HAL open science

Leverage the Average: an Analysis of KL Regularization in Reinforcement Learning

Nino Vieillard, Tadashi Kozuno, Bruno Scherrer, Olivier Pietquin, Rémi Munos, Matthieu Geist

To cite this version:

Nino Vieillard, Tadashi Kozuno, Bruno Scherrer, Olivier Pietquin, Rémi Munos, et al.. Leverage the Average: an Analysis of KL Regularization in Reinforcement Learning. NeurIPS - 34th Conference on Neural Information Processing Systems, Dec 2020, Vancouver / Online, Canada. hal-03137351

HAL Id: hal-03137351

https://inria.hal.science/hal-03137351v1

Submitted on 10 Feb 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Leverage the Average: an Analysis of KL Regularization in Reinforcement Learning

Nino Vieillard
Google Research, Brain Team
vieillard@google.com

Tadashi Kozuno*
Okinawa Institute of Science and Technology
tadashi.kozuno@oist.jp

Bruno Scherrer
Université de Lorraine, CNRS, Inria
IECL, F-54000 Nancy, France
bruno.scherrer@inria.fr

Olivier Pietquin
Google Research, Brain Team
pietquin@google.com

Rémi Munos
DeepMind
munos@google.com

Matthieu Geist
Google Research, Brain Team
mfgeist@google.com

Abstract

Recent Reinforcement Learning (RL) algorithms making use of Kullback-Leibler (KL) regularization as a core component have shown outstanding performance. Yet, only little is understood theoretically about why KL regularization helps, so far. We study KL regularization within an approximate value iteration scheme and show that it implicitly averages q-values. Leveraging this insight, we provide a very strong performance bound, the very first to combine two desirable aspects: a linear dependency to the horizon (instead of quadratic) and an error propagation term involving an averaging effect of the estimation errors (instead of an accumulation effect). We also study the more general case of an additional entropy regularizer. The resulting abstract scheme encompasses many existing RL algorithms. Some of our assumptions do not hold with neural networks, so we complement this theoretical analysis with an extensive empirical study.

1 Introduction

In Reinforcement Learning (RL), Kullback-Leibler (KL) regularization consists in penalizing a new policy from being too far from the previous one, as measured by the KL divergence. It is at the core of efficient deep RL algorithms, such as Trust Region Policy Optimization (TRPO) [37] (motivated by trust region constraints) or Maximum a Posteriori Policy Optimization (MPO) [2] (arising from the view of control as probabilistic inference [26, 16]), but without much theoretical guarantees. Recently, Geist et al. [20] have analyzed algorithms operating in the larger scope of regularization by Bregman divergences. They concluded that regularization doesn't harm in terms of convergence, rate of convergence, and propagation of errors, but these results are not better than the corresponding ones in unregularized approximate dynamic programming (ADP).

Building upon their formalism, we show that using a KL regularization implicitly averages the successive estimates of the q-function in the ADP scheme. Leveraging this insight, we

*Work done while at DeepMind.

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

provide a strong performance bound, the very first to combine two desirable aspects: 1) it has a linear dependency to the time horizon $(1-\gamma)^{-1}$, 2) it exhibits an error averaging property of the KL regularization. The linear dependency in the time horizon contrasts with the standard quadratic dependency of usual ADP, which is tight [35]. The only approaches achieving a linear dependency we are aware of make use of non-stationary policies [8, 35] and never led to practical deep RL algorithms. More importantly, the bound involves the norm of the average of the errors, instead of a discounted sum of the norms of the errors for classic ADP. This means that, while standard ADP is not guaranteed to converge for the ideal case of independent and centered errors, KL regularization allows convergence to the optimal policy in that case. The sole algorithms that also enjoy this compensation of errors are Dynamic Policy Programming (DPP) [7] and Speedy Q-learning (SQL) [6], that also build (implicitly) on KL regularization, as we will show for SQL. However, their dependency to the horizon is quadratic, and they are not well amenable to a deep learning setting [43].

We also study the case of an additional entropy regularization, usual in practical algorithms, and specifically the interplay between both regularizations. The resulting abstract framework encompasses a wide variety of existing RL algorithms, the connections between some of them being known [20], but many other being new, thanks to the implicit average of q-values. We highlight that, even though our analysis covers the case where only the entropy regularization is considered, it does not explain why it helps without an additional KL term. Some argue that having a higher entropy helps exploration [38], other that it has beneficial effects on the optimization landscape [3], but it also biases the solution of the MDP [20].

Our analysis requires some assumptions, notably that the regularized greedy step is done without approximation. If this is reasonable with discrete actions and a linear parameterization, it does not hold when neural networks are considered. Given their prevalence today, we complement our thorough analysis with an extensive empirical study, that aims at observing the core effect of regularization in a realistic deep RL setting.

2 Background and Notations

Let Δ_X be the set of probability distributions over a finite set X and Y^X the set of applications from X to the set Y. An MDP is a tuple $\{\mathcal{S}, \mathcal{A}, P, r, \gamma\}$ with $\mathcal{S}$ the finite state space, $\mathcal{A}$ the finite set of actions, $P \in \Delta_{\mathcal{S}}^{\mathcal{S}\times\mathcal{A}}$ the Markovian transition kernel, $r \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ the reward function bounded by $r_{\max}$, and $\gamma \in (0,1)$ the discount factor. For $\tau \geq 0$, we write $v_{\max}^{\tau} = \frac{r_{\max}+\tau \ln|\mathcal{A}|}{1-\gamma}$ and simply $v_{\max} = v_{\max}^{0}$. We write $\mathbf{1} \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ the vector whose components are all equal to 1. A policy $\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}$ associates a distribution over actions to each state. Its (state-action) value function is defined as $q_\pi(s,a) = \mathbb{E}_\pi\left[\sum_{t=0}^{\infty}\gamma^t r(S_t,A_t)|S_0=s, A_0=a\right]$, $\mathbb{E}_\pi$ being the expectation over trajectories induced by π. Any optimal policy satisfies $\pi_* \in \operatorname{argmax}_{\pi\in\Delta_{\mathcal{A}}^{\mathcal{S}}} q_\pi$ (all scalar operators applied on vectors should be understood point-wise), and $q_* = q_{\pi_*}$. The following notations will be useful. For $f_1, f_2 \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$,$\langle f_1, f_2\rangle = \left(\sum_a f_1(s,a)f_2(s,a)\right)_s \in \mathbb{R}^{\mathcal{S}}$. This will be used with q-values and (log) policies. We write P_π the stochastic kernel induced by π, and for $q \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ we have $P_\pi q = \left(\sum_{s'} P(s'|s,a)\sum_{a'}\pi(a'|s')q(s',a')\right)_{s,a} \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$. For $v \in \mathbb{R}^{\mathcal{S}}$, we also define $Pv = \left(\sum_{s'} P(s'|s,a)v(s')\right)_{s,a} \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$, hence $P_\pi q = P\langle \pi, q\rangle$.

The Bellman evaluation operator is $T_\pi q = r + \gamma P_\pi q$, its unique fixed point being q_π. The set of greedy policies w.r.t. $q \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ is $\mathcal{G}(q) = \operatorname{argmax}_{\pi\in\Delta_{\mathcal{A}}^{\mathcal{S}}}\langle q, \pi\rangle$. A classical approach to estimate an optimal policy is Approximate Modified Policy Iteration (AMPI) [34, 36],

$$\begin{cases}\pi_{k+1} \in \mathcal{G}(q_k) \\ q_{k+1} = (T_{\pi_{k+1}})^m q_k + \epsilon_{k+1}\end{cases},$$

which reduces to Approximate Value Iteration (AVI, $m=1$) and Approximate Policy Iteration (API, $m=\infty$) as special cases. The term ϵ_{k+1} accounts for errors made when applying the Bellman operator. For example, the classic DQN [27] is encompassed by this abstract ADP scheme, with $m=1$ and the error arising from fitting the neural network (regression step of DQN). The typical use of m-step rollouts in (deep) RL actually corresponds to an AMPI scheme with $m>1$. Next, we add regularization to this scheme.

3 Regularized MPI

In this work, we consider the entropy $\mathcal{H}(\pi) = -\langle \pi, \ln \pi \rangle \in \mathbb{R}^{\mathcal{S}}$ and the KL divergence $\mathrm{KL}(\pi_1 || \pi_2) = \langle \pi_1, \ln \pi_1 - \ln \pi_2 \rangle \in \mathbb{R}^{\mathcal{S}}$. First, we introduce a slight variation of the Mirror Descent MPI scheme [20] (handling both KL and entropy penalties, based on q-values).

Mirror Descent MPI. For $q \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ and an associated policy $\mu \in \Delta_{\mathcal{A}}^{\mathcal{S}}$, we define the regularized greedy policy as $\mathcal{G}_{\mu}^{\lambda,\tau}(q) = \mathrm{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \left(\langle \pi, q \rangle - \lambda \, \mathrm{KL}(\pi || \mu) + \tau \mathcal{H}(\pi) \right)$. Observe that with $\lambda = \tau = 0$, we get the usual greediness. Notice also that with $\lambda = 0$, the KL term disappears, so does the dependency to μ. In this case we write $\mathcal{G}^{0,\tau}$. We also account for the regularization in the Bellman evaluation operator. Recall that the standard operator is $T_\pi q = r + \gamma P \langle \pi, q \rangle$. Given the form of the regularized greediness, it is natural to replace the term $\langle \pi, q \rangle$ by the regularized one, giving $T_{\pi|\mu}^{\lambda,\tau} q = r + \gamma P \left(\langle \pi, q \rangle - \lambda \, \mathrm{KL}(\pi || \mu) + \tau \mathcal{H}(\pi) \right)$. These lead to the following MD-MPI(λ,τ) scheme. It is initialized with $q_0 \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ such that $\|q_0\|_\infty \leq v_{\max}$ and with π_0 the uniform policy, without much loss of generality (notice that the greedy policy is unique whenever $\lambda > 0$ or $\tau > 0$):

$$\begin{cases} \pi_{k+1} = \mathcal{G}_{\pi_k}^{\lambda,\tau}(q_k) \\ q_{k+1} = (T_{\pi_{k+1}|\pi_k}^{\lambda,\tau})^m q_k + \epsilon_{k+1} \end{cases} . \tag{1}$$

Dual Averaging MPI. We provide an equivalent formulation of scheme (1). This will be the basis of our analysis, and it also allows drawing connections to other algorithms, originally not introduced as using a KL regularization. All the technical details are provided in the Appendix, but we give an intuition here, for the case $\tau = 0$ (no entropy). Let $\pi_{k+1} = \mathcal{G}_{\pi_k}^{\lambda,0}(q_k)$. This optimization problem can be solved analytically, yielding $\pi_{k+1} \propto \pi_k \exp \frac{q_k}{\lambda}$. By direct induction, π_0 being uniform, we have $\pi_{k+1} \propto \pi_k \exp \frac{q_k}{\lambda} \propto \cdots \propto \exp \frac{1}{\lambda} \sum_{j=0}^{k} q_j$. This means that penalizing the greedy step with a KL divergence provides a policy being a softmax over the scaled sum of all past q-functions (no matter how they are obtained). This is reminiscent of dual averaging in convex optimization, hence the name.

We now introduce the Dual Averaging MPI (DA-MPI) scheme. Contrary to MD-MPI, we have to distinguish the cases $\tau = 0$ and $\tau \neq 0$. DA-MPI(λ,0) and DA-MPI(λ,$\tau > 0$) are

$$\begin{cases} \pi_{k+1} = \mathcal{G}^{0,\frac{\lambda}{k+1}}(h_k) \\ q_{k+1} = (T_{\pi_{k+1}|\pi_k}^{\lambda,0})^m q_k + \epsilon_{k+1} \\ h_{k+1} = \frac{k+1}{k+2} h_k + \frac{1}{k+2} q_{k+1} \end{cases} \text{and} \begin{cases} \pi_{k+1} = \mathcal{G}^{0,\tau}(h_k) \\ q_{k+1} = (T_{\pi_{k+1}|\pi_k}^{\lambda,\tau})^m q_k + \epsilon_{k+1} \\ h_{k+1} = \beta h_k + (1-\beta) q_{k+1} \text{ with } \beta = \frac{\lambda}{\lambda+\tau} \end{cases} , \tag{2}$$

with $h_0 = q_0$ for $\tau = 0$ and $h_{-1} = 0$ for $\tau > 0$. The following result is proven in Appx. C.1.

Proposition 1. *For any $\lambda > 0$, MD-MPI(λ,0) and DA-MPI(λ,0) are equivalent (but not in the limit $\lambda \to 0$). Moreover, for any $\tau > 0$, MD-MPI(λ,τ) and DA-MPI(λ,τ) are equivalent.*

Table 1: Algorithms encompassed by MD/DA-MPI (in italic if new compared to [20]).

	only entropy	only KL	both
reg. eval.	Soft Q-learning [17, 21], SAC [22], *Mellowmax* [5]	DPP [7], *SQL* [6]	*CVI* [25], *AL* [9, 11]
unreg. eval.	*softmax DQN* [41]	TRPO [37], MPO [2], *Politex* [1], *MoVI* [43]	*softened LSPI* [31], *MoDQN* [43]

Links to existing algorithms. Equivalent schemes (1) and (2) encompass (possibly variations of) many existing RL algorithms (see Tab. 1 and details below). Yet, we think important to highlight that many of them don't consider regularization in the evaluation step (they use $T_{\pi_{k+1}}$ instead of $T_{\pi_{k+1}|\pi_k}^{\lambda,\tau}$), something we abbreviate as "*w/o*". If it does not preclude convergence in the case $\tau = 0$ [20, Thm. 4], it is known for the case $\tau > 0$ and $\lambda = 0$ that the resulting Bellman operator may have multiple fixed points [5], which is not desirable. Therefore, we only consider a regularized evaluation for the analysis, but we will

compare both approaches empirically. Now, we present the approaches encompassed by scheme (1) (see also Appx. B.1). Soft Actor Critic (SAC) [22] and soft Q-learning [21] are variations of MD-MPI(0,τ), as is softmax DQN [41] but *w/o*. The Mellowmax policy [5] is equivalent to MD-MPI(0,τ). TRPO and MPO are variations of MD-MPI(λ,0), *w/o*. DPP [7] is almost a reparametrization of MD-MPI(λ,0), and Conservative Value Iteration (CVI) [25] is a reparametrization of MD-MPI$_1$(λ,τ), which consequently also generalizes Advantage Learning (AL) [9, 11]. Next, we present the approaches encompassed by schemes (2) (see also Appx. B.2). Politex [1] is a PI scheme for the average reward case, building upon prediction with expert advice. In the discounted case, it is DA-MPI(λ,0), *w/o*. Momentum Value Iteration (MoVI) [43] is a limit case of DA-MPI(λ,0), *w/o*, as $\lambda \to 0$, and its practical extension to deep RL momentum DQN (MoDQN) is a limit case of DA-MPI(λ,τ), *w/o*. SQL [6] is a limit case of DA-MPI(λ, 0) as $\lambda \to 0$. Softened LSPI [30] deals with zero-sum Markov games, but specialized to single agent RL it is a limit case of DA-MPI(λ,τ), *w/o*.

4 Theoretical Analysis

Here, we analyze the propagation of errors of MD-MPI, through the equivalent DA-MPI, for the case $m = 1$ (that is regularized VI, the extension to $m > 1$ remaining an open question). We provide component-wise bounds that assess the quality of the learned policy, depending on $\tau = 0$ or not. From these, ℓ_p-norm bounds could be derived, using [36, Lemma 5].

Analysis of DA-VI(λ,0). This corresponds to scheme (2), left, with $m = 1$. The following Thm. is proved in Appx. C.2.

Theorem 1. *Define* $E_k = -\sum_{j=1}^{k} \epsilon_j$, $A_k^1 = (I - \gamma P_{\pi_*})^{-1} - (I - \gamma P_{\pi_k})^{-1}$ *and* $g^1(k) = \frac{4}{1-\gamma}\frac{v_{max}^{\lambda}}{k}$. *Assume that* $\|q_k\|_\infty \leq v_{max}$. *We have* $0 \leq q_* - q_{\pi_k} \leq \left|A_k^1 \frac{E_k}{k}\right| + g^1(k)\mathbf{1}$.

Remark 1. *The assumption* $\|q_k\|_\infty \leq v_{max}$ *is not strong. It can be enforced by simply clipping the result of the evaluation step in* $[-v_{max}, v_{max}]$. *See also Appx. C.3.*

To ease the discussion, we express an ℓ_∞-bound as a direct corollary of Thm. 1:

$$\|q_* - q_{\pi_k}\|_\infty \leq \frac{2}{1-\gamma}\left\|\frac{1}{k}\sum_{j=1}^{k}\epsilon_j\right\|_\infty + \frac{4}{1-\gamma}\frac{v_{\max}^{\lambda}}{k}.$$

We also recall the typical propagation of errors of AVI without regularization (*e.g.* [36], we scale the sum by $1-\gamma$ to make explicit the normalizing factor of a discounted sum):

$$\|q_* - q_{\pi_k}\|_\infty \leq \frac{2\gamma}{(1-\gamma)^2}\left((1-\gamma)\sum_{j=1}^{k}\gamma^{k-j}\|\epsilon_j\|_\infty\right) + \frac{2}{1-\gamma}\gamma^k v_{\max}.$$

For each bound, the first term can be decomposed as a factor, the *horizon term* ($(1-\gamma)^{-1}$ is the average horizon of the MDP), scaling the *error term*, that expresses how the errors made at each iteration reflect in the final performance. The second term reflects the influence of the initialization over iterations, without errors it give the *rate of convergence* of the algorithms. We discuss these three terms.

Rate of convergence. It is slower for DA-VI(λ,0) than for AVI, $\gamma^k = o(\frac{1}{k})$. This was to be expected, as the KL term slows down the policy updates. It is not where the benefits of KL regularization arise. However, notice that for k small enough and γ close to 1, we may have $\frac{1}{k} \leq \gamma^k$. This term has also a linear dependency to λ (through $v_{\max}^{\lambda}$), suggesting that a lower λ is better. This is intuitive, a larger λ leads to smaller changes of the policy, and thus to a slower convergence.

Horizon term. We have a linear dependency to the horizon, instead of a quadratic one, which is very strong. Indeed, it is known that the square dependency to the horizon is tight for API and AVI [35]. The only algorithms based on ADP having a linear dependency we are aware of make use of non-stationary policies [35, 8], and have never led to practical (deep) RL algorithms. Minimizing directly the Bellman residual would also lead to a linear

dependency (*e.g.*, [32, Thm. 1]), but it comes with its own drawbacks [19] (*e.g.*, bias problem with stochastic dynamics, and it is not used in deep RL, as far as we know).

Error term. For AVI, the error term is a discounted *sum of the norms* of the successive estimation errors, while in our case it is the *norm of the average* of these estimation errors. The difference is fundamental, it means that the KL regularization allows for a compensation of the errors made at each iteration. Assume that the sequence of errors is a martingale difference. AVI would not converge in this case, while DA-VI(λ, 0) converges to the optimal policy ($\|\frac{1}{k}\sum_{j=1}^{k}\epsilon_j\|_\infty$ converges to 0 by the law of large numbers). As far as we know, only SQL and DPP have such an error term, but they have a worse dependency to the horizon.

Thm. 1 is the first result showing that an RL algorithm can benefit from both a linear dependency to the horizon and from an averaging of the errors, and we argue that this explains, at least partially, the beneficial effect of using a KL regularization. Notice that Thm. 4 of Geist et al. [20] applies to DA-VI(λ, 0), as they study more generally MPI regularized by a Bregman divergence. Although they bound a regret rather than $q_* - q_{\pi_k}$, their result is comparable to AVI, with a quadratic dependency to the horizon and a discounted sum of the norms of the errors. Therefore, our result significantly improves previous analyses.

We illustrate the bound with a simple experiment[2], see Fig. 1, left. We observe that AVI doesn't converge, while DA-VI(λ,0) does, and that higher values of λ slow down the convergence. Yet, they are also a bit more stable. This is not explained by our bound but is quite intuitive (policies changing less between iterations).

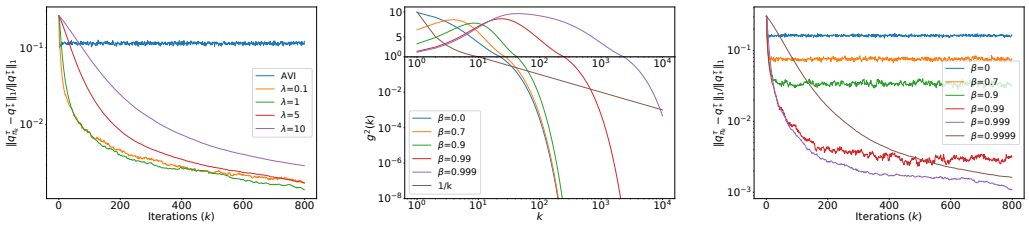

Figure 1: **Left**: behavior for Thm 1. **Middle**: function $g^2(k)$. **Right**: behavior for Thm 2.

