator components using the AIS loss, 2) the Q-function using the multi-step Q-learning loss. The full RQL-AIS algorithm is presented in Algorithm 1. In this section, we assume the AIS history generator $\sigma(h, z_0)$ is an RNN function which receives the initial hidden state z_0 and input history h and outputs z which is the hidden state corresponding to the sample at the end of the history sequence h.

Algorithm 1 Recurrent Q-learning with AIS (RQL-AIS)

```
 1: init $\sigma$, $\hat{P}^y$, $\hat{r}$, $Q_\theta$ to random networks, init $\theta' \leftarrow \theta$ and R2D2 buffer $\mathbf{D} \leftarrow \{\}$, L is the R2D2
    sequence length, $\phi$ is the pretrained observation encoder
 2: for episode $= 1, M$ do
 3:     start episode, init history $h \leftarrow \{\}$
 4:     while not done do
 5:         receive observation $y_t$, encode $y_t$ with $\phi$ and append to history h.
            /* $\epsilon$-greedy policy for data collection */
 6:         Select action $a_t = \text{argmax}_a Q_\theta(\sigma(h, \mathbf{0}), a)$ with probability $1 - \epsilon$ otherwise random action
            with probability $\epsilon$, append action to h
            /* Add data to buffer in regular intervals (Once every L steps)  */
 7:         Add sequence to $\mathbf{D}$
 8:         Sample batch of experience sequences $(h_b, z, y_{1:L}, r_{1:L}, a_{1:L})$ from $\mathbf{D}$
            /* Q-function and AIS component training */
 9:         Initialize $\sigma$ with z and burn-in history $h_b$ and detach gradients: $z' = \sigma(h_b, z)$
10:         Compute AIS loss for $(z', y_{1:L}, r_{1:L}, a_{1:L})$ and update $\sigma$, $\hat{P}^y$ and $\hat{r}$
11:         Compute n-step Q-learning loss for $(z', y_{1:L}, r_{1:L}, a_{1:L})$ and update $Q_\theta$
12:         Update target Q-function in regular intervals $\theta' \leftarrow \theta$
13:     end while
14: end for
```

D.1 Recurrent replay buffer

With non-recurrent DQN for fully-observable environments, individual single step samples (s, a, r, s') are saved in the replay buffer. With Recurrent Q-learning, full histories are required with each sample to allow for the accurate computation of the recurrent hidden states. The R2D2 method [Kap+19] suggests saving fixed length sequences of observation, action and rewards (y, a, r) in the replay buffer. Furthermore, to recreate the internal state of the RNN during training, it is suggested to store the following with each sample sequence: 1) A truncated preceding "Burn-In" history, 2) An initial RNN internal hidden state representing the internal RNN state at the start of the "Burn-In" history. The "Burn-In" sequence allows us to unroll the recurrent network on a truncated portion of preceding history with each sample sequence. The "Burn-In" portion is only used for unrolling the recurrent network and updates are only done on the main sequence. Additionally, the recurrent state at the start of this "Burn-In" history is also saved with each sample in the buffer. Both of these allow the recurrent network to recreate the hidden state at the start of the main sample sequence as accurately as possible. We utilize the same replay buffer for RQL-AIS and our ND-R2D2 baseline.

The problem of inaccuracy in recreated RNN hidden states is called representation drift and the effect of distributed training on this problem is investigated in the original R2D2 paper [Kap+19]. The representation drift problem is less severe when the data samples are more recent as their saved internal states comes from a version of the RNN which is closer to the most recently updated version. It is demonstrated that the distributed setting can mitigate this issue with higher number of actors [Kap+19]. The higher number of actors allows for a larger amount of generated experience, which results in data being added to the buffer more frequently, meaning older samples are less likely to be sampled by the learner unit, effectively alleviating the isssue. In our setup, distributed training is both unnecessary and infeasible. The environments used for our experimental setup require significantly less data to be successfully solved and also the computational resources available at our disposal do not allow for a distributed R2D2 implementation. In order to mitigate the representation drift problem in our setup, we choose a different set of R2D2 hyperparameters compared with [Kap+19].

