TOWARDS OPTIMAL POLICY IMPROVEMENT

Anonymous authors
Paper under double-blind review

ABSTRACT

Practical Reinforcement Learning (RL) algorithms learn to solve Markov Decision Processes (MDPs) through iterative policy improvement in the presence of approximate evaluation. We study policy improvement from first principles, defining optimal policy improvement as producing the best policy attainable in a single update under specified constraints. We show that optimal improvement restricted to a set of states is equivalent to solving an induced MDP, characterizing planning with an explicit or implicit model as a path towards optimal policy improvement. Because practical methods commonly solve such induced problems through iterative improvement in the form of greedification, we take steps towards optimal greedification under the central practical constraint of approximate evaluation. We formulate greedification under this constraint as probabilistic decision-making under uncertainty and derive a novel operator that is optimal with respect to the resulting objective. Empirically, the operator and its practical gradient-based approximations improve aggregate performance across GumbelAlphaZero, SAC, ReBRAC and Generalized Policy Iteration, in experiments spanning discrete and continuous actions, model-based and model-free, online and offline RL.

1 INTRODUCTION

The objective of Reinforcement Learning (RL) agents is to find well-performing policies for sequential decision problems, formalized as *Markov Decision Processes* (MDPs, Bellman, 1957). The performance of the policy is measured as its *expected return*, also called the policy's *value*. To find such policies, practical algorithms rely on the (approximately) monotonic learning process of iterative *Policy Improvement* (Sutton & Barto, 2018). Examples of algorithms that can be abstracted to this process are (Deep) Q-learners (Watkins & Dayan, 1992; Mnih et al., 2015), Actor-Critics (Konda & Tsitsiklis, 1999), learners-in-the-model (Hafner et al., 2023), planning-based learners (Silver et al., 2018; Schrittwieser et al., 2020) and many others. To drive policy improvement, RL algorithms use *Policy Improvement Operators* which take the agent's current policy and produce a better policy (Oren et al., 2025b; Chan et al., 2022). Prominent examples of policy improvement operators are Policy Gradient (Sutton et al., 2000), Monte Carlo Tree Search (Kocsis & Szepesvári, 2006) and the greedy operator used by Q-Learning.

A central design question in RL is which policy improvement operator best uses the information available to the agent under practical constraints. Approximate evaluation is particularly important because of both its prevalence and its adverse effects on learning (Baird, 1995; Tsitsiklis & Roy, 1997; Thrun & Schwartz, 2014; van Hasselt, 2010; van Hasselt et al., 2016; 2018). Popular algorithms mitigate the effects of evaluation error on policy improvement by constraining policy updates through different forms of regularization (Schulman et al., 2017; Haarnoja et al., 2018; Abdolmaleki et al., 2018; Danihelka et al., 2022; Hafner et al., 2023). These methods implicitly encode decisions about how uncertain evaluation should affect improvement. Making those decisions explicit and deriving operators that optimize the resulting objective remains an important avenue for research.

In this work we take steps towards optimal policy improvement under practical constraints such as approximate evaluation, in two central parts. First, we characterize optimal improvement under state-restricted policy updates. We show that constrained optimal improvement is equivalent to solving an induced MDP. These results connect optimal policy improvement to planning with explicit models such as world models and implicit models such as replay buffers.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

To solve the MDP in the model, practical RL algorithms rely on *greedification* for policy improvement (Sutton & Barto, 2018; Oren et al., 2025b). This motivates the second part of our analysis: optimal greedification under approximate evaluation. We model this problem explicitly as a problem of optimization under uncertainty formulated as Maximum a Posteriori optimization, and derive $\mathcal{I}_{\text{opt}}$, a novel greedification operator which is optimal for the derived optimization objective. $\mathcal{I}_{\text{opt}}$ generalizes an operator arising from Regularized Policy Optimization (Neu et al., 2017), Control as Inference (Levine, 2018), Mirror Descent (Tomar et al., 2022), and Trust Region Policy Optimization (Schulman et al., 2015), and used by popular algorithms such as GumbelAlpha/MuZero (Danihelka et al., 2022) and TDMPC variants (Hansen et al., 2024).

Theoretically, we investigate the soundness of $\mathcal{I}_{\text{opt}}$ for driving the RL process by establishing convergence in a Dynamic Programming setup underlying the general learning process of RL (Oren et al., 2025b). Empirically, $\mathcal{I}_{\text{opt}}$ and its practical gradient-based approximations improve performance across experiments encompassing RL and Dynamic Programming, discrete and continuous action spaces, model based and model free, classical and modern, online and offline algorithms. These include GumbelAlphaZero (Danihelka et al., 2022) in *9x9 Go* and *Chess*, Soft Actor-Critic (Haarnoja et al., 2018) in 10 classical DeepMind Control (Tunyasuvunakool et al., 2020) continuous-action environments, ReBRAC (Tarasov et al., 2023) in 4 classical offline RL domains of D4RL (Fu et al., 2020) and Policy Iteration algorithms (Sutton & Barto, 2018) across infinite horizon grid environments. These results support the design of improved policy improvement operators as a promising basis for better performing RL algorithms, and in addition, the construction of explicit, uncertainty-aware decision objectives as a promising basis for designing stronger policy improvement operators.

2 PRELIMINARIES

In RL, the environment is represented by an MDP $M = \langle \mathcal{S}, \mathcal{A}, \rho, R, P, \gamma \rangle \in \mathcal{M}$, where $\mathcal{S}$ is a set of states, $\mathcal{A}$ a set of actions, ρ an initial state distribution. For simplicity of presentation, we will focus on discrete and finite state and action spaces. $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is a bounded possibly stochastic reward function, and P is a transition distribution such that $P(s'|s, a)$ specifies the probability of transitioning from state s to state s' with action a. A policy $\pi \in \Pi$ is a stochastic mapping from states $s \in \mathcal{S}$ to probability distributions on the action space $\mathcal{A}$. We write $a \sim \pi(s)$ and denote the probability of a at s under π with $\pi(a|s)$. The objective of RL is to find a policy which maximizes *the expected discounted return* (or *value*) V^π:

$$\mathcal{J}^\pi := \mathbb{E}_{s\sim\rho}[V^\pi(s)] := \mathbb{E}\left[\sum_{t=1}^{H} \gamma^{t-1} R(s_t, a_t)\right], s_1 \sim \rho, a_t \sim \pi(\cdot|s_t), s_{t+1} \sim P(\cdot \mid s_t, a_t). \quad (1)$$

The discount factor $\gamma \in (0, 1)$ is introduced in the infinite-horizon setting ($H \to \infty$) to ensure that the value function V^π is well-defined and bounded. In the finite-horizon case, one typically sets $\gamma = 1$. The state-action value function (Q-function) associated with a policy π is defined by $Q^\pi(s, a) := \mathbb{E}\Big[R(s, a) + \gamma V^\pi(s')\Big]$, where $s' \sim P(\cdot \mid s, a)$. The optimal state-value function, and state-action value function are defined, respectively, by $V^*(s) := \sup_{\pi\in\Pi} V^\pi(s), Q^*(s, a) := \sup_{\pi\in\Pi} Q^\pi(s, a)$, and an optimal policy π^* satisfies $\forall s \in \mathcal{S} : V^{\pi^*}(s) = V^*(s)$. The advantage function is defined as:

$$A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s) = Q^\pi(s, a) - \textstyle\sum_{b\in\mathcal{A}} \pi(b|s) Q^\pi(s, b). \quad (2)$$

With access to the dynamics of the environment M, MDPs can be solved using Dynamic Programming (DP) methods, such as Value Iteration or Policy Iteration (Bertsekas, 2025). These methods can be cast as instances of *Generalized Policy Iteration* (GPI, Algorithm 1), which iterate *policy improvement* and *policy evaluation* using different *policy improvement operators* and underpin the learning process of many RL algorithms (Sutton & Barto, 2018; Oren et al., 2025b). The policy improvement operator is generally defined as follows (Sutton & Barto, 2018):

Definition 1 (Policy Improvement Operator)**.** *We call an operator $\mathcal{I} : \Pi \to \Pi$ a* policy improvement operator *if, $\forall \pi \in \Pi$ as long as the policy is not already optimal, it satisfies:*

$$\forall s \in \mathcal{S} : \; V^{\mathcal{I}(\pi)}(s) \geq V^\pi(s) \quad \textit{and} \quad \exists s \in \mathcal{S} : \; V^{\mathcal{I}(\pi)}(s) > V^\pi(s). \quad (3)$$

A popular class of policy improvement operators are *greedification operators*:

Definition 2 (Greedification Operator)**.** *We call* $\mathcal{I} : \Pi \times \mathcal{Q} \to \Pi$ *a greedification operator if* $\forall \pi \in \Pi$, $\forall Q \in \mathcal{Q}$, $\forall s \in \mathcal{S}$ *it satisfies:*

$$\sum_{a \in \mathcal{A}} \mathcal{I}(\pi, Q)(a|s) Q(s,a) \geq \sum_{a \in \mathcal{A}} \pi(a|s) Q(s,a), \tag{4}$$

and $\exists s \in \mathcal{S}$ *such that the inequality is strict, unless* π *is already greedy with respect to* Q*.*

$\mathcal{Q}$ denotes the set of all bounded functions $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, to indicate that greedification is well defined with respect to exact evaluation $Q = Q^\pi$ as well as approximate evaluation $Q = \hat{Q}^\pi \approx Q^\pi$. When $Q = Q^\pi$, *greedification* (Inequality 4) guarantees policy improvement (Sutton & Barto, 2018). A popular example of a greedification operator is the greedy operator $\mathcal{I}_{\arg\max}(\pi, Q)(s) = \arg\max_{a \in \mathcal{A}} Q(s,a)$ used by Q-learning and Value / Policy Iteration. We define $\arg\max_{a \in \mathcal{A}}$ generally as any distribution which has support only over maximizing actions. Another popular example is $\mathcal{I}_{reg}(\pi, Q)(a|s) \propto \pi(a|s) \exp(Q(s,a)/\eta)$ (Neu et al., 2017; Levine, 2018; Xiao, 2022) which satisfies Definition 2 when π has support over the entire domain $\mathcal{A}$.

3 A Path to Optimal Policy Improvement

Our aim in this work is to chart a path towards RL algorithms driven by policy improvement operators which are optimal under practical constraints. We first show that under a natural definition and a common constraint, optimal improvement is equivalent to solving an induced MDP in the model.

3.1 Planning as a Foundation for Optimal Policy Improvement

We begin with a definition of optimal improvement:

Definition 3 (Optimal Policy Improvement)**.** *We call* $\mathcal{I}^*$ *an* optimal policy improvement operator *if it satisfies the following:*

$$\forall \pi \in \Pi, \forall s \in \mathcal{S}: \quad \mathcal{I}^* \in \arg\inf_{\mathcal{I}} \left[V^*(s) - V^{\mathcal{I}(\pi)}(s) \right] = \arg\sup_{\mathcal{I}} \left[V^{\mathcal{I}(\pi)}(s) - V^\pi(s) \right]. \tag{5}$$

In fact, the constants V^* and V^π do not influence the optimization problem, and are included only to emphasize that *improvement* is equivalent to *minimizing the distance from the optimum* as well as *maximizing the increase over any policy* π. Without imposing constraints on $\mathcal{I}$, this natural definition immediately makes the optimal improvement equivalent to simply solving the MDP: $\mathcal{I}^*(\pi)(s) = \pi^*(s)$. Such operators are not artificial, and can be interpreted as finding the optimal policy using exhaustive planning (Osband et al., 2013; Marjani & Proutière, 2021) or training to global convergence with a replay buffer that contains every transition in the environment, which itself can be viewed as planning with a pseudo-model (Sutton & Barto, 2018; Young et al., 2023).

However, in practice improving the policy at all states at the same time is often impractical due to the size of $|\mathcal{S}|$ or lack of coverage of $\mathcal{S}$ in the data. In this case, we can think of operators as constrained to improving the policy at a batch of states $S \subseteq \mathcal{S}$. We find that *even when limiting the operator to improving only in* $S \subset \mathcal{S}$, optimal improvement is equivalent to solving a (semi-)MDP $M^{semi}_{S,\pi}$:

Theorem 1. *(Informal) Optimal policy improvement constrained by* $S \subset \mathcal{S}$*, such that* $\forall s \notin S$: $\mathcal{I}(\pi)(s) = \pi(s)$*, is equivalent to solving the induced semi-MDP* $M^{\text{semi}}_{S,\pi}$.

Intuition: The semi-MDP $M^{\text{semi}}_{S,\pi}$ can be understood as equal to M on the decision states in S. Whenever a transition leaves S, the resulting excursion is collapsed into a temporally extended transition: the agent follows π until it next returns to S, while preserving the discounted return accumulated along the excursion and the corresponding discounted continuation value. We provide formal definitions and proof in Appendix A.1. As any discounted SMDP admits an equivalent MDP representation (Sutton et al., 1999), we refer to solving this induced SMDP interchangeably as solving an induced MDP.

Constructing $M^{semi}_{S,\pi}$ requires knowledge of transitions outside of S, however. Thus, it does not directly represent the setting where the agent only has access to a dataset composed of transitions in the environment which were gathered during training and are memorized in a replay buffer or encoded in a learned model. More closely related to this setting, we can define the sub-MDP $M^{sub}_{S,\pi}$ on S as the MDP which maintains the original dynamics for transitions within S but terminates

whenever a transition leaves S with returns bootstrapped $R(s,a) + \gamma V^\pi(s')$. Thus, $M^{sub}_{S,\pi}$ optimizes within S, but does not model the trajectory after leaving S. In this case, we have a similar result:

Theorem 2. *(Informal) Optimal policy improvement constrained by $S \subset \mathcal{S}$ is equivalent to solving the sub-MDP $M^{sub}_{S,\pi}$, if leaving S precludes returning to S. More generally, solving the sub-MDP $M^{sub}_{S,\pi}$ lower bounds the optimum and upper bounds a single step greedification followed by the policy π, i.e. $\max_a Q^\pi(s,a)$.*

Intuition: If leaving S precludes returning to S, the sub and semi MDPs coincide. If it is possible to leave and then return to S, both planning as well as single step greedification can only account for the future outside of S through the value function. We provide a formal description of $M^{sub}_{S,\pi}$ and a full proof in Appendix A.2.

Such an operator can be interpreted as approximated by learning a model for transitions in S and conducting search inside this model, such as MuZero (Schrittwieser et al., 2020), or training in this model, such as Dreamer (Hafner et al., 2023), or repeatedly training on transitions in $S \times \mathcal{A}$ while bootstrapping from $V^\pi(s')$ when the successor lies outside S.

We conclude: *constrained optimal policy improvement is equivalent to solving an induced MDP*, directly connecting optimal improvement to planning with an explicit or implicit model.

3.2 Greedification as a Driver of Planning-Based Policy Improvement

Theorems 1 and 2 show that optimal improvement is equivalent to, or bounded by, solving MDPs induced by the model, but do not show us *how* to solve these MDPs. In the RL literature, the popular choice for driving policy improvement within the model is greedification (Definition 2) (Sutton & Barto, 2018; Oren et al., 2025b). Thus, a natural starting point for designing optimal policy improvement operators is the investigation of *optimal greedification*. We define *optimal greedification* as satisfying the objective:

$$\arg\max_{\pi'} \sum_{a\in\mathcal{A}} Q^\pi(s,a)\pi'(a|s) \;=\; \arg\max_{\pi'} \sum_{a\in\mathcal{A}} A^\pi(s,a)\pi'(a|s), \tag{6}$$

i.e. maximizing with respect to the exact evaluation $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$ at a state s. Under exact evaluation, Equation 6 defines an unambiguous greedification objective, whose solutions are greedy policies, with $\mathcal{I}_{\arg\max}$ naturally emerging as the optimal greedification operator (see Lemma 1 in Appendix A.4). When evaluation is approximate however, optimization should account for the evaluation uncertainty and optimal greedification must therefore be defined relative to specific decision formulations, which we address next.

4 Optimal Greedification under Approximate Evaluation

In this section we formulate greedification under approximate evaluation as a probabilistic decision problem, make the required modeling choices and approximations to arrive at a tractable decision objective and derive a novel greedification operator which is optimal with respect to the derived decision problem. We then proceed to establish that the operator is sound for driving the iterative learning process underlying practical RL algorithms by showing that it is sufficient for convergence to the optimal policy in a popular setup for such analysis (Tsitsiklis, 2002; Oren et al., 2025b).

4.1 Deriving the operator

Approximate evaluation $\hat{Q}^\pi \approx Q^\pi$ induces uncertainty over which candidate policy π' maximizes $\sum_{a\in\mathcal{A}} \pi'(a \mid s) A^\pi(s,a)$. Translating this uncertainty into a policy update requires specifying a decision problem. We develop an explicit formulation of such a decision problem based on *generalized Bayesian inference* (Bissiri et al., 2016): the current policy supplies a prior over candidate policies, while an uncertainty-marginalized utility factor represents the evidence favoring each candidate. This gives the generalized posterior:

$$p_G\Big(\pi' \mid \hat{A}^\pi(s,\cdot), \pi, s\Big) \propto \underbrace{\mathcal{L}(\pi'; \hat{A}^\pi(s,\cdot), \pi, s)}_{\text{Utility factor}} \underbrace{p(\pi' \mid \pi, s)}_{\text{Prior}}. \tag{7}$$

Modeling the Prior. The current policy π distills the behavior learned so far. Absent additional information, we favor policies close to π, modeled using a KL divergence with strength $\beta > 0$:

$$p(\pi' \mid \pi, s) \propto \exp\left(-\beta_0 \,\mathrm{KL}(\pi'(\cdot|s) \,\|\, \pi(\cdot|s))\right). \tag{8}$$

Modeling the Utility Factor. We model the evaluation error as an unbiased heteroscedastic Gaussian:

$$\hat{Q}^\pi(s,a) = Q^\pi(s,a) + \epsilon(s,a), \qquad \epsilon(s,a) \sim \mathcal{N}\Big(0, \sigma^2_{\hat{Q}^\pi}(s,a)\Big), \tag{9}$$

independent across states and actions. The modeling choice of heteroscedastic unbiased Gaussian error is motivated by more than analytical tractability. Hui et al. (2023) motivate Gaussian TD error through the Central Limit Theorem and empirically find a moment-matched heteroscedastic Gaussian to provide a suitable approximation. Classical MCTS (Kocsis & Szepesvári, 2006) was designed around unbiased sample means of bounded Monte Carlo returns, which are asymptotically Gaussian. Despite the DNN-based evaluation used in modern MCTS (Silver et al., 2018; Schrittwieser et al., 2020), MCTS remains highly successful. In Appendix F.2, we similarly find that the residual errors in our MCTS experiments are reasonably modeled by a heteroscedastic Gaussian. We discuss the modeling assumptions further in Section 4.2.