Analysis of DA-VI(λ,τ). This is scheme (2), right, with $m = 1$. Due to the non-vanishing entropy term in the greedy step, it cannot converge to the unregularized optimal q-function. Yet, without errors and with $\lambda = 0$, it would converge to the solution of the MDP regularized by the scaled entropy (that is, considering the reward augmented by the scaled entropy). Our bound will show that adding a KL penalty does not change this. To do so, we introduce a few notations. The proofs of the following claims can be found in [20], for example. We already have defined the operator $T_\pi^{0,\tau}$. It has a unique fixed point, that we write q_π^τ. The unique optimal q-function is $q_*^\tau = \max_\pi q_\pi^\tau$. We write $\pi_*^\tau = \mathcal{G}^{0,\tau}(q_*^\tau)$ the associated unique optimal policy, and $q_{\pi_*^\tau}^\tau = q_*^\tau$. The next result is proven in Appx. C.4.

Theorem 2. *For a sequence of policies $\pi_0, \dots, \pi_k$, we define $P_{k:j} = P_{\pi_k} P_{\pi_{k-1}} \dots P_{\pi_j}$ if $j \le k$, $P_{k:j} = I$ else. We define $A_{k:j}^2 = P_{\pi_*^\tau}^{k-j} + (I - \gamma P_{\pi_{k+1}})^{-1} P_{k:j+1}(I - \gamma P_{\pi_j})$. We define $g^2(k) = \gamma^k(1 + \frac{1-\beta}{1-\gamma})\sum_{j=0}^{k}(\frac{\beta}{\gamma})^j v_{max}^\tau$, with β as defined in Eq. (2). Finally, we define $E_k^\beta = (1-\beta)\sum_{j=1}^{k}\beta^{k-j}\epsilon_j$. With these notations: $0 \le q_*^\tau - q_{\pi_{k+1}}^\tau \le \sum_{j=1}^{k}\gamma^{k-j}\left|A_{k:j}^2 E_j^\beta\right| + g^2(k)\mathbf{1}$.*

[2] We illustrate the bounds in a simple tabular setting with access to a generative model. Considering random MDPs (called Garnets), at each iteration of DA-VI we sample a single transition for each state-action couple and apply the resulting sampled Bellman operator. The error ϵ_k is the difference between the sampled and the exact operators. The sequence of these estimation errors is thus a martingale difference w.r.t. its natural filtration [6] (one can think about bounded, centered and roughly independent errors). More details about this practical setting are provided in Appx. D.

Again, to ease the discussion, we express an ℓ_∞-bound as a direct corollary of Thm. 2:

$$\|q_*^\tau - q_{\pi_{k+1}}^\tau\|_\infty \leq \frac{2}{(1-\gamma)^2}\left((1-\gamma)\sum_{j=1}^{k}\gamma^{k-j}\|E_j^\beta\|_\infty\right) + \gamma^k(1+\frac{1-\beta}{1-\gamma})\sum_{j=0}^{k}\left(\frac{\beta}{\gamma}\right)^j v_{\max}^\tau.$$

There is a square dependency to the horizon, as for AVI. We discuss the other terms.

Rate of convergence. It is given by the function g^2, defined in Thm. 2. If $\beta = \gamma$, we have $g^2(k) = 2(k+1)\gamma^k v_{\max}^\tau$. If $\beta \neq \gamma$, we have $g^2(k) = (1+\frac{1-\beta}{1-\gamma})\frac{\beta^{k+1}-\gamma^{k+1}}{\beta-\gamma}$. In all cases, $g^2(k) = o(\frac{1}{k})$, so it is asymptotically faster than in Thm. 1, but the larger the β, the slower the initial convergence. This is illustrated in Fig. 1, middle (notice that it's a logarithmic plot, except for the upper part of the y-axis).

Error rate. As with AVI, the error term is a discounted sum of the norms of errors. However, contrary to AVI, each error term is not an iteration error, but a moving average of past iteration errors, $E_k^\beta = \beta E_{k-1}^\beta + (1-\beta)\epsilon_k$. In the ideal case where the sequence of these errors is a martingale difference with respect to the natural filtration, this term no longer vanishes, contrary to $\frac{1}{k}E_k$. However, it can reduce the variance. For simplicity, assume that the ϵ_j's are i.i.d. of variance 1. In this case, it is easy to see that the variance of E_k^β is bounded by $1-\beta < 1$, that tends toward 0 for β close to 1. Therefore, we advocate that DA-VI$_1$(λ,τ) allows for a better control of the error term than AVI (retrieved for $\beta = 0$). Notice that if asymptotically this error term predominates, the non-asymptotic behavior is also driven by the convergence rate g^2, which will be faster for β closer to 0. Therefore, there is a trade-off, illustrated in Fig. 1, right (for the same simple experiment[2]). Higher values of β lead to better asymptotic performance, but at the cost of slower initial convergence rate.

Interplay between the KL and the entropy terms. The l.h.s. of the bound of Thm. 2 solely depends on the entropy scale τ, while the r.h.s. solely depends on the term $\beta = \frac{\lambda}{\lambda+\tau}$. DA-VI($\lambda$,$\tau$) approximates the optimal policy of the regularized MDP, while we are usually interested in the solution of the original one. We have that $\|q_* - q_{\pi_*^\tau}\|_\infty \leq \frac{\tau \ln|\mathcal{A}|}{1-\gamma}$ [20], this bias can be controlled by setting an (arbitrarily) small τ. This does not affect the r.h.s. of the bound, as long as the scale of the KL term follows (such that $\frac{\lambda}{\lambda+\tau}$ remains fixed to the chosen value). So, Thm. 2 suggests to set τ to a very small value and to choose λ such that we have a given value of β. However, adding an entropy term has been proven efficient empirically, be it with arguments of exploration and robustness [22] or regarding the optimization landscape [3]. Our analysis does not cover this aspect. Indeed, it applies to $\lambda = \beta = 0$ (that is, solely entropy regularization), giving the propagation of errors of SAC, as a special case of [20, Thm. 3]. In this case, we retrieve the bound of AVI ($E_j^0 = \epsilon_j$, $g^2(k) \propto \gamma^k$), up to the bounded quantity. Thus, it does not show an advantage of using solely an entropy regularization, but it shows the advantage for considering an additional KL regularization, if the entropy is of interest for other reasons.

We end this discussion with some related works. The bound of Thm. 2 is similar to the one of CVI, despite a quite different proof technique. Notably, both involve a moving average of the errors. This is not surprising, CVI being a reparameterization of DA-VI. The core difference is that by bounding the distance to the regularized optimal q-function (instead of the unregularized one), we indeed show to what the algorithm converges without error. Shani et al. [40] study a variation of TRPO, for which they show a convergence rate of $\mathcal{O}(\frac{1}{\sqrt{k}})$, improved to $\mathcal{O}(\frac{1}{k})$ when an additional entropy regularizer is considered. This is to be compared to the convergence rate of our variation of TRPO, $\mathcal{O}(\frac{1}{k}) = o(\frac{1}{\sqrt{k}})$ (Thm. 1) improved to $g^2(k) = o(\frac{1}{k})$ with an additional entropy term (Thm. 2). Our rates are much better. However, this is only part of the story. We additionally show a compensation of errors in both cases, something not covered by their analysis. They also provide a sample complexity, but it is much worse than the one of SQL, that we would improve (thanks to the improved horizon term). Therefore, our results are stronger and more complete.

Limitations of our analysis. Our analysis provides strong theoretical arguments in favor of considering KL regularization in RL. Yet, it has also some limitations. First, it does not

provide arguments for using only entropy regularization, as already extensively discussed (even though it provides arguments for combining it with a KL regularization). Second, we study how the errors propagate over iterations, and show that KL allows for a compensation of these errors, but we say nothing about how to control these errors. This depends heavily on how the q-functions are approximated and on the data used to approximate them. We could easily adapt the analysis of Azar et al. [6] to provide sample complexity bounds for MD-VI in the case of a tabular representation and with access to a generative model, but providing a more general answer is difficult, and beyond the scope of this paper. Third, we assumed that the greedy step was performed exactly. This assumption would be reasonable with a linear parameterization and discrete actions, but not if the policy and the q-function are approximated with neural networks. In this case, the equivalence between MD-VI and DA-VI no longer holds, suggesting various ways of including the KL regularizer (explicitly, MD-VI, or implicitly, DA-VI). Therefore, we complement our thorough theoretical analysis with an extensive empirical study, to analyse the core effect of regularization in deep RL.

5 Empirical study

Before all, we would like to highlight that if regularization is a core component of successful deep RL algorithms (be it with entropy, KL, or both), it is never the sole component. For example, SAC uses a twin critic [18], TRPO uses a KL hard constraint rather than a KL penalty [39], or MPO uses retrace [29] for value function evaluation. All these further refinements play a role in the final performance. On the converse, our goal is to study the core effect of regularization, especially of KL regularization, in a deep RL context. To achieve this, we notice that DA-VI and MD-VI are extensions of AVI. One of the most prevalent VI-based deep RL algorithm being DQN [28], our approach is to start from a reasonably tuned version of it [15] and to provide the minimal modifications to obtain deep versions of MD-VI or DA-VI. Notably, we fixed the meta-parameters to the best values for DQN.

Practical algorithms. We describe briefly the variations we consider, a complementary high-level view is provided in Appx. E.1 and all practical details in Appx. E.2. We modify DQN by adding an actor. For the **evaluation step**, we keep the DQN loss, modified to account for regularization (that we'll call "*w/*", and that simply consists in adding the regularization term to the target q-network). Given that many approaches ignore the regularization there, we'll also consider the DQN loss (denoted "*w/o*" before, not covered by our analysis). For the **greedy step**, MD-VI and DA-VI are no longer equivalent. For MD-VI, there are two ways of approximating the regularized policy. The first one, denoted "*MD direct*", consists in directly solving the optimization problem corresponding to the regularized greediness, the policy being a neural network. This is reminiscent of TRPO (with a penalty rather than a constraint). The second one, denoted "*MD indirect*", consists in computing the analytical solution to the greedy step ($\pi_{k+1} \propto \pi_k^\beta \exp(\frac{1}{\lambda}\beta q_k)$) and to approximate it with a neural network. This is reminiscent of MPO. For DA-VI, we have to distinguish $\tau > 0$ from $\tau = 0$. In the first case, the regularized greedy policy can be computed analytically from an h-network, that can be computed by fitting a moving average of the online q-network and of a target h-network. This is reminiscent of MoDQN. If $\tau = 0$, DA-VI(λ,0) is not practical in a deep learning setting, as it requires averaging over iterations. Updates of target networks are too fast to consider them as new iterations, and a moving average is more convenient. So, we only consider the limit case $\lambda, \tau \to 0$ with $\beta = \frac{\lambda}{\lambda+\tau}$ kept constant. This is MoDQN with fixed β, and the evaluation step is necessarily unregularized ($\lambda = \tau = 0$). To sum up, we have six variations (three kinds of greediness, evaluation regularized or not), restricted to five variations for $\tau = 0$.

Research questions. Before describing the empirical study, we state the research questions we would like to address. The first is to know if regularization, without further refinements, helps, compared to the baseline DQN. The second one is to know if adding regularization in the evaluation step, something required by our analysis, provides improved empirical results. The third one is to compare the different kinds of regularized greediness, which are no longer equivalent with approximation. The last one is to study the effect of entropy, not covered by our analysis, and its interplay with the KL term.

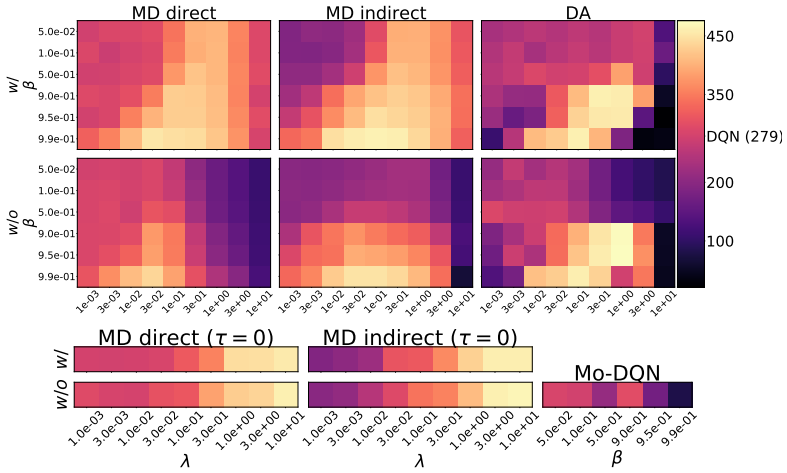

Figure 2: Cartpole.

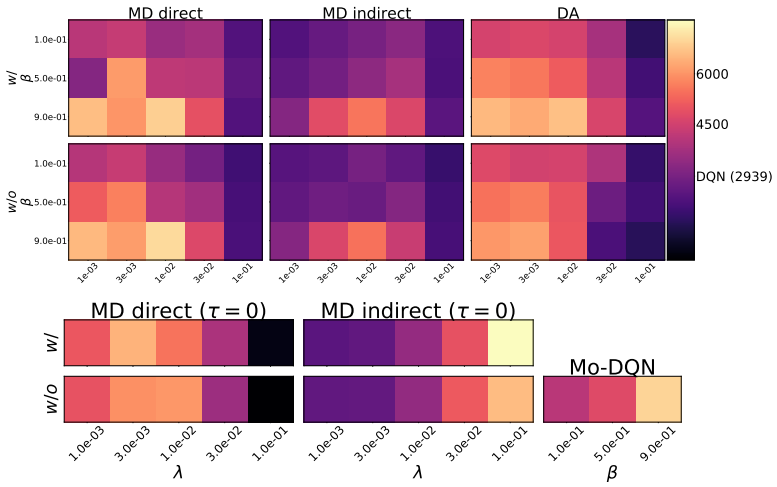

Figure 3: Asterix.

Environments. We consider two environments here (more are provided in Appx. E). The light Cartpole from Gym [14] allows for a large sweep over the parameters, and to average each result over 10 seeds. We also consider the Asterix Atari game [10], with sticky actions, to assess the effect of regularization on a large-scale problem. The sweep over parameters is smaller, and each result is averaged over 3 seeds.

Visualisation. For each environment, we present results as a table, the rows corresponding to the type of evaluation ($w/$ or w/o), the columns to the kind of greedy step. Each element of this table is a grid, varying β for the rows and τ for the columns. One element of this grid is the average undiscounted return per episode obtained during training, averaged over the number of seeds. On the bottom of this table, we show the limit cases with the same principle, varying with λ for MD-VI and with β for DA-VI (ony w/o, as explained before). The scale of colors is common to all these subplots, and the performance of DQN is indicated on this scale for comparison. Additional visualisations are provided in Appx. E.

Discussion. Results are provided in Fig. 2 and 3. First, we observe that regularization helps. Indeed, the results obtained by all these variations are better than the one of DQN, the baseline, for a large range of the parameters, sometime to a large extent. We also observe that, for a given value of τ, the results are usually better for medium to large values of

β (or λ), suggesting that **KL regularization is beneficial** (even though too large KL regularization can be harmful in some case, for example for MD direct, $\tau = 0$, on Asterix).

Then, we study the effect of regularizing the evaluation step, something suggested by our analysis. The effect of this can be observed by comparing the first row to the second row of each table. One can observe that the range of good parameters is larger in the first row (especially for large entropy), suggesting that **regularizing the evaluation step helps**. Yet, we can also observe that when $\tau = 0$ (no entropy), there is much less difference between the two rows. This suggests that adding the entropy regularization to the evaluation step might be more helpful (but adding the KL term too is costless and never harmful).

Next, we study the effect of the type of greediness. MD-direct shows globally better results than MD-indirect, but MD-indirect provides the best result on both environments (by a small margin), despite being more sensitive to the parameters. DA is more sensitive to parameters than MD for Cartpole, but less for Asterix, its best results being comparable to those of MD. This let us think that **the best choice of greediness is problem dependent**, something that goes beyond our theoretical analysis.

Last, we discuss the effect of entropy. As already noticed, for a given level of entropy, medium to large values of the KL parameter improve performance, suggesting that entropy works better in conjunction with KL, something appearing in our bound. Now, observing the table corresponding to $\tau = 0$ (no entropy), we observe that we can obtain comparable best performance with solely a KL regularization, especially for MD. This suggests that **entropy is better with KL, and KL alone might be sufficient**. We already explained that some beneficial aspects of entropy, like exploration or better optimization landscape, are not explained by our analysis. However, we hypothesize that KL might have similar benefits. For examples, entropy enforces stochastic policies, which helps for exploration. KL has the same effect (if the initial policy is uniform), but in an adaptive manner (exploration decreases with training time).

6 Conclusion

We provided an explanation of the effect of KL regularization in RL, through the implicit averaging of q-values. We provided a very strong performance bound for KL regularization, the very first RL bound showing both a linear dependency to the horizon and an averaging the estimation errors. We also analyzed the effect of KL regularization with an additional entropy term. The introduced abstract framework encompasses a number of existing approaches, but some assumptions we made do not hold when neural networks are used. Therefore, we complemented our thorough theoretical analysis with an extensive empirical study. It confirms that KL regularization is helpful, and that regularizing the evaluation step is never detrimental. It also suggests that KL regularization alone, without entropy, might be sufficient (and better than entropy alone).

The core issue of our analysis is that it relies heavily on the absence of errors in the greedy step, something we deemed impossible with neural networks. However, Vieillard et al. [42] proposed subsequently a reperameterization of our regularized approximate dynamic scheme. The resulting approach, called "Munchausen Reinforcement Learning", is simple and general, and provides agents outperforming the state of the art. Crucially, thanks to this reparameterization, there's no error in their greedy step and our bounds apply readily. More details can be found in [42].

Broader impact. Our core contribution is theoretical. We unify a large body of the literature under KL-regularized reinforcement learning, and provide strong performance bounds, among them the first one ever to combine a linear dependency to the horizon and an averaging of the errors. We complement these results with an empirical study. It shows that the insights provided by the theory can still be used in a deep learning context, when some of the assumptions are not satisfied. As such, we think the broader impact of our contribution to be the same as the one of reinforcement learning.

References

[1] Abbasi-Yadkori, Y., Bartlett, P., Bhatia, K., Lazic, N., Szepesvári, C., and Weisz, G. Politex: Regret bounds for policy iteration using expert prediction. In *International Conference on Machine Learning (ICML)*, 2019.

[2] Abdolmaleki, A., Springenberg, J. T., Tassa, Y., Munos, R., Heess, N., and Riedmiller, M. Maximum a posteriori policy optimisation. In *International Conference on Learning Representations (ICLR)*, 2018.

[3] Ahmed, Z., Le Roux, N., Norouzi, M., and Schuurmans, D. Understanding the impact of entropy on policy optimization. In *International Conference on Machine Learning (ICML)*, 2019.

[4] Archibald, T., McKinnon, K., and Thomas, L. On the generation of markov decision processes. *Journal of the Operational Research Society*, 46(3):354–361, 1995.

[5] Asadi, K. and Littman, M. L. An alternative softmax operator for reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2017.

[6] Azar, M. G., Munos, R., Ghavamzadeh, M., and Kappen, H. J. Speedy q-learning. In *Advances in neural information processing systems (NeurIPS)*, 2011.

[7] Azar, M. G., Gómez, V., and Kappen, H. J. Dynamic policy programming. *Journal of Machine Learning Research (JMLR)*, 13(Nov):3207–3245, 2012.

[8] Bagnell, J. A., Kakade, S. M., Schneider, J. G., and Ng, A. Y. Policy search by dynamic programming. In *Advances in neural information processing systems*, pp. 831–838, 2004.

[9] Baird III, L. C. *Reinforcement Learning Through Gradient Descent*. PhD thesis, US Air Force Academy, US, 1999.

[10] Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

[11] Bellemare, M. G., Ostrovski, G., Guez, A., Thomas, P. S., and Munos, R. Increasing the action gap: New operators for reinforcement learning. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2016.

[12] Boyd, S. and Vandenberghe, L. *Convex optimization*. Cambridge university press, 2004.

[13] Bradtke, S. J. and Barto, A. G. Linear least-squares algorithms for temporal difference learning. *Machine learning*, 22(1-3):33–57, 1996.

[14] Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[15] Castro, P. S., Moitra, S., Gelada, C., Kumar, S., and Bellemare, M. G. Dopamine: A research framework for deep reinforcement learning. *arXiv preprint arXiv:1812.06110*, 2018.

[16] Fellows, M., Mahajan, A., Rudner, T. G., and Whiteson, S. Virel: A variational inference framework for reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 7122–7136, 2019.

[17] Fox, R., Pakman, A., and Tishby, N. Taming the noise in reinforcement learning via soft updates. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2016.

[18] Fujimoto, S., Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pp. 1587–1596, 2018.

[19] Geist, M., Piot, B., and Pietquin, O. Is the bellman residual a bad proxy? In *Advances in Neural Information Processing Systems*, pp. 3205–3214, 2017.

[20] Geist, M., Scherrer, B., and Pietquin, O. A theory of regularized markov decision processes. In *International Conference on Machine Learning (ICML)*, 2019.

[21] Haarnoja, T., Tang, H., Abbeel, P., and Levine, S. Reinforcement learning with deep energy-based policies. In *International Conference on Machine Learning (ICML)*, 2017.

[22] Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning (ICML)*, 2018.

[23] Hiriart-Urruty, J.-B. and Lemaréchal, C. *Fundamentals of convex analysis*. Springer Science & Business Media, 2012.

[24] Kakade, S. and Langford, J. Approximately optimal approximate reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2002.

[25] Kozuno, T., Uchibe, E., and Doya, K. Theoretical analysis of efficiency and robustness of softmax and gap-increasing operators in reinforcement learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019.