First, we choose a significantly smaller main sequence length for our R2D2 buffer and second, we choose a much longer "Burn-In" sequence length compared with the original R2D2 implementation. This combination allows us to achieve a good performance with the R2D2 replay buffer while using a single actor setup. By using a shorter main sequence length and a larger batch size, single step samples are less correlated in each batch which helps with the performance of Q-learning. Furthermore, adjacent saved samples are no longer overlapping each other as they do in the original R2D2 implementation as we find that unnecessary considering the shorter main sequence length.

D.2 Q-learning

In the main text, we discussed the AIS loss used for training the AIS-generator. Assuming Z_t is the recreated AIS state representation for each single-step sample, we train the Q-function using the n-step Q-learning loss:

$$\left| \sum_{k=0}^{n-1} R_{t+k}\gamma^k + \gamma^n Q_{\theta^-}\left(Z_{t+n}, a^*\right) - Q_\theta\left(Z_t, A_t\right) \right|^2, \quad a^* = \arg\max_a Q_\theta\left(Z_{t+n}, a\right). \tag{32}$$

with Q_θ being the Q-function and Q_{θ^-} being the target Q-function. Also, a^* is chosen by taking the arg-max of $Q_\theta(Z_{t+n}, a)$ similar to the double Q-learning approach [HGS15]. The multi-step length n is set to 5 for all experiments. Following the DQN implementation, target Q-function parameters θ^- are periodically updated with the most recent value of θ. Gradients from the Q-learning loss do not backpropagate through the AIS generator parameters.

We can further utilize prioritized experience replay (PER) [Sch+15a] with RQL-AIS. With prioritized sampling, sequences in the R2D2 buffer are sampled with non-uniform probabilities. Each sequence's priority is the absolute value of the n-step Q-learning error averaged over all samples in the sequence. Similar to the original PER method [Sch+15a], we use importance-sampling (IS) weights to correct the change in the data distribution introduced by the non-uniform weights. The importance sampling weights are applied to rescale both the AIS loss and the Q-learning loss.

D.3 Environments

We evaluate the performance of RQL-AIS and other baseline algorithms on 22 environments form the Minigrid testbed [CWP18]. All environments are partially observed grid worlds where the agent has to solve a series of tasks. At each time, the agent observes a 7×7 view in front of it; each observed tile is encoded as a three-dimensional tuple of (OBJECT_ID, COLOR_IDX, STATE). The agent cannot see through walls and closed doors nor see inside closed boxes. The agent receives a positive reward of $+1$ on successful completion of the task and no intermediate rewards. The environments are grouped into eight major categories, as shown in Figure 4

Given the large (discrete) observation space, we follow the methodology of [Sub+22] and use a pre-trained auto-encoder to compress the observations. The auto-encoder is trained on a dataset of random agent observations, separately for each environment. During the RL phase, the auto-encoder is frozen and doesn't change.

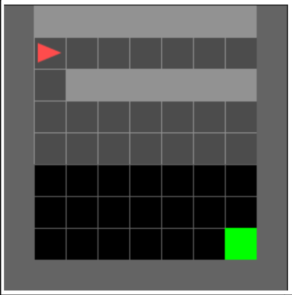
D.4 Algorithms tested

We evaluate the performance of the algorithms described below. Each algorithm has two variants: one using uniform sampling (denoted by "U") and another using prioritized experience replay (denoted by "PER")

- `RQL-AIS-U` and `RQL-AIS-PER`: Recurrent Q-learning algorithm with AIS representations.
- `ND-R2D2-U` and `ND-R2D2-PER`: Non-distributed R2D2
- `ND-R2D2-AE-U` and `ND-R2D2-AE-PER`: Non-distributed R2D2 where observations are compressed using pre-trained auto-encuders.

Note that both `RQL-AIS` and `ND-R2D2-AE` use the same pretrained auto-encoders.