The uncertain advantage $\hat{A}^\pi(s,a) = \hat{Q}^\pi(s,a) - \sum_{b\in\mathcal{A}} \pi(b|s)\hat{Q}^\pi(s,b)$ therefore satisfies the action-wise observation model:

$$\hat{A}^\pi(s,a) | A^\pi(s,a), \pi, s \sim \mathcal{N}\Big(A^\pi(s,a), \underbrace{(1-\pi(a|s))^2\sigma^2_{\hat{Q}^\pi}(s,a) + \sum_{b\neq a}\pi(b|s)^2\sigma^2_{\hat{Q}^\pi}(s,b)}_{=:\sigma^2_{\hat{A}^\pi}(s,a)}\Big). \tag{10}$$

For analytical tractability, we model the action-wise advantage prior independently across actions as not favoring any action over another $A^\pi(s,a) \sim \mathcal{N}(0,\tau^2)$. Applying Bayes' rule to the advantage likelihood and prior using Normal–Normal conjugacy gives the action-wise posterior:

$$p\Big(A^\pi(s,a) \mid \hat{A}^\pi(s,a), \pi, s\Big) = \mathcal{N}\Big(\frac{\tau^2}{\tau^2 + \sigma^2_{\hat{A}^\pi}(s,a)}\hat{A}^\pi(s,a), \frac{\tau^2\sigma^2_{\hat{A}^\pi}(s,a)}{\tau^2 + \sigma^2_{\hat{A}^\pi}(s,a)}\Big). \tag{11}$$

Equations 10 and 11 give action-wise posteriors conditioned on $\hat{A}^\pi(s,a)$. Although the Q-value errors are independent across actions, the advantage errors are correlated through their shared dependence on $\hat{Q}^\pi$. To combine the action-wise posteriors tractably, we adopt the approximate joint posterior across all actions:

$$\tilde{p}\Big(A^\pi(s,\cdot) \mid \hat{A}^\pi(s,\cdot), \pi, s\Big) := \prod_{a\in\mathcal{A}} p\Big(A^\pi(s,a) \mid \hat{A}^\pi(s,a), \pi, s\Big). \tag{12}$$

For a candidate policy π', we define its utility relative to π as $A^{\pi',\pi}(s) := \sum_{a\in\mathcal{A}} \pi'(a \mid s)A^\pi(s,a)$, the maximization objective of greedification. Following the widely used exponential transformation of utility in decision models (Train, 2009; Bissiri et al., 2016; Levine, 2018), we map this utility to an exponential score. Because the true advantage vector $A^\pi(s,\cdot)$ is unobserved, the utility factor entering the generalized posterior in Equation 7 is obtained by marginalizing this score over the approximate joint advantage posterior, where $\lambda > 0$ controls the scale of the exponentiated utility:

$$\mathcal{L}\Big(\pi'; \hat{A}^\pi(s,\cdot), \pi, s\Big) := \mathbb{E}_{A^\pi(s,\cdot)\sim\tilde{p}(\cdot|\hat{A}^\pi(s,\cdot),\pi,s)}\left[\exp\Big(\lambda A^{\pi',\pi}(s)\Big)\right]. \tag{13}$$

Evaluating this expectation under the factorized Gaussian posterior introduces quadratic terms $\lambda^2(\pi'(a \mid s))^2$ through the Gaussian moment-generating function, precluding a simple closed-form solution to the resulting MAP. To retain tractability, we adopt a certainty-equivalent approximation (van de Water & Willems, 1981), replacing each uncertain advantage by its posterior mean. We denote the certainty-equivalent approximation to the uncertainty-marginalized utility factor by:

$$\mathcal{L}_{\mathrm{CE}}\Big(\pi'; \hat{A}^\pi(s,\cdot), \pi, s\Big) := \exp\left(\lambda \sum_{a\in\mathcal{A}} \pi'(a|s)\frac{\tau^2}{\tau^2 + \sigma^2_{\hat{A}^\pi}(s,a)}\hat{A}^\pi(s,a)\right) \approx \mathcal{L}\Big(\pi'; \hat{A}^\pi(s,\cdot), \pi, s\Big). \tag{14}$$

The corresponding tractable generalized posterior is

$$p_{\mathrm{CE}}\Big(\pi' \mid \hat{A}^\pi(s,\cdot),\pi,s\Big) \propto \mathcal{L}_{\mathrm{CE}}\Big(\pi';\hat{A}^\pi(s,\cdot),\pi,s\Big)\, p(\pi' \mid \pi,s). \tag{15}$$

Only the ratio $\beta = \beta_0/\lambda$ affects the MAP solution, which we use in the rest of the derivation for simplicity of notation. The log-posterior then induces the following tractable decision objective:

$$\mathcal{J}_{\mathrm{CE}}\Big(\pi';\hat{A}^\pi(s,\cdot),\pi,s\Big) := \sum_{a\in\mathcal{A}} \pi'(a \mid s)\frac{\tau^2}{\tau^2+\sigma^2_{\hat{A}^\pi}(s,a)}\hat{A}^\pi(s,a) - \beta D_{\mathrm{KL}}(\pi'(\cdot|s)\,\|\,\pi(\cdot|s))\,. \tag{16}$$

We define $\mathcal{I}_{\mathrm{opt}}$ as the operator which solves this decision problem:

$$\mathcal{I}_{\mathrm{opt}}(\pi,\hat{Q}^\pi)(\cdot|s) := \arg\max_{\pi'\in\Delta(\mathcal{A})} \mathcal{J}_{\mathrm{CE}}\Big(\pi';\hat{A}^\pi(s,\cdot),\pi,s\Big)\,. \tag{17}$$

Assuming that π has full support over $\mathcal{A}$, the objective in Equation 16 is strictly concave in $\pi'(\cdot \mid s)$ and therefore has a unique global maximizer:

$$\mathcal{I}_{\mathrm{opt}}(\pi,\hat{Q}^\pi)(a|s) \propto \pi(a|s)\exp\left(\frac{\tau^2}{\big(\tau^2+\sigma^2_{\hat{A}^\pi}(s,a)\big)\beta}\hat{A}^\pi(s,a)\right) \tag{18}$$

$$= \pi(a|s)\exp\left(\frac{\hat{A}^\pi(s,a)}{\eta(s,a)}\right), \quad \text{where} \quad \eta(s,a) := \frac{\big(\tau^2+\sigma^2_{\hat{A}^\pi}(s,a)\big)\beta}{\tau^2}. \tag{19}$$

$\mathcal{I}_{\mathrm{opt}}$ is the unique MAP optimizer of the certainty-equivalent generalized posterior (Equation 15), which itself approximates the original posterior formulated for the decision objective (Equation 7). Equivalently, $\mathcal{I}_{\mathrm{opt}}$ is the optimal operator with respect to the derived objective (Equation 16). In the no-uncertainty setting, where $\forall a \in \mathcal{A} : \sigma^2_{\hat{A}^\pi}(s,a) = 0$ and $\beta \to 0$, $\mathcal{I}_{\mathrm{opt}}$ reduces to $\mathcal{I}_{\arg\max}$. For detailed derivations see Appendix B.

4.2 Theoretical Analysis

We proceed to establish that $\mathcal{I}_{\mathrm{opt}}$ is sound for driving the iterative learning process, in a setup used for analysis of the underlying learning behavior of RL algorithms (Tsitsiklis, 2002; Oren et al., 2025b) (GPI, Algorithm 1 in Appendix D). We begin by showing that $\mathcal{I}_{\mathrm{opt}}$ satisfies *greedification*:

Theorem 3. *For any $\beta > 0, \tau^2 > 0, \sigma^2_{\hat{Q}^\pi} \geq 0$ and policy π with full support over $\mathcal{A}$, $\mathcal{I}_{opt}(\pi,\hat{Q}^\pi)$ is a greedification operator (Definition 2).*

Intuition: $\mathcal{I}_{\mathrm{opt}}$ exclusively *increases* the unnormalized log-probability of *advantage-positive* and *decreases* the unnormalized log-probability of *advantage-negative* actions, which guarantees greedification (Danihelka et al., 2022). See Appendix A.5 for full proof.

Using the greedification property, we show that $\mathcal{I}_{\mathrm{opt}}$ is sufficient for convergence:

Theorem 4 (Convergence)**.** *Let $\beta > 0$ and $\tau^2 > 0$, π_0 have full support, the evaluation errors be fresh and independent conditional on the history $\mathcal{F}_i$: $\epsilon_i(s,a) \mid \mathcal{F}_i \sim \mathcal{N}(0,\sigma^2_{Q_i}(s,a))$, where the corresponding variances are supplied to $\mathcal{I}_{\mathrm{opt}}$ and satisfy $\sigma^2_{Q_i}(s,a) \to 0$. Then, in the finite state, action and horizon GPI process (Algorithm 1 with its $\delta \to 0$):*

$$V^{\pi_i} \xrightarrow[i\to\infty]{\text{a.s.}} V^*, \qquad \sum_{a\notin\arg\max_b Q^*(s,b)} \pi_i(a|s) \xrightarrow[i\to\infty]{\text{a.s.}} 0, \quad \forall s. \tag{20}$$

Intuition: $\mathcal{I}_{opt}$ satisfies a *Limit-Sufficient Greedification* condition, established by Oren et al. (2025b) to induce convergence, almost surely under vanishing evaluation noise. See Appendix A.6 for proof.

Remark. Optimality under approximate evaluation is necessarily relative to a decision formulation: different objectives, likelihoods, priors, utilities, or approximations can yield different operators. Within this family of possible formulations, we choose modeling assumptions and approximations that yield a practical operator which accounts for evaluation uncertainty, is inexpensive to compute, and admits theoretical analysis (Theorems 3 and 4). Our contribution is to instantiate an explicit, informed approach to operator design and solve the resulting tractable objective exactly. Possible interactions between a policy improvement operator which accounts for evaluation uncertainty and exploration are discussed in Appendix B.7.

5 Connections to Prior Work

$\mathcal{I}_{\text{opt}}$ generalizes the widely used shift-invariant $\mathcal{I}_{reg}$ operator parameterized by η:

$$\mathcal{I}_{reg}(\pi, Q)(a|s) \propto \pi(a|s) \exp\left(Q(s,a)/\eta\right) \propto \pi(a|s) \exp\left(A(s,a)/\eta\right).$$

$\mathcal{I}_{reg}$ itself emerges independently from several seminal directions: ***Regularized Policy Optimization*** (Neu et al., 2017); ***Control as Inference*** and ***"Optimal Control"*** (Levine, 2018; Kappen et al., 2012), which interpret $\pi'(s) = \mathcal{I}_{reg}(\pi, Q^{\pi'}_{soft})(s)$ as a "soft-optimal" policy and motivate SAC (Haarnoja et al., 2018), MPO (Abdolmaleki et al., 2018) and TDMPC (Hansen et al., 2022; 2024); The ***Natural Policy Gradient*** (Kakade, 2001), ***Trust Region Policy Optimization*** (Schulman et al., 2015) and ***Mirror Descent*** (Tomar et al., 2022) whose improvement steps have all been shown to connect to $\mathcal{I}_{reg}$ (Neu et al., 2017; Xiao, 2022). For additional detail see Appendix C.

In view of $\mathcal{I}_{\text{opt}}$, the regularization term used by Regularized Policy Optimization induces a prior over candidate policies. From the perspective of Control as Inference, $\mathcal{I}_{\text{reg}}$ is most directly interpreted as a form of optimal ***improvement***, rather than as approximating a globally optimal ***policy***. Moreover, $1/\eta(s,a)$ can be interpreted as an action-dependent step size which accounts for the uncertainty in the improvement signal. Conversely, when we do not have any information about the evaluation error we can model the uncertainty as uniformly bounded by a constant, use $\sigma^2_{\hat{A}}(s,a) = \sigma^2$ conservatively, reducing $\eta(s,a)$ to η and $\mathcal{I}_{\text{opt}}$ to $\mathcal{I}_{reg}$, which thus can be viewed as optimal with respect to the same decision objective as $\mathcal{I}_{\text{opt}}$, when we are uninformed about error.

Gumbel-AlphaZero/MuZero (GA/MZ) (Danihelka et al., 2022) use $\mathcal{I}_{reg}$ within MCTS, where the coefficient $1/\eta(s)$ increases with $N(s,a)$, the number of evaluations at each node in the search tree: $1/\eta(s) = (c_{\text{visit}} + \max_a N(s,a))c_{\text{scale}}$. This heuristic mechanism is inspired by the original MCTS assumptions which induce $N(s,a) \propto 1/\sigma^2_{\hat{Q}^\pi}(s,a)$ and increases as evaluation error reduces (Kocsis & Szepesvári, 2006). Rather than using $N(s,a)$ directly which does not guarantee policy improvement, GA/MZ use $\max_a N(s,a) \propto \max_a 1/\sigma^2_{\hat{Q}^\pi}(s,a)$: a per-***node*** surrogate for a quantity that is inherently per-***action***. $\mathcal{I}_{\text{opt}}$ can circumvent this heuristic and use $N(s,a)$ per action directly.

Several previous works incorporate uncertainty into policy optimization for specific algorithms or sources of uncertainty. POMBU (Zhou et al., 2020) uses Q-value uncertainty to make PPO updates more conservative. Queeney et al. (2021) (UA-TRPO) make TRPO robust in the estimated policy gradient to uncertainty sourced from finite sampling. UWAC (Wu et al., 2021), specifically addressing offline RL, uses epistemic uncertainty to down-weigh the contribution of out-of-distribution predictions in actor-critic training. Hui et al. (2023) model Q-function evaluation error probabilistically and derive a modified Q-learning objective. In contrast, we study policy improvement at the operator level rather than within a particular algorithmic framework. We formulate greedification under uncertain evaluation as an explicit optimization problem and derive an operator that is optimal for the resulting uncertainty-aware objective under the stated probabilistic modeling and approximation choices. The operator admits convergence guarantees in the analyzed GPI setting and can be incorporated modularly into algorithms with different policy-improvement mechanisms (see Section 6).

6 Experiments

$\mathcal{I}_{\text{opt}}$ exactly solves the decision objective derived in Equation 16. Our experiments investigate whether optimizing this local decision objective indeed facilitates improvement in the broader iterative learning process. We ask two questions. First, *does $\mathcal{I}_{\text{opt}}$ improve sample efficiency in a controlled Policy Iteration setting that matches its evaluation assumptions*? Second, *do $\mathcal{I}_{\text{opt}}$ and its gradient-based practical approximations improve established RL algorithms when uncertainty must be estimated during training*? We study these questions across Dynamic Programming, model-based and model-free RL, discrete and continuous actions, and online and offline learning.

$\mathcal{I}_{\text{opt}}$'s hyperparameters were tuned per baseline algorithm it was incorporated into and in a similar manner in which each baseline algorithm was itself tuned. For additional details see Appendix G.1.

Policy Iteration (Grid MDPs). We compare Policy Iteration algorithms with different policy improvement operators $\mathcal{I}$ (Algorithm 1 with $k \to \infty$) with evaluation error which coincides with the assumptions under which $\mathcal{I}_{\text{opt}}$ is designed. At each iteration i the exact evaluation Q^{π_i} is

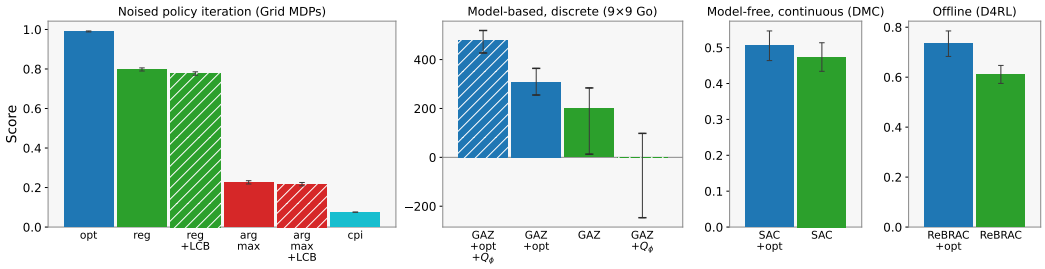

Figure 1: **Noised policy iteration:** normalised $V^\pi(s_0)$ at iteration 40 over 20 grid environments, 95% Gaussian CI. **Model-based, discrete:** Bayes Elo at 34M training frames, mean, min. and max. of 5 training seeds in 9x9 Go. **Model-free, continuous:** mean normalised returns at 1M steps on 10 DeepMind Control Suite tasks, mean and 95% Gaussian CI. **Offline:** normalised return at 1M steps on 4 D4RL medium-replay tasks, mean and 95% Gaussian CI.

computed analytically and corrupted by heteroscedastic noise $\hat{Q}^{\pi_i}(s,a) \sim \mathcal{N}(Q^{\pi_i}(s,a), \sigma^2_{Q_i}(s,a))$. The variances $\sigma^2_{Q_i}(s,a)$ are sampled each iteration from a decaying schedule. We compare $\mathcal{I}_{\text{opt}}$ to the following standard operators: $\mathcal{I}_{\arg\max}$ (standard policy iteration), $\mathcal{I}_{\text{reg}}$ and $\mathcal{I}_{\text{cpi}}$ (Pirotta et al., 2013), a variant of Conservative Policy Iteration (Kakade & Langford, 2002), a method designed for Policy Iteration in the presence of approximate evaluation. We include pessimistic variants of $\mathcal{I}_{\arg\max}$ and $\mathcal{I}_{\text{reg}}$ that use the lower-confidence bound (LCB) $\hat{Q}^{\pi_i}(s,a) - \lambda_{\text{LCB}}\sqrt{\sigma^2_{Q_i}(s,a)}$ to test whether $\mathcal{I}_{\text{opt}}$'s uncertainty-awareness merely reproduces standard LCB-based pessimism.

$\mathcal{I}_{\text{opt}}$ outperforms every operator tested (Figure 1, left) in the presence of the evaluation error of the type for which it was designed to account. The effect of $\mathcal{I}_{\text{opt}}$ is clearly different from that of LCB-based pessimism, which does not contribute to performance in this setting. See Appendix E.1 for implementation and experimental details and Appendix F.1 for additional results including the learning curves, an investigation of the effect of the magnitude of the noise, sensitivity to the hyperparameters τ, β and an experiment with biased evaluation rather than unbiased.

Model-based discrete action (GAZ on board games). In Section 3, we showed how greedification connects to search for policy improvement. With this experiment, we circle back to this connection and evaluate $\mathcal{I}_{\text{opt}}$'s contribution to search-based policy improvement with the search operator GumbelMCTS used by the modern AlphaZero variant GumbelAlphaZero (GAZ), which remains a popular and well-performing RL algorithm (Danihelka et al., 2022; Wang et al., 2024; Oren et al., 2026b). As discussed in Section 5, it is natural to replace the heuristic $\eta(s)$ GAZ uses in $\mathcal{I}_{\text{reg}}$ with the theoretically motivated $\mathcal{I}_{\text{opt}}$. This enables the algorithm to account for the variance using $\sigma^2_{Q_i}(s_t,a) \propto 1/N(s,a)$ at each node *and action* (s_t,a) independently, as was enabled by the PUCT search policy (Rosin, 2011; Silver et al., 2016) which originally inspired using $\mathcal{I}_{\text{reg}}$ in MCTS (Grill et al., 2020). We include a comparison to VanillaAZ with PUCT for reference in Figure 5 in Appendix F.2.

$\mathcal{I}_{\text{reg}}$ requires an estimate of $Q^\pi(s,a)$ for each state and action, which traditional MCTS architectures do not provide. GumbelMCTS instead uses the *completed Q-values* mechanism, which substitutes a prediction of $V^\pi(s)$ for $Q^\pi(s,a)$ for unsearched actions, composed of the predictions of all searched actions weighted under π. This substitution removes action-specific information, since every unsearched action receives the same value prediction. When action-wise prediction errors contain independent or weakly correlated components, this averaging reduces their variance. Moreover, actions with greater policy support are generally searched or played, and consequently trained, more frequently; weighting by π therefore tends to emphasize better-supported estimates while assigning less weight to weakly supported, higher-error actions. Consistent with this reasoning, using the empirical proxy described in Appendix F.3, we observe lower estimated error variance for the V^π-based predictions in our experiments. We therefore hypothesize that $\mathcal{I}_{\text{opt}}$, which is designed to exploit action-specific Q_ϕ predictions while accounting for their estimated uncertainty, will benefit more than baseline GAZ from incorporating a Q_ϕ critic. Such a critic is inexpensive to incorporate as a small additional head on the shared-body DNN of GAZ.

Figure 1 (second from the left) reports Bayes Elo (Coulom, 2010) on 9×9 Go after 34M frames of training, averaged over five training seeds per agent. $\mathcal{I}_{\text{opt}}$ benefits substantially from a Q_ϕ critic, while the baseline's performance *degrades* with a Q_ϕ critic. $\mathcal{I}_{\text{opt}}$ mean Elo is 107.8 higher than the

baseline's, even without the addition of the Q_ϕ critic. With Q_ϕ the difference increases to 277 Elo. See Appendix E.2 for additional experimental and implementation details and Appendix F.2 for additional results, including per-seed results, an investigation of the error behavior in MCTS and a fixed-network comparison along compute budgets M which includes the additional board game chess and the additional search policy PUCT. In the fixed-network comparison, GumbelMCTS with $\mathcal{I}_{\text{opt}}$ and the Q_ϕ critic achieves the highest playing strength among all tested GumbelMCTS and PUCT variants at every evaluated search budget in both 9×9 Go and chess (Figure 5).

Model free classical continuous control (SAC on DMC). We incorporate $\mathcal{I}_{\text{opt}}$ into the classical continuous-control RL algorithm SAC (Haarnoja et al., 2018). In practical implementations, SAC approximates a projection of the target soft distribution $\mathcal{I}_{reg}(\mu, Q_{soft})(a|s)$ (where μ is the uniform policy and Q_{soft} is the soft Q-value function) to the parameterized policy using empirical gradient descent. We incorporate $\mathcal{I}_{opt}$ into this traditional implementation of SAC similarly, with a gradient based approximation of $\mathcal{I}_{opt}$ (see Appendix E.3 for details). To estimate σ_Q^2 we use the rudimentary and cheap uncertainty estimation already provided by SAC's two critic value ensemble. Results are aggregated across 10 classical continuous control environments from the DeepMind Control suite (DMC, Tunyasuvunakool et al., 2020). Figure 1 (second from right) reports normalised returns after 1M steps. Even in the presence of the rudimentary uncertainty estimation provided by the value ensemble, which was not designed for explicit uncertainty estimation, the gradient-based approximation of $\mathcal{I}_{\text{opt}}$ is able to provide performance improvement over baseline SAC. See Appendix E.3 for implementation and experimental details and Appendix F.4 for per-environment results.

Offline RL (ReBRAC on D4RL). We evaluate $\mathcal{I}_{\text{opt}}$'s contribution to performance in offline RL, incorporating it into the recent ReBRAC algorithm (Tarasov et al., 2023) similarly using the two critic value ensemble (see Appendix E.4 for details). We evaluate on the medium-replay datasets of the D4RL Gym-MuJoCo suite (Fu et al., 2020) using four locomotion environments: *halfcheetah*, *hopper*, *walker2d* and *ant* with the *medium-replay* datasets. Scores are D4RL-normalized, where 0 corresponds to a random policy and 100 to an expert (SAC) policy. Each agent trains for 1M gradient steps. Figure 1 (right) reports the return of the final policy at the end of training. The gradient-based approximation of $\mathcal{I}_{\text{opt}}$ improves on the baseline in this domain as well. See Appendix E.4 for implementation and experimental details and Appendix F.5 for per-environment results.

7 Conclusions and Future Work

In this work, we took steps towards practical RL algorithms driven by optimal policy improvement operators. We characterized optimal policy improvement, showing that optimal improvement is itself equivalent to solving MDPs and fundamentally connecting optimal policy improvement to planning with an explicit and implicit model. To solve the MDP in the explicit or implicit model, practical methods rely on *greedification*, which lead us to focus on optimal greedification under practical constraints. We formulated greedification under approximate evaluation as probabilistic decision-making under uncertainty and derived a tractable decision objective. Solving for this objective, we derived $\mathcal{I}_{\text{opt}}$, a novel greedification operator which is optimal with respect to the probabilistic decision objective. We established convergence for algorithms driven by $\mathcal{I}_{\text{opt}}$ in a Generalized Policy Iteration setting and shown that $\mathcal{I}_{\text{opt}}$ generalizes a popular greedification operator in literature, $\mathcal{I}_{\text{reg}}$, extending it with explicit accounting for evaluation uncertainty. Empirically, $\mathcal{I}_{\text{opt}}$ and its practical gradient-based approximations improved aggregate performance over the respective GumbelAlphaZero, SAC, and ReBRAC baselines in board games, online continuous control, and offline continuous control, respectively. In a noised Policy Iteration experiment which coincides with the assumptions under which $\mathcal{I}_{\text{opt}}$ was designed, the novel operator outperformed all other tested operators. $\mathcal{I}_{\text{opt}}$ presents a new, attractive choice for greedification, unified across different algorithms, and lays a path towards optimal policy improvement through the design of operators which are optimal with respect to practical RL constraints.