[26] Levine, S. Reinforcement Learning and Control as Probabilistic Inference: Tutorial and Review. *arXiv preprint arXiv:1805.00909*, 2018.

[27] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[28] Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2016.

[29] Munos, R., Stepleton, T., Harutyunyan, A., and Bellemare, M. Safe and efficient off-policy reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[30] Perolat, J., Scherrer, B., Piot, B., and Pietquin, O. Approximate dynamic programming for two-player zero-sum markov games. In *International Conference on Machine Learning (ICML)*, 2015.

[31] Pérolat, J., Piot, B., Geist, M., Scherrer, B., and Pietquin, O. Softened approximate policy iteration for markov games. In *International Conference on Machine Learning (ICML)*, 2016.

[32] Piot, B., Geist, M., and Pietquin, O. Difference of convex functions programming for reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 2519–2527, 2014.

[33] Puterman, M. L. *Markov Decision Processes.: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 2014.

[34] Puterman, M. L. and Shin, M. C. Modified policy iteration algorithms for discounted markov decision problems. *Management Science*, 24(11):1127–1137, 1978.

[35] Scherrer, B. and Lesner, B. On the use of non-stationary policies for stationary infinite-horizon markov decision processes. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2012.

[36] Scherrer, B., Ghavamzadeh, M., Gabillon, V., Lesner, B., and Geist, M. Approximate modified policy iteration and its application to the game of tetris. *Journal of Machine Learning Research (JMLR)*, 16:1629–1676, 2015.

[37] Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. Trust region policy optimization. In *International Conference on Machine Learning (ICML)*, 2015.

[38] Schulman, J., Chen, X., and Abbeel, P. Equivalence between policy gradients and soft q-learning. *arXiv preprint arXiv:1704.06440*, 2017.

[39] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[40] Shani, L., Efroni, Y., and Mannor, S. Adaptive trust region policy optimization: Global convergence and faster rates for regularized MDPs. In *AAAI Conference on Artificial Intelligence*, 2020.

[41] Song, Z., Parr, R., and Carin, L. Revisiting the softmax bellman operator: New benefits and new perspective. In *International Conference on Machine Learning (ICML)*, 2019.

[42] Vieillard, N., Pietquin, O., and Geist, M. Munchausen Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[43] Vieillard, N., Scherrer, B., Pietquin, O., and Geist, M. Momentum in reinforcement learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2020.

Content. These appendices complement the core paper with the following:

- Appx. A is a warm-up that states a few facts about the Legendre-Fenchel transform, useful all along the derivations.
- Appx. B justifies the connections drawn in Sec. 3 between MD-MPI or DA-MPI and the literature.
- Appx. C provides the proofs of all stated theoretical results, as well as some necessary lemmata.
- Appx. D provides details about the experiment used to illustrate the bounds in Sec. 4.
- Appx. E provides additional details regarding the practical algorithms and the experiments, as well as additional experiments and visualisations.

A Convex Conjugacy for KL and Entropy Regularization

Let $q \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ and $\mu \in \Delta_{\mathcal{A}}^{\mathcal{S}}$, and consider the general greedy step $\pi' \in \mathcal{G}_{\mu}^{\lambda,\tau}$, the optimization being understood here state-wise.

$$\pi' \in \operatorname*{argmax}_{\pi\in\Delta_{\mathcal{A}}^{\mathcal{S}}} \left(\langle \pi, q\rangle - \lambda\,\mathrm{KL}(\pi||\mu) + \tau\mathcal{H}(\pi)\right). \tag{3}$$

The function $\lambda\,\mathrm{KL}(\pi||\mu) - \tau\mathcal{H}(\pi)$ being convex in π, this optimization problem is related to the Legendre-Fenchel transform (*e.g.*, Hiriart-Urruty & Lemaréchal [23, Ch. E]), or convex conjugate (which is the maximum rather than the maximizer). First, we consider a simple case, $\lambda = 0$ and $\tau = 1$. It is well known in this case that the maximum (the convex conjugate) is the log-sum-exp function and the maximizer (the gradient of the convex conjugate) is the softmax (*e.g.*, Boyd & Vandenberghe [12, Ex. 3.25]):

$$\max_{\pi\in\Delta_{\mathcal{A}}^{\mathcal{S}}} \left(\langle \pi, q\rangle + \mathcal{H}(\pi)\right) = \ln\langle \mathbf{1}, \exp q\rangle \in \mathbb{R}^{\mathcal{S}},$$

$$\operatorname*{argmax}_{\pi\in\Delta_{\mathcal{A}}^{\mathcal{S}}} \left(\langle \pi, q\rangle + \mathcal{H}(\pi)\right) = \frac{\exp q}{\langle \mathbf{1}, \exp q\rangle} \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}},$$

with $\mathbf{1} \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ the vector of which all components are equal to 1. We made use of the notations introduced in Sec. 2, and overload $v \in \mathbb{R}^{\mathcal{S}}$ to $v \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ as $v(s,a) = v(s)$. To make things clear, it gives

$$\left[\ln\langle \mathbf{1}, \exp q\rangle\right](s) = \ln \sum_{a\in\mathcal{A}} \exp q(s,a)$$

$$\text{and } \left[\frac{\exp q}{\langle \mathbf{1}, \exp q\rangle}\right](s,a) = \frac{\exp q(s,a)}{\sum_{a'\in\mathcal{A}} q(s,a')}.$$

Notice also that a direct consequence of this is that

$$\ln\langle \mathbf{1}, \exp q\rangle = \langle \pi', q\rangle + \mathcal{H}(\pi') \text{ with } \pi' = \frac{\exp q}{\langle \mathbf{1}, \exp q\rangle}.$$

From this simple case, we can easily handle the general case. We have

$$\begin{aligned}
\langle \pi, q\rangle - \lambda\,\mathrm{KL}(\pi||\mu) + \tau\mathcal{H}(\pi) &= \langle \pi, q\rangle - \lambda\langle \pi, \ln\pi - \ln\mu\rangle - \tau\langle \pi, \ln\pi\rangle \\
&= \langle \pi, q + \lambda\ln\mu\rangle - (\lambda+\tau)\langle \pi, \ln\pi\rangle \\
&= (\lambda+\tau)\left(\left\langle \pi, \frac{q+\lambda\ln\mu}{\lambda+\tau}\right\rangle + \mathcal{H}(\pi)\right).
\end{aligned}$$

From this, we can deduce directly that the maximum of (3) is

$$\begin{aligned}
\max_{\pi\in\Delta_{\mathcal{A}}^{\mathcal{S}}} \left(\langle \pi, q\rangle - \lambda\,\mathrm{KL}(\pi||\mu) + \tau\mathcal{H}(\pi)\right) &= (\lambda+\tau)\ln\left\langle 1, \exp\frac{q+\lambda\ln\mu}{\lambda+\tau}\right\rangle \\
&= (\lambda+\tau)\ln\left\langle \mu^{\frac{\lambda}{\lambda+\tau}}, \exp\frac{q}{\lambda+\tau}\right\rangle \\
&= (\lambda+\tau)\left(\ln\sum_{a\in\mathcal{A}} \mu(a|s)^{\frac{\lambda}{\lambda+\tau}} \exp\frac{q(s,a)}{\lambda+\tau}\right)_{s\in\mathcal{S}},
\end{aligned} \tag{4}$$

and that the maximizer of (3) is

$$\underset{\pi\in\Delta_{\mathcal{A}}^{\mathcal{S}}}{\operatorname{argmax}}\left(\langle\pi,q\rangle-\lambda\,\mathrm{KL}(\pi||\mu)+\tau\mathcal{H}(\pi)\right)=\frac{\exp\frac{q+\lambda\ln\mu}{\lambda+\tau}}{\langle\mathbf{1},\exp\frac{q+\lambda\ln\mu}{\lambda+\tau}\rangle}=\frac{\mu^{\frac{\lambda}{\lambda+\tau}}\exp\frac{q}{\lambda+\tau}}{\langle\mathbf{1},\mu^{\frac{\lambda}{\lambda+\tau}}\exp\frac{q}{\lambda+\tau}\rangle} \tag{5}$$

$$=\left(\frac{\mu(a|s)^{\frac{\lambda}{\lambda+\tau}}\exp\frac{q(s,a)}{\lambda+\tau}}{\sum_{a'\in\mathcal{A}}\mu(a'|s)^{\frac{\lambda}{\lambda+\tau}}\exp\frac{q(s,a')}{\lambda+\tau}}\right)_{(s,a)\in\mathcal{S}\times\mathcal{A}}$$

Again, the relationship between the maximum and the maximizer gives

$$(\lambda+\tau)\ln\left\langle\mu^{\frac{\lambda}{\lambda+\tau}},\exp\frac{q}{\lambda+\tau}\right\rangle=\langle\pi',q\rangle-\lambda\,\mathrm{KL}(\pi'||\mu)+\tau\mathcal{H}(\pi') \tag{6}$$

$$\text{with }\pi'=\frac{\mu^{\frac{\lambda}{\lambda+\tau}}\exp\frac{q}{\lambda+\tau}}{\langle\mathbf{1},\mu^{\frac{\lambda}{\lambda+\tau}}\exp\frac{q}{\lambda+\tau}\rangle}.$$

B Connections to existing algorithms

In this section, we justify the connections stated in Sec. 3 between the considered regularized ADP schemes and the literature.

B.1 Connection of MD-MPI(λ,τ) to other algorithms

Connection to SAC. We stated that SAC [22] is a variation of MD-MPI(0,τ). SAC was introduced as PI scheme ($m=\infty$), while it is practically implemented as VI scheme ($m=1$). We keep the VI viewpoint for this discussion. The MD-VI(0,τ) scheme is given by

$$\begin{cases}\pi_{k+1}=\mathcal{G}^{0,\tau}(q_k)\\ q_{k+1}=T^{0,\tau}_{\pi_{k+1}}q_k+\epsilon_{k+1}\end{cases}. \tag{7}$$

The regularized Bellman operator can be rewritten as follows:

$$\begin{aligned}T^{0,\tau}_{\pi_{k+1}}q_k&=T_{\pi_{k+1}}q_k+\gamma P\tau\mathcal{H}(\pi_{k+1})=r+\gamma P\left(\langle\pi_{k+1},q_k\rangle-\tau\langle\pi_{k+1},\ln\pi_{k+1}\rangle\right)\\&=r+\gamma P\langle\pi_{k+1},q_k-\tau\ln\pi_{k+1}\rangle.\end{aligned}$$

This is exactly the Bellman operator considered in SAC. For the greedy step, we have directly from Eq. (5) that $\pi_{k+1}\propto\exp\frac{q_k}{\tau}$. In SAC, continuous actions are considered, so the policy cannot be computed (due to the partition function). Therefore, it is approximated with a neural network by minimizing a reverse KL divergence (that allows getting rid of the partition function) between the neural policy and the target policy (the solution of the original greedy step):

$$\pi_{k+1}=\underset{\pi_\theta}{\operatorname{argmin}}\,\mathbb{E}_s[\mathrm{KL}(\pi_\theta||\pi^*_{k+1})]=\underset{\pi_\theta}{\operatorname{argmin}}\,\mathbb{E}_s[\mathrm{KL}(\pi_\theta||\exp\frac{q}{\tau})]\text{ with }\pi^*_{k+1}=\frac{\exp\frac{q_k}{\tau}}{\langle\mathbf{1},\exp\frac{q_k}{\tau}\rangle}.$$

Connection to Soft Q-learning. We stated that Soft Q-learning [17, 21] is also a variation of MD-MPI(0,τ). It is indeed a VI scheme, so a variation of MD-VI(0,τ) depicted in Eq. (7). As a direct consequence of Eq. (6), $\pi_{k+1}\propto\exp\frac{q_k}{\tau}$ being the maximizer, we have

$$\langle\pi_{k+1},q_k\rangle+\tau\mathcal{H}(\pi_{k+1})=\tau\ln\langle\mathbf{1},\exp\frac{q_k}{\tau}\rangle.$$

This allows rewriting the evaluation step as follows:

$$\begin{aligned}q_{k+1}&=T^{0,\tau}_{\pi_{k+1}}q_k+\epsilon_{k+1}\\&=r+\gamma P\left(\langle\pi_{k+1},q_k\rangle+\tau\mathcal{H}(\pi_{k+1})\right)+\epsilon_{k+1}\\\Leftrightarrow q_{k+1}&=r+\gamma P\left(\tau\ln\langle\mathbf{1},\exp\frac{q_k}{\tau}\rangle\right)+\epsilon_{k+1}.\end{aligned} \tag{8}$$

Eq. (8) is equivalent to Eq. (7), and it is the Bellman operator upon which Soft Q-learning is built (replacing the hard maximum by the log-sum-exp). Haarnoja et al. [21] additionally handle continuous actions, which requires some refinements.

Connection to Softmax DQN. We stated that Softmax DQN [41] is a variation of MD-MPI(0,τ), but *w/o* (without regularization in the evaluation step). Therefore, it is scheme (7), but replacing $T^{0,\tau}_{\pi_{k+1}}$ by $T_{\pi_{k+1}}$:

$$\begin{cases} \pi_{k+1} = \mathcal{G}^{0,\tau}(q_k) \\ q_{k+1} = T_{\pi_{k+1}} q_k + \epsilon_{k+1} \end{cases}.$$

Given that $\pi_{k+1} \propto \exp \frac{q_k}{\tau}$, this amounts to iterating the following so called softmax operator

$$\begin{aligned} q_{k+1} &= T_{\pi_{k+1}} q_k + \epsilon_{k+1} \\ &= r + \gamma P \left\langle \frac{\exp \frac{q_k}{\tau}}{\langle \mathbf{1}, \exp \frac{q_k}{\tau} \rangle}, q_k \right\rangle + \epsilon_{k+1}, \end{aligned}$$

which is the core update rule of softmax DQN. Notice that this operator might not be a contraction (depending on the value of τ), and that it can have multiple fixed points [5].

Connection to the mellowmax policy. Asadi & Littman [5] introduced a so-called mellowmax policy as a convergent alternative to the softmax operator. This can be indeed seen as an alternative way of regularizing the evaluation step. We explain here why. To do so, we reframe the mellowmax idea with our notations. Asadi & Littman [5] introduced the mellowmax operator as

$$\mathrm{mm}_\tau(q) = \tau \ln \left\langle \mathbf{1}, \frac{1}{|\mathcal{A}|} \exp \frac{q}{\tau} \right\rangle.$$

One can easily see that it is indeed the convex conjugate of the KL with respect to the uniform policy (that behaves like the entropy). Indeed, from Eq. (4), we have directly that

$$\mathrm{mm}_\tau(q) = \max_{\pi \in \Delta^{\mathcal{S}}_{\mathcal{A}}} \left(\langle \pi, q \rangle - \tau \, \mathrm{KL}(\pi || \pi_U) \right),$$

with π_U the uniform policy. From Geist et al. [20], we know that the following equivalent schemes,

$$\begin{cases} \pi_{k+1} = \mathrm{argmax}_{\pi \in \Delta^{\mathcal{S}}_{\mathcal{A}}} \left(\langle \pi, q \rangle - \tau \, \mathrm{KL}(\pi || \pi_U) \right) \\ q_{k+1} = T_{\pi_{k+1}} q_k - \gamma P \tau \, \mathrm{KL}(\pi_{k+1} || \pi_U) \end{cases} \Leftrightarrow q_{k+1} = r + \gamma P \, \mathrm{mm}_\tau(q_k),$$

are convergent (MDP regularized with $\tau \, \mathrm{KL}(\cdot || \pi_U)$, the equivalence being from Eq. (6)). This is not the viewpoint of Asadi & Littman [5]. They try to find a policy π'_{k+1} such that $q_{k+1} = r + \gamma P \, \mathrm{mm}_\tau(q_k) = r + \gamma P \langle \pi'_{k+1}, q_k \rangle$. To account for the possible existence of multiple policies, they look for the one with maximal entropy and solve (numerically) for

$$\pi'_{k+1} = \max_{\pi \in \Delta^{\mathcal{S}}_{\mathcal{A}} : \langle \pi, q_k \rangle = \mathrm{mm}_\tau(q_k)} \mathcal{H}(\pi).$$

Then, they apply $q_{k+1} = r + \gamma P \langle \pi'_{k+1}, q_k \rangle$. If there is no error when computing π'_{k+1}, this is equivalent to adding the regularization to the evaluation step.

Connection to TRPO. We stated that TRPO [37] is a variation of MD-MPI(λ, 0), *w/o*. More precisely, it is a variation of MD-PI(λ, 0):

$$\begin{cases} \pi_{k+1} = \mathcal{G}^{\lambda,0}(q_k) \\ q_{k+1} = T^{\infty}_{\pi_{k+1}} q_k + \epsilon_k = q_{\pi_{k+1}} + \epsilon_k \end{cases}. \tag{9}$$

In TRPO, the q-function is evaluated using Monte Carlo rollouts. The greedy policy is approximated with a neural network by directly solving the expected greedy step:

$$\pi_{k+1} = \mathrm{argmin}_{\pi_\theta} \, \mathbb{E}_s [\langle \pi_\theta, q_k \rangle - \lambda \, \mathrm{KL}(\pi_\theta || \pi_k)].$$

TRPO is indeed a bit different, as it uses importance sampling to sample actions according to π_k (which is especially useful for continuous actions, but does not change the objective function), it uses a constraint based on the KL rather than a regularization, and it considers the KL in the other direction:

$$\pi_{k+1} = \mathrm{argmin}_{\pi_\theta : \mathbb{E}_s[\mathrm{KL}(\pi_k || \pi_\theta)] \leq \epsilon} \, \mathbb{E}_s [\mathbb{E}_{a \sim \pi_k(.|s)} [\langle \frac{\pi_\theta}{\pi_k}, q_k \rangle]].$$

However, from an abstract viewpoint, TRPO is close to scheme (9).

Connection to MPO. We stated that MPO [2] is also a variation of MD-MPI(λ, 0), *w/o*:

$$\begin{cases} \pi_{k+1} = \mathcal{G}^{\lambda,0}(q_k) \\ q_{k+1} = T^m_{\pi_{k+1}} q_k + \epsilon_k \end{cases}. \tag{10}$$

The evaluation step is done by combining a TD approach with eligibility traces (a geometric average of m-step returns), rather than using m-step returns (that amounts to using the T^m_π operator). For the greedy step, the analytic solution can be computed for any state-action couple, and generalized to the whole state-action space by minimizing a KL between this analytical solution and a neural network:

$$\pi_{k+1} = \underset{\pi_\theta}{\operatorname{argmin}}\, \mathbb{E}_s[\mathrm{KL}(\pi^*_{k+1}||\pi_\theta)] = \underset{\pi_\theta}{\operatorname{argmax}}\, \mathbb{E}_s[\mathbb{E}_{a\sim\pi^*_{k+1}(.|s)}[\ln \pi_\theta(a|s)]]$$

$$\text{with } \pi^*_{k+1} = \frac{\pi_k \exp \frac{q_k}{\lambda}}{\langle \mathbf{1}, \pi_k \exp \frac{q_k}{\lambda}\rangle}.$$

The greedy step of MPO is indeed a bit different, the algorithm being derived from an expectation-maximization principle based on a probabilistic inference view of RL. The term λ is not fixed but learnt by the minimization of a convex dual function (coming from viewing the KL term as a constraint rather than a regularization), and an additional KL penalty is added (not necessarily redundant with the initial one, as the KL there is in the other direction):

$$\pi_{k+1} = \underset{\pi_\theta : \mathbb{E}_s[\mathrm{KL}(\pi_k||\pi_\theta)]\leq\epsilon}{\operatorname{argmax}} \mathbb{E}_s[\mathbb{E}_{a\sim\pi^*_{k+1}(.|s)}[\ln \pi_\theta(a|s)]].$$

However, from an abstract viewpoint, MPO is close to scheme (10).

Connection to DPP. We stated that DPP [7] is a variation of MD-MPI(λ, 0). More precisely, it is close to be a reparameterization of MD-VI(λ, 0), the difference being mainly the error term:

$$\begin{cases} \pi_{k+1} = \mathcal{G}^{\lambda,0}(q_k) \\ q_{k+1} = T^{\lambda,0}_{\pi_{k+1}} q_k + \epsilon_k \end{cases}. \tag{11}$$

To derive the DPP update rule from Eq. (11), we consider $\epsilon_k = 0$. The greedy policy is, according to (5),

$$\pi_{k+1} = \frac{\pi_k \exp \frac{q_k}{\lambda}}{\langle \mathbf{1}, \pi_k \exp \frac{q_k}{\lambda}\rangle}.$$

Define v_{k+1} as (the second equality coming from Eq. (6))

$$v_{k+1} = \langle \pi_{k+1}, q_k\rangle - \lambda\, \mathrm{KL}(\pi_{k+1}||\pi_k) = \lambda \ln\langle \pi_k, \exp \frac{q_k}{\lambda}\rangle.$$

With this, we have

$$q_{k+1} = T^{\lambda,0}_{\pi_{k+1}} q_k = r + \gamma P(\langle \pi_{k+1}, q_k\rangle - \lambda\, \mathrm{KL}(\pi_{k+1}||\pi_k)) = r + \gamma P v_{k+1}$$

Let us define $\psi_{k+1} \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ as

$$\psi_{k+1} = \lambda \ln\left(\pi_k \exp \frac{q_k}{\lambda}\right) = r + \gamma P v_k + \lambda \ln \pi_k. \tag{12}$$

Thus, we have

$$\pi_k = \frac{\exp \frac{\psi_k}{\lambda}}{\langle \mathbf{1}, \exp \frac{\psi_k}{\lambda}\rangle} \tag{13}$$

$$\text{and } v_k = \lambda \ln\langle \mathbf{1}, \frac{\psi_k}{\lambda}\rangle. \tag{14}$$

Injecting Eqs. (13) and (14) into (12), we get

$$\psi_{k+1} = r + \gamma P \lambda \ln\langle \mathbf{1}, \frac{\psi_k}{\lambda}\rangle + \psi_k - \lambda \ln\langle \mathbf{1}, \frac{\psi_k}{\lambda}\rangle.$$

This is how DPP is justified from a DP viewpoint [7, Appx. A]. It is a bit different from the DPP algorithm analyzed by Azar et al. [7], for which $\ln\langle \mathbf{1}, \frac{\psi_k}{\lambda}\rangle$ is replaced by $\langle \pi_k, \psi_k\rangle$ (both terms being equal in the limit $\lambda \to 0$), and that consider an estimation error ϵ'_{k+1}:

$$\psi_{k+1} = r + \gamma P\langle \pi_k, \psi_k\rangle + \psi_k - \langle \pi_k, \psi_k\rangle + \epsilon'_{k+1}.$$

We advocate that the error ϵ'_k is usually harder to control than ϵ_k (or equivalently that q_k is easier to estimate than ψ_k), because the function ψ_* (the optimal ψ-function for the MDP) is equal to $-\infty$ for any suboptimal action [7, Cor. 4].