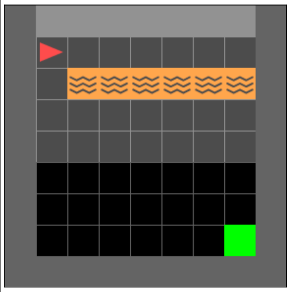
Simple Crossing



Navigate multiple corridors to reach the goal state.

Four variants of this environment with varying sizes are tested.

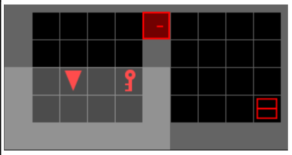
Lava Crossing



Navigate through lava fields to reach the goal while avoiding lava.

Four variants of this environment with varying sizes and number of lava fields are tested.

Unlock Pickup

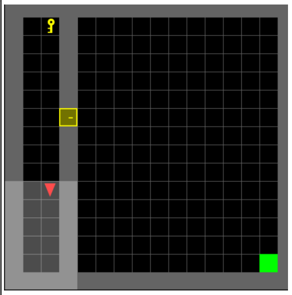


There are two variants:

Unlock: Pick up a key and unlock the door.

Unlock Pickup: Additional step of picking up a box.

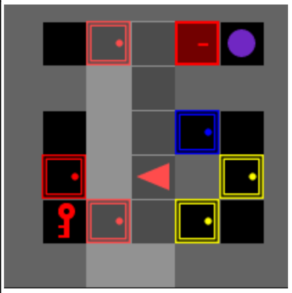
Door Key



Start in one room, find a key to unlock the door to the second room, navigate in the second room to reach the goal state.

Three variants of this environment with increasing room sizes are tested.

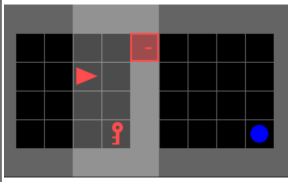
Key Corridor



Find the key behind a series of unlocked but closed doors. Use this key to open a door which leads to the goal state.

Three variants of this environment with increasing room sizes and larger number of doors and corridors are tested.

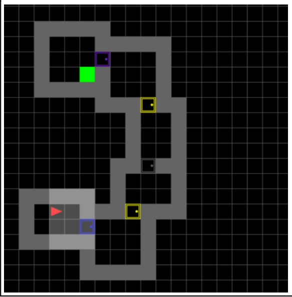
Obstructed Maze



Find a key and use it to unlock the door to the second room and navigate to reach the goal state.

In the second variant of this environment, the key is hidden in a box.

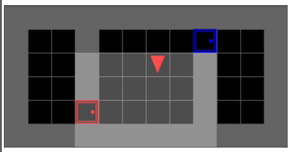
Multi-room



Navigate a series of rooms each connected to the next via unlocked but closed doors. The goal state is in the final room.

Two variants with varying number of rooms are tested.

Red Blue Doors



First open the red door, then the blue door.

Two variants with varying room sizes are tested.

Figure 4: Details of the different MiniGrid environments

We train all algorithms for $T = 4 \cdot 10^6$ environment steps for $N = 5$ seeds. The agents are trained using epsilon-greedy approach [Mni+13] to allow for exploration. The value of epsilon is chosen to smoothly decay between ϵ_{start} and ϵ_{end} according to:

$$\epsilon_{\text{expl},t} = \epsilon_{\text{end}} + (\epsilon_{\text{start}} - \epsilon_{\text{end}}) \exp\left(-\frac{t}{\epsilon_{\text{decay}}}\right).$$