AI use statement

In this work, we used generative AI tools to assist with writing and checking proofs, developing and implementing experimental code, analyzing and plotting results, and supporting literature research. We did not use generative AI for any other task requiring disclosure. We reviewed all AI-assisted

content and verified and tested all AI-generated code. We take full responsibility for the final manuscript, including all text, claims, code, and other artifacts produced with AI assistance.

REFERENCES

Abbas Abdolmaleki, Jost Tobias Springenberg, Yuval Tassa, Remi Munos, Nicolas Heess, and Martin Riedmiller. Maximum a posteriori policy optimisation. *arXiv preprint arXiv:1806.06920*, 2018.

Kenneth J Arrow. *Social choice and individual values*, volume 12. Yale university press, 2012.

Leemon Baird. Residual algorithms: Reinforcement learning with function approximation. In *Machine Learning, Proceedings of the Twelfth International Conference on Machine Learning*, 1995. doi: 10.1016/B978-1-55860-377-6.50013-X.

Gary S Becker. *The economic approach to human behavior*, volume 803. University of Chicago press, 1976.

Richard Bellman. A Markovian Decision Process. *Journal of Mathematics and Mechanics*, 6(5): 679–684, 1957.

Dimitri P Bertsekas. Neuro-dynamic programming. In *Encyclopedia of optimization*, pp. 1–6. Springer, 2025.

Pier Giovanni Bissiri, Chris C. Holmes, and Stephen G. Walker. A general framework for updating belief distributions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 78 (5):1103–1130, 2016. doi: 10.1111/rssb.12158.

Alan Chan, Hugo Silva, Sungsu Lim, Tadashi Kozuno, A. Rupam Mahmood, and Martha White. Greedification Operators for Policy Optimization: Investigating Forward and Reverse KL Divergences. *Journal of Machine Learning Research*, 23(253):1–79, 2022.

Rémi Coulom. Bayesian Elo rating. `https://www.remi-coulom.fr/Bayesian-Elo/`, 2010.

Ivo Danihelka, Arthur Guez, Julian Schrittwieser, and David Silver. Policy improvement by planning with Gumbel. In *The Tenth International Conference on Learning Representations*, 2022.

Joery A. de Vries, Jinke He, Yaniv Oren, and Matthijs T. J. Spaan. Trust-Region Twisted Policy Improvement. In *Forty-second International Conference on Machine Learning*, 2025.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Chao Gao, Martin Müller, and Ryan Hayward. Three-head neural network architecture for monte carlo tree search. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, pp. 1814–1820. International Joint Conferences on Artificial Intelligence Organization, 2018. doi: 10.24963/ijcai.2018/523.

Jakob Gawlikowski, Cedrique Rovile Njieutcheu Tassi, Mohsin Ali, Jongseok Lee, Matthias Humt, Jianxiang Feng, Anna Kruspe, Rudolph Triebel, Peter Jung, Ribana Roscher, Muhammad Shahzad, Wen Yang, Richard Bamler, and Xiao Xiang Zhu. A survey of uncertainty in deep neural networks. *Artificial Intelligence Review*, 56:1513–1589, 2023. doi: 10.1007/s10462-023-10562-9.

Jean-Bastien Grill, Florent Altché, Yunhao Tang, Thomas Hubert, Michal Valko, Ioannis Antonoglou, and Remi Munos. Monte-Carlo Tree Search as Regularized Policy Optimization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pp. 3769–3778. PMLR, 2020.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pp. 1861–1870. PMLR, 2018.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Peter Hall and Christopher C. Heyde. *Martingale Limit Theory and Its Application*. Academic Press, New York, 1980.

Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, Robust World Models for Continuous Control. In *International Conference on Learning Representations*, 2024.

Nicklas A Hansen, Hao Su, and Xiaolong Wang. Temporal Difference Learning for Model Predictive Control. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, pp. 8387–8406. PMLR, 2022.

Shengyi Huang, Rousslan Fernand Julien Dossa, Chang Ye, Jeff Braga, Dipam Chakraborty, Kinal Mehta, and João G.M. Araújo. Cleanrl: High-quality single-file implementations of deep reinforcement learning algorithms. *Journal of Machine Learning Research*, 23(274):1–18, 2022. URL `http://jmlr.org/papers/v23/21-1342.html`.

David Yu-Tung Hui, Aaron C Courville, and Pierre-Luc Bacon. Double gumbel q-learning. *Advances in Neural Information Processing Systems*, 36:2580–2616, 2023.

Eyke Hüllermeier and Willem Waegeman. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine Learning*, 110(3):457–506, 2021.

Sham M. Kakade. A natural policy gradient. In *Advances in Neural Information Processing Systems 14*, 2001.

Sham M. Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*, 2002.

Hilbert J Kappen, Vicenç Gómez, and Manfred Opper. Optimal control as a graphical model inference problem. *Machine learning*, 87(2):159–182, 2012.

Zohar Karnin, Tomer Koren, and Oren Somekh. Almost Optimal Exploration in Multi-Armed Bandits. In *Proceedings of the 30th International Conference on Machine Learning*, volume 28, pp. 1238–1246. PMLR, 2013.

Toshinori Kitamura and Ryo Yonetani. Shinrl: A library for evaluating rl algorithms from theoretical and practical perspectives. *arXiv preprint arXiv:2112.04123*, 2021.

Levente Kocsis and Csaba Szepesvári. Bandit based Monte-Carlo planning. In Johannes Fürnkranz, Tobias Scheffer, and Myra Spiliopoulou (eds.), *17th European Conference on Machine*, 2006. doi: 10.1007/11871842_29.

Vijay Konda and John Tsitsiklis. Actor-critic algorithms. *Advances in neural information processing systems*, 12, 1999.

Sotetsu Koyamada, Shinri Okano, Soichiro Nishimori, Yu Murata, Keigo Habara, Haruka Kita, and Shin Ishii. Pgx: Hardware-accelerated parallel game simulators for reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2023.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, 2017.

Sergey Levine. Reinforcement Learning and Control as Probabilistic Inference: Tutorial and Review. arXiv:1805.00909, 2018.

R Duncan Luce et al. *Individual choice behavior*, volume 4. Wiley New York, 1959.

Aymen Al Marjani and Alexandre Proutière. Adaptive sampling for best policy identification in markov decision processes. In *Proceedings of the 38th International Conference on Machine Learning*, Proceedings of Machine Learning Research, 2021.

Daniel McFadden. Conditional logit analysis of qualitative choice behavior. *Frontier in Econometrics*, 1973.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Gergely Neu, Anders Jonsson, and Vicenç Gómez. A unified view of entropy-regularized markov decision processes. *arXiv preprint arXiv:1705.07798*, 2017.

Yaniv Oren, Viliam Vadocz, Matthijs T. J. Spaan, and Wendelin Boehmer. Epistemic Monte Carlo Tree Search. In *The Thirteenth International Conference on Learning Representations*, 2025a.

Yaniv Oren, Moritz A Zanger, Pascal R Van der Vaart, Mustafa Mert Çelikok, Matthijs TJ Spaan, and Wendelin Boehmer. Value Improved Actor Critic Algorithms. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025b.

Yaniv Oren, Joery A de Vries, Pascal R van der Vaart, Matthijs TJ Spaan, and Wendelin Böhmer. Twice sequential monte carlo for tree search. *ICML*, 2026a.

Yaniv Oren, Viliam Vadocz, Joery A de Vries, Wendelin Böhmer, Matthijs TJ Spaan, and Hendrik Baier. Pmcts: Particle monte carlo tree search for principled parallelized inference time scaling. *arXiv preprint arXiv:2605.08982*, 2026b.

Ian Osband, Daniel Russo, and Benjamin Van Roy. (more) efficient reinforcement learning via posterior sampling. In *27th Annual Conference on Neural Information Processing Systems*, 2013.

Matteo Pirotta, Marcello Restelli, Alessio Pecorino, and Daniele Calandriello. Safe policy iteration. In *Proceedings of the 30th International Conference on Machine Learning*, 2013.

James Queeney, Ioannis Ch. Paschalidis, and Christos G. Cassandras. Uncertainty-aware policy optimization: A robust, adaptive trust region approach. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021. doi: 10.1609/aaai.v35i11.17130.

Christopher D Rosin. Multi-armed bandits with episode context. *Annals of Mathematics and Artificial Intelligence*, 61(3):203–230, 2011.

Leonard J Savage. *The foundations of statistics*. Courier Corporation, 2012.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering Atari, Go, chess and shogi by planning with a learned model. *Nature*, 588 (7839):604–609, 2020. doi: 10.1038/s41586-020-03051-4.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. PMLR, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms. arXiv:1707.06347, 2017.

David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016. doi: 10.1038/nature16961.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*, 362(6419):1140–1144, 2018. doi: 10.1126/science.aar6404.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. A Bradford Book, 2nd edition, 2018.

Richard S Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

Richard S. Sutton, David McAllester, Satinder P. Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, volume 12, pp. 1057–1063, 2000.

Denis Tarasov, Vladislav Kurenkov, Alexander Nikulin, and Sergey Kolesnikov. Revisiting the minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 2023.

Sebastian Thrun and Anton Schwartz. Issues in using function approximation for reinforcement learning. In *Proceedings of the 1993 connectionist models summer school*, pp. 255–263. Psychology Press, 2014.

Manan Tomar, Lior Shani, Yonathan Efroni, and Mohammad Ghavamzadeh. Mirror descent policy optimization. In *The Tenth International Conference on Learning Representations*, 2022.

Kenneth E. Train. *Discrete Choice Methods with Simulation*. Cambridge University Press, 2 edition, 2009. doi: 10.1017/CBO9780511805271.

John N. Tsitsiklis. On the convergence of optimistic policy iteration. *J. Mach. Learn. Res.*, 3:59–72, 2002.

John N. Tsitsiklis and Benjamin Van Roy. An analysis of temporal-difference learning with function approximation. *IEEE Trans. Autom. Control.*, 42(5):674–690, 1997. doi: 10.1109/9.580874.

Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020. ISSN 2665-9638.

Henk van de Water and Jan C. Willems. The certainty equivalence property in stochastic control theory. *IEEE Transactions on Automatic Control*, 26(5):1080–1087, 1981. doi: 10.1109/TAC.1981.1102781.

Hado van Hasselt. Double q-learning. In John D. Lafferty, Christopher K. I. Williams, John Shawe-Taylor, Richard S. Zemel, and Aron Culotta (eds.), *Advances in Neural Information Processing Systems*, 2010.

Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In Dale Schuurmans and Michael P. Wellman (eds.), *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, 2016. doi: 10.1609/AAAI.V30I1.10295.

Hado van Hasselt, Yotam Doron, Florian Strub, Matteo Hessel, Nicolas Sonnerat, and Joseph Modayil. Deep reinforcement learning and the deadly triad. *CoRR*, abs/1812.02648, 2018. URL http://arxiv.org/abs/1812.02648.

John von Neumann and Oskar Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 2007. ISBN 978-0-691-13061-3.

Shengjie Wang, Shaohuai Liu, Weirui Ye, Jiacheng You, and Yang Gao. Efficientzero V2: mastering discrete and continuous control with limited data. In *Forty-first International Conference on Machine Learning*, Proceedings of Machine Learning Research, 2024.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3):279–292, 1992.

B. L. Welch. The generalization of 'student's' problem when several different population variances are involved. *Biometrika*, 34(1–2):28–35, 1947. doi: 10.1093/biomet/34.1-2.28.

Yue Wu, Shuangfei Zhai, Nitish Srivastava, Joshua M. Susskind, Jian Zhang, Ruslan Salakhutdinov, and Hanlin Goh. Uncertainty weighted actor-critic for offline reinforcement learning. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 11319–11328. PMLR, 2021.

Lin Xiao. On the convergence rates of policy gradient methods. *Journal of Machine Learning Research*, 23(282):1–36, 2022.

Kenny John Young, Aditya A. Ramesh, Louis Kirsch, and Jürgen Schmidhuber. The benefits of model-based generalization in reinforcement learning. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 40254–40276. PMLR, 2023.

Qi Zhou, HouQiang Li, and Jie Wang. Deep model-based reinforcement learning via estimated uncertainty and conservative policy optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020. doi: 10.1609/aaai.v34i04.6177.

APPENDIX CONTENTS

A Proofs **16**

A.1 Optimal policy improvement as solving an induced MDP 16

A.2 Solving sub-MDPs is a lower bound on optimal policy improvement 17

A.3 Even when $|S| = 1$, greedification is not generally optimal policy improvement . . 19

A.4 Optimal greedification under exact evaluation 20

A.5 Greedification 21

A.6 Convergence 21

B Derivations and Discussion **25**

B.1 Derivation of the advantage posterior in the likelihood 25

B.2 Modeling the Utility Factor using Rational Choice and Expected Utility Theory . . 26

B.3 The utility as the local improvement of π' over π 27

B.4 Derivation of the certainty-equivalent MAP objective 27

B.5 Derivation of the solution for the certainty-equivalent MAP objective 27

B.6 Modeling evaluation uncertainty with the Normal distribution 28

B.7 $\mathcal{I}_{opt}$ and exploration 29

C Connections to prior work in more detail **30**

D Pseudo Code **31**

E Experimental and Implementation Details **31**

E.1 Policy Iteration Experiments 31

E.2 Discrete-Action Model Based RL (GumbelAlphaZero on board games) 32

E.3 Classical Control Model-Free RL (SAC on DMC) 35

E.4 Offline RL (ReBRAC on D4RL) 36

F Additional Results **37**

F.1 Dynamic Programming additional results 37

F.2 Discrete-action model-based, additional results 39

F.3 Investigation of error in MCTS with DNNs 40

F.4 Learning curves on model-free online continuous control (SAC on DMC) 41

F.5 Offline RL: per-environment results 41

G Hyperparameters **42**

G.1 Hyperparameter tuning 42

G.2 Hyperparameter tables 43

A PROOFS

A.1 OPTIMAL POLICY IMPROVEMENT AS SOLVING AN INDUCED MDP

Let $S \subseteq \mathcal{S}$ denote the subset of states at which the policy is permitted to change, and let π denote the current policy. We construct an induced semi-Markov decision process (SMDP) $M^{\text{semi}}_{S,\pi}$ whose decision states are exactly the states in S.

Starting from a decision state $s_0 = s \in S$, the agent first selects an action $a_0 = a$. The environment then evolves according to the original transition dynamics P. Whenever the resulting trajectory leaves S, actions are subsequently sampled from the fixed policy π until the process first returns to S. Define this first-return time as

$$\tau := \inf\{t \geq 1 : s_t \in S\}. \tag{21}$$

If the process transitions directly from one state in S to another, then $\tau = 1$.

Each state-action pair (s, a) in the induced SMDP therefore corresponds to a temporally extended transition in the original MDP: take action a at s, and, whenever the process leaves S, follow π until the next state in S is reached.

The reward of this temporally extended transition is the expected discounted return accumulated before the next decision state:

$$R_{M^{\text{semi}}_{S,\pi}}(s, a) := \mathbb{E}\left[\sum_{t=0}^{\tau-1} \gamma^t R(s_t, a_t) \,\middle|\, s_0 = s,\ a_0 = a,\ \pi\right]. \tag{22}$$

Here and below, if the process never returns to S, we take $\tau = \infty$, so that the trajectory contributes its complete discounted return to Equation 22 and no continuation value.

The transition dynamics of the induced SMDP are described jointly by the next decision state and the duration of the temporally extended transition:

$$P_{M^{\text{semi}}_{S,\pi}}(s', k|s, a) := \Pr\left(s_\tau = s',\ \tau = k \,|\, s_0 = s,\ a_0 = a,\ \pi\right), \qquad s, s' \in S,\ k \geq 1. \tag{23}$$

Consequently, the Bellman optimality equation of $M^{\text{semi}}_{S,\pi}$ is

$$V^{*,\text{semi}}(s) = \max_{a \in \mathcal{A}} \left[R_{M^{\text{semi}}_{S,\pi}}(s, a) + \sum_{s' \in S} \sum_{k=1}^{\infty} P_{M^{\text{semi}}_{S,\pi}}(s', k|s, a)\gamma^k V^{*,\text{semi}}(s')\right]. \tag{24}$$

The factor γ^k accounts for the possibly variable duration of the temporally extended transition.

We can now re-describe Theorem 1 more formally.

Theorem 5. *Let*

$$\Pi_S(\pi) := \{\pi' \in \Pi : \pi'(\cdot|s) = \pi(\cdot|s),\ \forall s \notin S\} \tag{25}$$

denote the set of policies that may differ from π only on $S \subseteq \mathcal{S}$. Let $\pi^{*,\text{semi}}_S$ be an optimal policy of $M^{\text{semi}}_{S,\pi}$, and extend it to the original MDP by

$$\bar{\pi}(a|s) := \begin{cases} \pi^{*,\text{semi}}_S(a|s), & s \in S, \\ \pi(a|s), & s \notin S. \end{cases} \tag{26}$$

Then $\bar{\pi}$ is an optimal policy among all policies in $\Pi_S(\pi)$. In particular,

$$\forall s \in \mathcal{S}: \qquad V^{\bar{\pi}}(s) = \sup_{\pi' \in \Pi_S(\pi)} V^{\pi'}(s). \tag{27}$$

Equivalently, since $V^*(s) \geq V^{\pi'}(s)$ for every policy π',

$$\forall s \in \mathcal{S}: \qquad \bar{\pi} \in \arg\inf_{\pi' \in \Pi_S(\pi)} \left[V^*(s) - V^{\pi'}(s)\right]. \tag{28}$$

Thus, solving $M^{\text{semi}}_{S,\pi}$ is equivalent to optimal policy improvement under the constraint that the policy may be modified only on S.

Proof. Consider any policy $\pi' \in \Pi_S(\pi)$. Whenever the process is in a state $s \in S$, π' selects an action according to $\pi'(\cdot|s)$. After this action, if the trajectory leaves S, the policies π' and π coincide by definition of $\Pi_S(\pi)$. Hence, until the trajectory next reaches a state in S, its action distribution is exactly the one used in the construction of $M^{\text{semi}}_{S,\pi}$.

Therefore, every policy $\pi' \in \Pi_S(\pi)$ induces a policy on $M^{\text{semi}}_{S,\pi}$ given by its restriction to S. Conversely, every policy on $M^{\text{semi}}_{S,\pi}$ uniquely induces a policy in $\Pi_S(\pi)$ by using that policy on S and π on $\mathcal{S} \setminus S$.

By Equations 22 and 23, each temporally extended transition in $M^{\text{semi}}_{S,\pi}$ preserves both the discounted reward accumulated during the corresponding trajectory segment in the original MDP and the discounted continuation value upon returning to S. Thus, under the policy correspondence above,

$$V^{\pi'}_M(s) = V^{\pi'|_S}_{M^{\text{semi}}_{S,\pi}}(s), \qquad \forall s \in S. \tag{29}$$

It follows that an optimal policy of $M^{\text{semi}}_{S,\pi}$ maximizes the value in the original MDP among all policies that differ from π only on S, for every $s \in S$.

Finally, consider any state $s \notin S$. Every policy in $\Pi_S(\pi)$ follows the same policy π until either the process terminates or first reaches S. Therefore, the distribution of rewards and of the first state reached in S before that point is independent of the choice of policy within S. Since $\bar{\pi}$ maximizes the continuation value at every state in S, it also maximizes the value from every state $s \notin S$. Hence,

$$V^{\bar{\pi}}(s) = \sup_{\pi' \in \Pi_S(\pi)} V^{\pi'}(s), \qquad \forall s \in \mathcal{S}. \tag{30}$$

Therefore, solving $M^{\text{semi}}_{S,\pi}$ is equivalent to solving the original MDP under the constraint that the policy equals π on $\mathcal{S} \setminus S$, proving the result. □

A.2 Solving sub-MDPs is a lower bound on optimal policy improvement

Let $S \subseteq \mathcal{S}$ denote the subset of states at which the policy is permitted to change, and let π denote the current policy. We define the sub-MDP $M^{sub}_{S,\pi}$ by preserving the original dynamics within S and terminating trajectories whenever they leave S.

Specifically, for a transition (s, a, s') with $s \in S$ and $s' \in S$, the original transition probability and reward are preserved. For a transition with $s \in S$ and $s' \notin S$, the transition is made terminal and bootstrapped using the value of the current policy at the realized successor state s'. Thus, the value associated with such an exit transition is

$$R(s, a) + \gamma V^{\pi}(s'). \tag{31}$$

Importantly, this boundary value depends on the realized successor s', rather than on the unconditional state-action value $Q^{\pi}(s, a)$.

Equivalently, the Bellman optimality operator of $M^{sub}_{S,\pi}$ is, for every $s \in S$,

$$\left(\mathcal{T}^{sub}_{S,\pi} V\right)(s) := \max_{a \in \mathcal{A}} \left[R(s, a) + \gamma \sum_{s' \in S} P(s'|s, a) V(s') + \gamma \sum_{s' \notin S} P(s'|s, a) V^{\pi}(s') \right]. \tag{32}$$

Let $V^{*,sub}_{S,\pi}$ denote the unique fixed point of $\mathcal{T}^{sub}_{S,\pi}$.

The difference between $M^{sub}_{S,\pi}$ and the induced semi-MDP $M^{semi}_{S,\pi}$ of the previous section is therefore that $M^{semi}_{S,\pi}$ remains connected through the behavior of π outside S, whereas $M^{sub}_{S,\pi}$ terminates immediately upon leaving S and bootstraps from V^{π} at the boundary.