Connection to CVI. We stated that CVI is a reparametrization of MD-VI(λ,τ), that we recall (without the error term, to do the reparameterization):

$$\begin{cases} \pi_{k+1} = \mathcal{G}^{\lambda,\tau}(q_k) \\ q_{k+1} = T^{\lambda,\tau}_{\pi_{k+1}} q_k \end{cases}.$$

We now show how to derive the CVI update rule from this. The regularized greedy policy is, thanks to Eq. (5), and writing $\beta = \frac{\lambda}{\lambda+\tau}$:

$$\pi_{k+1} = \frac{\pi_k^\beta \exp \frac{\beta q_k}{\lambda}}{\langle \mathbf{1}, \pi_k^\beta \exp \frac{\beta q_k}{\lambda}\rangle}.$$

Similarly to DPP, we can define v_{k+1} as (still using Eq. (6) for the second equality):

$$v_{k+1} = \langle \pi_{k+1}, q_k\rangle - \lambda\,\mathrm{KL}(\pi_{k+1}||\pi_k) + \tau\mathcal{H}(\pi_{k+1}) = \frac{\lambda}{\beta}\ln\langle \pi_k^\beta, \exp\frac{\beta q_k}{\lambda}\rangle.$$

With this, we have

$$q_{k+1} = T^{\lambda,0}_{\pi_{k+1}} q_k = r + \gamma P(\langle \pi_{k+1}, q_k\rangle - \lambda\,\mathrm{KL}(\pi_{k+1}||\pi_k) + \tau\mathcal{H}(\pi_{k+1})) = r + \gamma P v_{k+1}.$$

Let us define $\psi_{k+1} \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ as

$$\psi_{k+1} = \frac{\lambda}{\beta}\ln\left(\pi_k^\beta \exp\frac{\beta q_k}{\lambda}\right) = r + \gamma P v_k + \lambda \ln \pi_k. \tag{15}$$

Thus, we have

$$\pi_k = \frac{\exp\frac{\beta\psi_k}{\lambda}}{\langle \mathbf{1}, \exp\frac{\beta\psi_k}{\lambda}\rangle} \tag{16}$$

$$\text{and } v_k = \frac{\lambda}{\beta}\ln\langle \mathbf{1}, \frac{\beta\psi_k}{\lambda}\rangle. \tag{17}$$

Injecting Eqs. (16) and (17) into (15), we get

$$\psi_{k+1} = r + \gamma P\frac{\lambda}{\beta}\ln\langle \mathbf{1}, \frac{\beta\psi_k}{\lambda}\rangle + \beta(\psi_k - \frac{\lambda}{\beta}\ln\langle \mathbf{1}, \frac{\beta\psi_k}{\lambda}\rangle).$$

This is exactly the CVI update rule. Notice that setting $\beta = 1$, i.e., $\tau = 0$ (no entropy term), we retrieve DPP (which was to be expected). As we obtain CVI, by considering $\lambda + \tau \to 0$ while keeping $\beta = \frac{\lambda}{\lambda+\tau}$ constant, we retrieve advantage learning in the limit [9, 11], that DA-VI(λ,τ) thus generalizes.

B.2 Connection of DA-MPI(λ,τ) to other algorithms

Connection to Politex. Politex [1] addresses the average reward criterion. It is a PI scheme, up to the fact that the policy, instead of being greedy according to the last q-function, is softmax according to the sum of all past q-function. In the discounted reward case considered here, this is exactly DA-PI(λ,0), *w/o* (without regularization in the evaluation step):

$$\begin{cases} \pi_{k+1} = \mathcal{G}^{0,\frac{\lambda}{k+1}}(h_k) \\ q_{k+1} = T^\infty_{\pi_{k+1}} q_k + \epsilon_{k+1} = q_{\pi_{k+1}} + \epsilon_{k+1} \\ h_{k+1} = \frac{k+1}{k+2}h_k + \frac{1}{k+2}q_{k+1} \end{cases}$$

Indeed, by definition $h_k = \frac{1}{k+1}\sum_{j=0}^{k} q_j$ and the greedy policy is

$$\pi_{k+1} = \mathcal{G}^{0,\frac{\lambda}{k+1}}(h_k) = \frac{\exp\frac{(k+1)h_k}{\lambda}}{\langle \mathbf{1}, \exp\frac{(k+1)h_k}{\lambda}\rangle} = \frac{\exp\frac{\sum_{j=0}^{k} q_j}{\lambda}}{\langle \mathbf{1}, \exp\frac{\sum_{j=0}^{k} q_j}{\lambda}\rangle}.$$

This is exactly the Politex algorithm, but for the discounted reward case (that changes how the q-function is defined, and thus estimated).

Connection to MoVI. MoVI [43] is a VI scheme, up to the fact that the policy, instead of being greedy according to the last q-function, is greedy according to the average of past q-functions. It is indeed is a limiting case of DA-VI(λ, 0), *w/o*:

$$\begin{cases} \pi_{k+1} = \mathcal{G}^{0,\frac{\lambda}{k+1}}(h_k) \\ q_{k+1} = T_{\pi_{k+1}} q_k + \epsilon_{k+1} \\ h_{k+1} = \frac{k+1}{k+2} h_k + \frac{1}{k+2} q_{k+1} \end{cases}.$$

It is well known that the limit of a softmax, when the temperatures goes to zero, is the greedy policy: $\mathcal{G}^{0,\frac{\lambda}{k+1}}(h_k) \to \mathcal{G}(h_k)$ as $\lambda \to 0$. So, DA-VI($\lambda \to 0$, 0), *w/o*, is the following scheme,

$$\begin{cases} \pi_{k+1} \in \mathcal{G}(h_k) \\ q_{k+1} = T_{\pi_{k+1}} q_k + \epsilon_{k+1} \\ h_{k+1} = \frac{k+1}{k+2} h_k + \frac{1}{k+2} q_{k+1} \end{cases},$$

that is exactly MoVI. Notice that it is different from MD-VI($\lambda \to 0$, 0), *w/o*, which is AVI (see also Prop. 1).

Connection to momentum DQN. Momentum DQN [43] was introduced as a practical heuristic to MoVI, changing the exact average by a moving average (more amenable to optimization with deep networks). We show below that it is indeed a limiting case of DA-VI(λ,τ), *w/o* (without regularized greedy step), that is:

$$\begin{cases} \pi_{k+1} = \mathcal{G}^{0,\tau}(h_k) \\ q_{k+1} = T_{\pi_{k+1}} q_k + \epsilon_{k+1} \\ h_{k+1} = \beta h_k + (1-\beta) q_{k+1} \text{ with } \beta = \frac{\lambda}{\lambda+\tau} \end{cases}.$$

Fix $\beta \in (0,1)$, we can consider $\lambda, \tau \to 0$ with $\beta = \frac{\lambda}{\lambda+\tau}$ kept constant. In this case, the regularized greedy operator tends to the usual greedy one: $\mathcal{G}^{0,\tau}(h_k) \to \mathcal{G}(h_k)$ as $\tau \to 0$. In the limit, we obtain the following scheme,

$$\begin{cases} \pi_{k+1} = \mathcal{G}(h_k) \\ q_{k+1} = T_{\pi_{k+1}} q_k + \epsilon_{k+1} \\ h_{k+1} = \beta h_k + (1-\beta) q_{k+1} \end{cases}$$

for a chosen β, which is exactly momentum DQN with fixed β.

Connection to Speedy Q-learning. We stated that Speedy Q-learning [6] is a limiting case of DA-VI(λ,0), which we recall (without the error term here):

$$\begin{cases} \pi_{k+1} = \mathcal{G}^{0,\frac{\lambda}{k+1}}(h_k) \\ q_{k+1} = T^{\lambda,0}_{\pi_{k+1}} q_k \\ h_{k+1} = \frac{k+1}{k+2} h_k + \frac{1}{k+2} q_{k+1} \end{cases}.$$

As shown in Lemma 2 in Appx. C.2, we have

$$T^{\lambda,0}_{\pi_{k+1}|\pi_k} q_k = (k+1) T^{0,\frac{\lambda}{k+1}}_{\pi_{k+1}} h_k - k T^{0,\frac{\lambda}{k}}_{\pi_k} h_{k-1}.$$

With this, DA-VI$_1$(λ,0) can be expressed solely in terms of h_k and π_k:

$$\begin{cases} \pi_{k+1} = \mathcal{G}^{0,\frac{\lambda}{k+1}}(h_k) \\ h_{k+1} = \frac{k+1}{k+2} h_k + \frac{1}{k+2}\left((k+1) T^{0,\frac{\lambda}{k+1}}_{\pi_{k+1}} h_k - k T^{0,\frac{\lambda}{k}}_{\pi_k} h_{k-1}\right). \end{cases} \tag{18}$$

As before, as $\lambda \to 0$, the regularized greedy step tends to the greedy step, $\mathcal{G}^{0,\frac{\lambda}{k+1}}(h_k) \to \mathcal{G}(h_k)$. Regarding the evaluation step, we can write, by definition of the regularized Bellman operator and using Eq. (6),

$$\begin{aligned} T^{0,\frac{\lambda}{k+1}}_{\pi_{k+1}} h_k &= r + \gamma P \left(\langle \pi_{k+1}, h_k \rangle + \frac{\lambda}{k+1} \mathcal{H}(\pi_{k+1}) \right) \\ &= r + \gamma P \left(\frac{\lambda}{k+1} \ln \langle \mathbf{1}, \exp \frac{(k+1)h_k}{\lambda} \rangle \right). \end{aligned}$$

It is a classical result that the convex conjugate of the entropy tends to the hard maximum as the associated temperature goes to zero. For any $s \in \mathcal{S}$,

$$\lim_{\lambda \to 0} \frac{\lambda}{k+1} \ln \sum_{a \in \mathcal{A}} \exp \frac{(k+1)h_k(s,a)}{\lambda} = \frac{1}{k+1} \max_{a \in \mathcal{A}} \left((k+1)h_k(s,a) \right) = \max_{a \in \mathcal{A}} h_k(s,a).$$

Writing T_* the Bellman optimality operator, defined as $T_* q = \max_\pi T_\pi q$, we thus have

$$\lim_{\lambda \to 0} T^{0,\frac{\lambda}{k+1}}_{\pi_{k+1}} h_k = T_* h_k.$$

Thus, writing the limit of scheme (18) as $\lambda \to 0$, we obtain

$$h_{k+1} = (1 - \frac{1}{k+2}) h_k + \frac{1}{k+2} \left((k+1) T_* h_k - k T_* h_{k-1} \right),$$

which is exactly the Speedy Q-learning update rule.

Connection to softened LSPI. [31] address the problem of learning a Nash equilibria in zero-sum Markov games. They show that state of the art algorithms can be derived by minimizing the norm of the (projected) Bellman residual using a Newton descent, and propose more stable algorithms by using instead a quasi-Newton descent. Single agent reinforcement learning is a special case of zero-sum Markov games, and in this case the algorithm they propose can be written as follows, in an abstract way[3]:

$$\begin{cases} \pi_{k+1} \in \mathcal{G}(h_k) \\ q_{k+1} = T^\infty_{\pi_{k+1}} q_k + \epsilon_{k+1} = q_{\pi_{k+1}} + \epsilon_{k+1} \\ h_{k+1} = \beta h_k + (1-\beta) q_{k+1} \end{cases}.$$

Using the same arguments as for the connection to momentum DQN, this is a limit case of DA-PI(λ,τ), *w/o*, as $\lambda, \tau \to 0$ with $\beta = \frac{\lambda}{\lambda+\tau}$ kept constant. It is also closely related to Politex (the policy is greedy instead of being softmax, moving average of the q-values instead of an average).

C Proofs of Theoretical Results

In this section, we prove the results stated in the paper.

C.1 Proof of Proposition 1

Sketch of proof. As explained in the paper, the optimization problem $\pi_{k+1} = \mathcal{G}^{\lambda,0}_{\pi_k}(q_k)$ can be solved analytically, yielding $\pi_{k+1} \propto \pi_k \exp \frac{q_k}{\lambda}$. By direct induction, π_0 being uniform, we have $\pi_{k+1} \propto \pi_k \exp \frac{q_k}{\lambda} \propto \cdots \propto \exp \frac{1}{\lambda} \sum_{j=0}^k q_j$. Thus, the policy is indeed softmax according to the sum of q-values. Defining h_k as the average of past q-values basically provides the stated DA-VI(λ,0). The case with an additionnal entropy term is a bit more involved, but the principle is the same.

[3] Specialized to single agent RL, their algorithm adopts a linear parameterization of the q-function and estimate $q_{\pi_{k+1}}$ either with LSTD [13] or by minimizing the norm of the Bellman residual.

Proof. We start by proving the equivalence for the case $\tau = 0$. Recall that we assumed, with little loss of generality, that π_0 is the uniform policy. We recall MD-MPI(λ,0):

$$\begin{cases} \pi_{k+1} = \mathcal{G}^{\lambda,0}_{\pi_k}(q_k) \\ q_{k+1} = (T^{\lambda,0}_{\pi_{k+1}|\pi_k})^m q_k + \epsilon_{k+1} \end{cases}. \tag{19}$$

Let us define $h_0 = q_0$ and h_k for $k \geq 1$ as the average of past q-functions.

$$h_k = \frac{1}{k+1}\sum_{j=0}^{k} q_j = \frac{k}{k+1}h_{k-1} + \frac{1}{k+1}q_k.$$

As a direct consequence of Eq. (5), we have that $\pi_{k+1} \propto \pi_k \exp \frac{q_k}{\lambda}$. By direct induction,

$$\pi_{k+1} \propto \pi_k \exp \frac{q_k}{\lambda} \propto \pi_{k-1} \exp \frac{q_k + q_{k-1}}{\lambda} \propto \cdots \propto \exp \frac{\sum_{j=0}^{k} q_j}{\lambda} = \exp \frac{(k+1)h_k}{\lambda}.$$

Still thanks to Eq. (5), this means that π_{k+1} satisfies

$$\pi_{k+1} = \underset{\pi \in \Delta^{\mathcal{S}}_{\mathcal{A}}}{\operatorname{argmax}} \left(\langle \pi, h_k \rangle + \frac{\lambda}{k+1}\mathcal{H}(\pi) \right) = \mathcal{G}^{0,\frac{\lambda}{k+1}}(h_k).$$

This shows that Eq. (19) is equivalent to

$$\begin{cases} \pi_{k+1} = \mathcal{G}^{0,\frac{\lambda}{k+1}}(h_k) \\ q_{k+1} = (T^{\lambda,0}_{\pi_{k+1}|\pi_k})^m q_k + \epsilon_{k+1} \\ h_{k+1} = \frac{k+1}{k+2}h_k + \frac{1}{k+2}q_{k+1} \end{cases}, \tag{20}$$

which is DA-MPI(λ,0), and this shows the first part of the result. In the limit $\lambda \to 0$, the regularized greediness becomes the usual greediness (hard maximum over q-values) and the (regularized) evaluation operator becomes the standard one. However, notice that schemes are not equivalent in the limit: scheme (19) tends to classic VI, while scheme (20) tends to Speedy Q-learning [6] (see the justification of the connection to SQL in Appx. B.2).

Next, we prove the equivalence for the case $\tau > 0$. We recall MD-MPI(λ,τ):

$$\begin{cases} \pi_{k+1} = \mathcal{G}^{\lambda,\tau}_{\pi_k}(q_k) \\ q_{k+1} = (T_{\pi_{k+1}|\pi_k})^m q_k + \epsilon_{k+1} \end{cases}. \tag{21}$$

Thanks to Eq. (5), we have that $\pi_{k+1} \propto \exp \frac{q_k + \lambda \ln \pi_k}{\lambda+\tau}$. We define $\beta = \frac{\lambda}{\lambda+\tau}$ (and thus $1-\beta = \frac{\tau}{\lambda+\tau}$ and $\frac{\beta}{\lambda} = \frac{1}{\lambda+\tau}$). By induction, we have (writing "cst" any function depending solely on states, not necessarily the same for different lines):

$$\begin{aligned} \ln \pi_{k+1} &= \frac{\beta}{\lambda}q_k + \beta \ln \pi_k + \text{cst} \\ &= \frac{\beta}{\lambda}\left(q_k + \beta q_{k-1} + \beta^2 q_{k-2} + \dots\right) + \text{cst} \\ &= \frac{\beta}{\lambda(1-\beta)}\left((1-\beta)(q_k + \beta q_{k-1} + \beta^2 q_{k-2} + \dots)\right) + \text{cst}\,. \end{aligned}$$

We now define h_k as the moving average of past q-values, with $h_{-1} = 0$:

$$h_k = \beta h_{k-1} + (1-\beta)q_k = (1-\beta)\sum_{j=0}^{k} \beta^{k-j} q_j. \tag{22}$$

Noticing also that $\frac{\beta}{\lambda(1-\beta)} = \frac{1}{\tau}$, this shows that

$$\pi_{k+1} \propto \exp \frac{h_k}{\tau}.$$

As before, this means that π_{k+1} is the solution of an entropy regularized greedy step with respect to h_k:

$$\pi_{k+1} = \underset{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}}{\operatorname{argmax}} \left(\langle \pi, h_k \rangle + \tau \mathcal{H}(\pi) \right) = \mathcal{G}^{0,\tau}(h_k).$$

This means that Eq. (21) is equivalent to

$$\begin{cases} \pi_{k+1} = \mathcal{G}^{0,\tau}(h_k) \\ q_{k+1} = (T^{\lambda,\tau}_{\pi_{k+1}|\pi_k})^m q_k + \epsilon_{k+1} \\ h_{k+1} = \beta h_k + (1-\beta) q_{k+1} \text{ with } \beta = \frac{\lambda}{\lambda+\tau} \end{cases},$$

which is the DA-MPI(λ,τ) scheme. This concludes the proof.

C.2 Proof of Theorem 1

Here, we provide the bound for DA-VI(λ,0), which we recall:

$$\begin{cases} \pi_{k+1} = \mathcal{G}^{0,\frac{\lambda}{k+1}}(h_k) \\ q_{k+1} = T^{\lambda,0}_{\pi_{k+1}|\pi_k} q_k + \epsilon_{k+1} \\ h_{k+1} = \frac{k+1}{k+2} h_k + \frac{1}{k+2} q_{k+1} \end{cases}. \tag{23}$$

Sketch of proof. The quantity of interest is $q_* - q_{\pi_{k+1}}$, it can be decomposed as $q_* - q_{\pi_{k+1}} = q_* - h_k + h_k - q_{\pi_{k+1}}$. Lemma 1 allows expressing the quantity of interest essentially as a function of the Bellman residual $T_{\pi_{k+1}} h_k - h_k$. Controlling this residual is the key to state our bound. To achieve this, we first derive Lemma 2 that expresses the evaluation step (the update of the q-function) as a difference of Bellman operators applied to successive h-functions (the averages of q-values). Thanks to this, we're able to derive a Bellman-like recursion for h_k in Lemma 3, using notably Lemma 2 and a telescoping argument. The rest of the proof consists in exploiting this Bellman-like recursion to control the residual and eventually boud the quantity of interest.

Proof. We start by stating a useful lemma.