	RQL-AIS		ND-R2D2			
Environment	U	PER	U	PER	AE-U	AE-PER
SimpleCrossingS9N1	0.789 ± 0.166	0.767 ± 0.111	0.926 ± 0.044	0.953 ± 0.005	0.950 ± 0.016	0.952 ± 0.003
SimpleCrossingS9N2	0.944 ± 0.007	0.912 ± 0.022	0.757 ± 0.423	0.949 ± 0.001	0.935 ± 0.023	0.904 ± 0.083
SimpleCrossingS9N3	0.934 ± 0.006	0.905 ± 0.031	0.933 ± 0.019	0.932 ± 0.016	0.941 ± 0.006	0.935 ± 0.009
SimpleCrossingS11N5	0.835 ± 0.072	0.927 ± 0.011	0.285 ± 0.140	0.210 ± 0.159	0.535 ± 0.322	0.156 ± 0.135
LavaCrossingS9N1	0.853 ± 0.054	0.830 ± 0.054	0.955 ± 0.005	0.950 ± 0.009	0.955 ± 0.004	0.953 ± 0.002
LavaCrossingS9N2	0.926 ± 0.014	0.844 ± 0.044	0.934 ± 0.034	0.561 ± 0.512	0.943 ± 0.005	0.605 ± 0.461
LavaCrossingS9N3	0.871 ± 0.063	0.850 ± 0.061	0.920 ± 0.029	0.860 ± 0.045	0.942 ± 0.007	0.699 ± 0.394
LavaCrossingS11N5	0.316 ± 0.145	0.489 ± 0.138	0.310 ± 0.254	0.052 ± 0.116	0.777 ± 0.093	0.272 ± 0.365
Unlock	0.956 ± 0.011	0.965 ± 0.005	0.969 ± 0.002	0.967 ± 0.005	0.946 ± 0.042	0.968 ± 0.001
UnlockPickup	0.517 ± 0.474	0.847 ± 0.066	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000
DoorKey-5x5	0.926 ± 0.053	0.928 ± 0.043	0.964 ± 0.001	0.962 ± 0.002	0.957 ± 0.016	0.964 ± 0.000
DoorKey-6x6	0.954 ± 0.005	0.910 ± 0.076	0.967 ± 0.005	0.952 ± 0.030	0.951 ± 0.039	0.965 ± 0.007
DoorKey-8x8	0.942 ± 0.038	0.945 ± 0.026	0.371 ± 0.508	0.760 ± 0.427	0.000 ± 0.000	0.189 ± 0.422
KeyCorridorS3R1	0.937 ± 0.026	0.945 ± 0.013	0.944 ± 0.012	0.928 ± 0.037	0.950 ± 0.001	0.948 ± 0.003
KeyCorridorS3R2	0.885 ± 0.038	0.631 ± 0.365	0.000 ± 0.000	0.000 ± 0.000	0.183 ± 0.410	0.722 ± 0.404
KeyCorridorS3R3	0.155 ± 0.347	0.759 ± 0.119	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000
ObstructedMaze-1Dl	0.916 ± 0.020	0.916 ± 0.022	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.186 ± 0.415
ObstructedMaze-1Dlh	0.000 ± 0.000	0.911 ± 0.022	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000
MultiRoom-N2-S4	0.790 ± 0.049	0.787 ± 0.044	0.839 ± 0.010	0.837 ± 0.004	0.842 ± 0.003	0.834 ± 0.006
MultiRoom-N4-S5	0.438 ± 0.400	0.431 ± 0.394	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000
RedBlueDoors-6x6	0.865 ± 0.075	0.889 ± 0.084	0.977 ± 0.011	0.982 ± 0.000	0.976 ± 0.013	0.979 ± 0.004
RedBlueDoors-8x8	0.977 ± 0.009	0.922 ± 0.050	0.962 ± 0.018	0.756 ± 0.423	0.963 ± 0.012	0.950 ± 0.046

Table 2: Comparison of RQL-AIS with the different variants of ND-R2D2 on the MiniGrid benchmark.

The trained agents are evaluated at regular intervals. During evaluation, the agents are tested for 10 episodes and actions are chosen greedily with respect to the trained Q-function.

D.5 Training results

The mean and the standard deviations of the final performance for all algorithms is shown in Table 2. Figure 5 shows the training curves for all 22 tested environments.