Consequently, constructing $M^{sub}_{S,\pi}$ does not require a transition model beyond transitions originating from state-action pairs in $S \times \mathcal{A}$. It does, however, require the boundary values $V^{\pi}(s')$ for successor states $s' \notin S$. In model-free RL, the former may correspond to transitions available in a replay buffer, while the latter may be supplied by a value estimator.

Let $\pi_S^{*,sub}$ denote an optimal policy of $M_{S,\pi}^{sub}$. We extend this policy to the original MDP as

$$\bar{\pi}(a|s) := \begin{cases} \pi_S^{*,sub}(a|s), & s \in S, \\ \pi(a|s), & s \notin S. \end{cases} \tag{33}$$

The corresponding policy-improvement operator is denoted by $\mathcal{I}_{\text{solve } M_{S,\pi}^{sub}}$.

We can now state Theorem 2 more formally.

Theorem 6. *Let*

$$\Pi_S(\pi) := \{\pi' \in \Pi : \pi'(\cdot|s) = \pi(\cdot|s),\ \forall s \notin S\}. \tag{34}$$

Then, for every $s \in S$,

$$\max_{a\in\mathcal{A}} Q^{\pi}(s,a) \ \le\ V_{S,\pi}^{*,sub}(s) \ \le\ V^{\bar{\pi}}(s) \ \le \sup_{\pi'\in\Pi_S(\pi)} V^{\pi'}(s). \tag{35}$$

If leaving S precludes returning to S, then, for every $s \in S$,

$$V_{S,\pi}^{*,\text{sub}}(s) = V^{\bar{\pi}}(s) = \sup_{\pi'\in\Pi_S(\pi)} V^{\pi'}(s). \tag{36}$$

Thus, under this condition, solving $M_{S,\pi}^{\text{sub}}$ is equivalent to optimal policy improvement constrained by S. Consequently, since $\sup_{\pi'\in\Pi_S(\pi)} V^{\pi'}(s) \le V^*(s)$,

$$\left|V^*(s) - \max_{a\in\mathcal{A}} Q^{\pi}(s,a)\right| \ge \left|V^*(s) - V^{\mathcal{I}_{\text{solve } M_{S,\pi}^{sub}}(\pi)}(s)\right| \ge \inf_{\pi'\in\Pi_S(\pi)} \left|V^*(s) - V^{\pi'}(s)\right|. \tag{37}$$

Thus, solving $M_{S,\pi}^{sub}$ yields an improvement that is bounded below by one-step greedification $\max_a Q^{\pi}(s,a)$ and above by optimal policy improvement under the constraint that the policy may be modified only on S.

Proof. First observe that restricting the original policy π to S yields exactly its original value on the sub-MDP. Indeed, for every $s \in S$,

$$V^{\pi}(s) = \sum_{a\in\mathcal{A}} \pi(a|s)\left[R(s,a) + \gamma \sum_{s'\in S} P(s'|s,a)V^{\pi}(s') + \gamma \sum_{s'\notin S} P(s'|s,a)V^{\pi}(s')\right]. \tag{38}$$

Hence $V^{\pi}|_S$ is the value of the policy $\pi|_S$ in $M_{S,\pi}^{sub}$, and therefore

$$V_{S,\pi}^{*,sub}(s) \ge V^{\pi}(s), \qquad \forall s \in S. \tag{39}$$

Applying the optimality operator of the sub-MDP to V^{π} gives

$$\left(\mathcal{T}_{S,\pi}^{sub} V^{\pi}\right)(s) = \max_{a\in\mathcal{A}} \left[R(s,a) + \gamma \sum_{s'\in S} P(s'|s,a)V^{\pi}(s') + \gamma \sum_{s'\notin S} P(s'|s,a)V^{\pi}(s')\right] \tag{40}$$

$$= \max_{a\in\mathcal{A}} Q^{\pi}(s,a). \tag{41}$$

Since $\mathcal{T}_{S,\pi}^{sub}$ is monotone and $V_{S,\pi}^{*,sub}$ is its optimal fixed point,

$$V_{S,\pi}^{*,sub}(s) \ge \left(\mathcal{T}_{S,\pi}^{sub} V^{\pi}\right)(s) = \max_{a\in\mathcal{A}} Q^{\pi}(s,a), \qquad \forall s \in S. \tag{42}$$

We next relate the value of the sub-MDP to the value obtained by extending its optimal policy to the original MDP according to Equation 33. Define

$$W(s) := \begin{cases} V_{S,\pi}^{*,sub}(s), & s \in S, \\ V^{\pi}(s), & s \notin S. \end{cases} \tag{43}$$

For $s \in S$, optimality of $\pi_S^{*,sub}$ in the sub-MDP gives

$$\left(\mathcal{T}^{\bar{\pi}} W\right)(s) = W(s). \tag{44}$$

For $s \notin S$, $\bar{\pi}$ coincides with π. Moreover, Equation 39 implies $W \geq V^{\pi}$ everywhere. By monotonicity of the Bellman operator,

$$\left(\mathcal{T}^{\bar{\pi}} W\right)(s) = \left(\mathcal{T}^{\pi} W\right)(s) \tag{45}$$

$$\geq \left(\mathcal{T}^{\pi} V^{\pi}\right)(s) = V^{\pi}(s) = W(s), \qquad s \notin S. \tag{46}$$

Therefore,

$$\mathcal{T}^{\bar{\pi}} W \geq W. \tag{47}$$

Repeated application of the monotone Bellman operator $\mathcal{T}^{\bar{\pi}}$, together with its convergence to $V^{\bar{\pi}}$, yields

$$V^{\bar{\pi}}(s) \geq W(s), \tag{48}$$

and hence, in particular,

$$V^{\bar{\pi}}(s) \geq V_{S,\pi}^{*,sub}(s), \qquad \forall s \in S. \tag{49}$$

It remains to establish the equivalence when leaving S precludes returning to S. Consider any $\pi' \in \Pi_S(\pi)$. While the process remains in S, the original MDP and $M_{S,\pi}^{\text{sub}}$ have identical dynamics and rewards. If the process leaves S at a successor state $s' \notin S$, it follows π thereafter, since π' and π coincide outside S. By assumption, it never returns to a state at which the two policies may differ. Its exact continuation value is therefore $V^{\pi}(s')$, precisely the boundary value used by $M_{S,\pi}^{\text{sub}}$.

Consequently, for every $\pi' \in \Pi_S(\pi)$ and every $s \in S$,

$$V_M^{\pi'}(s) = V_{M_{S,\pi}^{\text{sub}}}^{\pi'|_S}(s). \tag{50}$$

Optimizing both sides over $\pi' \in \Pi_S(\pi)$ gives

$$V_{S,\pi}^{*,\text{sub}}(s) = \sup_{\pi' \in \Pi_S(\pi)} V^{\pi'}(s). \tag{51}$$

Since $\bar{\pi}|_S = \pi_S^{*,\text{sub}}$, the same correspondence gives

$$V^{\bar{\pi}}(s) = V_{S,\pi}^{*,\text{sub}}(s), \tag{52}$$

which proves Equation 36.

Finally, $\bar{\pi} \in \Pi_S(\pi)$ by construction. It follows immediately that

$$V^{\bar{\pi}}(s) \leq \sup_{\pi' \in \Pi_S(\pi)} V^{\pi'}(s). \tag{53}$$

Combining this inequality with Equations 42 and 49 proves Equation 35, and therefore also Equation 37. □

The first inequality is strict at any state $s \in S$ for which

$$V_{S,\pi}^{*,sub}(s) > \left(\mathcal{T}_{S,\pi}^{sub} V^{\pi}\right)(s) = \max_{a \in \mathcal{A}} Q^{\pi}(s,a). \tag{54}$$

Thus, whenever a single greedy Bellman backup from V^{π} does not attain the optimal sub-MDP value, solving the sub-MDP can provide a strictly stronger improvement than single-step greedification followed by followed by π.

A.3 Even when $|S| = 1$, greedification is not generally optimal policy improvement

Perhaps surprisingly, the greedy operator is *not* an optimal policy improvement operator in general even when $|S| = 1$. This arises from the fact that the greedy operator does not account for the benefits of taking better decisions $\pi'(s)$ at s. We will show this with an example. If the agent cannot return

from s back to s in the MDP (e.g. in non-cyclical MDPs), this is not a problem and indeed in this case the greedy operator is an optimal policy improvement operator on $S = \{s\}$.

Consider the following Markov Decision Process (MDP): $\mathcal{S} = \{s, s_{terminal}\}, \mathcal{A} = \{a_0, a_1, a_2\}$. Discount factor: $\gamma = 0.9$. Deterministic rewards: $R(s, a_0) = 0, R(s, a_1) = 10, R(s, a_2) = 9$. Deterministic transitions: $P(s_{terminal}|s, a_0) = 1, P(s_{terminal}|s, a_1) = 1, P(s|s, a_2) = 1$ and otherwise 0. The optimal policy in this MDP $\pi^*(s)$ always chooses a_2 yielding a return of $\sum_{t=0}^{\infty} 0.9^t \cdot 9 = \frac{9}{1-0.9} = 90$.

Let the current policy $\pi(s)$ be a policy that always chooses a_0. Therefore, $V^\pi(s) = R(s, a_0) + \gamma V^\pi(s_{terminal}) = 0$. Let us compute $Q^\pi(s, \cdot)$:

$$Q^\pi(s, a_0) = 0 + 0.9V^\pi(s_{\text{terminal}}) = 0 + 0 = 0, \tag{55}$$

$$Q^\pi(s, a_1) = 10 + 0.9V^\pi(s_{\text{terminal}}) = 10 + 0 = 10, \tag{56}$$

$$Q^\pi(s, a_2) = 9 + 0.9V^\pi(s) = 9 + 0.9 \cdot 0 = 9. \tag{57}$$

If we use the greedy operator to improve the policy, we have:

$$\pi'(s) = \arg\max_{a \in \mathcal{A}} Q^\pi(s, a), \quad P_{\pi'(s)}(a = a_1) = 1, \quad V^{\pi'}(s) = 10. \tag{58}$$

Which is not the optimal policy. In this MDP, *optimal policy improvement* will always select a_2 at state s.

We conclude that if the MDP is cyclical, the greedy operator is *not guaranteed* to provide optimal policy improvement *even in an MDP where the policy is permitted to be modified only at one state*.

A.4 Optimal greedification under exact evaluation

Lemma 1 (Optimal greedification under exact evaluation)**.** *Fix a policy $\pi \in \Pi$ and a state $s \in \mathcal{S}$. If $\widehat{Q}^\pi = Q^\pi$, then any distribution $\mu^*(\cdot \mid s)$ supported on $\arg\max_{a \in \mathcal{A}} Q^\pi(s, a)$ satisfies*

$$\mu^* \in \arg\max_{\mu \in \Delta(\mathcal{A})} \sum_{a \in \mathcal{A}} \mu(a \mid s) Q^\pi(s, a) = \arg\max_{\mu \in \Delta(\mathcal{A})} \sum_{a \in \mathcal{A}} \mu(a \mid s) A^\pi(s, a).$$

Consequently, $\mathcal{I}_{\arg\max}$ is an optimal greedification operator under exact evaluation.

Proof. Both objectives are linear in μ. Moreover,

$$\sum_a \mu(a \mid s) A^\pi(s, a) = \sum_a \mu(a \mid s) Q^\pi(s, a) - V^\pi(s),$$

and $V^\pi(s)$ is constant with respect to μ. Hence the two objectives have the same maximizers, namely the distributions supported on $\arg\max_a Q^\pi(s, a)$. □

Lemma 2 (When optimal greedification is optimal one-state improvement)**.** *Fix $\pi \in \Pi$ and $s \in \mathcal{S}$, and define*

$$\Pi_{\{s\}}(\pi) := \{\pi' \in \Pi : \pi'(\cdot \mid x) = \pi(\cdot \mid x),\ \forall x \neq s\}.$$

Assume that, after leaving s, the process cannot revisit s under the continuation policy π. Then, for every $\pi' \in \Pi_{\{s\}}(\pi)$,

$$V^{\pi'}(s) - V^\pi(s) = \sum_{a \in \mathcal{A}} \pi'(a \mid s) A^\pi(s, a).$$

Therefore,

$$\arg\max_{\pi' \in \Pi_{\{s\}}(\pi)} V^{\pi'}(s) = \arg\max_{\pi' \in \Pi_{\{s\}}(\pi)} \sum_{a \in \mathcal{A}} \pi'(a \mid s) Q^\pi(s, a),$$

and $\mathcal{I}_{\arg\max}$ is also an optimal policy-improvement operator within $\Pi_{\{s\}}(\pi)$.

Proof. Because s cannot be revisited, after the initial action the policies π' and π coincide on every state that can subsequently be visited. Hence

$$V^{\pi'}(s) = \sum_{a \in \mathcal{A}} \pi'(a \mid s) Q^\pi(s, a).$$

Subtracting $V^\pi(s) = \sum_a \pi(a \mid s) Q^\pi(s, a)$ gives the first identity. The remaining claims follow by maximizing the resulting linear objective. □

The second lemma does not generally hold when s can be revisited, as demonstrated by the counterexample in Appendix A.3.

A.5 GREEDIFICATION

Proof. Write $w(s,a) := 1/\sigma^2(s,a) = \tau^2/\big((\tau^2 + \sigma_A^2(s,a))\beta\big)$, which is strictly positive for every a because $\beta > 0$, $\tau^2 > 0$ and $\sigma_A^2(s,a) \geq 0$. Let $f(a) := \exp\big(w(s,a)A(s,a)\big) > 0$ and $Z := \sum_{b\in\mathcal{A}} \pi(b|s)f(b) > 0$, so that $\mathcal{I}_{opt}(\pi,Q)(a|s) = \pi(a|s)f(a)/Z$.

Because $A(s,a) = Q(s,a) - \sum_{b\in\mathcal{A}} \pi(b|s)Q(s,b)$, the advantage satisfies

$$\sum_{a\in\mathcal{A}} \pi(a|s)A(s,a) = 0. \tag{59}$$

Subtracting a constant from $Q(s,\cdot)$ shifts both sides of Definition 2 equally, so it suffices to show $\sum_a \mathcal{I}_{opt}(\pi,Q)(a|s)A(s,a) \geq 0$.

Compare $\pi(a|s)f(a)A(s,a)$ with $\pi(a|s)A(s,a)$ term by term. Since $w(s,a) > 0$: if $A(s,a) > 0$ then $f(a) > 1$ and hence $f(a)A(s,a) > A(s,a)$; if $A(s,a) < 0$ then $f(a) < 1$ and hence $f(a)A(s,a) > A(s,a)$; and if $A(s,a) = 0$ both sides vanish. In every case $f(a)A(s,a) \geq A(s,a)$, strictly whenever $A(s,a) \neq 0$. Multiplying by $\pi(a|s) \geq 0$, summing, and applying equation 59,

$$\sum_{a\in\mathcal{A}} \pi(a|s)f(a)A(s,a) \;\geq\; \sum_{a\in\mathcal{A}} \pi(a|s)A(s,a) \;=\; 0. \tag{60}$$

Dividing by $Z > 0$ gives $\sum_a \mathcal{I}_{opt}(\pi,Q)(a|s)A(s,a) \geq 0$, with strict inequality as soon as some a with $\pi(a|s) > 0$ has $A(s,a) \neq 0$, i.e. unless π is already indifferent among the actions it plays. □ □

Remark 1 (why the advantage, and not Q, is essential)**.** *For $\mathcal{I}_{reg}$ the substitution $A \leftrightarrow Q$ is harmless: a single scalar temperature makes the softmax shift-invariant. For $\mathcal{I}_{opt}$ it is not. With per-action weights $w(s,a)$, adding a constant c to every $Q(s,a)$ changes the operator, because $\exp\big(w(s,a)(Q(s,a)+c)\big) \propto \exp\big(w(s,a)c\big)$ is not a common factor. The proof above uses Equation 59 and therefore genuinely requires the advantage; implementations must centre by V^π before applying the weights.*

The following counter-example makes the point concrete and shows that the Q-based variant is not a greedification operator. Take $\mathcal{A} = \{a_1, a_2\}$, $Q(s,a_1) = 1$, $Q(s,a_2) = 2$, $\pi = [0.3, 0.7]$, and weights $w(s,a_1) = 10$, $w(s,a_2) = 1$. Here $V^\pi(s) = 1.7$ and $A(s,\cdot) = [-0.7, +0.3]$. The advantage-based operator gives

$$\pi' \propto [\,0.3e^{-7},\; 0.7e^{0.3}\,] \approx [0.0003,\, 0.9997], \qquad \textstyle\sum_a \pi'(a|s)Q(s,a) \approx 2.00 > 1.7, \tag{61}$$

so greedification holds, as guaranteed above. Substituting Q for A gives instead

$$\pi' \propto [\,0.3e^{10},\; 0.7e^{2}\,] \approx [0.999,\, 0.001], \qquad \textstyle\sum_a \pi'(a|s)Q(s,a) \approx 1.00 < 1.7, \tag{62}$$

which moves mass onto the worse *action.*

A.6 CONVERGENCE

We prove convergence of Generalized Policy Iteration (Algorithm 1) driven by $\mathcal{I}_{opt}$ using the finite-horizon analysis of Tsitsiklis (2002) and the machinery developed by Oren et al. (2025b). In particular, the convergence proof of Oren et al. (2025b) relies on the *Limit-Sufficient Greedification* condition (Definition 5 therein). We first show that $\mathcal{I}_{opt}$ satisfies the corresponding condition almost surely in the presence of fresh, unbiased Gaussian evaluation noise whose variance vanishes over iterations.

A.6.1 LIMIT-SUFFICIENT GREEDIFICATION

Lemma 3 (Limit-Sufficient Greedification under Vanishing Evaluation Noise)**.** *Let $Q_0, Q_1, \cdots \in \mathcal{Q}$ be a sequence of bounded functions such that, for every $s \in \mathcal{S}$ and $a \in \mathcal{A}$,*

$$Q_i(s,a) \xrightarrow[i\to\infty]{\text{a.s.}} Q(s,a) \tag{63}$$

for some $Q \in \mathcal{Q}$.

Let $\mathcal{F}_i$ denote the history prior to drawing the evaluation noise at iteration i. Assume that Q_i and π_i are $\mathcal{F}_i$-measurable and that, conditional on $\mathcal{F}_i$, the evaluation errors are fresh, independent zero-mean Gaussian random variables,

$$\epsilon_i(s,a) \mid \mathcal{F}_i \sim \mathcal{N}\left(0, \sigma^2_{Q_i}(s,a)\right), \tag{64}$$

where

$$\lim_{i\to\infty} \sigma^2_{Q_i}(s,a) = 0, \qquad \forall s \in \mathcal{S},\ a \in \mathcal{A}. \tag{65}$$

Let

$$\hat{Q}_i(s,a) := Q_i(s,a) + \epsilon_i(s,a), \tag{66}$$

and let

$$\pi_{i+1} = \mathcal{I}_{opt}(\pi_i, \hat{Q}_i). \tag{67}$$

Assume that $\mathcal{A}$ is finite and that π_0 has full support over $\mathcal{A}$.

Then, for every state $s \in \mathcal{S}$, if

$$\mathcal{A}^*(s) := \arg\max_{a\in\mathcal{A}} Q(s,a), \tag{68}$$

the policy asymptotically concentrates on the maximizing action set:

$$\sum_{a\in\mathcal{A}^*(s)} \pi_i(a|s) \xrightarrow[i\to\infty]{\text{a.s.}} 1. \tag{69}$$

Consequently,

$$\sum_{a\in\mathcal{A}} \pi_i(a|s) Q_i(s,a) \xrightarrow[i\to\infty]{\text{a.s.}} \max_{a\in\mathcal{A}} Q(s,a), \tag{70}$$

which is the Limit-Sufficient Greedification condition of Oren et al. (2025b), extended here to vanishing stochastic evaluation error.

Proof. Fix a state $s \in \mathcal{S}$ and suppress s from the notation where convenient. Define the true and noisy advantages at iteration i by

$$A_i(a) := Q_i(s,a) - \sum_{b\in\mathcal{A}} \pi_i(b \mid s) Q_i(s,b), \tag{71}$$

$$\hat{A}_i(a) := \hat{Q}_i(s,a) - \sum_{b\in\mathcal{A}} \pi_i(b \mid s) \hat{Q}_i(s,b). \tag{72}$$

Let

$$v_i(a) := \sigma^2_{\hat{A}_i}(s,a) \tag{73}$$

$$:= \left(1 - \pi_i(a \mid s)\right)^2 \sigma^2_{Q_i}(s,a) + \sum_{b\neq a} \pi_i(b \mid s)^2 \sigma^2_{Q_i}(s,b), \tag{74}$$

and denote the denominator in the exponent of $\mathcal{I}_{\text{opt}}$ by

$$d_i(a) := \beta \frac{\tau^2 + v_i(a)}{\tau^2}. \tag{75}$$

Because

$$0 \le v_i(a) \le \sum_{b\in\mathcal{A}} \sigma^2_{Q_i}(s,b), \tag{76}$$

the finiteness of $\mathcal{A}$ and the assumption $\sigma^2_{Q_i}(s,b) \to 0$ imply

$$d_i(a) \to \beta \tag{77}$$

uniformly over $a \in \mathcal{A}$. Moreover, $d_i(a) \geq \beta > 0$ for every i and a.

The policy update can now be written as

$$\pi_{i+1}(a \mid s) \propto \pi_i(a \mid s) \exp\left(\frac{\hat{A}_i(a)}{d_i(a)}\right). \tag{78}$$

Because π_0 has full support and the multiplicative factor is strictly positive, π_i has full support for every finite i.