Lemma 1. *For any $q \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ and $\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}$, we have*

$$q_\pi - q = (I - \gamma P_\pi)^{-1}(T_\pi q - q).$$

Proof. This result is classic, and appears many times in the literature (*e.g.*, Kakade & Langford [24]). We provide a one line proof for completeness, relying on basic properties of the Bellman operator:

$$q_\pi - q = T_\pi q_\pi - T_\pi q + T_\pi q - q = \gamma P_\pi (q_\pi - q) + T_\pi q - q \Leftrightarrow q_\pi - q = (I - \gamma P_\pi)^{-1}(T_\pi q - q).$$

□

The aim is to bound the quantity $q_* - q_{\pi_{k+1}}$, the difference between the optimal value function and the value function computed by DA-VI(λ,0). Thanks to Lemma 1, we can decompose this term as

$$\begin{aligned} q_* - q_{\pi_{k+1}} &= q_* - h_k + h_k - q_{\pi_{k+1}} \\ &= (I - \gamma P_{\pi_*})^{-1}(T_{\pi_*} h_k - h_k) - (I - \gamma P_{\pi_{k+1}})^{-1}(T_{\pi_{k+1}} h_k - h_k). \end{aligned} \tag{24}$$

Notice that $q_* = q_{\pi_*}$ for any optimal policy π_*. There exists an optimal deterministic policy [33], so we will consider a deterministic π_*. As for any deterministic policy, $\mathcal{H}(\pi_*) = 0$. Using the definition of π_{k+1}, we have

$$\begin{aligned} \pi_{k+1} = \mathcal{G}^{0,\frac{\lambda}{k+1}}(h_k) &\Rightarrow \langle \pi_{k+1}, h_k \rangle + \frac{\lambda}{k+1}\mathcal{H}(\pi_{k+1}) \geq \langle \pi_*, h_k \rangle + \frac{\lambda}{k+1}\underbrace{\mathcal{H}(\pi_*)}_{=0} \\ &\Rightarrow r + \gamma P\left(\langle \pi_{k+1}, h_k \rangle + \frac{\lambda}{k+1}\mathcal{H}(\pi_{k+1}) \right) \geq r + \gamma P \langle \pi_*, h_k \rangle \\ &\Rightarrow T^{0,\frac{\lambda}{k+1}}_{\pi_{k+1}} h_k = T_{\pi_{k+1}} h_k + \gamma \frac{\lambda}{k+1} P \mathcal{H}(\pi_{k+1}) \geq T_{\pi_*} h_k. \end{aligned}$$

Injecting this into Eq. (24), we obtain, using the fact that for any π the matrix $(I-\gamma P_\pi)^{-1} = \sum_{t\geq 0}\gamma^t P_\pi^t$ is positive,

$$q_* - q_{\pi_{k+1}} \leq (I-\gamma P_{\pi_*})^{-1}(T_{\pi_{k+1}}^{0,\frac{\lambda}{k+1}} h_k - h_k) - (I-\gamma P_{\pi_{k+1}})^{-1}(T_{\pi_{k+1}}^{0,\frac{\lambda}{k+1}} h_k - h_k - \gamma \frac{\lambda}{k+1} P\mathcal{H}(\pi_{k+1})). \tag{25}$$

So, what we have to do is to control the residual $T_{\pi_{k+1}}^{0,\frac{\lambda}{k+1}} h_k - h_k$.

To do so, the following lemma will be useful.

Lemma 2. *For any $k \geq 1$, we have that*

$$T_{\pi_{k+1}|\pi_k}^{\lambda,0} q_k = (k+1) T_{\pi_{k+1}}^{0,\frac{\lambda}{k+1}} h_k - k T_{\pi_k}^{0,\frac{\lambda}{k}} h_{k-1}.$$

For $k=0$, we have

$$T_{\pi_1|\pi_0}^{\lambda,0} q_0 = T_{\pi_1}^{0,\lambda} h_0 - \gamma\lambda P\mathcal{H}(\pi_0).$$

Proof. To prove this result, we will start by working on the optimization problem related to the regularized greedy step $\mathcal{G}_{\pi_k}^{\lambda,0} q_k$:

$$\langle \pi, q_k\rangle - \lambda\,\mathrm{KL}(\pi||\pi_k) = \langle \pi, q_k\rangle - \lambda\langle \pi, \ln\pi - \ln\pi_k\rangle = \langle \pi, q_k + \lambda\ln\pi_k\rangle - \lambda\langle \pi, \ln\pi\rangle.$$

For DA-VI$_1$(λ,0), $\pi_{k+1} \in \mathcal{G}^{0,\frac{\lambda}{k+1}}(h_k)$ (see Eq. (23)), so according to Eq. (5), $\pi_{k+1} \propto \exp\frac{(k+1)h_k}{\lambda}$. Therefore, we have, using also the definition of h_k

$$\begin{aligned} q_k + \lambda\ln\pi_k &= q_k + \lambda(\frac{k}{\lambda}h_{k-1} - \ln\langle 1, \exp\frac{kh_{k-1}}{\lambda}\rangle) \\ &= (k+1)h_k - \lambda\ln\langle 1, \exp\frac{kh_{k-1}}{\lambda}\rangle. \end{aligned}$$

Therefore, we have

$$\langle \pi, q_k\rangle - \lambda\,\mathrm{KL}(\pi||\pi_k) = \langle \pi, (k+1)h_k\rangle - \lambda\langle \pi, \ln\pi\rangle - \lambda\ln\langle \mathbf{1}, \exp\frac{kh_{k-1}}{\lambda}\rangle.$$

The maximizer is π_{k+1}, obviously. It is also the maximizer of $\langle \pi, (k+1)h_k\rangle - \lambda\langle \pi, \ln\pi\rangle$ (the third term not depending on π), and the associated maximum is, according to Eq. (4), $\lambda\ln\langle \mathbf{1}, \exp\frac{(k+1)h_k}{\lambda}\rangle$. This gives

$$\begin{aligned} \langle \pi_{k+1}, q_k\rangle - \lambda\,\mathrm{KL}(\pi_{k+1}||\pi_k) &= \lambda\ln\langle \mathbf{1}, \exp\frac{(k+1)h_k}{\lambda}\rangle - \lambda\ln\langle \mathbf{1}, \exp\frac{kh_{k-1}}{\lambda}\rangle \\ &= (k+1)\frac{\lambda}{k+1}\ln\langle \mathbf{1}, \exp\frac{(k+1)h_k}{\lambda}\rangle - k\frac{\lambda}{k}\ln\langle \mathbf{1}, \exp\frac{kh_{k-1}}{\lambda}\rangle. \end{aligned}$$

Still from Eq. (4), we know that $\frac{\lambda}{k+1}\ln\langle 1, \exp\frac{(k+1)h_k}{\lambda}\rangle$ is the maximum of $\langle \pi, h_k\rangle + \frac{\lambda}{k+1}\mathcal{H}(\pi)$, the associated maximizer being again π_{k+1}, so using Eq. (6), we can conclude that

$$\langle \pi_{k+1}, q_k\rangle - \lambda\,\mathrm{KL}(\pi_{k+1}||\pi_k) = (k+1)\left(\langle \pi_{k+1}, h_k\rangle + \frac{\lambda}{k+1}\mathcal{H}(\pi_{k+1})\right) - k\left(\langle \pi_k, h_{k-1}\rangle + \frac{\lambda}{k}\mathcal{H}(\pi_k)\right).$$

Noticing that $r = (k+1)r - kr$, we have the first part of the result:

$$T_{\pi_{k+1}|\pi_k}^{\lambda,0} q_k = (k+1) T_{\pi_{k+1}}^{0,\frac{\lambda}{k+1}} h_k - k T_{\pi_k}^{0,\frac{\lambda}{k}} h_{k-1}.$$

This only holds for $k \geq 1$. For $k=0$, using the fact that $h_0 = q_0$,

$$\begin{aligned} T_{\pi_1|\pi_0}^{\lambda,0} q_0 &= r + \gamma P(\langle \pi_1, q_0\rangle - \lambda\,\mathrm{KL}(\pi_1||\pi_0)) \\ &= r + \gamma P(\langle \pi_1, h_0\rangle - \lambda\langle \pi_1, \ln\pi_1 - \ln\pi_0\rangle) \\ &= r + \gamma P(\langle \pi_1, h_0\rangle + \lambda\mathcal{H}(\pi_1) + \lambda\langle \pi_1, \ln\pi_0\rangle) \\ &= T_{\pi_1}^{0,\lambda} h_0 - \gamma\lambda P\mathcal{H}(\pi_0), \end{aligned}$$

where we used in the last line the fact that, π_0 being uniform,

$$\langle \pi_1, \ln\pi_0\rangle = \langle \pi_1, \ln\frac{1}{|\mathcal{A}|}\rangle = -\ln|\mathcal{A}|\langle \pi_1, 1\rangle = -\ln|\mathcal{A}| = -\mathcal{H}(\pi_0).$$

This concludes the proof. □

Using this lemma, we can provide a Bellman-like induction on h_k.

Lemma 3. *Define $E_k = -\sum_{j=1}^{k} \epsilon_j$. For any $k \geq 1$, we have that*

$$h_{k+1} = \frac{k+1}{k+2} T^{0,\frac{\lambda}{k+1}}_{\pi_{k+1}} h_k + \frac{1}{k+2} \left(q_0 - E_{k+1} - \gamma\lambda P\mathcal{H}(\pi_0) \right).$$

Proof. Using the definition of h_k, Lemma 2, the fact that $q_{k+1} = T^{\lambda,0}_{\pi_{k+1}|\pi_k} q_k + \epsilon_{k+1}$, and the definition $E_k = -\sum_{j=1}^{k} \epsilon_j$, we have

$$\begin{aligned}
(k+2)h_{k+1} &= \sum_{j=0}^{k+1} q_j \\
&= q_0 + q_1 + \sum_{j=1}^{k} q_{j+1} \\
&= q_0 + T^{\lambda,0}_{\pi_1|\pi_0} q_0 + \epsilon_1 + \sum_{j=1}^{k} \left(T^{\lambda,0}_{\pi_{j+1}} q_j + \epsilon_{j+1} \right) \\
&= q_0 + T^{0,\lambda}_{\pi_1} h_0 - \gamma\lambda P\mathcal{H}(\pi_0) + \sum_{j=1}^{k} \left((j+1) T^{0,\frac{\lambda}{j+1}}_{\pi_{j+1}} h_j - j T^{0,\frac{\lambda}{j}}_{\pi_j} h_{j-1} \right) - E_{k+1} \\
&= q_0 - E_{k+1} - \gamma\lambda P\mathcal{H}(\pi_0) + (k+1) T^{0,\frac{\lambda}{k+1}}_{\pi_{k+1}} h_k \\
\Leftrightarrow h_{k+1} &= \frac{k+1}{k+2} T^{0,\frac{\lambda}{k+1}}_{\pi_{k+1}} h_k + \frac{1}{k+2} \left(q_0 - E_{k+1} - \gamma\lambda P\mathcal{H}(\pi_0) \right).
\end{aligned}$$

□

We now have the tools to work on the residual of interest. Starting from Lemma 3, and using the fact that $(k+2)h_{k+1} = (k+1)h_k + q_{k+1}$,

$$\begin{aligned}
h_{k+1} &= \frac{k+1}{k+2} T^{0,\frac{\lambda}{k+1}}_{\pi_{k+1}} h_k + \frac{1}{k+2} \left(q_0 - E_{k+1} - \gamma\lambda P\mathcal{H}(\pi_0) \right) \\
\Leftrightarrow (k+1)h_k + q_{k+1} &= q_0 - E_{k+1} - \gamma\lambda P\mathcal{H}(\pi_0) + (k+1) T^{0,\frac{\lambda}{k+1}}_{\pi_{k+1}} h_k \\
\Leftrightarrow T^{0,\frac{\lambda}{k+1}}_{\pi_{k+1}} h_k - h_k &= \frac{1}{k+1} \left(q_{k+1} - q_0 + E_{k+1} + \gamma\lambda P\mathcal{H}(\pi_0) \right).
\end{aligned}$$

Injecting this last result into decomposition (25), we get

$$\begin{aligned}
q_* - q_{\pi_{k+1}} &\leq (I - \gamma P_{\pi_*})^{-1} (T^{0,\frac{\lambda}{k+1}}_{\pi_{k+1}} h_k - h_k) - (I - \gamma P_{\pi_{k+1}})^{-1} (T^{0,\frac{\lambda}{k+1}}_{\pi_{k+1}} h_k - h_k - \gamma\lambda P\mathcal{H}(\pi_{k+1})) \\
&= (I - \gamma P_{\pi_*})^{-1} \left(\frac{1}{k+1} \left(q_{k+1} - q_0 + E_{k+1} + \gamma\lambda P\mathcal{H}(\pi_0) \right) \right) \\
&\quad - (I - \gamma P_{\pi_{k+1}})^{-1} \left(\frac{1}{k+1} \left(q_{k+1} - q_0 + E_{k+1} + \gamma\lambda P\mathcal{H}(\pi_0) \right) - \gamma \frac{\lambda}{k+1} P\mathcal{H}(\pi_{k+1}) \right) \\
&\leq (I - \gamma P_{\pi_*})^{-1} \left(\frac{1}{k+1} \left(q_{k+1} - q_0 + E_{k+1} + \gamma\lambda P\mathcal{H}(\pi_0) \right) \right) \\
&\quad - (I - \gamma P_{\pi_{k+1}})^{-1} \left(\frac{1}{k+1} \left(q_{k+1} - q_0 + E_{k+1} - \gamma\lambda P\mathcal{H}(\pi_{k+1}) \right) \right),
\end{aligned}$$

where we used for the last inequality the fact that $-(I-\gamma P_{\pi_{k+1}})^{-1}P\mathcal{H}(\pi_0) \leq 0$. Next, using the fact that $q_* - q_{\pi_{k+1}} \geq 0$ and rearranging terms, we have

$$\begin{aligned} q_* - q_{\pi_{k+1}} \leq & \left| \left((I-\gamma P_{\pi_*})^{-1} - (I-\gamma P_{\pi_{k+1}})^{-1} \right) \frac{E_{k+1}}{k+1} \right| \\ & + (I-\gamma P_{\pi_*})^{-1} \left| \frac{q_{k+1} - q_0 + \gamma\lambda P\mathcal{H}(\pi_0)}{k+1} \right| \\ & + (I-\gamma P_{\pi_{k+1}})^{-1} \left| \frac{q_{k+1} - q_0 + \gamma\lambda P\mathcal{H}(\pi_{k+1})}{k+1} \right|. \end{aligned}$$

We assumed that $\|q_{k+1}\|_\infty \leq v_{\max} \leq v_{\max}^\lambda$ (see also Rk. 1). When introducing the algorithm, we assumed that $\|q_0\|_\infty \leq v_{\max}$. Therefore, $\|q_0 - \gamma\lambda P\mathcal{H}(\pi_0)\|_\infty \leq v_{\max}^\lambda$. Writing $\mathbf{1}$ the vector whose components are all 1, we get $|q_{k+1} - q_0 + \gamma\lambda P\mathcal{H}(\pi_0)| \leq 2v_{\max}^\lambda \mathbf{1}$. Notice that for any policy π, we have that $P_\pi \mathbf{1} = \mathbf{1}$. Therefore, we have

$$(I-\gamma P_{\pi_*})^{-1} \left| \frac{q_{k+1} - q_0 + \gamma\lambda P\mathcal{H}(\pi_0)}{k+1} \right| \leq \frac{2}{1-\gamma} \frac{v_{\max}^\lambda}{k+1} \mathbf{1}.$$

With the same arguments, we have that

$$(I-\gamma P_{\pi_{k+1}})^{-1} \left| \frac{q_{k+1} - q_0 + \gamma\lambda P\mathcal{H}(\pi_{k+1})}{k+1} \right| \leq \frac{2}{1-\gamma} \frac{v_{\max}^\lambda}{k+1} \mathbf{1}.$$

We finally have

$$q_* - q_{\pi_{k+1}} \leq \left| \left((I-\gamma P_{\pi_*})^{-1} - (I-\gamma P_{\pi_{k+1}})^{-1} \right) \frac{E_{k+1}}{k+1} \right| + \frac{4}{1-\gamma} \frac{v_{\max}^\lambda}{k+1} \mathbf{1},$$

which is the stated result.

C.3 About Remark 1

We stated in Rk. 1, in the context of DA-VI(λ,0), that the assumption $\|q_k\|_\infty \leq v_{\max}$ is not strong with approximation, as this just requires clipping the q-values. Indeed, without approximation, it's not even necessary to clip the q-values.

No approximation. We will proceed by induction. Assume that $\|q_k\|_\infty \leq v_{\max}$. We assumed generally that $\|q_0\|_\infty \leq v_{\max}$. Without error, the considered scheme is

$$\begin{cases} \pi_{k+1} = \mathcal{G}^{0,\frac{\lambda}{k+1}}(h_k) \\ q_{k+1} = T^{\lambda,0}_{\pi_{k+1}|\pi_k} q_k \\ h_{k+1} = \frac{k+1}{k+2} h_k + \frac{1}{k+2} q_{k+1} \end{cases} \Leftrightarrow \begin{cases} \pi_{k+1} = \mathcal{G}^{\lambda,0}(q_k) \\ q_{k+1} = T^{\lambda,0}_{\pi_{k+1}|\pi_k} q_k \end{cases}.$$

As $\pi_{k+1} = \mathcal{G}^{\lambda,0}(q_k)$, we have that

$$q_{k+1} = T^{\lambda,0}_{\pi_{k+1}|\pi_k} q_k \geq T^{\lambda,0}_{\pi_k|\pi_k} q_k = T_{\pi_k} q_k \geq -v_{\max}\mathbf{1},$$

The inequality making use of the induction argument. On the other hand, making use of the positiveness of the KL divergence, we have that

$$q_{k+1} = T^{\lambda,0}_{\pi_{k+1}|\pi_k} q_k \leq T_{\pi_{k+1}} q_k \leq v_{\max}\mathbf{1},$$

where again the inequality comes from the induction argument. This allows concluding, $\|q_{k+1}\|_\infty \leq v_{\max}$.

With approximation. Knowing a bound of the q-values without approximation, we can clip q_k such that it satisfies the bound, the effect of the clipping being part of the error. For example, assume that the evaluation step is approximated with a least-squares problems, a parameterized q-function, the target being a sampling of $T^{\lambda,0}_{\pi_{k+1}|\pi_k} q_k$, q_k being the previous approximation (for example the target network). We can clip the result of the least-squares in $[-v_{\max}, +v_{\max}]$ and call the resulting function q_{k+1}. The resulting error is defined as $\epsilon_{k+1} = q_{k+1} - T^{\lambda,0}_{\pi_{k+1}|\pi_k} q_k$.

C.4 Proof of Theorem 2

In this section, we provide a bound for DA-VI(λ,τ). First, we recall the scheme:

$$\begin{cases} \pi_{k+1} = \mathcal{G}^{0,\tau}(h_k) \\ q_{k+1} = T^{\lambda,\tau}_{\pi_{k+1}|\pi_k} q_k + \epsilon_{k+1} \\ h_{k+1} = \beta h_k + (1-\beta) q_{k+1} \text{ with } \beta = \frac{\lambda}{\lambda+\tau} \end{cases}.$$

We recall that due to the entropy term, this scheme cannot converge to the unregularized optimal q_* function. Yet, without errors and with $\lambda = 0$, it would converge to the solution of the MDP regularized by the scaled entropy [20] (optimizing for the reward augmented by the scaled entropy). Our bound will show that adding a KL penalty does not change this. We recall the notations introduced in the main paper. We already have defined the operator $T^{0,\tau}_\pi$. It has a unique fixed point, which we write q^τ_π. The unique optimal q-function is $q^\tau_* = \max_\pi q^\tau_\pi$. We write $\pi^\tau_* = \mathcal{G}^{0,\tau}(q^\tau_*)$ the associated unique optimal policy, and $q^\tau_{\pi^\tau_*} = q^\tau_*$.

Sketch of proof. The proof is similar to the one of Thm. 1, albeit a bit more technical. Thanks to Lemma 4 (that generalizes Lemma 1), we decompose the quantity of interest $q^\tau_* - q^\tau_{\pi_{k+1}}$ as a function of $q^\tau_* - h_k$ and of $T^{0,\tau}_{\pi_{k+1}} h_k - h_k$, to be respectively upper-bounded and lower-bounded. To achieve this, we first derive Lemma 5 that expresses the evaluation step as a difference of Bellman operators applied to successive h-functions (similarly to Lemma 2). Thanks to this, we're able to derive a Bellman-like recursion for h_k in Lemma 6, using notably Lemma 5 and a telescoping argument (similarly to Lemma 3). The end of the proof is then close to the classic propagation of errors of AVI, involving moving averages of the errors instead of the errors, as well as some additional terms.

Proof. The following lemma, generalizing Lemma 1 to the regularized Bellman operator, will be useful:

Lemma 4. *Let $\tau \geq 0$. For any $q \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ and $\pi \in \Delta^{\mathcal{S}}_{\mathcal{A}}$, we have*

$$q^\tau_\pi - q = (I - \gamma P_\pi)^{-1}(T^{0,\tau}_\pi q - q).$$

Proof. The proof is the same as the one of Lemma 1, relying on the fact that the regularized Bellman operator has the same properies as the Bellman operator [20]:

$$q^\tau_\pi - q = T^{0,\tau}_\pi q^\tau_\pi - T^{0,\tau}_\pi q + T^{0,\tau}_\pi q - q = \gamma P_\pi(q^\tau_\pi - q) + T^{0,\tau}_\pi q - q \Leftrightarrow q^\tau_\pi - q = (I - \gamma P_\pi)^{-1}(T^{0,\tau}_\pi q - q).$$

□

We will bound the quantity $q^\tau_* - q^\tau_{\pi_{k+1}}$, using the following decomposition, based on Lemma 4:

$$\begin{aligned} q^\tau_* - q^\tau_{\pi_{k+1}} &= q^\tau_* - h_k + h_k - q^\tau_{\pi_{k+1}} \\ &= (q^\tau_* - h_k) - (I - \gamma P_{\pi_{k+1}})^{-1}(T^{0,\tau}_{\pi_{k+1}} h_k - h_k). \end{aligned} \tag{26}$$

To do so, we will upper-bound $q^\tau_* - h_k$ and lower-bound $T^{0,\tau}_{\pi_{k+1}} h_k - h_k$ (we recall that the matrix $(I - \gamma P_{\pi_{k+1}})^{-1}$ is non-negative). This requires a Bellman-like induction on h_k. For this, the following intermediate lemma, similar to Lemma 2, will be useful.