Prioritized experience replay can offer a significant boost to both algorithms in more difficult environments and we see RQL-AIS-PER to show the best performance among all tested algorithms. This variant of RQL-AIS is capable of solving all of the tested environments. We observe that variants of RQL-AIS and ND-R2D2 with prioritized sampling generally perform worse in simpler environments. Prioritized sampling relies on priority weights obtained from the absolute value of the n-step Q-learning loss. In the initial phases of training, Q value estimates are inaccurate, making prioritized sampling hurt performance initially while the uniform sampling variants of these algorithms are able to quickly solve these environments. We observe that AIS state representations are very effective in solving the more difficult environments and RQL-AIS to show superior performance in more difficult environments regardless of the sampling approach used for the replay buffer.

We are reporting two additional variants of R2D2 which work with the pretrained encoders (similar to RQL-AIS). The main ND-R2D2 baseline used in the main text does not use pretrained encoders as we believe pretrained encoders are not an essential component of ND-R2D2. Nevertheless, we report results for ND-R2D2 using these pretrained encoders so that their effect can be decoupled from the other factors. We observe that their performance is very similar to ND-R2D2 working on raw observations.

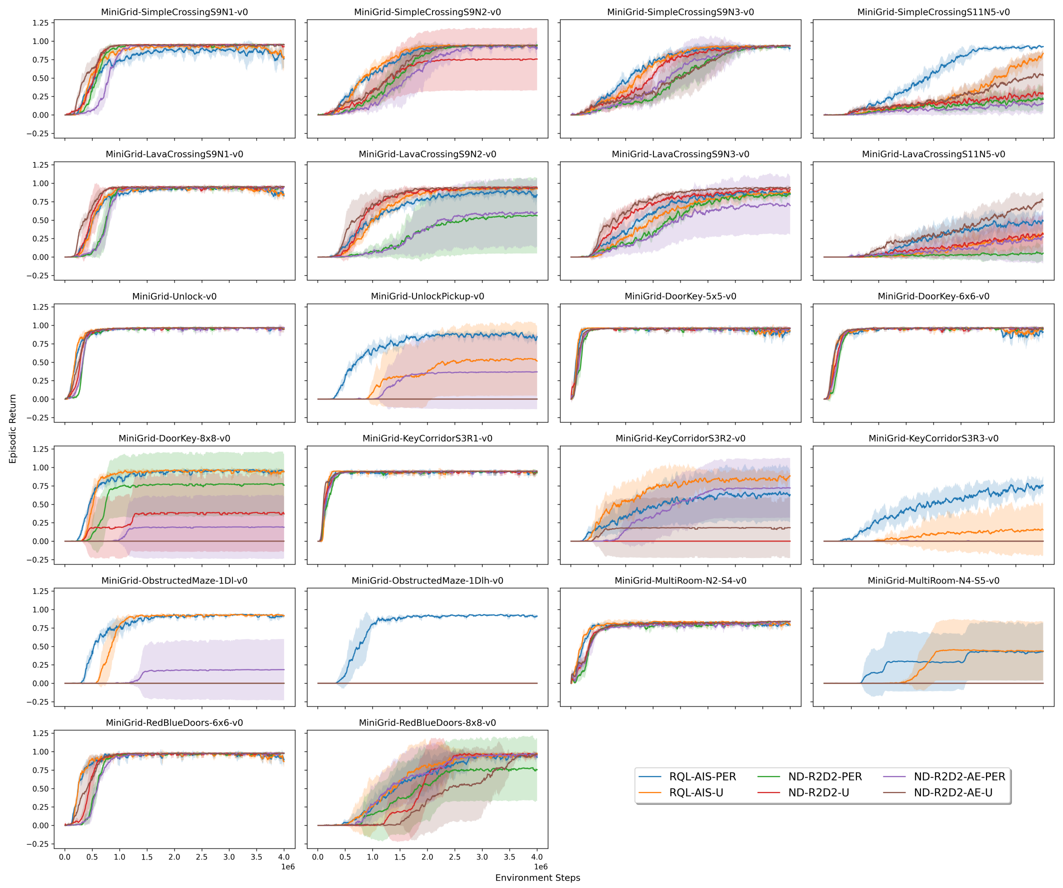


Figure 5: Training curves from all 22 tested MiniGrid environments. Results for both uniform sampling and prioritized sampling variants of RQL-AIS and ND-R2D2 are provided. Two additional variants of R2D2 are also included which use the pretrained encoders.