Fix an optimal action $a^\star \in \mathcal{A}^\star(s)$ and a suboptimal action $a \notin \mathcal{A}^\star(s)$, and define their limiting action gap

$$\Delta_a := Q(s, a^\star) - Q(s, a) > 0. \tag{79}$$

Taking the ratio of their probabilities eliminates the normalization constant:

$$\log \frac{\pi_{i+1}(a^\star \mid s)}{\pi_{i+1}(a \mid s)} = \log \frac{\pi_i(a^\star \mid s)}{\pi_i(a \mid s)} + Y_i, \tag{80}$$

$$Y_i := \frac{\hat{A}_i(a^\star)}{d_i(a^\star)} - \frac{\hat{A}_i(a)}{d_i(a)}. \tag{81}$$

Decompose this increment into its conditional mean and a zero-mean remainder:

$$g_i := \mathbb{E}[Y_i \mid \mathcal{F}_i], \qquad \xi_i := Y_i - g_i. \tag{82}$$

Because the evaluation errors have conditional mean zero,

$$g_i = \frac{A_i(a^\star)}{d_i(a^\star)} - \frac{A_i(a)}{d_i(a)}. \tag{83}$$

Writing

$$B_i := \sum_{b \in \mathcal{A}} \pi_i(b \mid s) Q_i(s, b), \tag{84}$$

we equivalently have

$$g_i = \frac{Q_i(s, a^\star)}{d_i(a^\star)} - \frac{Q_i(s, a)}{d_i(a)} + B_i \left(\frac{1}{d_i(a)} - \frac{1}{d_i(a^\star)}\right). \tag{85}$$

Since $Q_i(s, \cdot) \to Q(s, \cdot)$ over the finite action set, the sequence $\{B_i\}$ is bounded. Together with the uniform convergence $d_i(\cdot) \to \beta$, this gives

$$g_i \xrightarrow[i \to \infty]{\text{a.s.}} \frac{Q(s, a^\star) - Q(s, a)}{\beta} = \frac{\Delta_a}{\beta} > 0. \tag{86}$$

Consequently, by the elementary Cesàro lemma,

$$\frac{1}{n} \sum_{i=0}^{n-1} g_i \xrightarrow[n \to \infty]{\text{a.s.}} \frac{\Delta_a}{\beta}. \tag{87}$$

By construction, $\mathbb{E}[\xi_i \mid \mathcal{F}_i] = 0$, so $\{\xi_i\}$ is a martingale-difference sequence. Each ξ_i is a finite linear combination of the evaluation errors. Because $d_i(a) \geq \beta > 0$ and the convergent variance sequences $\{\sigma^2_{Q_i}(s, b)\}_i$ are bounded, there exists a finite constant C such that

$$\mathbb{E}[\xi_i^2 \mid \mathcal{F}_i] \leq C \qquad \forall i. \tag{88}$$

Hence,

$$\sum_{i=0}^{\infty} \frac{\mathbb{E}[\xi_i^2 \mid \mathcal{F}_i]}{(i+1)^2} < \infty. \tag{89}$$

The strong law for martingale-difference sequences (Hall & Heyde, 1980, Theorem 2.18) therefore gives

$$\frac{1}{n}\sum_{i=0}^{n-1}\xi_i \xrightarrow[n\to\infty]{\text{a.s.}} 0. \tag{90}$$

Unrolling the log-ratio recursion and dividing by n now yields

$$\frac{1}{n}\log\frac{\pi_n(a^\star \mid s)}{\pi_n(a \mid s)} = \frac{1}{n}\log\frac{\pi_0(a^\star \mid s)}{\pi_0(a \mid s)} + \frac{1}{n}\sum_{i=0}^{n-1} g_i + \frac{1}{n}\sum_{i=0}^{n-1}\xi_i \tag{91}$$

$$\xrightarrow[n\to\infty]{\text{a.s.}} \frac{\Delta_a}{\beta} > 0. \tag{92}$$

Therefore,

$$\frac{\pi_n(a \mid s)}{\pi_n(a^\star \mid s)} \xrightarrow[n\to\infty]{\text{a.s.}} 0. \tag{93}$$

Since $\pi_n(a^\star \mid s) \le 1$, this implies

$$\pi_n(a \mid s) \xrightarrow[n\to\infty]{\text{a.s.}} 0 \qquad \forall a \notin \mathcal{A}^\star(s). \tag{94}$$

Because $\mathcal{A}$ is finite,

$$\sum_{a\in\mathcal{A}^\star(s)} \pi_n(a \mid s) \xrightarrow[n\to\infty]{\text{a.s.}} 1, \tag{95}$$

which proves the first claim.

Finally, let $Q_{\max}(s) := \max_a Q(s,a)$. Then

$$0 \le Q_{\max}(s) - \sum_{a\in\mathcal{A}} \pi_i(a \mid s) Q(s,a) \tag{96}$$

$$= \sum_{a\notin\mathcal{A}^\star(s)} \pi_i(a \mid s)\big(Q_{\max}(s) - Q(s,a)\big) \xrightarrow[i\to\infty]{\text{a.s.}} 0. \tag{97}$$

Moreover,

$$\left|\sum_{a\in\mathcal{A}} \pi_i(a \mid s) Q_i(s,a) - Q_{\max}(s)\right| \le \|Q_i(s,\cdot) - Q(s,\cdot)\|_\infty \tag{98}$$

$$+ \left|\sum_{a\in\mathcal{A}} \pi_i(a \mid s) Q(s,a) - Q_{\max}(s)\right| \xrightarrow[i\to\infty]{\text{a.s.}} 0. \tag{99}$$

This is the required Limit-Sufficient Greedification condition. □

A.6.2 Convergence using Limit-Sufficient Greedification

Proof. We extend the Generalized Policy Iteration setup of Algorithm 1 by allowing the evaluation supplied to the policy improvement operator at iteration i to be corrupted according to

$$\hat{Q}_i(s,a) = Q_i(s,a) + \epsilon_i(s,a), \tag{100}$$

where the errors satisfy the assumptions of Lemma 3. The underlying evaluation update of Algorithm 1 remains unchanged; the additional error affects the evaluation supplied to the policy-improvement operator.

The convergence proof of Theorem 3 in Oren et al. (2025b) proceeds by induction backwards from terminal states in a finite-horizon MDP. Its Limit-Sufficient Greedification step requires that,

whenever the sequence of action-value estimates at a state converges to some limiting function Q, the corresponding policy sequence satisfies

$$\sum_{a\in\mathcal{A}} \pi_i(a|s)Q_i(s,a) \longrightarrow \max_{a\in\mathcal{A}} Q(s,a). \tag{101}$$

Lemma 3 establishes exactly Equation 101 almost surely under the additional evaluation noise considered here.

Condition on the probability-one event on which Equation 101 holds at every state. The finite-horizon induction of Oren et al. (2025b) then applies unchanged: convergence at terminal states implies convergence of the action values at the preceding time step; Limit-Sufficient Greedification implies that the policy becomes greedy with respect to those limiting action values; and the argument proceeds recursively backwards through the horizon. Since the state and action spaces and the horizon are finite, the intersection of the corresponding probability-one events is itself a probability-one event.

Therefore, Generalized Policy Iteration driven by $\mathcal{I}_{opt}$ converges almost surely to the optimal value:

$$Q_i \xrightarrow[i\to\infty]{\text{a.s.}} Q^*, \qquad V^{\pi_i} \xrightarrow[i\to\infty]{\text{a.s.}} V^*. \tag{102}$$

Moreover, at every state,

$$\sum_{a\in\arg\max_b Q^*(s,b)} \pi_i(a|s) \xrightarrow[i\to\infty]{\text{a.s.}} 1. \tag{103}$$

Thus, the policy converges to the set of optimal policies in the sense that all probability mass assigned to strictly suboptimal actions vanishes. If every state has a unique optimal action, then π_i itself converges almost surely to the corresponding optimal policy. □

B Derivations and Discussion

B.1 Derivation of the advantage posterior in the likelihood

For a fixed state s, we model the approximate evaluation as

$$\hat{Q}^\pi(s,a) = Q^\pi(s,a) + \epsilon(s,a), \qquad \epsilon(s,a) \sim \mathcal{N}\Big(0, \sigma^2_{\hat{Q}^\pi}(s,a)\Big), \tag{104}$$

with the errors independent across actions.

The corresponding approximate advantage is

$$\begin{aligned}\hat{A}^\pi(s,a) &:= \hat{Q}^\pi(s,a) - \sum_{b\in\mathcal{A}} \pi(b \mid s)\hat{Q}^\pi(s,b) \\ &= A^\pi(s,a) + \big(1-\pi(a \mid s)\big)\epsilon(s,a) - \sum_{b\neq a} \pi(b \mid s)\epsilon(s,b).\end{aligned} \tag{105}$$

Because Equation 105 is an affine transformation of independent Gaussian errors,

$$\hat{A}^\pi(s,a) \mid A^\pi(s,a), \pi, s \sim \mathcal{N}\big(A^\pi(s,a), \sigma^2_{\hat{A}^\pi}(s,a)\big), \tag{106}$$

where

$$\sigma^2_{\hat{A}^\pi}(s,a) := \big(1-\pi(a \mid s)\big)^2 \sigma^2_{\hat{Q}^\pi}(s,a) + \sum_{b\neq a} \pi(b \mid s)^2 \sigma^2_{\hat{Q}^\pi}(s,b). \tag{107}$$

For analytical tractability, we place the action-wise prior

$$A^\pi(s,a) \sim \mathcal{N}(0,\tau^2). \tag{108}$$

Taken jointly across actions, the corresponding product prior ignores the exact constraint

$$\sum_{a\in\mathcal{A}} \pi(a \mid s)A^\pi(s,a) = 0.$$

We make this modeling approximation because it yields a tractable action-wise posterior and expresses that, absent evaluation evidence, the model does not favor any action over another.

By Bayes' rule,

$$p\Big(A^\pi(s,a) \mid \hat{A}^\pi(s,a), \pi, s\Big) \propto \exp\left[-\frac{1}{2}\left(\frac{(A^\pi(s,a))^2}{\tau^2} + \frac{\big(\hat{A}^\pi(s,a) - A^\pi(s,a)\big)^2}{\sigma^2_{\hat{A}^\pi}(s,a)}\right)\right]. \tag{109}$$

Completing the square gives

$$p\Big(A^\pi(s,a) \mid \hat{A}^\pi(s,a), \pi, s\Big) = \mathcal{N}(m^\pi(s,a), v^\pi(s,a)), \tag{110}$$

with

$$m^\pi(s,a) = \frac{\tau^2}{\tau^2 + \sigma^2_{\hat{A}^\pi}(s,a)} \hat{A}^\pi(s,a), \tag{111}$$

$$v^\pi(s,a) = \frac{\tau^2 \sigma^2_{\hat{A}^\pi}(s,a)}{\tau^2 + \sigma^2_{\hat{A}^\pi}(s,a)}. \tag{112}$$

This posterior is exact for the scalar action-wise observation model. In the main text, we approximate the joint posterior over $A^\pi(s,\cdot)$ by the product of these action-wise posteriors.

B.2 Modeling the Utility Factor using Rational Choice and Expected Utility Theory

Rational Choice Theory is a foundational framework widely popular in the fields of economics, political science, and behavioral sciences for modeling complex, goal-oriented decision-making (Becker, 1976). Rational Choice Theory posits that decision-makers consistently select actions that maximize their subjective preferences (Arrow, 2012). Under conditions of uncertainty, this is formalized by Expected Utility Theory (EUT), which demonstrates that a rational agent's preferences can be mathematically represented by the expectation of a scalar utility function over possible outcomes (von Neumann & Morgenstern, 2007; Savage, 2012). To operationalize EUT within probabilistic frameworks, deterministic utility maximization is standardly relaxed into probabilistic choice models. Rooted in Luce's choice axiom (Luce et al., 1959) and random utility theory (McFadden, 1973), these models construct likelihood functions where the probability of selecting an alternative is proportional to its exponentiated utility.

This approach allows us to model *policy optimization* as reasoning over which policy π' is preferred based on its expected utility $A^{\pi',\pi}(s)$. We model the policy's *expected utility for improvement* $A^{\pi',\pi}(s)$ with its local improvement surrogate over π. The derived Bayesian posterior $p(A^\pi(s,a)|\hat{A}^\pi(s,a),\pi,s)$ for the uncertain advantage allows us to account for the evaluation uncertainty within policy optimization and yields the tractable certainty-equivalent objective in Equation 16, for which $\mathcal{I}_{\text{opt}}$ is the exact MAP optimizer.

B.3 THE UTILITY AS THE LOCAL IMPROVEMENT OF π' OVER π

We have:

$$A^{\pi',\pi}(s) = \sum_{a\in A} \pi'(a|s) A^{\pi}(s,a) \tag{113}$$

$$= \sum_{a\in A} \pi'(a|s)(Q^{\pi}(s,a) - V^{\pi}(s)) \tag{114}$$

$$= \sum_{a\in A} \pi'(a|s) Q^{\pi}(s,a) - \sum_{a\in A} \pi'(a|s) V^{\pi}(s) \tag{115}$$

$$= \sum_{a\in A} \pi'(a|s) Q^{\pi}(s,a) - V^{\pi}(s) \sum_{a\in A} \pi'(a|s) \tag{116}$$

$$= \sum_{a\in A} \pi'(a|s) Q^{\pi}(s,a) - V^{\pi}(s) \tag{117}$$

$$= V^{\pi'}(s) - V^{\pi}(s). \tag{118}$$

Equation 118 holds when π' differs from π only at s and s cannot be revisited after leaving it. In particular, this holds for a one-state policy update in a non-cyclical MDP. However, we always have:

$$\arg\max_{\pi'} A^{\pi',\pi}(s) = \arg\max_{\pi'} \sum_{a\in A} \pi'(a|s) A^{\pi}(s,a) = \arg\max_{\pi'} \sum_{a\in A} \pi'(a|s) Q^{\pi}(s,a) \tag{119}$$

because $V^{\pi}(s)$ does not depend on a. In other words, this objective is always sound as it shares the same maximizer as $\arg\max_{\pi'} \sum_{a\in A} \pi'(a|s) Q^{\pi}(s,a)$.

We use $A^{\pi',\pi}(s) = \sum_{a\in A} \pi'(a|s) A^{\pi}(s,a)$ instead of $\arg\max_{\pi'} \sum_{a\in A} \pi'(a|s) Q^{\pi}(s,a)$, because the advantage has important properties for policy improvement. Specifically, it is *informed* with respect to π, guaranteeing that the sign is always positive $A^{\pi}(s,a) > 0$ for actions that are *better* than those chosen by the policy (e.g. *improved*) and negative $A^{\pi}(s,a) < 0$ for actions that are *worse*.

B.4 DERIVATION OF THE CERTAINTY-EQUIVALENT MAP OBJECTIVE

Writing β_0 for the coefficient of the policy prior in Equation 8, the certainty-equivalent posterior is

$$p_{\mathrm{CE}}\Big(\pi' \mid \hat{A}^{\pi}(s,\cdot), \pi, s\Big) \propto \exp\left(\lambda \sum_{a\in\mathcal{A}} \pi'(a \mid s) \frac{\tau^2}{\tau^2 + \sigma^2_{\hat{A}^{\pi}}(s,a)} \hat{A}^{\pi}(s,a)\right) \times \exp\left(-\beta_0 D_{\mathrm{KL}}(\pi'(\cdot \mid s) \,\|\, \pi(\cdot \mid s))\right). \tag{120}$$

Therefore, because $\lambda > 0$,

$$\arg\max_{\pi'} p_{\mathrm{CE}}\Big(\pi' \mid \hat{A}^{\pi}(s,\cdot), \pi, s\Big) = \arg\max_{\pi'} \left\{ \sum_{a\in\mathcal{A}} \pi'(a \mid s) \frac{\tau^2}{\tau^2 + \sigma^2_{\hat{A}^{\pi}}(s,a)} \hat{A}^{\pi}(s,a) - \frac{\beta_0}{\lambda} D_{\mathrm{KL}}(\pi'(\cdot \mid s) \,\|\, \pi(\cdot \mid s)) \right\}. \tag{121}$$

Defining $\beta := \beta_0/\lambda$ recovers the certainty-equivalent objective in Equation 16.

B.5 DERIVATION OF THE SOLUTION FOR THE CERTAINTY-EQUIVALENT MAP OBJECTIVE

We derive $\mathcal{I}_{\text{opt}}$, the solution to the optimization problem:

$$\mathcal{I}_{opt}(\pi, \hat{Q}^{\pi})(s) = \arg\max_{\pi'} \left(\sum_a \pi'(a|s) w(s,a) \hat{A}^{\pi}(s,a) - \beta \sum_a \pi'(a|s) \log \frac{\pi'(a|s)}{\pi(a|s)} \right), \tag{122}$$

where $w(s,a) = \frac{\tau^2}{\tau^2 + \sigma^2_{\hat{A}^{\pi}}(s,a)}$, below.

Introduce a Lagrange multiplier ν for the constraint $\sum_a \pi'(a|s) = 1$:

$$\mathcal{L} = \sum_a \pi'(a|s) w(s,a) \hat{A}^\pi(s,a) - \beta \sum_a \pi'(a|s) \log \frac{\pi'(a|s)}{\pi(a|s)} + \nu \left(\sum_a \pi'(a|s) - 1 \right). \tag{123}$$

Take derivative w.r.t. $\pi'(a|s)$ and set to zero:

$$0 = w(s,a) \hat{A}^\pi(s,a) - \beta \left(\log \frac{\pi'(a|s)}{\pi(a|s)} + 1 \right) + \nu. \tag{124}$$

Rearranging,

$$\log \frac{\pi'(a|s)}{\pi(a|s)} = \frac{1}{\beta} w(s,a) \hat{A}^\pi(s,a) + \frac{\nu}{\beta} - 1. \tag{125}$$

Exponentiating,

$$\pi'(a|s) = \pi(a|s) \exp\left(\frac{1}{\beta} w(s,a) \hat{A}^\pi(s,a) \right) \exp\left(\frac{\nu}{\beta} - 1 \right). \tag{126}$$

To guarantee that π' is a proper probability distribution $\sum_{a \in A} \pi'(a|s) = 1$, ν is chosen such that $\exp\left(\frac{\nu}{\beta} - 1 \right)$ is the normalization constant:

$$\exp\left(\frac{\nu}{\beta} - 1 \right) = \left[\sum_b \pi(b|s) \exp\left(\frac{1}{\beta} w(s,b) \hat{A}^\pi(s,b) \right) \right]^{-1}. \tag{127}$$

Thus, we have:

$$\pi'(a|s) = \frac{\pi(a|s) \exp\left(\frac{1}{\beta} w(s,a) \hat{A}^\pi(s,a) \right)}{\sum_b \pi(b|s) \exp\left(\frac{1}{\beta} w(s,b) \hat{A}^\pi(s,b) \right)} \tag{128}$$

$$\propto \pi(a|s) \exp\left(\frac{\tau^2}{(\tau^2 + \sigma^2_{\hat{A}^\pi}(s,a))\beta} \hat{A}^\pi(s,a) \right) \tag{129}$$

B.6 Modeling evaluation uncertainty with the Normal distribution

In this section we further discuss the suitability of a Gaussian distribution to model the evaluation error, beyond its advantages for analytical tractability, and how we estimate the error in practice. As discussed in Section 4, Hui et al. (2023) motivate Gaussian TD error through the Central Limit Theorem and empirically find a moment-matched heteroscedastic Gaussian to provide a suitable approximation throughout training. More generally, evaluation error in modern RL can arise from stochastic returns, bootstrapped targets, sampled actions and function approximation. These sources need not be independent or exactly Gaussian, so we use the heteroscedastic Gaussian as an approximation rather than an exact description of modern evaluation error.

The better we can estimate the resulting state-action evaluation uncertainty, the better $\mathcal{I}_{\text{opt}}$ can be expected to perform. Uncertainty estimation remains an active and major field of research for AI, ML and RL, and we expect the potential of $\mathcal{I}_{\text{opt}}$ to grow with the quality of the estimator. To strengthen the significance of the contribution of $\mathcal{I}_{\text{opt}}$ to RL, in this work we incorporate $\mathcal{I}_{\text{opt}}$ into algorithms that already use some form of uncertainty estimation, and rely on this source - even when it is rudimentary.

In Policy Iteration, we noise the evaluation with Gaussian noise with known variance, and provide the agent with this exact variance. As expected, $\mathcal{I}_{\text{opt}}$ is the best performer in this setting.

In model-based RL, we incorporate $\mathcal{I}_{\text{opt}}$ into MCTS. Classical MCTS estimates action values from sample means of bounded Monte-Carlo returns (Kocsis & Szepesvári, 2006). Under fixed-distribution independent sampling, these means are unbiased, their variance decreases as $1/N(s,a)$, and their error becomes asymptotically Gaussian. Modern DNN-based MCTS (Silver et al., 2018; Schrittwieser

et al., 2020) departs substantially from this idealized sampling setting yet remains highly successful, suggesting that exact satisfaction of these assumptions is not necessary for the corresponding design principles to remain useful. We therefore use $\sigma^2_{\hat{Q}^\pi}(s,a) \propto 1/N(s,a)$ as a simple variance proxy and directly examine the resulting DNN residuals in Figure 6.

In model free RL (SAC and ReBRAC) we use the two-critic value ensemble. Value ensembles are a standard approach for value-uncertainty estimation in RL (Lakshminarayanan et al., 2017; Gawlikowski et al., 2023) and as we see, they are sufficient to enable $\mathcal{I}_{\text{opt}}$ to provide improvement over the baseline, both in offline as well as online RL (Figures 1 and 7, and Table 1).

B.7 $\mathcal{I}_{opt}$ AND EXPLORATION

The operator $\mathcal{I}_{opt}$ can be expected to have an effect on the exploration of RL algorithms along multiple lines. Due to the complexity of exploration as its own independent research field, we divide the discussion into three parts. First, we will motivate why exploration does not need to be considered by $\mathcal{I}_{opt}$ at all. Second, we will motivate what effects $\mathcal{I}_{opt}$ *could* have on exploration. Last, we will discuss how exploration should be incorporated into algorithms utilizing $\mathcal{I}_{opt}$.