Lemma 5. *For any $k \geq 0$, we have that*

$$T^{\lambda,\tau}_{\pi_{k+1}|\pi_k} q_k = \frac{1}{1-\beta}\left(T^{0,\tau}_{\pi_{k+1}} h_k - \beta T^{0,\tau}_{\pi_k} h_{k-1}\right).$$

Proof. We have that, for any π,

$$\begin{aligned} \langle \pi, q_k \rangle - \lambda \,\mathrm{KL}(\pi||\pi_k) + \tau \mathcal{H}(\pi) &= \langle \pi, q_k \rangle - \lambda \langle \pi, \ln \pi - \ln \pi_k \rangle - \tau \langle \pi, \ln \pi \rangle \\ &= \langle \pi, q_k + \lambda \ln \pi_k \rangle - (\lambda + \tau) \langle \pi, \ln \pi \rangle. \end{aligned}$$

As $\pi_{k+1} \propto \exp \frac{h_k}{\tau}$, using also the fact that $\beta = \frac{\lambda}{\lambda+\tau}$ and $1-\beta = \frac{\tau}{\lambda+\tau}$, as well as the definition of h_k (22), we have

$$\begin{aligned} q_k + \lambda \ln \pi_k &= q_k + \lambda \left(\frac{h_{k-1}}{\tau} - \ln \langle \mathbf{1}, \exp \frac{h_{k-1}}{\tau} \rangle \right) \\ &= \frac{1}{1-\beta} \left((1-\beta) q_k + \beta h_{k-1} - \beta\tau \ln \langle \mathbf{1}, \exp \frac{h_{k-1}}{\tau} \rangle \right) \\ &= \frac{1}{1-\beta} \left(h_k - \beta\tau \ln \langle \mathbf{1}, \exp \frac{h_{k-1}}{\tau} \rangle \right). \end{aligned}$$

Hence, injecting this in the previous result, we get

$$\begin{aligned} \langle \pi, q_k + \lambda \ln \pi_k \rangle - (\lambda+\tau) \langle \pi, \ln \pi \rangle &= \langle \pi, q_k + \lambda \ln \pi_k \rangle - \frac{\tau}{1-\beta} \langle \pi, \ln \pi \rangle \\ &= \frac{1}{1-\beta} \left(\langle \pi, h_k \rangle - \tau \langle \pi, \ln \pi \rangle - \beta\tau \ln \langle \mathbf{1}, \exp \frac{h_{k-1}}{\tau} \rangle \right). \end{aligned}$$

Now, as $\pi_{k+1} \propto \exp \frac{h_k}{\tau}$, we have that $\langle \pi_{k+1}, h_k \rangle + \tau \mathcal{H}(\pi_{k+1}) = \tau \ln \langle \mathbf{1}, \exp \frac{h_k}{\tau} \rangle$ (again from Eq. (6)), therefore

$$\begin{aligned} &\langle \pi_{k+1}, q_k \rangle - \lambda \, \mathrm{KL}(\pi_{k+1} || \pi_k) + \tau \mathcal{H}(\pi_{k+1}) \\ =& \frac{1}{1-\beta} \left(\langle \pi_{k+1}, h_k \rangle + \tau \mathcal{H}(\pi_{k+1}) - \beta (\langle \pi_k, h_{k-1} \rangle + \tau \mathcal{H}(\pi_k)) \right). \end{aligned}$$

The result follows by the definition of $T^{\lambda,\tau}_{\pi_{k+1}|\pi_k} q_k = r + \gamma P(\langle \pi_{k+1}, q_k \rangle - \lambda \, \mathrm{KL}(\pi_{k+1}||\pi_k) + \tau \mathcal{H}(\pi_{k+1}))$, and noticing that $r = \frac{1}{1-\beta}(r - \beta r)$. □

This result allows to build the lemma stating a Bellman-like induction for h_k.

Lemma 6. *Define $E^\beta_{k+1} = -(1-\beta) \sum_{j=1}^{k+1} \beta^{k+1-j} \epsilon_j = \beta E^\beta_k + (1-\beta)\epsilon_{k+1}$ (with $E^\beta_0 = 0$). For any $k \geq 0$, we have that*

$$h_{k+1} = T^{0,\tau}_{\pi_{k+1}} h_k - E_{k+1} - \beta^{k+1} (T^{0,\tau}_{\pi_0} h_{-1} - h_0).$$

Proof. Using the definition of h_k, Eq. (22), the relationship between q_{k+1} and q_k, and Lemma 5, we have

$$\begin{aligned} h_{k+1} &= (1-\beta) \sum_{j=0}^{k+1} \beta^{k+1-j} q_k \\ &= (1-\beta)\beta^{k+1} q_0 + (1-\beta) \sum_{j=1}^{k+1} \beta^{k+1-j} q_j \\ &= (1-\beta)\beta^{k+1} q_0 + (1-\beta) \sum_{j=0}^{k} \beta^{k-j} q_{j+1} \\ &= (1-\beta)\beta^{k+1} q_0 + (1-\beta) \sum_{j=0}^{k} \beta^{k-j} \left(T^{\lambda,\tau}_{\pi_{j+1}|\pi_j} q_j + \epsilon_{j+1} \right) \\ &= (1-\beta)\beta^{k+1} q_0 + (1-\beta) \sum_{j=0}^{k} \beta^{k-j} \left(\frac{1}{1-\beta} \left(T^{0,\tau}_{\pi_{j+1}} h_j - \beta T^{0,\tau}_{\pi_j} h_{j-1} \right) + \epsilon_{j+1} \right). \end{aligned}$$

Let define E^β_{k+1} as

$$\begin{aligned}
E_{k+1} &= -(1-\beta)\sum_{j=0}^{k}\beta^{k-j}\epsilon_{j+1} \\
&= -(1-\beta)\sum_{j=1}^{k+1}\beta^{k+1-j}\epsilon_{j} \\
&= \beta E^\beta_k + (1-\beta)\epsilon_{k+1} \text{ with } E_0 = 0.
\end{aligned}$$

We also have

$$\begin{aligned}
&(1-\beta)\sum_{j=0}^{k}\beta^{k-j}\left(\frac{1}{1-\beta}\left(T^{0,\tau}_{\pi_{j+1}}h_j - \beta T^{0,\tau}_{\pi_j}h_{j-1}\right)\right) \\
=&\sum_{j=0}^{k}\beta^{k-j}\left(T^{0,\tau}_{\pi_{j+1}}h_j - \beta T^{0,\tau}_{\pi_j}h_{j-1}\right) \\
=&\sum_{j=1}^{k+1}\beta^{k+1-j}T^{0,\tau}_{\pi_j}h_{j-1} - \sum_{j=0}^{k}\beta^{k+1-j}T^{0,\tau}_{\pi_j}h_{j-1} \\
=&T^{0,\tau}_{\pi_{k+1}}h_k - \beta^{k+1}T^{0,\tau}_{\pi_0}h_{-1}.
\end{aligned}$$

Notice also that $h_0 = (1-\beta)q_0$. Putting all these parts together, we obtain

$$\begin{aligned}
h_{k+1} &= \beta^{k+1}h_0 - E^\beta_{k+1} + T^{0,\tau}_{\pi_{k+1}}h_k - \beta^{k+1}T^{0,\tau}_{\pi_0}h_{-1} \\
&= T^{0,\tau}_{\pi_{k+1}}h_k - E^\beta_{k+1} - \beta^{k+1}(T^{0,\tau}_{\pi_0}h_{-1} - h_0),
\end{aligned}$$

which is the stated result. □

Thanks to this result, we can now bound the terms of interest.

Upper-bounding $q^\tau_* - h_k$. Write $e_k = E^\beta_k + \beta^k(T^{0,\tau}_{\pi_0}h_{-1} - h_0)$, we have from Lemma 6 that $h_{k+1} = T^{0,\tau}_{\pi_{k+1}}h_k - e_{k+1}$. Then, we have :

$$\begin{aligned}
q^\tau_* - h_{k+1} &= q^\tau_* - T^{0,\tau}_{\pi_{k+1}}h_k + e_{k+1} \\
&= \underbrace{T^{0,\tau}_{\pi^\tau_*}q^\tau_* - T^{0,\tau}_{\pi^\tau_*}h_k}_{=\gamma P_{\pi^\tau_*}(q^\tau_* - h_k)} + \underbrace{T^{0,\tau}_{\pi^\tau_*}h_k - T^{0,\tau}_{\pi_{k+1}}h_k}_{\leq 0 \text{ as } \pi_{k+1} = \mathcal{G}^{0,\tau}(h_k)} + e_{k+1} \\
&\leq \gamma P_{\pi^\tau_*}(q^\tau_* - h_k) + e_{k+1}.
\end{aligned}$$

By direct induction, we obtain

$$\begin{aligned}
q^\tau_* - h_{k+1} &\leq (\gamma P_{\pi^\tau_*})^{k+1}(q^\tau_* - h_0) + \sum_{j=1}^{k+1}(\gamma P_{\pi^\tau_*})^{k+1-j}e_j \\
&= (\gamma P_{\pi^\tau_*})^{k+1}(q^\tau_* - h_0) + \sum_{j=1}^{k+1}(\gamma P_{\pi^\tau_*})^{k+1-j}\left(E^\beta_j + \beta^j(T^{0,\tau}_{\pi_0}h_{-1} - h_0)\right). \quad (27)
\end{aligned}$$

This is the desired upper-bound.

Lower-bounding $T^{0,\tau}_{\pi_{k+1}}h_k - h_k$. Using the same notation e_k, we have

$$\begin{aligned}
T^{0,\tau}_{\pi_{k+1}}h_k - h_k &= \underbrace{T^{0,\tau}_{\pi_{k+1}}h_k - T^{0,\tau}_{\pi_k}h_k}_{\geq 0 \text{ as } \pi_{k+1} = \mathcal{G}^{0,\tau}(h_k)} + T^{0,\tau}_{\pi_k}h_k - h_k \\
&\geq T^{0,\tau}_{\pi_k}h_k - h_k \\
&= T^{0,\tau}_{\pi_k}\left(T^{0,\tau}_{\pi_k}h_{k-1} - e_k\right) - \left(T^{0,\tau}_{\pi_k}h_{k-1} - e_k\right) \text{ by Lemma 6} \\
&= \gamma P_{\pi_k}\left(T^{0,\tau}_{\pi_k}h_{k-1} - h_{k-1}\right) - (I - \gamma P_{\pi_k})^{-1}e_k.
\end{aligned}$$

We define $P_{k:j} = P_{\pi_k} P_{\pi_{k-1}} \dots P_{\pi_{j+1}} P_{\pi_j}$ for $j \leq k$, with the convention $P_{k:k+1} = I$. By direct induction, the preceding inequality gives

$$\begin{aligned} T^{0,\tau}_{\pi_{k+1}} h_k - h_k &\geq \gamma^k P_{k:1}(T^{0,\tau}_{\pi_1} h_0 - h_0) - \sum_{j=1}^{k} \gamma^{k-j} P_{k:j+1}(I - \gamma P_{\pi_j}) e_j \\ &= \gamma^k P_{k:1}(T^{0,\tau}_{\pi_1} h_0 - h_0) - \sum_{j=1}^{k} \gamma^{k-j} P_{k:j+1}(I - \gamma P_{\pi_j})(E^\beta_j + \beta^j (T^{0,\tau}_{\pi_0} h_{-1} - h_0)). \end{aligned} \tag{28}$$

Putting things together. Plugging Eqs. (27) and (28) into Eq. (26), we obtain

$$\begin{aligned} q^\tau_* - q^\tau_{\pi_{k+1}} \leq & (\gamma P_{\pi^\tau_*})^k (q^\tau_* - h_0) + \sum_{j=1}^{k} (\gamma P_{\pi^\tau_*})^{k-j} \left(E^\beta_j + \beta^j (T^{0,\tau}_{\pi_0} h_{-1} - h_0) \right) \\ & + (I - \gamma P_{\pi_{k+1}})^{-1} \Bigg(- \gamma^k P_{k:1}(T^{0,\tau}_{\pi_1} h_0 - h_0) \\ & + \sum_{j=1}^{k} \gamma^{k-j} P_{k:j+1}(I - \gamma P_{\pi_j})(E^\beta_j + \beta^j (T^{0,\tau}_{\pi_0} h_{-1} - h_0)) \Bigg). \end{aligned}$$

Using the fact that $q^\tau_* - q^\tau_{\pi_{k+1}} \geq 0$, rearranging terms, we have

$$\begin{aligned} q^\tau_* - q^\tau_{\pi_{k+1}} \leq & \sum_{j=1}^{k} \left| (\gamma P_{\pi^\tau_*})^{k-j} + (I - \gamma P_{\pi_{k+1}})^{-1} \gamma^{k-j} P_{k:j+1} \left(I - \gamma P_{\pi_j} \right) E^\beta_j \right| \\ & + (\gamma P_{\pi^\tau_*})^k |q^\tau_* - h_0| + \sum_{j=1}^{k} (\gamma P_{\pi^\tau_*})^{k-j} \beta^j |T^{0,\tau}_{\pi_0} h_{-1} - h_0| \\ & + (I - \gamma P_{\pi_{k+1}})^{-1} \gamma^k P_{k:1} |T^{0,\tau}_{\pi_1} h_0 - h_0| \\ & + (I - \gamma P_{\pi_{k+1}})^{-1} \sum_{j=1}^{k} \gamma^{k-j} P_{k:j+1}(I + \gamma P_{\pi_j}) \beta^j |T^{0,\tau}_{\pi_0} h_{-1} - h_0|. \end{aligned} \tag{29}$$

The first term is related to the error, the others to the initialisation. We'll work on each of these other terms.

Recall that we assumed that $\|q_0\|_\infty \leq v_{\max} = \frac{r_{\max}}{1-\gamma}$. Therefore, $\|q_0\|_\infty \leq v^\tau_{\max} = \frac{r_{\max} + \tau \ln |\mathcal{A}|}{1-\gamma}$. As $h_0 = (1-\beta) q_0$, we have $\|h_0\|_\infty \leq (1-\beta) v^\tau_{\max}$. From obvious properties of regularized MDPs [20], we have $\|q^\tau_*\|_\infty \leq v^\tau_{\max}$. Therefore, writing $\mathbf{1} \in \mathbb{R}^{\mathcal{S} \times \mathcal{A}}$ the vector with all components equal to 1, we have $|q^\tau_* - h_0| \leq (2-\beta) v^\tau_{\max} \mathbf{1}$. Notice that for any policy π, we have $P_\pi \mathbf{1} = \mathbf{1}$, thus

$$(\gamma P_{\pi^\tau_*})^k |q^\tau_* - h_0| \leq \gamma^k (2-\beta) v^\tau_{\max} \mathbf{1}.$$

We also have that $\|T^{0,\tau}_{\pi_1} h_0\|_\infty \leq r_{\max} + \tau \ln |\mathcal{A}| + \gamma(1-\beta) v^\tau_{\max} = (1-\gamma\beta) v^\tau_{\max}$, so

$$(I - \gamma P_{\pi_{k+1}})^{-1} \gamma^k P_{k:1} |T^{0,\tau}_{\pi_1} h_0 - h_0| \leq \gamma^k \frac{2 - (1+\gamma)\beta}{1-\gamma} v^\tau_{\max} \mathbf{1}.$$

By definition $h_{-1} = 0$, so we have $\|T^{0,\tau}_{\pi_0} h_{-1}\|_\infty = \|r + \gamma P \tau \mathcal{H}(\pi_0)\|_\infty \leq r_{\max} + \tau \ln |\mathcal{A}| = (1-\gamma) v^\tau_{\max}$, so $\|T^{0,\tau}_{\pi_0} h_{-1} - h_0\|_\infty \leq (2 - \gamma - \beta) v^\tau_{\max}$. Therefore, we have the following bound:

$$\sum_{j=1}^{k} (\gamma P_{\pi^\tau_*})^{k-j} \beta^j |T^{0,\tau}_{\pi_0} h_{-1} - h_0| \leq \gamma^k \sum_{j=1}^{k} \left(\frac{\beta}{\gamma} \right)^j (2 - \beta - \gamma) v^\tau_{\max} \mathbf{1}.$$

Similarly, for the last term we have

$$(I - \gamma P_{\pi_{k+1}})^{-1} \sum_{j=1}^{k} \gamma^{k-j} P_{k:j+1}(I + \gamma P_{\pi_j}) \beta^j |T^{0,\tau}_{\pi_0} h_{-1} - h_0| \leq \frac{1+\gamma}{1-\gamma} \gamma^k \sum_{j=1}^{k} \left(\frac{\beta}{\gamma} \right)^j (2 - \beta - \gamma) v^\tau_{\max} \mathbf{1}.$$

Summing these four upper bounds, we obtain

$$\gamma^k(2-\beta)v_{\max}^\tau\mathbf{1}+\gamma^k\frac{2-(1+\gamma)\beta}{1-\gamma}v_{\max}^\tau\mathbf{1}+\gamma^k\sum_{j=1}^k\left(\frac{\beta}{\gamma}\right)^j(2-\beta-\gamma)v_{\max}^\tau\mathbf{1}$$
$$+\frac{1+\gamma}{1-\gamma}\gamma^k\sum_{j=1}^k\left(\frac{\beta}{\gamma}\right)^j(2-\beta-\gamma)v_{\max}^\tau\mathbf{1}$$
$$=2\gamma^k\frac{2-\beta-\gamma}{1-\gamma}\sum_{j=0}^k\left(\frac{\beta}{\gamma}\right)^j v_{\max}^\tau\mathbf{1}=2\gamma^k\left(1+\frac{1-\beta}{1-\gamma}\right)\sum_{j=0}^k\left(\frac{\beta}{\gamma}\right)^j v_{\max}^\tau\mathbf{1}.$$

Plugging this result into Eq. (29), we obtain the stated result:

$$q_*^\tau-q_{\pi_{k+1}}^\tau\leq\sum_{j=1}^k\left|(\gamma P_{\pi_*^\tau})^{k-j}+(I-\gamma P_{\pi_{k+1}})^{-1}\gamma^{k-j}P_{k:j+1}\left(I-\gamma P_{\pi_j}\right)E_j^\beta\right|$$
$$+\gamma^k\left(1+\frac{1-\beta}{1-\gamma}\right)\sum_{j=0}^k\left(\frac{\beta}{\gamma}\right)^j v_{\max}^\tau\mathbf{1}.$$

D Empirical illustration of the bounds

We have illustrated the bounds of Sec. 2 (Fig. 1) in a simple tabular setting with acces to a generative model. We provide more details about this setting here.

We consider MDPs with small state and action spaces, such that a tabular representation of the q-function is possible. We also assume to have access to a generative model, allowing us to sample a transition for any state-action couple. We then consider sampled MD-VI(λ,τ), depicted in Alg. 1. At each iteration of MD-VI, we sample a single transition for each state-action couple and apply the resulting sampled Bellman operator. The error ϵ_k is the difference between the sampled and the exact operators. The sequence of these estimation errors is thus a martingale difference w.r.t. its natural filtration [6] (one can think about bounded, centered and roughly i.i.d. errors).

We run this algorithm on randomized MDPs called Garnets. A Garnet [4] is an abstract MDP, built from three parameters (N_S, N_A, N_B), with N_S and N_A respectively the number of states and actions, and N_B the branching factor. The principle is to directly build the transition kernel P that represents the MDP. For each $(s,a)\in\mathcal{S}\times\mathcal{A}$, N_B states $(s_1,\dots s_{N_B})$ are drawn uniformly from $\mathcal{S}$ without replacement. Then, N_B-1 numbers are drawn uniformly in $(0,1)$ and sorted as $(p_0=0,p_1,\dots p_{N_B-1},p_{N_B}=1)$. The transition kernel is then defined as $P(s_k|s,a)=p_k-p_{k-1}$ for each $1\leq k\leq N_B$. The reward function is drawn uniformly in $(0,1)$ for 10% of the states, these states being drawn uniformly without replacement.

For the experiments shown in Fig. 1, we set $N_S=30$, $N_A=4$, $N_B=4$ and $\gamma=0.9$. We generate 100 Garnets and run MD-VI once for each of these Garnets, for $K=800$ iterations. The results in Fig. 1 shows the normalized average performance, $\frac{\|q_*^\tau-q_{\pi_k}^\tau\|_1}{\|q_*^\tau\|_1}$. For sampled DA-VI($\lambda$, 0), we show the behavior for various values of λ. For DA-VI(λ, τ), we fix τ to a small value ($\tau=10^{-3}$) and show the behavior for various values of $\beta=\frac{\lambda}{\lambda+\tau}$. Notice that considering a large value of τ would not be interesting. In this case, the regularized optimal policy would be close to be uniform, so close to the initial policy.

E Algorithms and experimental details

This appendix provides additional details about the algorithms and the experiments:

- Appx. E.1 provides a complementary high level view of algorithms sketched in Sec. 5.
- Appx. E.2 provides implementation details of these algorithms, including a pseudo-code.