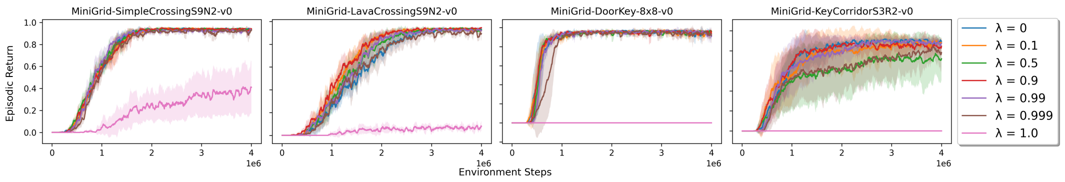


Figure 6: Performance plots of RQL-AIS using different values of the λ hyperparameter.

E Effect of the hyperparameter λ

In this section, we report the effect of the λ hyperparameter on performance for four candidate environments. The plots show the performance of RQL-AIS with uniform sampling over time. We observe that the performance of RQL-AIS is robust with respect to the choice of the λ hyperparameter with minor changes in performance seen between the different variants except for the $\lambda = 1.0$ case which represents training the AIS using only the reward loss. Very high values of λ can diminish the gradients coming from the observation prediction component and hurt performance. We believe that the observation prediction component of the AIS loss has a big impact on the performance of RQL-AIS. This is understandable as MiniGrid environments are sparse reward and zero rewards are received on every environment interaction except the terminal steps leading to a successful completion of the task. Observation prediction can provide a much richer learning signal allowing the agent to learn more meaningful AIS state representations. The performance of RQL-AIS with different values of λ is reported in Figure 6. We choose $\lambda = 0.5$ for all experiments involving RQL-AIS.

Number of Seeds	5
Number of Environment steps	4×10^6
Discount Factor γ	0.99
ϵ_{start}	1.0
ϵ_{end}	0.05
ϵ_{decay}	400000
Evaluation Interval	5000 environment steps
R2D2 sequence length	10
R2D2 Burn-In sequence length	50
Optimizer	Adam [KB14]
AIS λ (for RQL-AIS)	0.5
AIS learning rate (for RQL-AIS)	10^{-3}
Q-function learning rate (for RQL-AIS)	10^{-3}
learning rate (for ND-R2D2)	10^{-3}
Minibatch Size	256
Network Update Interval	10 environment steps
Target network update interval	100 updates to main network
Priority Exponent (PER α)	0.6
Importance Sampling Exponent (PER β)	$[0.4, 1.0]$
LSTM hidden size (d_Z)	128

Table 3: Hyperparameter values.

F Implementation details

In this section, we first present the hyperparameters used for implementing RQL-AIS and ND-R2D2. These hyperparameters are shared between the two algorithms (unless specifically stated). We report the hyperparameter values in Table 3.

In all MiniGrid environments, the agent receives $7 \times 7 \times 3$ sized observation vectors. In order to compress the observations into a more compact representation, we are using pretrained encoders. These are simple auto-encoders [Kra91] that are trained on datasets of random agent observations. The encoders are trained for 100 epochs on this dataset. We use MLP layers with ReLU nonlinearities [Fuk75] for the encoder and the decoder architecture. The encoder weights are frozen during the RL phase. The encoder and decoder architectures are as follows:

Encoder	Decoder
Linear $(147, 96)$	Linear $(64, 96)$
ReLU	ReLU
Linear $(96, 64)$	Linear $(96, 147)$
	Tanh