B.7.1 WHY WE NEED NOT WORRY ABOUT THE EFFECTS $\mathcal{I}_{opt}$ HAS ON EXPLORATION

Although the simplest and most popular approach is to use trained *exploitation* policy π_θ to act in the environment, and implicitly to explore, there is no principle necessitating that. π_θ need not be used for exploration at all. Oren et al. (2025a) showed that one could use in a principled manner a policy trained explicitly for exploration to benefit the exploitation policy learned by π_θ.

Due to this principled separation between policies, we need not even consider explicitly the effects $\mathcal{I}_{opt}$ has on exploration - because the policy it is used to train π_θ need not be used for exploration at all.

B.7.2 WHAT EFFECTS $\mathcal{I}_{opt}$ *could* HAVE ON EXPLORATION

Of course, in practice, the simplest approach would be to replace the existing greedification operator used by an RL algorithm with $\mathcal{I}_{\text{opt}}$ and act with the trained exploitation policy π_θ in the environment (which is indeed how many popular, practical algorithms work in practice (Danihelka et al., 2022; Hansen et al., 2024; Haarnoja et al., 2018)). Despite not being the only alternative, it is popular in practice for several reasons. Not only is it simpler, but in complex environments such as Chess or Go it is not immediately clear which principled exploration policy is best suited, and acting with the trained policy is known to work well in practice (Silver et al., 2018; Danihelka et al., 2022). From this perspective, it may still be interesting to reason over what effects $\mathcal{I}_{opt}$ *could* have on exploration. We provide a detailed set of conjectures below:

1. Over-estimation bias should reduce, because the operator is more stable, because it propagates less error, because it favors high-error values less. As a result of reduced over-estimation bias, exploration may reduce.
2. During acting, exploration may also be reduced as a result of not favoring high-error, high-value actions.
3. However, because *high*-error, *low*-value actions are less unfavored, exploration can overall *increase*, as well.

Due to these conflicting possible effects, it is hard to predict whether acting with π_θ trained with $\mathcal{I}_{opt}$ increase, decrease or have other effects on exploration in aggregate. However, due to the last point in this list, we would expect $\mathcal{I}_{opt}$ in practice to simply induce a better learning behavior over algorithms that do not employ explicit directed exploration in the majority of domains. This result is supported by the results of our experiments in online RL (GumbelAlphaZero and SAC).

B.7.3 HOW TO UTILIZE $\mathcal{I}_{opt}$ FOR PRINCIPLED EXPLORATION

As shown by Oren et al. (2025a), a principled approach to training an exploration policy π_e is using policy improvement operators. However, instead of the improvement being with respect to Q or

A, it is with respect to an *exploration statistic*. A popular and principled exploration statistic is the *maximum upper confidence bound* (UCB), to drive classical, principled UCB exploration. This statistic can be modeled as $U^\pi_M(s,a) = Q^\pi_M(s,a) + \beta\sqrt{\mathbb{V}_{epistemic}[Q^\pi_M]}$ where $\mathbb{V}_{epistemic}$ here denotes the *epistemic* uncertainty with respect to Q^π_M. The random variable is the model M, and the uncertainty stems from the model not necessarily being aligned with reality. We refer to Oren et al. (2025a) for more detail.

Similarly, $\mathcal{I}_{opt}$ can be used to maximize $\arg\max_{\pi'} \sum_{a\in\mathcal{A}} U^\pi(s,a)\pi'(a|s)$ in the same manner it is used to maximize $\arg\max_{\pi'} \sum_{a\in\mathcal{A}} Q^\pi(s,a)\pi'(a|s)$. We leave investigating the effect of such operator on the exploration policy to future work. However, we can predict that the operator has the same underlying effect: actions with *certain* high UCB will be favored, actions with *certain* low UCB will be unfavored, and actions with *uncertain* UCB will not be strongly modified in the exploration policy.

C Connections to prior work in more detail

Control as Inference (CAI) and "Optimal Control" (Levine, 2018; Kappen et al., 2012) formulate finding a policy that maximizes expected return as a probabilistic inference problem in a graphical model. Trajectories are treated as latent variables generated by the dynamics P and a prior policy π. Rewards are incorporated as likelihood terms by introducing binary optimality variables O_t such that $p(O_t = 1|s_t, a_t) \propto \exp(r(s_t, a_t)/\eta)$. Inference in this model leads to the "soft-optimal" state value function $V^{\pi'}_{\text{soft}}(s)$ and action-value function $Q^{\pi'}_{\text{soft}}(s,a)$:

$$Q_{\text{soft}}(s,a) = r(s,a) + \gamma\mathbb{E}_{s'\sim P(\cdot|s,a)}[V_{\text{soft}}(s')], \tag{130}$$

$$V_{\text{soft}}(s) = \eta \log \sum_a \pi(a|s) \exp\left(Q_{\text{soft}}(s,a)/\eta\right). \tag{131}$$

The so-called "soft-optimal" policy π' is given by:

$$\pi'(a|s) = \mathcal{I}_{\text{cai}}(\pi, Q^{\pi'})(a|s) = \pi(a|s)\exp\big(\frac{Q^{\pi'}_{\text{soft}}(s,a) - V^{\pi'}_{\text{soft}}(s)}{\eta}\big) = \pi(a|s)\exp\big(\frac{A^{\pi'}_{\text{soft}}(s,a)}{\eta}\big).$$

In practice, the agent does not have access to $Q^{\pi'}_{\text{soft}}, V^{\pi'}_{\text{soft}}$ and therefore $Q^{\pi}_{\text{soft}} \approx Q^{\pi'}_{\text{soft}}, V^{\pi}_{\text{soft}} \approx V^{\pi'}_{\text{soft}}$ is used. This is equivalent to $\mathcal{I}_{\text{reg}}$ with A^π (except for softness, which is often omitted in practice (Hansen et al., 2024; de Vries et al., 2025)), and equivalently to $\mathcal{I}_{\text{opt}}$ with $\sigma^2_{A^\pi}(s,a) = \sigma$, a notion of uniform uncertainty across all state-action pairs. This is also the operator used by MPO (Abdolmaleki et al., 2018).

Section 4 suggests that it is more accurate to interpret $\mathcal{I}_{\text{cai}}$ as producing a notion of *optimal improvement* with respect to π rather than as producing an approximation of the (soft) *optimal policy*. It also shows how this operator emerges naturally from modeling policy improvement under evaluation uncertainty, without necessitating the modified objective, soft-values and non-trivial additional complexity resulting from modeling RL as a probabilistic inference problem.

Mirror Descent (MD) and The Natural Policy Gradient (NPG) (Tomar et al., 2022; Kakade, 2001) connect to $\mathcal{I}_{\text{reg}}$ in the finite tabular setting with direct policy parameterization. For a KL-Bregman divergence, the exact PMD update with step size $\alpha_k > 0$ has the state-wise form (Xiao, 2022)

$$\begin{aligned}\pi_{k+1}(a \mid s) &\propto \pi_k(a \mid s)\exp\left(\alpha_k Q^{\pi_k}(s,a)\right)\\ &\propto \pi_k(a \mid s)\exp\left(\alpha_k A^{\pi_k}(s,a)\right).\end{aligned} \tag{132}$$

Xiao (2022) identifies this exact policy-space update with NPG. Thus, using our inverse-step-size notation $\eta = 1/\alpha_k$, the update is precisely $\mathcal{I}_{\text{reg}}$.

Trust Region Policy Optimization (Schulman et al., 2015; 2017) bounds the approximation error between the true RL objective and a local surrogate by constraining the policy update within a trust region defined by $KL(\pi'|\pi) \le \delta$. Neu et al. (2017) analyze an exact, additively KL-regularized TRPO surrogate in which the current policy is fully evaluated and the current-policy state distribution is held fixed:

$$\pi_{k+1} \in \arg\max_{\pi'} \sum_s d^{\pi_k}(s)\left[\sum_a \pi'(a \mid s)A^{\pi_k}(s,a) - \eta D_{\text{KL}}(\pi'(\cdot \mid s) \,\|\, \pi_k(\cdot \mid s))\right]. \tag{133}$$

Its exact tabular solution is

$$\pi_{k+1}(a \mid s) \propto \pi_k(a \mid s) \exp\left(A^{\pi_k}(s,a)/\eta\right). \tag{134}$$

which is exactly $\mathcal{I}_{\text{reg}}$.

Regularized Policy Optimization is motivated from the perspective of greedy updates often being unstable in practice (Neu et al., 2017). It models policy optimization using a weighted regularizer $\eta\mathcal{R}(\pi', \pi)$ (left), which is often chosen to be $\eta KL(\pi'|\pi)$, recovering (right, Grill et al., 2020):

$$\arg\max_{\pi'} \sum_{a\in\mathcal{A}} \pi'(a|s)Q^\pi(s,a) - \eta\mathcal{R}(\pi'(s), \pi(s)),\ \mathcal{I}_{reg}(\pi, Q^\pi)(a|s) = \pi(a|s)\exp(Q^\pi(s,a)/\eta).$$

In view of $\mathcal{I}_{opt}$, we can see that $\mathcal{R}(\pi', \pi)$ connects to a Bayesian prior $p(\pi'|\pi)$. Similarly, $\eta(s,a)$ allows $\mathcal{I}_{opt}$ to regularize each state action *differently*, as a function of the certainty in the advantage $A(s,a)$.

D Pseudo Code

Algorithm 1 Generalized Policy Iteration

1: For starting functions $q \in \mathcal{Q}$, $\pi \in \Pi$ greedification operator $\mathcal{I}$, $k \geq 1$ and $\delta > 0$
2: **while** $\max_{s\in\mathcal{S}} \left| \sum_{a\in\mathcal{A}} \pi(a|s)q(s,a) - \max_b q(s,b) \right| > \delta$ **or** $\max_{(s,a)\in\mathcal{S}\times\mathcal{A}} |q(s,a) - \mathcal{T}^* q(s,a)| > \delta$ **do**
3: $\quad q(s,a) \leftarrow (\mathcal{T}^\pi)^k q(s,a),\ \forall (s,a) \in \mathcal{S}\times\mathcal{A}$
4: $\quad \pi(s) \leftarrow \mathcal{I}(\pi, q)(s),\ \forall s \in \mathcal{S}$
5: **end while**

The Bellman update $\mathcal{T}^\pi$ is defined as: $\mathcal{T}^\pi q_i(s,a) = \mathbb{E}[R(s,a)] + \gamma \mathbb{E}_{s'\sim P}[\sum_{a'\in\mathcal{A}} \pi(a'|s')q_i(s',a')]$. The notation $(\mathcal{T}^\pi)^k$ defines k repeating Bellman updates of this form with the same policy π, repeatedly updating the value q. $\mathcal{T}^*$ denotes the Bellman optimality operator, $\mathcal{T}^* q_i(s,a) = \mathbb{E}[R(s,a)] + \gamma \mathbb{E}_{s'\sim P}[\max_{a'\in\mathcal{A}} q(s',a')]$.

Algorithm 2 Simplified Model Based RL Training Loop with Modular Search

Require: Search algorithm (planner) $\mathcal{P}$, neural networks $\pi_{\theta_1}, v_{\phi_1}$, replay buffer $\mathcal{D}_1 = \emptyset$, environment's dynamics model $\mathcal{M} = (P, R)$ and budget parameters B.

for episode $n = 1$ to N **do**
  Sample starting state $s_1 \sim \rho$.
  for step $t = 1$ in the environment to termination or timeout **do**
    $\pi_{search}(s_t), V_{search}(s_t) \leftarrow \mathcal{P}(\pi_{\theta_n}, v_{\phi_n}, \mathcal{M}, B)(s_t)$.
    $a_t \sim \pi_{search}(s_t)$.
    $s_{t+1} \sim P(\cdot|s_t, a_t), \quad r_t \sim R(s_t, a_t)$.
    Append $(s_t, a_t, r_t, s_{t+1}, \pi_{search}(s_t), V_{search}(s_t))$ to buffer $\mathcal{D}_n$.
  end for
  Update policy params θ_{n+1} with SGD and CE loss on targets π_{search} from $\mathcal{D}_n$.
  Update value params ϕ_{n+1} with SGD and MSE loss on TD-λ targets using V_{search} from $\mathcal{D}_n$.
  Set $\mathcal{D}_{n+1} \leftarrow \mathcal{D}_n$.
end for

E Experimental and Implementation Details

E.1 Policy Iteration Experiments

The generalized Policy Iteration algorithm which accepts different policy improvement operators is described in Algorithm 1. In these experiments, we vary the improvement operator $\mathcal{I}$ used by the

algorithm across different types of evaluation. Two evaluation methods are used: *exact evaluation* ($k \to \infty$) computed via matrix inversion and a single Bellman backup per iteration ($k = 1$), similar to Value Iteration and TD-learning. Following both evaluation steps, the value is noised with independent zero-mean Gaussian noise with state-action-specific variances. The experiments' results are presented in Figure 1 and Appendix F.

Environments. The environments used in this experiment are infinite horizon grid-world mazes, using the ShinRL framework (Kitamura & Yonetani, 2021). To vary the transition and reward dynamics we generate 20 fixed randomly generated MDPs and run several trials per algorithm in each (Table 3). Every trial starts from the uniform policy $\pi_0(a|s) = 1/|\mathcal{A}|$.

Noise schedule. At each iteration every state–action pair draws a fresh variance:

$$c_i \sim \mathcal{U}\big(0,\, a_0\gamma_{\text{dec}}^i\big), \quad \sigma^2_{Q_i}(s,a) \sim \mathcal{U}\big(c_i,\, c_i + b\big), \quad \hat{Q}^{\pi_i}(s,a) \sim \mathcal{N}\big(Q^{\pi_i}(s,a),\, \sigma^2_{Q_i}(s,a)\big). \tag{135}$$

All constants are specified in Table 3. A random half of the pairs instead keeps $\sigma^2_{Q_i} \sim \mathcal{U}(0,1)$ at every iteration to keep the variances of very mixed sizes, simulating the larger-error predictions from poorly or newly visited areas of the state space, as well as lower-error predictions of well visited, in RL. The floor c_i decays as well, further simulating the reduction in evaluation error over the training process.

Operators compared. All operators act on the noisy $\hat{Q}^{\pi_i}$ and its advantage $\hat{A}^{\pi_i}(s,a) = \hat{Q}^{\pi_i}(s,a) - \sum_b \pi_i(b|s)\hat{Q}^{\pi_i}(s,b)$. Writing $\sigma^2_{\hat{A}}$ for the advantage variance of Equation 10, the different operators are defined as follows:

$$\mathcal{I}_{\text{opt}}(\pi_i, \hat{Q}^{\pi_i})(a|s) \propto \pi_i(a|s) \exp\Big(\tfrac{\tau^2}{(\tau^2 + \sigma^2_{\hat{A}}(s,a))\beta}\hat{A}^{\pi_i}(s,a)\Big), \tag{136}$$

$$\mathcal{I}_{\text{reg}}(\pi_i, \hat{Q}^{\pi_i})(a|s) \propto \pi_i(a|s) \exp\big(\hat{A}^{\pi_i}(s,a)/\eta\big), \tag{137}$$

$$\mathcal{I}_{\arg\max}(\pi_i, \hat{Q}^{\pi_i})(a|s) = \arg\max_a \hat{Q}^{\pi_i}(s,a), \tag{138}$$

$$\mathcal{I}_{\text{cpi}}(\pi_i, \hat{Q}^{\pi_i})(s) = (1-\alpha_i)\pi_i(s) + \alpha_i\, \arg\max_a \hat{Q}^{\pi_i}(s,a). \tag{139}$$

$\mathcal{I}_{\text{cpi}}$ (Kakade & Langford, 2002) interpolates towards the greedy policy rather than jumping to it. For α_i we use Pirotta et al. (2013)'s more recent form, $\alpha_i \propto \frac{(1-\gamma)^2}{\gamma} \cdot \frac{|\mathcal{S}|^{-1}\sum_s B(s)}{\max_s B(s) - \min_{s'} B(s')}$ with $B(s) = \sum_a (\pi_{\text{greedy}}(a|s) - \pi_i(a|s))\hat{Q}^{\pi_i}(s,a)$.

In addition, we include pessimistic (LCB) variants of $\mathcal{I}_{\arg\max}$ and $\mathcal{I}_{\text{reg}}$ to investigate the differences between the effect of $\mathcal{I}_{\text{opt}}$'s uncertainty awareness and standard pessimism. They receive the same $\sigma^2_{Q_i}$ of Equation 135 but spend it as a value penalty, $\hat{Q}_i^{\text{LCB}}(s,a) = \hat{Q}^{\pi_i}(s,a) - \lambda\sqrt{\sigma^2_{Q_i}(s,a)}$, before the unmodified operator is applied.

Score. Returns are min–max normalised per environment, $V^{\pi_i}_{\text{norm}}(s_0) = \frac{V^{\pi_i}(s_0) - V^{\pi_0}(s_0)}{V^*(s_0) - V^{\pi_0}(s_0)}$, where V^{π_0} is the uniform policy and V^* is obtained per seed by running a noise-free $\arg\max$ operator to convergence. Comparison and tuning use the normalised area under the curve, $\text{AUC} = T^{-1}\sum_{i=1}^{T} V^{\pi_i}_{\text{norm}}(s_0)$. Curves aggregate all $N_{\text{envs}} \cdot N_{\text{trials}} = 2000$ scores per iteration. Every aggregate interval holds the environments fixed and treats only seeds as random, so that $\text{Var} = N_{\text{envs}}^{-2}\sum_e \sigma_e^2/n_e$ with σ_e^2 the across-seed variance inside environment e. The hyperparameters for all operators were tuned with an extensive grid search, described in more detail in Appendix G.

E.2 Discrete-Action Model Based RL (GumbelAlphaZero on board games)

GumbelAlphaZero uses GumbelMCTS to drive policy improvement and action selection in RL. The GumbelMCTS algorithm is a modern variant of the MCTS algorithm which uses greedification operators directly within the search to drive policy improvement at the root. In broad strokes, MCTS is an iterative algorithm that constructs a search tree and uses statistics to direct its search towards increasingly more promising directions. In each iteration, the algorithm traverses the tree from the root using a *search policy* $\pi_i = \mathcal{I}_{\text{reg}}(\pi_\theta, Q_i)$, where π_θ, often called the *prior policy*, the policy trained by the RL algorithm, and Q_i is the aggregated value observed during search. When search arrives at an as-of-yet un-expanded action a at a node s_T (a *leaf* in the tree), the node is expanded, evaluating $s_{T+1} \sim P(\cdot|s_T, a), Q_i(s_T, a) = r_T + \gamma V_\psi(s_{T+1})$. The reward and transition models are both

usually assumed to be deterministic. After the expansion step, the new information $r_T + \gamma V_\psi(s_{T+1})$ observed in the expansion is backpropagated throughout the entire search trajectory all the way back to the root, the iteration has completed, and the next iteration starts from the top. The number of iterations, which is also the number of expansions, M, is the common compute budget considered for MCTS, as in many cases the DNN forward-pass-based expansion is the cost's bottleneck.

The GumbelAlphaZero (GAZ) algorithm is a natural test-bed for $\mathcal{I}_{\text{opt}}$: its visit counts provide a natural state-action evaluation-variance estimator under the classical sample-mean interpretation of MCTS and its DNN-based evaluations test $\mathcal{I}_{\text{opt}}$ in fully the modern deep-RL setting. Finally, it already has a clear interpretation as using iterative greedification within the tree for policy improvement outside the tree, with $\mathcal{I}_{\text{reg}}$.

Evaluation in board game, a standard, hard, benchmark domain for RL and MCTS, is very expensive: the agents take relatively large number of steps before they have well functioning policies. To achieve high-quality training curves, the evaluation requires comparing every checkpoint to every other checkpoint across a large number of games, where each agent uses MCTS variants with the checkpoint's DNN. Due to the high cost of this evaluation, we run two different experiments, answering two faces of the same question - *how does GumbelAlphaZero perform when $\mathcal{I}_{reg}$ is replaced with $\mathcal{I}_{opt}$?*

In the first experiment we compare 4 agents: with and without a Q_ϕ critic, and with and without $\mathcal{I}_{\text{opt}}$ in place of $\mathcal{I}_{\text{reg}}$. This experiment allows us to evaluate *which operator, in and out of the presence of the additional critic, produces better learning?* This is the experiment presented in Figure 1. In the second experiment we compare 6 agents - the same 4 with additionally a vanilla-AlphaZero (Silver et al., 2018) PUCT-based (Silver et al., 2016; Rosin, 2011) agent, with and without a Q_ϕ critic. In this experiment, we use *the same* DNNs ($\pi_\theta, V_\psi, Q_\phi$) for all agents ($Q_\phi$ when relevant). Rather than learning curves, this produces the playing strength of MCTS with each operator. This answers the question *what is the direct contribution of each operator to action selection - playing strength - when all else is equal?* We also repeat this experiment across different planning budgets M and in two environments: chess and 9x9 Go, allowing us to answer this question much more comprehensively.

The experiments' results are presented in Figure 1 in Section 6 and Figures 4 and 5 in Appendix F.