Algorithm 1 Sampled MD-VI(λ, τ)

Require: K number of iterations, P the transition kernel.
set $\beta = \frac{\lambda}{\lambda+\tau}$
set q_0 to the null vector
set π_0 to be the uniform policy
  for $1 \leq k \leq K$ **do**
    for $(s,a) \in \mathcal{S} \times \mathcal{A}$ **do**
      $\pi_k(a|s) = \frac{\pi_{k-1}(a|s)^\beta \exp \frac{q_{k-1}(s,a)}{\lambda+\tau}}{\sum_{b\in\mathcal{A}} \pi_{k-1}(b|s)^\beta \exp \frac{q_{k-1}(s,b)}{\lambda+\tau}}$
    end for
    for $(s,a) \in \mathcal{S} \times \mathcal{A}$ **do**
      $s' \sim P(\cdot|s,a)$
      $q_k(s,a) = r(s,a) + \gamma \sum_{b\in\mathcal{A}} \pi_k(b|s') \left(q_{k-1}(s',b) - \lambda \ln \frac{\pi_k(b|s')}{\pi_{k-1}(b|s')} - \tau \ln \pi_k(b|s') \right)$
    end for
  end for
output π_K

- Appx. E.3 provides all hyperparameters used in our experiments.
- Appx. E.4 provides additionnal experiments (one additional gym environment, Lunar Lander, and two additional Atari games, Breakout and Seaquest), as well as additional visualisations (including all training curves on Atari games).

E.1 High level view of practical algorithms

DA-VI and MD-VI are extensions of VI. One of the most prevalent VI-based deep RL algorithm is probably DQN [27]. Thus, our approach consists in modifying the DQN algorithm to study regularization. To complement the sketch of Sec. 5, We present the different variations we consider with a high level viewpoint here, all practical details being just after.

DQN maintains a replay buffer and a target network q_k, and computes q_{k+1} by minimizing the loss (recall that '*w/o*' stands for "without regularization"):

$$\mathcal{L}_{w/o}(q) = \hat{\mathbb{E}}_{s,a} \left[\left([\hat{T}_{\pi_{k+1}} q_k](s,a) - q(s,a) \right)^2 \right], \tag{30}$$

with q a neural network, $\pi_{k+1} \in \mathcal{G}(q_k)$ the greedy policy computed analytically from q_k, $[\hat{T}_{\pi_{k+1}} q_k](s,a) = r(s,a) + \gamma \langle \pi_{k+1}, q_k \rangle(s')$ the sampled Bellman operator (with $s' \sim P(\cdot|s,a)$), and where the empirical expectation $\mathbb{E}_{s,a}$ is according to the transitions in the buffer. DQN is an optimistic AVI scheme, in the sense that only a few steps of stochastic gradient descent are performed before updating the target network. We modify DQN by adding a policy network and possibly modifying the evaluation step. For the moment, we consider $\tau > 0$.

Greedy step. As explained before, when the greedy step is approximated, MD-VI and DA-VI are no longer equivalent. We start with MD-VI. A natural way to learn the policy network is to optimize directly for the greedy step. Let π_k be the target policy network and q_k the target q-network, it corresponds to ('dir' stands for direct):

$$\mathcal{L}_{\text{dir}}(\pi) = \hat{\mathbb{E}}_s \left[\langle \pi, q_k \rangle(s) - \lambda \, \text{KL}(\pi || \pi_k)(s) + \tau \mathcal{H}(\pi)(s) \right]. \tag{31}$$

Maximizing this loss over networks gives π_{k+1}. This is reminiscent of TRPO (see Appx. B.1).

One can also compute analytically the policy π_{k+1} (see Appx. A), but it would require remembering all past networks. Thus, another solution is to approximate this analytical solution by a neural network ('ind' stands for indirect):

$$\mathcal{L}_{\text{ind}}(\pi) = \hat{\mathbb{E}}_s \left[\text{KL}(\pi^*_{k+1} || \pi)(s) \right] \text{ with } \pi^*_{k+1} \propto \pi_k^\beta \exp \frac{\beta q_k}{\lambda}.$$

Minimizing this loss over networks gives π_{k+1}. This is reminiscent of MPO (see Appx. B.1), up to the fact that we consider the KL in the reverse order. Indeed, MPO (or SAC) would optimise for $\hat{\mathbb{E}}_s[\mathrm{KL}(\pi||\pi^*_{k+1})(s)]$. The motivation to do so is to get ride of the partition function. Yet, this is equivalent to what we call the "direct" approach, writing $Z_k \in \mathbb{R}^{\mathcal{S}}$ the partition function:

$$\begin{aligned}
\mathrm{KL}(\pi||\pi^*_{k+1}) &= \langle \pi, \ln \frac{\pi_k^\beta \exp \frac{\beta q_k}{\lambda}}{Z_k} - \ln \pi \rangle \\
&= \langle \pi, \ln(\pi_k^\beta \exp \frac{\beta q_k}{\lambda}) \rangle - \langle \pi, \ln Z_k \rangle - \langle \pi, \ln \pi \rangle \\
&= \frac{\beta}{\lambda} \langle \pi, q_k \rangle + \beta \langle \pi, \ln \pi_k \rangle - \langle \pi, \ln \pi \rangle - \ln Z_k \\
&= \frac{\beta}{\lambda} \left(\langle \pi, q_k \rangle + \lambda \langle \pi, \ln \pi_k \rangle - (\lambda + \tau) \langle \pi, \ln \pi \rangle - (\lambda + \tau) \ln Z_k \right) \\
&= \frac{\beta}{\lambda} \left(\langle \pi, q_k \rangle - \lambda \, \mathrm{KL}(\pi||\pi_k) + \tau \mathcal{H}(\pi) - \ln Z_k \right).
\end{aligned}$$

So, up to the scaling $\frac{\beta}{\lambda} = \frac{1}{\lambda+\tau}$ and to the term $\ln Z_k$, which is a constant regarding the optimized policy π and can thus safely be ignored, we obtain the loss of Eq. (31).

When considering DA-VI, the policy can be computed analytically, $\pi_{k+1} = \mathcal{G}^{0,\tau}(h_k)$, but h_k has to be approximated (and can be seen as the logits of the policy). With h_{k-1} and q_k the target networks:

$$\mathcal{L}_{\mathrm{da}}(h) = \hat{\mathbb{E}}_{s,a} \left[\left([\beta h_{k-1} + (1-\beta) q_k](s,a) - h(s,a) \right)^2 \right].$$

Minimizing this loss over networks h gives h_k. This is reminiscent of momentum-DQN (see Appx. B.2).

Evaluation step. Given one of the three ways of doing the greedy step, one can choose between regularizing the evaluation step (*w/*, as suggested by the theory) or not (*w/o*, as often done empirically). This second case is already depicted in Eq. (30) (changing the considered policy) and the first case is given by

$$\mathcal{L}_{w/}(q) = \hat{\mathbb{E}}_{s,a} \left[\left([\hat{T}^{\lambda,\tau}_{\pi_{k+1}} q_k](s,a) - q(s,a) \right)^2 \right].$$

So combining one of the two evaluation steps (*w/* or *w/o*) with one of the three greedy steps (MD-dir, MD-ind or DA), we get six variations. We discuss also the limit case without entropy.

When $\tau = 0$. For MD-VI, one can set $\tau = 0$. However, recall that for DA-VI, the resulting algorithm is different. DA-VI(λ, 0) is not practical in a deep learning setting, as it requires averaging over iterations. Indeed, updates of target networks are too fast to consider them as new iterations, and a moving average is more convenient. Vieillard et al. [43] used a decay on β to mimic this behavior, but this is a heuristic that needs to be tuned. Therefore, for DA-VI we will only consider the limit case $\lambda + \tau \to 0$ with $\beta = \frac{\lambda}{\lambda+\tau}$ kept constant (that is, momentum-DQN with fixed β). In this case, type 1 and 2 are equivalent. We offer additional visualisations in Appx. E.4.

E.2 More on practical algorithms

We now detail the losses presented in the previous section, giving equations that are closer to implementation, and providing a detailed pseudo-code in Algorithm 2. Firts, let us introduce some notations.The q-value is represented by a neural network Q_θ of parameters θ, and the policy is represented by a network Π_ϕ of parameters ϕ. During training, the algorithms interact with an environment, and collect transitions (s, a, r, s') that are stored in a FIFO replay buffer $\mathcal{B}$. The parameters of the networks are copied regularly into old versions of themselves, with target weights $\bar{\theta}$ and $\bar{\phi}$. The weights θ are optimized during the evaluation step, and ϕ during the greedy step.

E.2.1 Evaluation step

All the actor-critics we consider have the same update rule of their critic – the Q-network. We consider two regressions targets, corresponding to regularizing the evaluation step or not. If not regularized, we define a regression target as

$$\hat{Q}_{w/o}(r, s') = r + \gamma \sum_{b \in \mathcal{A}} Q_{\bar{\theta}}(s', b) \Pi_{\phi}(b|s'),$$

and if regularized,

$$\hat{Q}_{w/}(r, s') = \hat{Q}_2(r, s') - \lambda \, \mathrm{KL}\left(\Pi_{\phi} \| \Pi_{\bar{\phi}}\right)(s') + \tau \mathcal{H}\left(\Pi_{\phi}\right)(s').$$

The weights θ are then updated by minimizing the following regression loss with a variant of SGD

$$\mathcal{L}_{w/\text{-}w/o}(\theta) = \hat{E}_{\mathcal{B}}\left[\left(Q_{\theta}(s, a) - \hat{Q}_{w/\text{-}w/o}(r, s')\right)^2\right]. \tag{32}$$

Note that if Π_{ϕ} was greedy with respect to $Q_{\bar{\theta}}$, using $\mathcal{L}_{w/o}$ would reduce to Deep q-networks (DQN) [27].

E.2.2 Greedy step

Let us re-write in detail the three equations from Section E.1 that define three ways of performing the greedy step.

MD-dir. The Direct MD update tackles directly the optimization problem derived from the greedy step. For convenience, we define a loss (the opposite of what we would like to maximize) that we minimize with SGD

$$\mathcal{L}_{\text{dir}}(\phi) = \hat{E}_{\mathcal{B}}\left[-\sum_{b \in \mathcal{A}} Q_{\bar{\theta}}(s, b) \Pi_{\phi}(b|s) + \lambda \, \mathrm{KL}\left(\Pi_{\phi} \| \Pi_{\bar{\phi}}\right)(s') - \tau \mathcal{H}\left(\Pi_{\phi}\right)(s')\right]. \tag{33}$$

MD-ind. The indirect version is based on the analytical result of the optimization problem corresponding to the greedy step. We show in Appendix B.1 that , at iteration k of MD-VI(λ, τ), we have $\pi_{k+1} = \mathcal{G}^{\lambda,\tau}_{\pi_k}(q_k) \propto \pi_k^{\beta} \exp \frac{q_k}{\tau+\lambda}$. Hence, we would need to fit a target that approximates this maximizer, by defining $\hat{\Pi}(a|s)$ as

$$\hat{\Pi}(a|s) = \Pi_{\bar{\phi}}(a|s)^{\beta} \exp \frac{Q_{\bar{\theta}}(s, a)}{\lambda + \tau} \left(\sum_{b \in \mathcal{A}} \Pi_{\bar{\phi}}(b|s)^{\beta} \exp \frac{Q_{\bar{\theta}}(s, b)}{\lambda + \tau}\right)^{-1}.$$

However, the exponential term can cause numerical problems, so what we optimize during the evaluation step is actually the logarithm of the policy. To work around this, we define a network L_{ϕ} that represents the log-probabilities of a policy, and we define a regression target

$$\hat{L}(s, a) = \frac{\lambda L_{\bar{\phi}}(a|s) + Q_{\bar{\theta}}(s, a)}{\lambda + \tau} - \ln \sum_{b \in \mathcal{A}} \frac{\lambda L_{\bar{\phi}}(b|s) + Q_{\bar{\theta}}(s, b)}{\lambda + \tau},$$

and then we have $\hat{\Pi}(a|s) = \exp\left(\hat{L}(s, a)\right)$ and $\Pi_{\phi}(a|s) = \exp\left(L_{\phi}(a|s)\right)$. We then define a loss on the parameters ϕ,

$$\mathcal{L}_{\text{ind}}(\phi) = \hat{E}_{\mathcal{B}}\left[\mathrm{KL}\left(\hat{\Pi} \| \Pi_{\phi}\right)(s)\right]. \tag{34}$$

DA. The dual averaging version is inspired by the DA-VI formulation. Instead of representing directly the policy, we estimate a moving average of the q-values, and then compute its softmax. The moving average is estimated via a network H_{ϕ}, which fits a regression target

$$\hat{H}(s, a) = \beta H_{\bar{\phi}}(s, a) + (1 - \beta) Q_{\bar{\theta}}(s, a),$$

and the policy is defined as softmax over $H_\phi(s, \cdot)$,

$$\Pi_\phi(a|s) = \exp \frac{H_\phi(s,a)}{\tau} \left(\sum_b \exp \frac{H_\phi(s,b)}{\tau} \right)^{-1}.$$

The weights ϕ are optimized by minimizng the loss

$$\mathcal{L}_{\text{da}}(\phi) = \hat{E}_{\mathcal{B}} \left[\left(H_\phi(s,a) - \hat{H}(s,a) \right)^2 \right]. \tag{35}$$

E.2.3 Pseudo code

We give a general pseudo-code of the deep RL algorithms we used in Alg. 2. Notice that for a policy π, we define the e-greedy policy with respect to π as the policy that takes a random action (uniformly on $\mathcal{A}$) with probability e, and follows π with probability $1 - e$.

Algorithm 2 (MD-dir | MD-ind | DA)

Require: $L_q(\theta)$ and $L_\pi(\phi)$, two losses, respectively for the evaluation and the greediness. The choice of these losses determines the algorithm, see Table 2.
Require: $K \in \mathbb{N}^*$ the number of steps, $C \in \mathbb{N}^*$ the update period, $F \in \mathbb{N}^*$ the interaction period.
set θ, ϕ at random
set Q_θ the q-value network, Π_ϕ the policy network, as defined in Sec. E.2.
set $\mathcal{B} = \{\}$
set Π_{ϕ,e_k} the policy e_k-greedy w.r.t. Π_ϕ
  $\bar{\theta} = \theta, \bar{\phi} = \phi$
  for $1 \leq k \leq K$ **do**
    Collect a transition $t = (s, a, r, s')$ from Π_{ϕ,e_k}
    $\mathcal{B} \leftarrow \mathcal{B} \cup \{t\}$
    if $k \mod F == 0$ **then**
      On a random batch of transitions $B_{q,k} \subset \mathcal{B}$, update θ with one step of SGD on L_q
      On a random batch of transitions $B_{h,k} \subset \mathcal{B}$, update ϕ with one step of SGD on L_π
    end if
    if $k \mod C == 0$ **then**
      $\bar{\theta} \leftarrow \theta$, $\bar{\phi} \leftarrow \phi$
    end if
  end for
output Π_ϕ

Table 2: Resulting algorithms given the choice of losses in Algorithm 2

	L_π		
L_q	$\mathcal{L}_{\text{dir}}$ (Eq.(34))	$\mathcal{L}_{\text{ind}}$ (Eq. (33))	$\mathcal{L}_{\text{da}}$ (Eq. (35))
$\mathcal{L}_{w/}$ (Eq. (32))	MD-dir *w/*	MD-ind *w/*	DA *w/*
$\mathcal{L}_{w/o}$ (Eq. (32))	MD-dir *w/o*	MD-ind *w/o*	DA *w/o*

E.3 Hyperparameters

We provide the hyperparameters used on the Atari environments in Table 3, and on the Gym environments in Table 4. We use the following notations to describe neural networks: FC n is a fully connected layer with n neurons; $\text{Conv}^d_{a,b}\, c$ is a 2d convolutional layer with c filters of size $a \times b$ and a stride of d. All hyperparameters are the one found in the Dopamine code base. We only tuned the learning rate and the update period of DQN on Lunar Lander (not provided in Dopamine).

Table 3: Parameters used on Atari. Both the Q-network and policy-network have the same structure. n_A is the number of actions available in a given game.

Parameter	Value
K (number of steps)	$5 * 10^7$
C (update period)	8000
F (interaction period)	4
γ (discount)	0.99
$\lvert\mathcal{B}\rvert$ (replay buffer size)	10^6
$\lvert B_{\pi,k}\rvert$ and $\lvert B_{q,k}\rvert$ (batch size)	32
e_k (random actions rate)	$e_0 = 0.01$, linear decay of period $2.5 \cdot 10^5$ steps
networks structure	$\text{Conv}^4_{8,8}\, 32 - \text{Conv}^2_{4,4}\, 64 - \text{Conv}^1_{3,3}\, 64 - \text{FC}\, 512 - \text{FC}\, n_A$
activations	Relu
optimizers	RMSprop ($lr = 0.00025$)

Table 4: Parameters used on CartPole and Lunar Lander . Both the Q-network and policy-network have the same structure. We have $n_A = 2$ on CartPole, and $n_A = 8$ on Lunar Lander.

Parameter	Value
K (number of steps)	$5 * 10^5$
C (update period)	100 (Cartpole), 2500 (Lunar Lander)
F (interaction period)	4
γ (discount)	0.99
$\lvert\mathcal{B}\rvert$ (replay buffer size)	$5 * 10^4$
$\lvert B_{\pi,k}\rvert$ and $\lvert B_{q,k}\rvert$ (batch size)	128
e_k (random actions rate)	0.01 (constant with k)
networks structure	$\text{FC}\, 512 - \text{FC}\, 512 - \text{FC}\, n_A$
activations	Relu
optimizers	Adam ($lr = 0.001$)

E.4 Additional results

Additional environment. In addition to the environments considered in Sec. 5, we provide three additional environments: Lunar Lander (from gym), Breakout and Seaquest (from Atari). The comments on these environments are similar to the discussion of Sec. 5

Full tables. We also provide the full results of the experiments (those from Section 5 and the new ones). The same plots are reported, expect that we add the exact value of each grid cell for completeness. Results for Carpole and Lunarlander are provided in Figs. 4 and 5, while results for the considered Atari games (Asterix, Breakout and Seaquest) are reported in Figs. 6, 7 and 8.

Training curves. We also report training curves on Atari. We report training curves of DA, MD-dir and MD-ind in Fig. 9 for Asterix, on Fig. 10 for Breakout, and on Fig. 11 for Seaquest. We report the training curves of the limit cases on these three games on Figs. 12, 13 and 14. In these figures, an *iteration* corresponds to 250000 training steps, and we report every iteration the undiscounted reward averaged over the last 100 episodes (the *averaged score*). The training curves are averaged over 3 random seeds.

The training curves give more hindsights on the performance of the algorithms. Indeed, the metric we used in the tables (the averaged score over all iteration) is partly flawed, because it could give a high score to an algorithm with a performance drop at the end of training. For example, the MD-dir method on Atari seems to benefit from regularizing the evaluation step (as unregularized evaluation suffers from a performance drop), which is less visible from the score tables. In almost all the cases, we do not observe such behaviour, which validates the use of our metric.

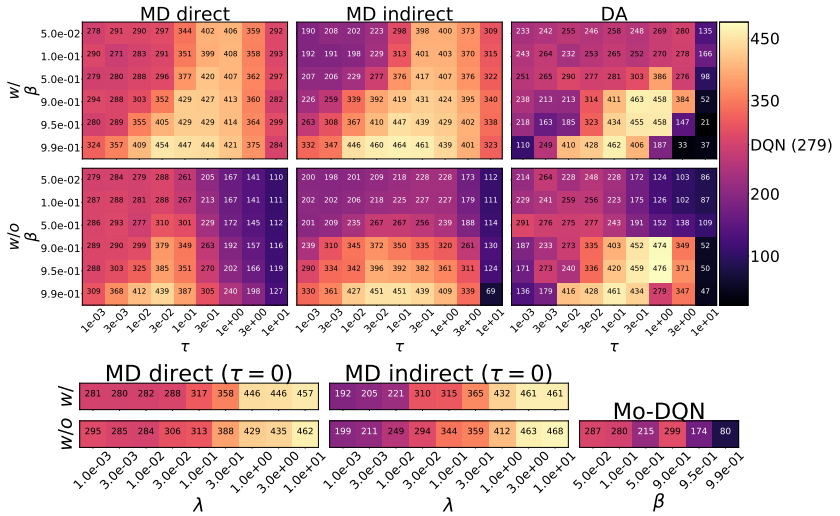

Figure 4: Cartpole with complete values.

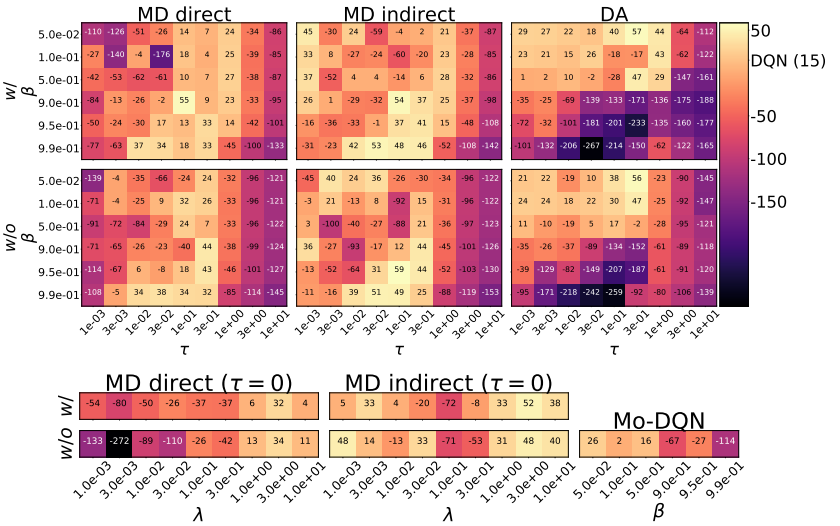

Figure 5: Lunar Lander with complete values.