RQL-AIS has the following four components: the recurrent history compression function $\hat{\sigma}$, the reward prediction function $\hat{r}$, the observation prediction function $\hat{P}^y$ and the action-value function $\hat{Q}_\theta$. During our experimentation, we use Exponential Linear Units (ELUs) [CUH15] for the nonlinear layers. For the recurrent component, we are using an LSTM [KT01] function. Linear layers with input sizes n and output sizes m are denoted by Linear(n, m) and an LSTM cell with input of size n and a hidden vector size of m is denoted by LSTM(n, m). n_O and n_A denote the observation vector size and the action space size. The neural network architectures of the four components are:

$\hat{\sigma}$	$\hat{r}$	$\hat{Q}_\theta$	$\hat{P}^y$
Linear $\left(n_O + n_A, d_{\hat{Z}}\right)$	Linear $\left(n_A + d_{\hat{Z}}, \frac{1}{2}d_{\hat{Z}}\right)$	Linear $\left(d_{\hat{Z}}, d_{\hat{Z}}\right)$	Linear $\left(n_A + d_{\hat{Z}}, \frac{1}{2}d_{\hat{Z}}\right)$
ELU	ELU	ELU	ELU
LSTM $\left(d_{\hat{Z}}, d_{\hat{Z}}\right)$	Linear $\left(\frac{1}{2}d_{\hat{Z}}, 1\right)$	Linear $\left(d_{\hat{Z}}, d_{\hat{Z}}\right)$	Linear $\left(\frac{1}{2}d_{\hat{Z}}, n_O\right)$
		ELU	
		Linear $\left(d_{\hat{Z}}, n_A\right)$	

For the experiments with ND-R2D2 we have divided the architecture into two components: the recurrent unit $\hat{\sigma}$ and the Q-function $\hat{Q}_\theta$. This is done for simplicity of implementation and consistency with the RQL-AIS notation but both are trained with the Q-learning loss. The Q-learning variant which does not use the pretrained autoencoders has an extra layer in its recurrent unit to allow for extra processing of the higher-dimensional inputs.

$\hat{\sigma}(AE)$	$\hat{\sigma}(\text{raw})$	$\hat{Q}_\theta$
Linear $\left(n_O + n_A, d_{\hat{Z}}\right)$	Linear $\left(n_O + n_A, d_{\hat{Z}}\right)$	Linear $\left(d_{\hat{Z}}, d_{\hat{Z}}\right)$
ELU	ELU	ELU
LSTM $\left(d_{\hat{Z}}, d_{\hat{Z}}\right)$	Linear $\left(d_{\hat{Z}}, d_{\hat{Z}}\right)$	Linear $\left(d_{\hat{Z}}, d_{\hat{Z}}\right)$
	ELU	ELU
	LSTM $\left(d_{\hat{Z}}, d_{\hat{Z}}\right)$	Linear $\left(d_{\hat{Z}}, n_A\right)$

G Limitations

In this work, we have conducted an extensive theoretical analysis into the convergence of recurrent Q-learning. Our analysis is restricted to a setting where the true state space and the recurrent state space are both finite and therefore the tabular setup can be used to represent the Q-function. In practice, the state space is either continuous or large so that some form of function approximation is needed and the recurrent state is continuous.

In addition, we assume that the AIS-generator functions including the recurrent history compression function are fixed throughout the Q-learning iterations. In a practical algorithm such as `RQL-AIS`, the representation is learnt in parallel with the Q-function.

Our results are derived under a slightly strong assumption on the learning rates **(A3)**, while most convergence results (for MDPs) use the weaker assumption **(A3')**.

In our empirical evaluation, we train the representation using an observation prediction as a proxy for the AIS-predictor. It is shown in [Sub+22, Proposition 8] that having a good observation predictor along with a recursively updateable AIS is a sufficient condition for having a good AIS-predictor. However, predicting all the observations might not be necessary and using an AIS-predictor may, in principle, need a smaller representation.