E.2.1 THE OPERATOR

GumbelMCTS (Danihelka et al., 2022) uses $\mathcal{I}_{reg}$ for three purposes: (I) Policy improvement at the root of the tree, once search has concluded. (II) To drive the search at the root, using Sequential Halving (Karnin et al., 2013). (III) To drive the search within the tree, deciding which action to select at each state (each node) in the search tree. All three use the *completed Q-values* mechanism and a single node-level coefficient:

$$\mathcal{I}_{reg}(\pi_\theta, Q_i)(a|s) \propto \pi_\theta(a|s) \exp\Big(\frac{1}{\eta(s)} Q_i(s,a)\Big), \;\; \frac{1}{\eta(s)} = c_{scale}\big(c_{visit} + \max_{b \in \mathcal{A}} N(s,b)\big), \quad (140)$$

where $N(s,a)$ is the number of simulations that have passed through (s,a) and Q_i is the value estimated by the search tree at search iteration i. This coefficient is a heuristic for the variance reduction produced by repeated evaluation: under a fixed rollout distribution and independent sampling, the variance of a sample mean decreases as $1/N(s,a)$, so the inverse visit count is proportional to the variance.

We replace every use of $\mathcal{I}_{\text{reg}}$ with $\mathcal{I}_{\text{opt}}$, setting $\sigma_Q^2(s,a) \propto 1/N(s,a)$, the corresponding classical evaluation variance estimator, obtain $\sigma_A^2(s,a)$ from Equation 10, and $\eta(s,a)$ from Equation 19:

$$\mathcal{I}_{\text{opt}}(\pi_\theta, Q_i)(a|s) \propto \pi_\theta(a|s) \exp\Big(\frac{1}{\eta(s,a)} A_i(s,a)\Big). \quad (141)$$

To avoid division by zero, actions the search has not visited take $N(s,a) = 1$, and all other actions add additionally the number of evaluations through the node. The value of actions the search has not visited is completed by the network's per-action $Q_\phi(s,a)$ where a Q head is available and otherwise by the completed-Q value mechanism used by Danihelka et al. (2022).

E.2.2 Training the Actor and the Critics

The actor π_θ is trained in the standard GumbelAlphaZero manner, with a cross entropy loss with the target $\pi_{\text{search}}(s) = \mathcal{I}(\pi_\theta, Q_M)$, the improved policy at the root of the search tree at the last iteration M (Danihelka et al., 2022). Baseline GumbelAlphaZero uses $\mathcal{I}_{\text{reg}}$, while our version uses $\mathcal{I}_{\text{opt}}$.

The state-value critic $V_\psi(s)$ is trained with MSE loss with the modern model-based target $V_{\text{search}}(s) = \sum_{a \in B} \bar{\pi}_{\text{search}}(a|s) Q_M(s,a)$ (Oren et al., 2025a; de Vries et al., 2025; Oren et al., 2026a;b). The set B denotes the set of all *searched* actions at the root, and the $\bar{\pi}_{\text{search}}$ the improved policy at the root, renormalized only over the searched actions.

MCTS does not traditionally train a Q-head, although it has been tried before (Gao et al., 2018). However, since the popular, canonical approaches, AlphaZero (Silver et al., 2018), MuZero (Schrittwieser et al., 2020) and GumbelAlpha/MuZero (Danihelka et al., 2022) from DeepMind do not use a Q head, there is no one accepted canonical approach. Building on recent ideas and developments from MCTS and AlphaZero/MuZero literature (Oren et al., 2025a; 2026a) we train the Q-network as follows. The search at the root splits the action set in three, each receiving a different target:

1. **Searched actions** ($N(s,a) > 0$) take the tree's own backed-up value Q_i at the end of search.

2. **Bootstrapped actions:** L of the unsearched legal actions are sampled from the improved policy at the end of search, without replacement and given the one-step TD-style target $T(s,a) = r(s,a) + \gamma\, V_\psi(s')$.

3. **All other actions** receive no target and contribute zero gradient.

The error vs. the target is measured with MSE.

Each searched action is weighted by $\sqrt{1 + N(s,a)}$, i.e. by inverse *standard error* rather than inverse variance. Precision weighting $(1 + N)$ is maximum-likelihood only for unbiased targets, and these are not: an action is most-visited partly because its early estimates looked good, so its backup carries a winner's curse that does not shrink like $1/N$. Bootstrapped actions get unit weight; their noise is that of $V_\psi(s')$ and is not on the $\sqrt{1+N}$ scale.

The weighted regression constrains only actions that received a target, so we add one scalar constraint per state: the improved policy at the root π_{search}-weighted mean of Q_ϕ must match the same mean over the tree's completed values, with the target treated as a constant, providing some learning signal to all actions.

The total loss is $L = L_\pi + L_v + \lambda_q L_q$, with L_π the cross-entropy against the improved policy over legal actions, L_v the squared error on the value head, and L_q the sum of the weighted regression and the calibration term. Hyperparameters are specified in Table 4.

E.2.3 Evaluation across training

Training. We use `pgx` (Koyamada et al., 2023) implementation of 9×9 Go and chess. All agents follow the training scheme of AlphaZero (Silver et al., 2018; Danihelka et al., 2022, Algorithm 2). Actions are sampled randomly from the acting policy for the first 8 moves and then acting is deterministic. The acting policy follows the conventions of AlphaZero and GumbelAlphaZero (visits at the root, and / or the action chosen by Sequential Halving in the GumbelAlphaZero case, see (Danihelka et al., 2022) and (Karnin et al., 2013)). All agents share every hyperparameter except the operator and the Q head, detailed in Table 4.

Evaluation. Evaluation with Bayes elo is conducted by playing every DNN checkpoint and search pair against every other. We train 5 seeds per agent and evaluate all of them in one tournament, so every checkpoint of every seed plays every other and the whole set is fitted on a single Elo scale. The results are presented in Figure 4. We report the mean over seeds and the minimum and maximum elo observed over seeds. BayesElo's own interval is reported per seed.

E.2.4 Evaluation at a fixed network

Setup. In the equalized-DNNs experiment, all agents use the same checkpoint trained with GAZ + Q_ϕ, to make sure no advantage is given to $\mathcal{I}_{\text{opt}}$ from training its own networks.

Evaluation. Agents play a round robin from a fixed opening book, with every pairing played in both colours so that a first-move advantage term is not identifiable. Elo is fitted with BayesElo holding the *colour-advantage* at zero.

E.3 Classical Control Model-Free RL (SAC on DMC)

The Operator. SAC can be interpreted as policy iteration with the regularized policy-improvement operator $\pi_{i+1}(a|s) = \mathcal{I}_{\text{reg}}\left(\mu, \hat{Q}^{\pi_i}_{\text{soft}}\right)(a|s)$. In a tabular or discrete-action setting, this improved policy can be computed explicitly over the action space. With a parameterized policy π_θ, the improved policy can be projected onto the policy class directly using a KL loss with respect to the target π_{i+1}. In continuous-action settings, explicit construction and projection of the improved policy $\pi_{i+1}(a|s)$ can also be performed using sampled actions, as in MPO (Abdolmaleki et al., 2018).

This is not, however, the approach taken by commonly used continuous-action implementations of SAC. In popular implementations (Huang et al. (2022)'s implementation for example which we use in the experiments), the actor is optimized directly using the reparameterized loss

$$L(\theta) = \eta \log \pi_\theta(a_\theta|s) - A_{\text{soft}}(s, a_\theta), \tag{142}$$

where $a_\theta = f_\theta(\epsilon; s)$, with ϵ denoting sampled noise. The reparameterization trick allows gradients to propagate through the sampled action to the policy parameters (Haarnoja et al., 2018). Here, using A_{soft} instead of Q_{soft} is equivalent for the actor optimization, since $A_{\text{soft}}(s, a) = Q_{\text{soft}}(s, a) - V_{\text{soft}}(s)$, and $V_{\text{soft}}(s)$ is independent of the action and therefore does not affect the gradient with respect to θ.

Thus, rather than explicitly constructing and projecting onto the improved policy, SAC directly optimizes the regularized objective whose unconstrained policy-space optimum is $\mathcal{I}_{\text{reg}}$.

Our objective is to evaluate whether $\mathcal{I}_{\text{opt}}$ can directly improve existing RL algorithms while retaining their practical design choices and tuning procedures. We therefore preserve the direct gradient-based actor optimization used by SAC. To incorporate $\mathcal{I}_{\text{opt}}$ while retaining SAC's self-tuned temperature parameter, we set $\eta = \frac{\beta}{\tau^2}$ and incorporate the uncertainty-dependent weighting prescribed by $\mathcal{I}_{\text{opt}}$. Specifically, we use

$$L_{\mathcal{I}_{\text{opt}}}(\theta) = \eta \log \pi_\theta(a_\theta|s) - \frac{A_{\text{soft}}(s, a_\theta)}{\tau^2 + \sigma^2_{A_{\text{soft}}}(s, a_\theta)}. \tag{143}$$

This leaves τ^2 as the only additional hyperparameter introduced by $\mathcal{I}_{\text{opt}}$.

For the uncertainty estimate, rather than combining the variances of the two value estimates as in Equation 10, which neglects their covariance, we compute the advantage separately using each critic and estimate its variance from their disagreement:

$$\sigma^2_{A_{\text{soft}}}(s, a_\theta) = \text{Var}_i\left[Q_{\phi_i}(s, a_\theta) - \mathbb{E}_{b\sim\pi_\theta(\cdot|s)}\left[Q_{\phi_i}(s, b)\right]\right]. \tag{144}$$

The inner expectation is each critic's own state value $V_{\phi_i}(s)$, estimated by Monte Carlo with 32 actions drawn from $\pi_\theta(\cdot|s)$ at that state. Each critic therefore forms its own advantage of the sampled action a_θ over the policy's value, and the variance across critics of those advantage estimates is the uncertainty estimate. Centring each critic on its own value removes that critic's state-dependent bias, so what remains is disagreement about the action rather than about the state; with the two critics used here the estimator reduces to $\frac{1}{2}(A_1 - A_2)^2$. This requires no additional networks, at the cost of 32 further action samples and critic evaluations per state.

The implementation weights $Q_{\text{soft}}(s, a_\theta)$ rather than $A_{\text{soft}}(s, a_\theta)$ in Equation 143; since the detached weight is treated as constant during differentiation, the two give identical actor gradients, by the argument given above for Equation 142.

In this implementation, gradients are stopped through the uncertainty estimate $\sigma^2_{A_{\text{soft}}}(s, a_\theta)$. Consequently, uncertainty determines the weighting assigned to each sampled state-action gradient, but does not contribute directly to the gradient through its dependence on the action. In particular, the resulting update retains the term:

$$\frac{1}{\tau^2 + \sigma^2_{A_{\text{soft}}}(s, a_\theta)} \nabla_\theta A_{\text{soft}}(s, a_\theta), \tag{145}$$

while omitting the gradient of the uncertainty-dependent weight itself. Because gradients are stopped through the uncertainty estimate, uncertainty introduces no additional action-space gradient through $\nabla_a \sigma^2_{A_{\text{soft}}}$. For each sampled state-action pair, it only positively rescales the ordinary advantage gradient in Equation 143. Thus, under direct optimization of a single action at a fixed state, it does not change the stationary points or maximizers of the Q term. As demonstrated in our experiments, this approximation is sufficient to improve performance while retaining the original SAC optimization procedure.

Setup. We use ten popular DeepMind Control Suite tasks (Tunyasuvunakool et al., 2020) and train the agents up to $1M$ steps. The environments are: `cheetah-run`, `humanoid-run`, `quadruped-run`, `fish-swim`, `humanoid-stand`, `quadruped-walk`, `hopper-hop`, `humanoid-walk`, `walker-walk` and `hopper-stand`.

Metric and aggregation. Returns are min–max normalised per task over all seeds and then aggregated per agent across domains. We report the performance after $1M$ steps.

Hyperparameters are specified in Appendix G, Table 5.

E.4 Offline RL (ReBRAC on D4RL)

The Operator. ReBRAC (Tarasov et al., 2023) trains a deterministic actor to maximize $Q_\phi(s, \pi_\theta(s))$ under a behavior-cloning penalty. This penalty serves as the behavior-regularization term corresponding to the divergence from the prior in Equation 16. A deterministic actor does not provide a tractable Boltzmann policy target to project onto, as in the SAC case (Appendix E.3). We therefore apply the coefficient prescribed by Equation 19 directly to ReBRAC's policy-improvement term:

$$L_{\mathcal{I}_{\text{opt}}}(\theta) = -\frac{\lambda}{\beta} \frac{\tau^2}{\tau^2 + \sigma^2_{\hat{A}}(s, a)} Q_\phi\big(s, \pi_\theta(s)\big) + \beta_1 \left\| \pi_\theta(s) - a \right\|_2^2 . \tag{146}$$

The advantage with respect to a policy π and its value functions Q^π and V^π is defined as $A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)$. In ReBRAC, however, the policy being evaluated and the policy serving as the behavior-regularization reference differ. The critic $Q_\phi \approx Q^{\pi_\theta}$ evaluates the learned policy π_θ, while the behavior policy π_b, represented by the actions in the offline dataset, serves as the reference toward which the actor is regularized.

We therefore use:

$$\hat{A}(s, a) = Q_\phi\big(s, \pi_\theta(s)\big) - Q_\phi(s, a), \qquad Q_\phi(s, \cdot) = \min_i Q_{\phi_i}(s, \cdot), \tag{147}$$

as the improvement quantity for constructing the uncertainty estimate. This quantity measures the predicted improvement of the learned action relative to ReBRAC's behavior reference, with both actions evaluated by the learned critic. Since the dataset action a does not depend on the actor parameters, we have:

$$\nabla_\theta \hat{A}(s, a) = \nabla_\theta Q_\phi\big(s, \pi_\theta(s)\big), \tag{148}$$

and hence $\hat{A}$ induces exactly the same actor gradient as ReBRAC's original Q-maximization term when their coefficients are fixed. Estimating uncertainty in this difference therefore measures critic disagreement about the improvement of the learned action relative to the behavior action, rather than disagreement about their absolute values.

To estimate this uncertainty, rather than combining the variances of the two value estimates as in Equation 10, which neglects their covariance, we compute the improvement separately using each

critic and estimate its variance from their disagreement:

$$\sigma^2_{\hat{A}}(s,a) = \mathrm{Var}_i\left[Q_{\phi_i}\big(s,\pi_\theta(s)\big) - Q_{\phi_i}(s,a)\right]. \tag{149}$$

That is, each critic evaluates both the current policy action $\pi_\theta(s)$ and the dataset action a, and the variance across critics of the resulting improvement estimates is used as the uncertainty estimate. Evaluating both actions with the same critic also allows errors shared between the two evaluations to cancel in their difference, rather than treating the two value estimates as independent sources of uncertainty. The resulting quantity therefore captures critic-ensemble disagreement about the estimated improvement of the current policy action over the action observed in the data. This requires no additional networks: with two critics, each critic evaluates both action inputs, in addition to the single actor evaluation required to obtain $\pi_\theta(s)$.

Writing

$$w(s,a) = \frac{\tau^2}{\tau^2 + \sigma^2_{\hat{A}}(s,a)} \tag{150}$$

for the uncertainty-dependent coefficient, which is treated as constant with respect to the actor gradient, we apply $w(s,a)$ directly to ReBRAC's original Q-maximization term. ReBRAC's Q-scale normalizer λ is correspondingly computed from the weighted term as it actually enters the loss:

$$\lambda = \left(\mathbb{E}_{(s,a)\sim\mathcal{D}}\left[\left|w(s,a)Q_\phi\big(s,\pi_\theta(s)\big)\right|\right]\right)^{-1}. \tag{151}$$

This replaces ReBRAC's original normalizer

$$\left(\mathbb{E}_{(s,a)\sim\mathcal{D}}\left[\left|Q_\phi\big(s,\pi_\theta(s)\big)\right|\right]\right)^{-1}, \tag{152}$$

so that ReBRAC's Q-maximization term remains scale-normalized after introducing the uncertainty weighting.

Although $\hat{A}(s,a)$ and $Q_\phi(s,\pi_\theta(s))$ induce identical actor gradients when their coefficients are fixed, their numerical values generally differ and they are therefore not interchangeable inside the scale normalizer. Computing λ from $\mathbb{E}[|w(s,a)\hat{A}(s,a)|]$ instead would generally produce a different λ, thereby changing the relative scale of the policy-improvement gradient and the behavior-cloning gradient. Equation 151 instead preserves ReBRAC's normalization convention by forming λ from the weighted Q-maximization term that actually enters Equation 146. In particular, when $w(s,a) = 1$, Equation 151 recovers ReBRAC's original normalizer.

The critic, both behavior-cloning coefficients, and all other components of ReBRAC are left unchanged.

Setup. Four D4RL (Fu et al., 2020) Gym-MuJoCo medium-replay datasets: *halfcheetah*, *hopper*, *walker2d* and *ant*. Each run trains for 1M gradient steps.

Metric. The reported metric is the final policy evaluated over 56 episodes. The offline bar of Figure 1 is min–max normalised per environment over the runs of both arms and then averaged over the four environments, the normalisation the SAC bar of the same figure uses. Table 1 instead reports raw D4RL points (0 = random, 100 = expert).

F Additional Results

In this section we provide additional results across the different experiment settings presented in Figure 1 in Section 6. These include the individual results aggregated in Figure 1, the behavior of the hyperparameters introduced with $\mathcal{I}_{\text{opt}}$ and additional experiments to answer specific research questions and further investigate the contribution of $\mathcal{I}_{\text{opt}}$ to performance in different domains and algorithms.

F.1 Dynamic Programming additional results

F.1.1 Generalized Policy Iteration with $k \to \infty$

We include a set of additional experiments and results in the Policy Iteration setup presented in Figure 1. In Figure 2 left we provide the full learning curves (improvement of the value of the starting state across iterations), summarized in Figure 1.

In Figure 2 center we show the sample efficiency, measured as *area under the curve* (AUC), across increasing starting values of the noise schedule. We see that the dominance of $\mathcal{I}_{\text{opt}}$ grows significantly as the error grows.

In Figure 2 right we investigate the dependence of the operator on the hyperparameters β, τ, which show a smooth ridge behavior, concentrating around smaller β and τ. This also supports the operator's ability to exploit σ_A, as the effect of σ_A reduces with the size of τ, and the larger τ the worse the performance.

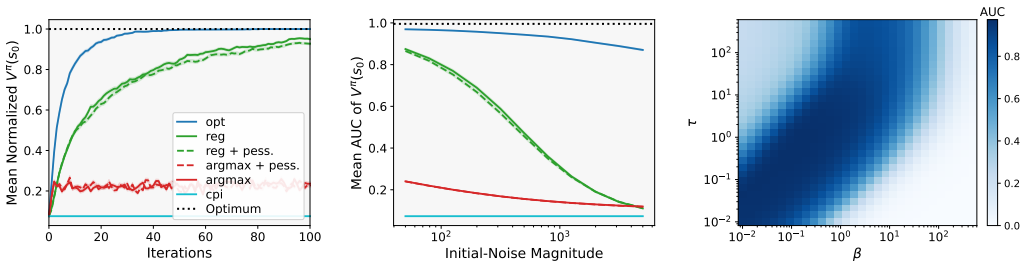

Figure 2: Policy Iteration experiments with different policy improvement operators and uncertain evaluation, mean and 95% Gaussian CI. **Left:** Example learning curves. **Center:** Sample-efficiency vs. starting noise magnitude, measured as Area Under the Curve (AUC) of learning curves until 100 iterations (higher is better). **Right:** AUC vs. hyperparameter combination. The performance landscape exhibits a single broad ridge of high performance, with performance degrading smoothly away from the ridge.

F.1.2 Generalized Policy Iteration with $k = 1$

In the majority of practical RL settings, we would not expect the evaluation to be truly unbiased (Hüllermeier & Waegeman, 2021). To investigate the behavior of $\mathcal{I}_{\text{opt}}$ in a setting that violates this idealized unbiased-evaluation assumption while remaining controlled (decoupled from additional effects such as exploration, whether there are true Gaussian components to the error, etc.), we repeat the Policy Iteration experiment presented in Figure 2 exactly but with $k = 1$. That is, rather than exact evaluation followed by Gaussian noise, we use a single Bellman backup followed by the same noise, introducing approximation and bootstrapping error relative to Q^{π_i} in the manner of Value Iteration and TD learning. The results are presented in Figure 3.

In Figure 3 left we see a single set of learning curves, where the conclusion holds: $\mathcal{I}_{\text{opt}}$ remains the best performer, albeit with a smaller advantage.

In Figure 3 center we see the behavior with growing error schedules. Although $\mathcal{I}_{\text{opt}}$ remains the best performer, the advantage, which previously increased with error, now remains much more constant.

In Figure 3 right we see that the behavior of the hyperparameters remains similar in that it is roughly a ridge, however the values of the optimal τ and β are now much larger.

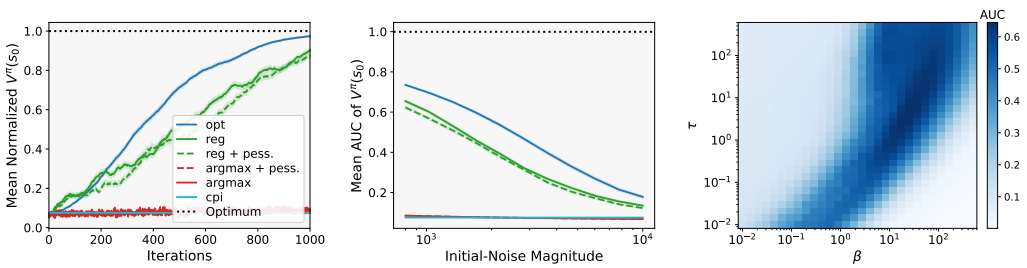

Figure 3: Policy Iteration with a 1-step Bellman update ($k = 1$), otherwise identical to Figure 2.