5

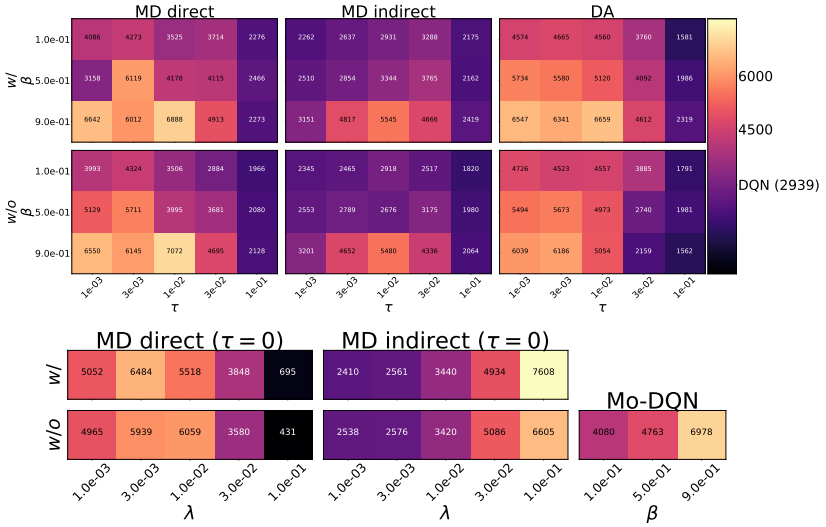

Figure 6: Asterix with complete values.

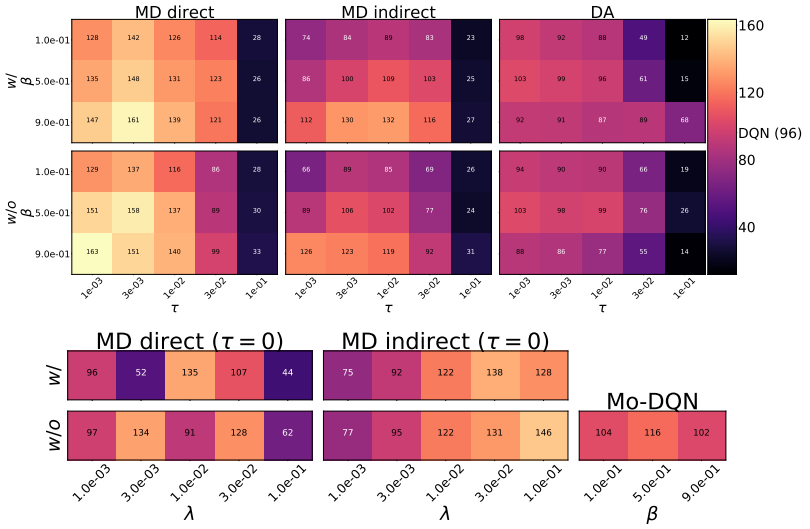

Figure 7: Breakout with complete values.

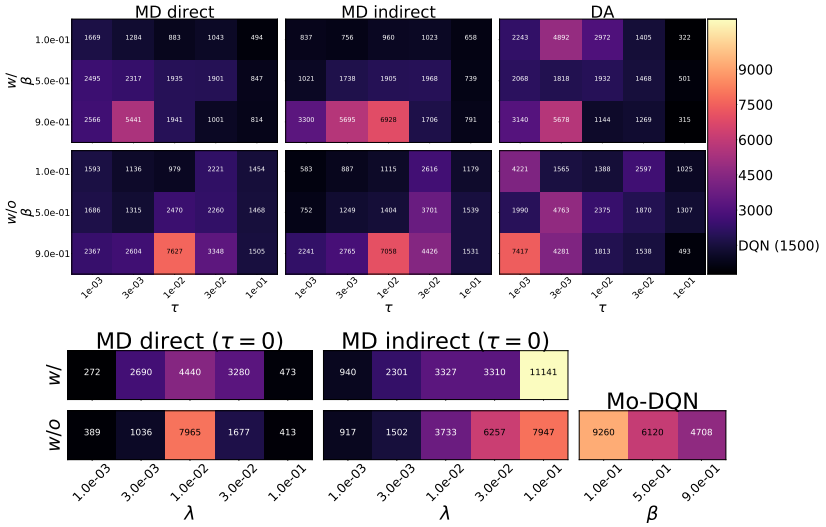

Figure 8: Seaquest with complete values.

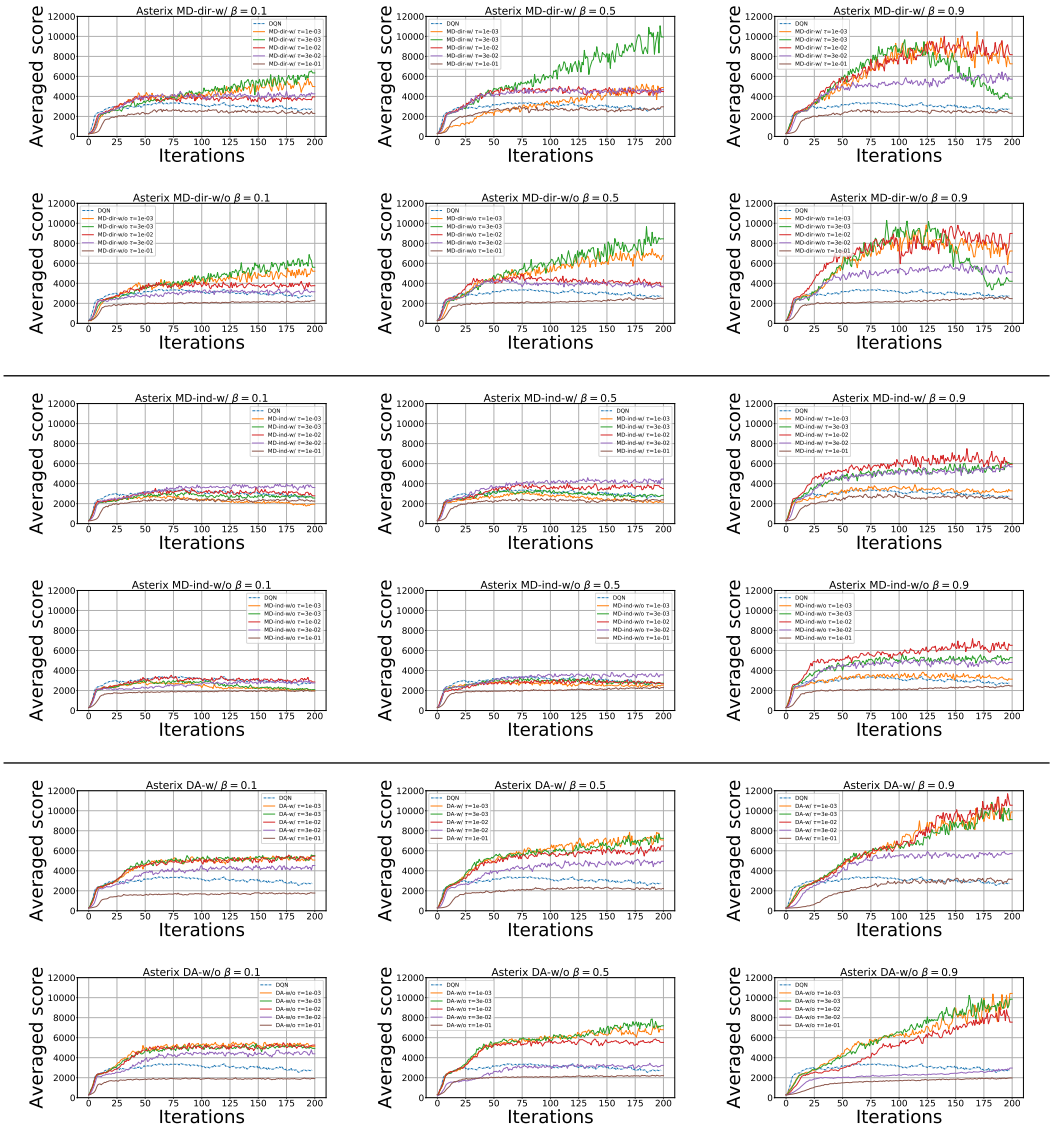

Figure 9: All averaged training scores of MD-dir (top), MD-ind (middle) and DA (bottom), *w/* and *w/o*, on Asterix, for several values of β and τ. Each plot corresponds to one value of β (in the titles). In each plot, a curve corresponds to a value of τ: $1e-3$ (orange), $3e-3$ (green), $1e-02$ (red), $3e-2$ (blue), $1e-1$ (brown). The blue dotted line is DQN.

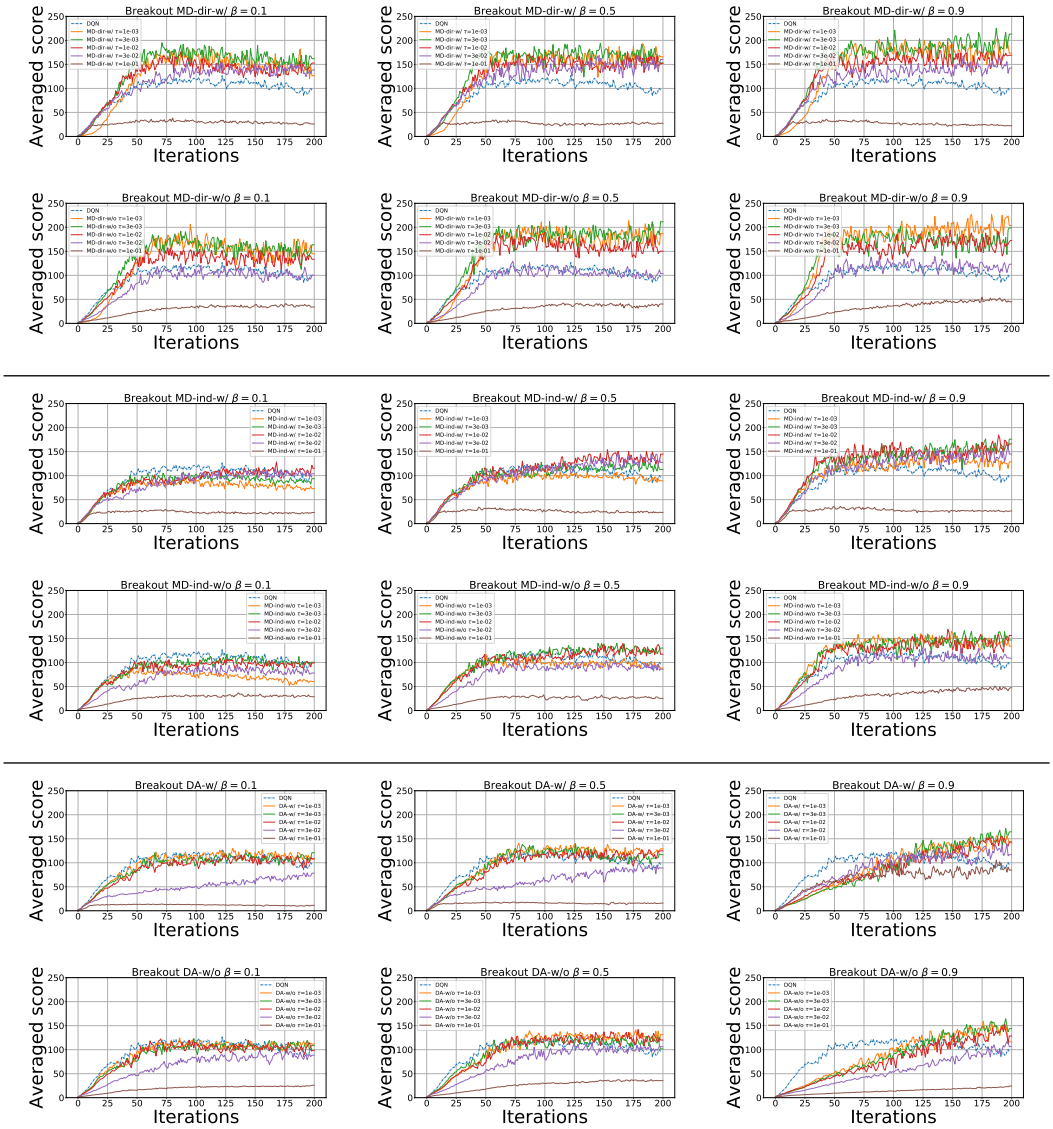

Figure 10: All averaged training scores of MD-dir (top), MD-ind (middle) and DA (bottom), *w/* and *w/o*, on Breakout, for several values of β and τ. Each plot corresponds to one value of β (in the titles). In each plot, a curve corresponds to a value of τ: $1e-3$ (orange), $3e-3$ (green), $1e-02$ (red), $3e-2$ (blue), $1e-1$ (brown). The blue dotted line is DQN.

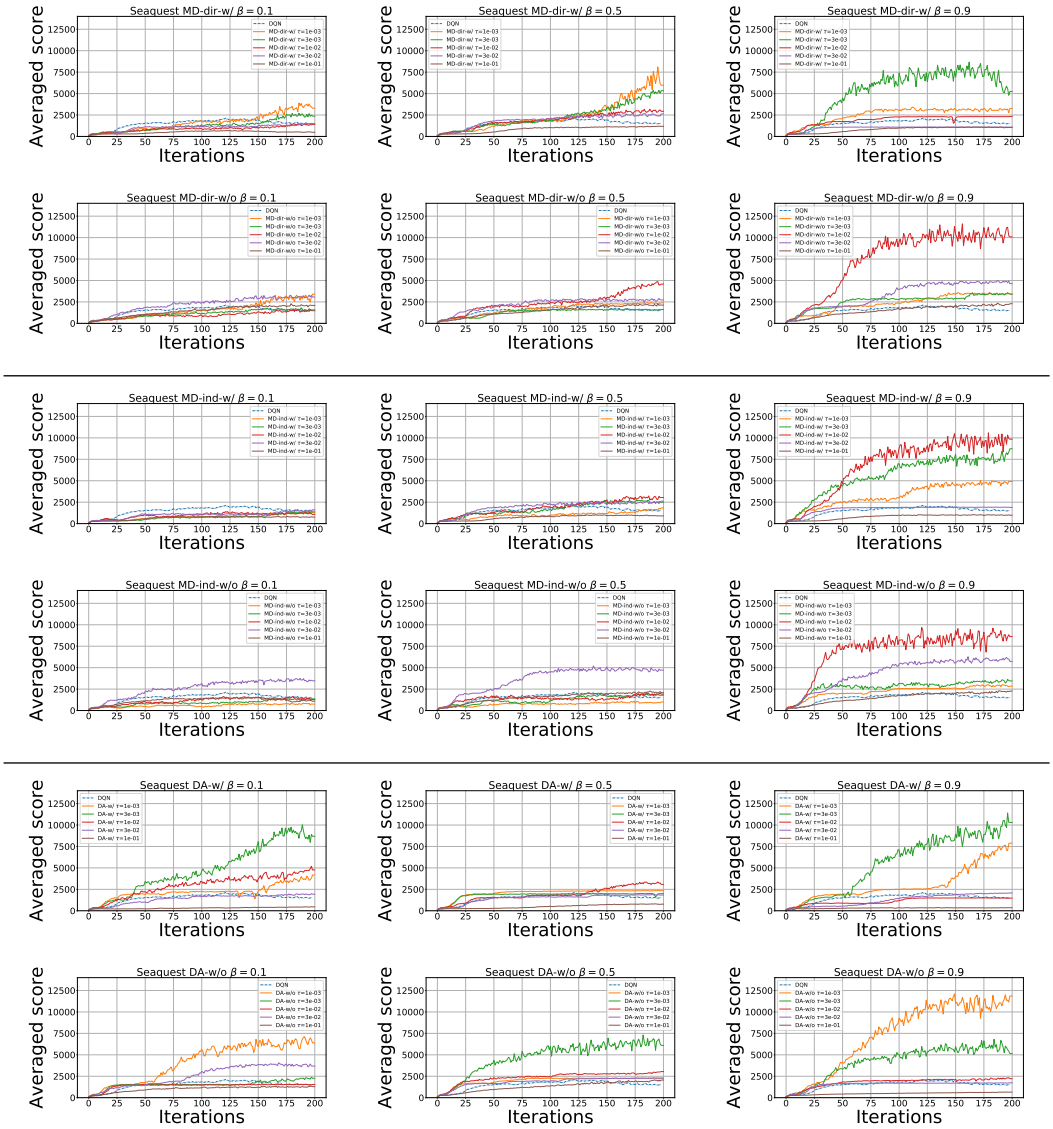

Figure 11: All averaged training scores of MD-dir (top), MD-ind (middle) and DA (bottom), *w/* and *w/o*, on Seaquest, for several values of β and τ. Each plot corresponds to one value of β (in the titles). In each plot, a curve corresponds to a value of τ: $1e-3$ (orange), $3e-3$ (green), $1e-02$ (red), $3e-2$ (blue), $1e-1$ (brown). The blue dotted line is DQN.

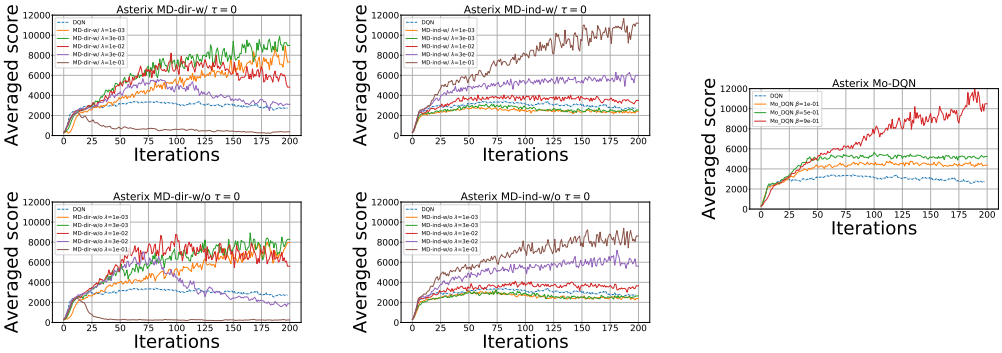

Figure 12: All averaged training scores of limit cases on Asterix, for several values of β and λ. In each plot, a curve corresponds to a value of λ for MD-ind and MD-dir, and to a value of β for Mo-DQN. The blue dotted line is DQN.

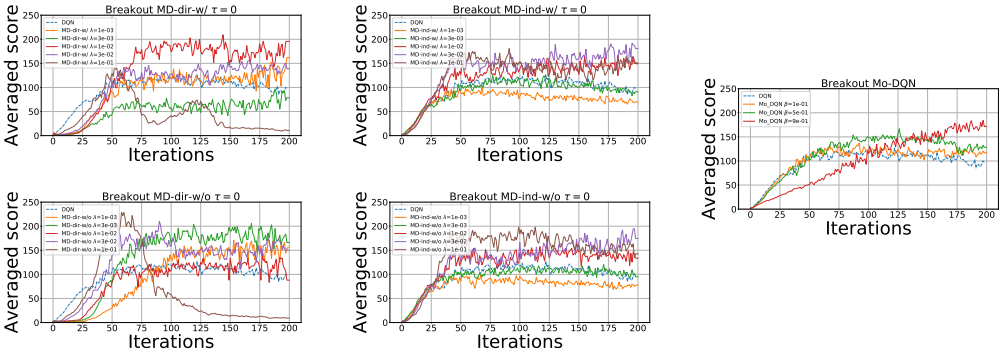

Figure 13: All averaged training scores of limit cases on Breakout, for several values of β and λ. In each plot, a curve corresponds to a value of λ for MD-ind and MD-dir, and to a value of β for Mo-DQN. The blue dotted line is DQN.

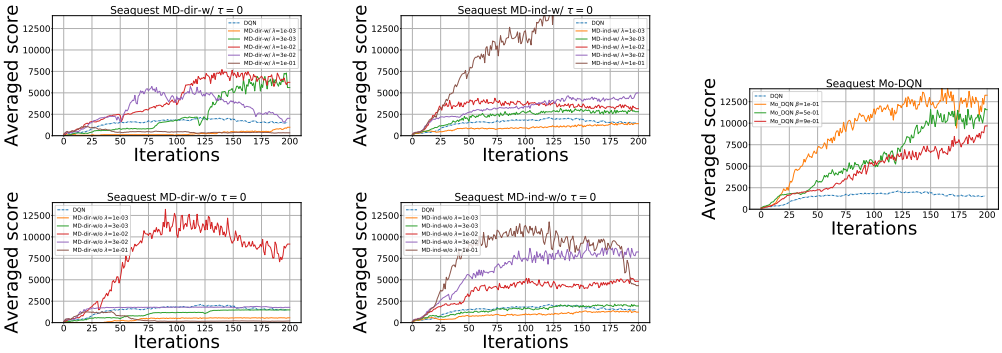

Figure 14: All averaged training scores of limit cases on Seaquest, for several values of β and λ. In each plot, a curve corresponds to a value of λ for MD-ind and MD-dir, and to a value of β for Mo-DQN. The blue dotted line is DQN.