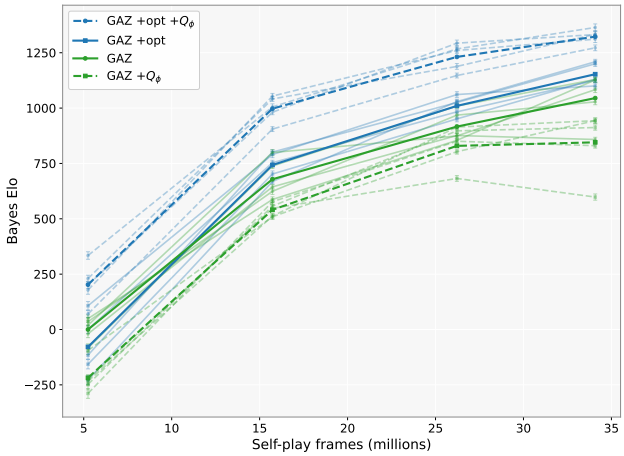

Figure 4: Playing strength in training on 9×9 Go with $M = 64$. Every checkpoint of every agent and every seed plays every other resulting in a single set of Elo scores. Bold curves are the mean over 5 training seeds per agent. Faint curves are the seeds themselves, with 95% BayesElo CI.

F.2 Discrete-action model-based, additional results

The contribution of $\mathcal{I}_{\text{opt}}$ to search-based policy improvement can be evaluated in different ways. In Figure 1 we include the comprehensive evaluation, comparing the strength of the agent that learns with each search variant, and then evaluates with it. We include checkpoints during training of all seeds of all agents in a single tournament below. Additionally, the contribution of $\mathcal{I}_{\text{opt}}$ to search based policy improvement can be evaluated more directly by comparing the playing strength of different MCTS variants using a *single* set of DNNs $\pi_\theta, V_\psi, Q_\phi$ in all variants. We include this comparison second. Since this comparison is much less compute intensive, we are able to expand it along an additional MCTS variant, PUCT, which is used by vanilla AlphaZero and in an additional environment, chess, and along different search-budgets M. Additionally, we include an investigation of the behavior of the error in MCTS with DNNs, where we find that indeed the Q_ϕ residuals have higher variance than the V_ψ residuals.

F.2.1 Elo-based learning curves across agents and seeds

We include the complete learning curves across seeds in Figure 4. All training checkpoints of all agents play against all other checkpoints and each other, forming true Bayes-Elo curves.

F.2.2 Playing strength across budgets M and games 9x9 Go and chess

We evaluate the effect of different operators used for search in MCTS for playing strength directly. We use the same set of DNNs for all agents to isolate the effect of the search operator and avoid confounding it with differences in network training. The DNNs were trained with GumbelAlphaZero rather than with $\mathcal{I}_{\text{opt}}$. The results are presented in Figure 5.

Across all M and both games, $\mathcal{I}_{\text{opt}}$ paired with Q_ϕ outperforms all other variants. Across most values of M, $\mathcal{I}_{\text{opt}}$ matches or outperforms $\mathcal{I}_{\text{reg}}$. However, this does not hold for small values of $M = 8, 24$, demonstrating a certain apparent dependence of $\mathcal{I}_{\text{opt}}$. We note that only one set of hyperparameters was used for all agents, so we would not extract the conclusion here that one agent is better than the others in a certain M, but rather that the target budget should be taken into account when tuning the hyper parameters of each agent (see the next subsection in this Appendix, discussing tuning the parameters of $\mathcal{I}_{\text{opt}}$).

PUCT is the worst performer in almost all domains, but benefits much more from Q_ϕ than $\mathcal{I}_{\text{reg}}$, which could be expected due to its explicit dependence on a Q_ϕ prediction per state and action, and lack of the completed-Q-values mechanism developed by Danihelka et al. (2022) for GumbelMCTS. Nonetheless, PUCT + Q_ϕ never competed with $\mathcal{I}_{\text{opt}}$ in playing strength in these experiments.

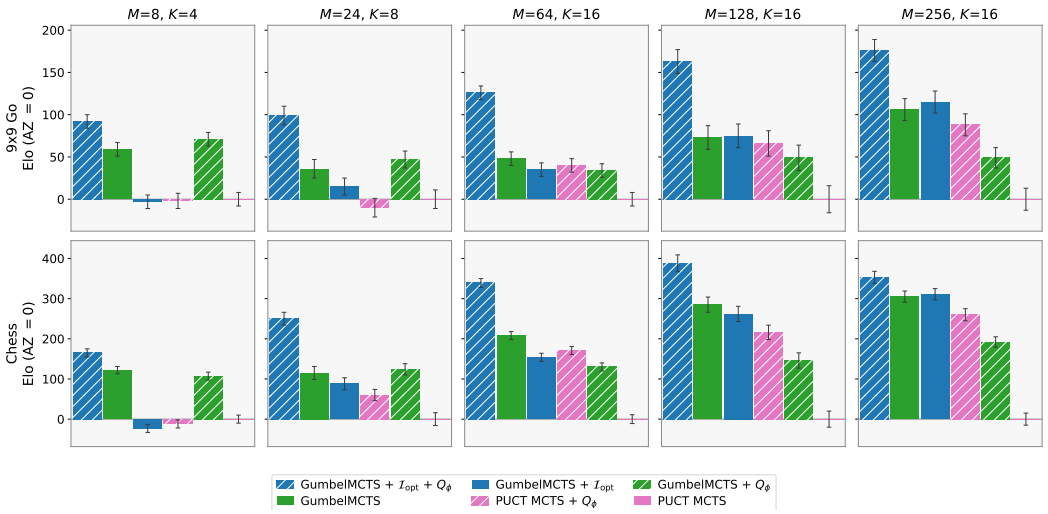

Figure 5: Playing strength per agent with the same set of DNNs. Bayes elo and 95% CIs.

F.3 Investigation of error in MCTS with DNNs

In Section 6 we hypothesize that $\mathcal{I}_{\text{opt}}$ can be expected to benefit significantly from a Q_ϕ critic over the completed-Q-values mechanism used by baseline GumbelAlphaZero (GAZ), for two complementary reasons. On the one hand, the increased information provided by a unique Q^{π_θ} prediction per state and action, compared to a single prediction of $V^{\pi_\theta}(s)$ replacing the prediction of the value of every action. On the other hand, the presence of larger prediction variance in Q_ϕ for which $\mathcal{I}_{\text{opt}}$ is designed to account and $\mathcal{I}_{\text{reg}}$ is not. In this section, we investigate the behavior of the error empirically and whether this hypothesis is supported.

In addition, we investigate to what extent the DNN evaluation residuals agree with the heteroscedastic Gaussian model used by $\mathcal{I}_{\text{opt}}$. Since we do not have access to the ground-truth Q^{π_θ} or V^{π_θ} in 9x9 Go, we use one-step TD-style residuals as a proxy for evaluation error. We take a single set of DNNs $\pi_\theta, Q_\phi, V_\psi$ and 8,192 positions drawn without replacement from the opening book. For every legal action we compute the one-step bootstrap target $T(s,a) = r(s,a) + \gamma V_\psi(s')$ and measure the signed residuals e_V, e_Q as follows:

$$e_V(s) \;=\; \textstyle\sum_a \pi(a|s)\, T(s,a) \;-\; V_\psi(s), \qquad e_Q(s,a) \;=\; T(s,a) \;-\; Q_\phi(s,a), \tag{153}$$

one per state and one per action at the state, respectively. The results are reported in Figure 6.

On the left, we plot the variance, computed as $\frac{1}{|S|}\sum_{(s,a)\in S} e_Q(s,a)^2 - \left(\frac{1}{|S|}\sum_{(s,a)\in S} e_Q(s,a)\right)^2$ over each action subset S, and the same over states for e_V. As expected, the variance is larger for Q_ϕ than for V_ψ, and increases as the degree of search and the quality (and weight) of the targets provided to the Q_ϕ critic reduces.

On the right, we center and scale e_Q per position and compare the standardized residual distribution to $\mathcal{N}(0,1)$. The bulk is approximately Gaussian, with heavier tails, supporting the heteroscedastic Gaussian model as a useful approximation.

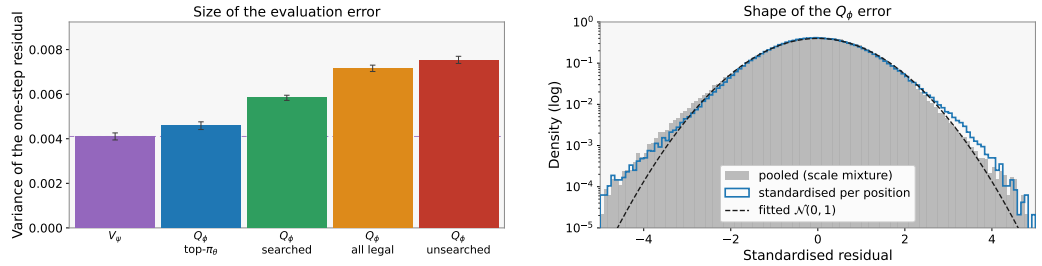

Figure 6: **Left:** Variance based on one-step TD residual proxies for evaluation error on different opening-book positions under one network on 9×9 Go with 95% percentile bootstrap over positions. **Right:** the shape of the Q_ϕ residual, on a log density scale. In grey the raw pooled residual, a scale mixture over actions of differing σ. In blue is the same data centered and scaled per position; dashed is the fitted $\mathcal{N}(0, 1)$.

F.4 Learning curves on model-free online continuous control (SAC on DMC)

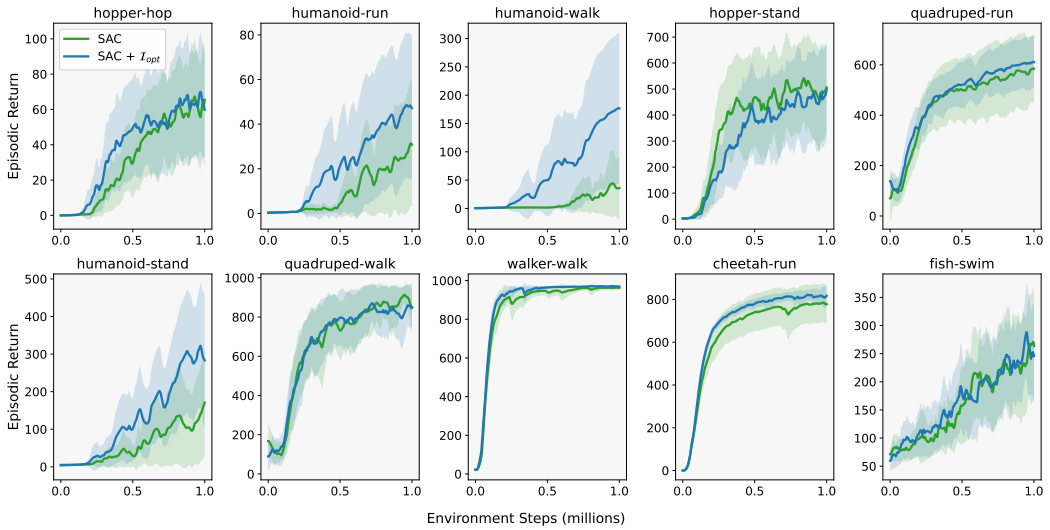

Figure 7: Per-environment learning curves on the 10 DeepMind Control Suite tasks, showing unnormalised episodic return against environment steps. Mean across 10 seeds and 95% interval.

F.5 Offline RL: per-environment results

We report the final performance per environment in Table 1. We include results with hyperparameters optimized per datasets, as is chosen by the baseline, as well as with one set of hyperparameters across all datasets, to demonstrate that the $+\mathcal{I}_{\text{opt}}$ algorithm does not appear to be sensitive to a unique per-environment configuration.

Table 1: Per-environment offline RL results on the four D4RL medium-replay datasets. Entries are the D4RL score of the final policy evaluated over 56 episodes, mean and 95% Gaussian CI over 20 runs per arm per environment (10 for the single-pair setting on halfcheetah). Bold marks the highest mean in each row together with every entry that is not significantly worse than it, at $p < 0.05$ under a two-sided Welch t-test over runs (Welch, 1947); several bold entries in a row therefore indicate a tie.

	ReBRAC $+\mathcal{I}_{\text{opt}}$		
Dataset	per-dataset τ^2, β	single pair $\tau^2 = 2, \beta = 0.5$	**ReBRAC**
halfcheetah-medium-replay	$\mathbf{50.86 \pm 1.09}$	$\mathbf{50.30 \pm 1.43}$	49.65 ± 0.28
hopper-medium-replay	$\mathbf{87.00 \pm 8.80}$	$\mathbf{87.00 \pm 8.80}$	$\mathbf{88.24 \pm 6.80}$
walker2d-medium-replay	$\mathbf{84.69 \pm 3.19}$	$\mathbf{84.69 \pm 3.19}$	78.75 ± 3.91
ant-medium-replay	$\mathbf{40.17 \pm 7.89}$	$\mathbf{40.17 \pm 7.89}$	17.09 ± 3.05

G HYPERPARAMETERS

G.1 HYPERPARAMETER TUNING

To keep the comparisons as meaningful as possible, we use each baseline's pre-tuned hyperparameters and network architecture unchanged for both the baseline and its $+\mathcal{I}_{\text{opt}}$ variant, apart from the additional $\mathcal{I}_{\text{opt}}$ hyperparameters. This controls the comparison by changing only the operator. It also leaves open the possibility that the $+\mathcal{I}_{\text{opt}}$ variants could perform better if the baseline hyperparameters were retuned jointly with τ^2 and β.

As a result, outside of the policy iteration experiments, where all hyperparameters were tuned from scratch, the only hyperparameters that were tuned in this work are $\mathcal{I}_{\text{opt}}$'s τ^2, β. To further adhere to each algorithm's standard tuning conditions τ^2, β are tuned per environment for algorithms which are tuned per-environment (ReBRAC), and one set of hyperparameters τ^2, β is tuned and used for an entire domain for the algorithms which are tuned in this way (all the rest: SAC, GumbelAlphaZero and the Policy Iteration experiments).

In the Policy Iteration experiments, all hyperparameters for all algorithms were tuned with an extensive grid search across the 20 environments and the single set of hyperparameters that performed best was chosen. The experiment was then repeated for 100 seeds in all environments and only the repeated seeds are shown.

GumbelAlphaZero uses the same set of GumbelMCTS hyperparameters across different board games, for which it was tuned. Following this choice, we have used one set of hyperparameters in all environments. To tune τ^2 and β we have chosen parameters that induce growth $\frac{1}{\eta(s,a)}$ with $N(s,a)$, which approximately match the growth of the baseline. With this starting point, we've then chosen a set of hyperparameters τ^2, β that result in roughly matching growth within the range of popular evaluation resources used for MCTS $M = (8, 200)$, from below. Our logic was that as $\frac{1}{\sigma^2(s,a)}$ increases independently per action, rather than for all actions at the same time, the growth rate should be lower than that of $\frac{1}{\eta(s)}$. We've then conducted an elo evaluation sweep across hyperparameter combinations around these values with $M = 64$ in 9x9 Go, chosen one set of uniformly well performing hyperparameter, and repeated all experiments with them.

The SAC baseline uses one set of hyperparameters across all environments. In accordance with this choice, we've done a sweep across hyperparameter combinations for $\mathcal{I}_{\text{opt}}$ across a subset of the environments. The single set of dominating hyperparameters was chosen and then the experiment was repeated for 10 seeds in all environments.

In the offline RL experiments, the baseline's hyperparameters were tuned per environment. Following the baselines choice, τ^2, β were tuned using a sweep across hyperparameter combinations per environment. The best performing hyperparameter combination was chosen per environment and the experiments were repeated for 20 additional seeds for this set of hyperparameters. However, we have also found that the same set of hyperparameters performed comparably to a unique set of hyperparameters per environment, and we present these results as well in Table 1 in Appendix F.5.

Table 2 collects how many independent runs stand behind every number in Figure 1.

Table 2: Number of seeds included in each experiment.

Experiment	**Number of Seeds per Agent**
Policy iteration	100 seeds $\times$ 20 environments
Model-based, discrete	5 training seeds
Model-free, continuous	10 seeds $\times$ 10 environments
Offline	20 seeds $\times$ 4 environments

G.2 Hyperparameter tables

Table 3: Hyperparameters for the Policy Iteration experiments.

Component	**Parameter**	**Value**
$\mathcal{I}_{\text{opt}}$	Advantage prior variance (τ^2)	$(0.17)^2$
	KL regularization strength (β)	0.07
$\mathcal{I}_{\text{reg}}$	Temperature (η)	4.89
Environments	Grid size ($\lvert\mathcal{S}\rvert \times \lvert\mathcal{A}\rvert$)	25×5
Noise schedule (Eq. 135)	Initial noise bound (a_0)	50
	Decay (γ_{dec})	0.998
	Variance width (b)	5

Table 4: Hyperparameters for the discrete-action model-based GAZ experiments.

Component	Parameter	Value
$\mathcal{I}_{\text{opt}}$	Advantage prior variance (τ^2)	0.08
	KL regularization strength (β)	0.01
$\mathcal{I}_{reg}$ (GumbelMCTS)	c_{visit}	50
	c_{scale}	1.0
PUCT (AlphaZero)	c_{init}	1.25
	c_{base}	19652
	Dirichlet root noise fraction	0.25
	Dirichlet root noise concentration (α)	0.3
Search	Budget M, training runs	64
	Budget M, fixed-network sweep	$8, 24, 64, 128, 256$
	Num. searched actions at the root (K)	16 (unless otherwise specified)
	Particles (N)	1
	Root Gumbel scale	0.0
	Discount (γ)	0.997
Q_ϕ head	Loss weight (λ_q)	0.1
	Bootstrapped actions (L)	16
Network	Residual blocks	6
	Channels	128
	Convolution kernel	3×3, stride 1
	Normalisation / activation	batch norm / ReLU
Training	Self-play frames per iteration (1024 games $\times$ 256 steps)	262,144
	Sampled opening plies	8
	Training minibatch	256
Optimiser	Algorithm	AdamW
	Learning rate	10^{-3} (constant)
	First-moment decay	0.9
	Other Adam settings	`optax` defaults

K (the number of actions searched by SH, see (Danihelka et al., 2022)) is capped at 16; at the two smallest budgets sequential halving's divisibility constraints force it lower, giving $K = 4$ at $M = 8$ and $K = 8$ at $M = 24$. The root Gumbel scale is 0 following Table 2 of Danihelka et al. (2022), whose board-game evaluations use 0; the value of 1 in their Table 4 applies to training self-play rather than evaluation.

The body follows Silver et al. (2018): a batch-normalised, rectified 3×3 convolution, then 6 residual blocks of two batch-normalised, rectified 3×3 convolutions with a skip connection, all at 128 channels. The value head is a 1×1 convolution to one channel, batch norm and ReLU, a linear layer of width 128, and a $\tanh$ linear layer to a scalar. The policy and Q_ϕ heads have the same shape as each other and differ by game: on 9×9 Go a 3×3 convolution, batch norm and ReLU, then a linear layer to the 82 actions; on chess a 3×3 convolution, batch norm and ReLU, then a second 3×3 convolution to the 73 plane-encoded actions. Q_ϕ is $\tanh$, one value per action, and is built in every arm but trained only where a Q head is used. Inputs are `pgx`'s encodings: 17 planes for 9×9 Go and 119 for chess.

Table 5: Hyperparameters for the online classical continuous control SAC experiments

Component	**Parameter**	**Value**
$\mathcal{I}_{\text{opt}}$	Advantage prior variance (τ^2)	0.4
	Num. actions sampled for $V_{\phi_i}(s)$	32
SAC	Optimizer	Adam
	Actor learning rate	$3 \cdot 10^{-4}$
	Critic and entropy learning rate	10^{-3}
	Batch size	256
	Replay buffer size	10^6
	Steps before learning starts	10^4
	Discount (γ)	0.99
	Target smoothing coefficient	0.005
	Actor update interval	2
	Target network update interval	1
	Entropy coefficient (α)	autotuned
	Target entropy	$-\dim(\mathcal{A})$
	Critics	2
Network	Actor trunk	2×256, ReLU
	Actor heads	mean and $\log \sigma$
	$\log \sigma$ range	$[-5, 2]$
	Action squashing	$\tanh$
	Critic trunk (input $[s, a]$)	2×256, ReLU
	Normalisation	none
	Initialisation	PyTorch defaults

Table 6: Hyperparameters for the offline RL ReBRAC experiments.

Component	Parameter	Value
$\mathcal{I}_{\text{opt}}$	Advantage prior variance (τ^2), halfcheetah-medium-replay	8
	Advantage prior variance (τ^2), hopper-medium-replay	2
	Advantage prior variance (τ^2), walker2d-medium-replay	2
	Advantage prior variance (τ^2), ant-medium-replay	2
	KL regularization strength (β), all datasets	0.5
ReBRAC	Critics (`num_critics`)	2
	Optimizer	Adam ($\epsilon = 10^{-5}$)
	Learning rate (actor and critics)	10^{-3}
	Batch size	1024
	Gradient steps	10^6
	Discount (γ)	0.99
	Target smoothing coefficient	0.005
	Target policy noise	0.2
	Target policy noise clip	0.5
	Critic updates per actor update	2
Network	Actor trunk	3×256, ReLU
	Actor output	$\dim(\mathcal{A})$, $\tanh$
	Critic trunk (input $[s, a]$)	3×256, ReLU, LayerNorm
	Critic weight sharing	none
	Observation normalisation	no
	Hidden bias initialisation	0.1
	Output initialisation, actor	$\mathcal{U}(\pm 10^{-3})$
	Output initialisation, critics	$\mathcal{U}(\pm 3 \cdot 10^{-3})$
	Target networks	actor and critics
Actor BC coefficient	β_1, halfcheetah-medium-replay	0.01
	β_1, hopper-medium-replay	0.05
	β_1, walker2d-medium-replay	0.05
	β_1, ant-medium-replay	0.001
Critic BC coefficient	β_2, halfcheetah-medium-replay	0.001
	β_2, hopper-medium-replay	0.5
	β_2, walker2d-medium-replay	0.01
	β_2, ant-medium-replay	0.01