Google Research

Regularization in Reinforcement Learning

Matthieu Geist (Google Research, Brain Team)

Overview

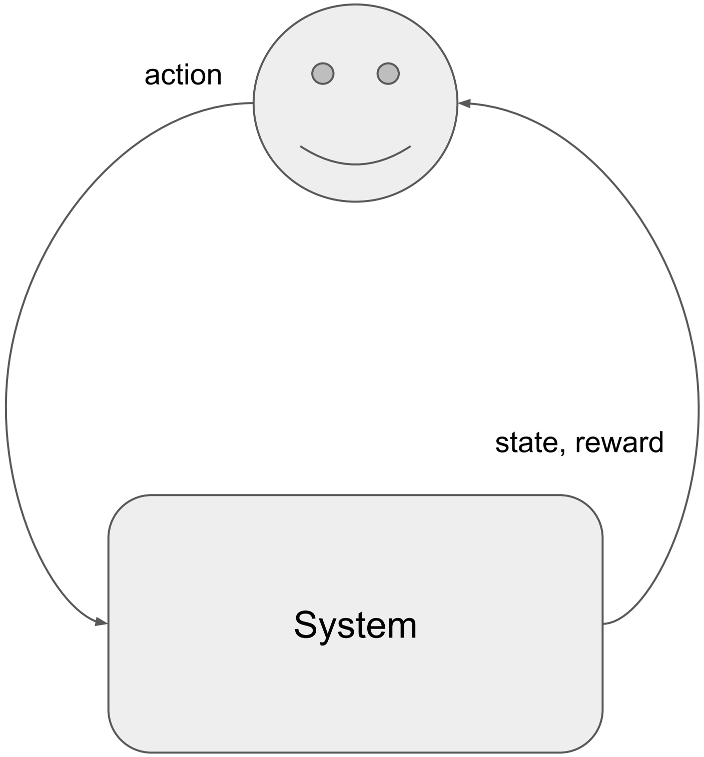
- Warm up
  - From value iteration to DQN, and back to approximate DP
- Regularized Approximate Dynamic Programming
  - A general view of regularization in RL
- Case studies
  - Entropy regularization
  - KL regularization
- The many ways to do regularization
  - A quick overview
- The issue with (KL) regularization
  - For deep RL
- A remedy
  - Munchausen RL

Warm up

From dynamic programming to (deep) RL (and back)

Reinforcement Learning

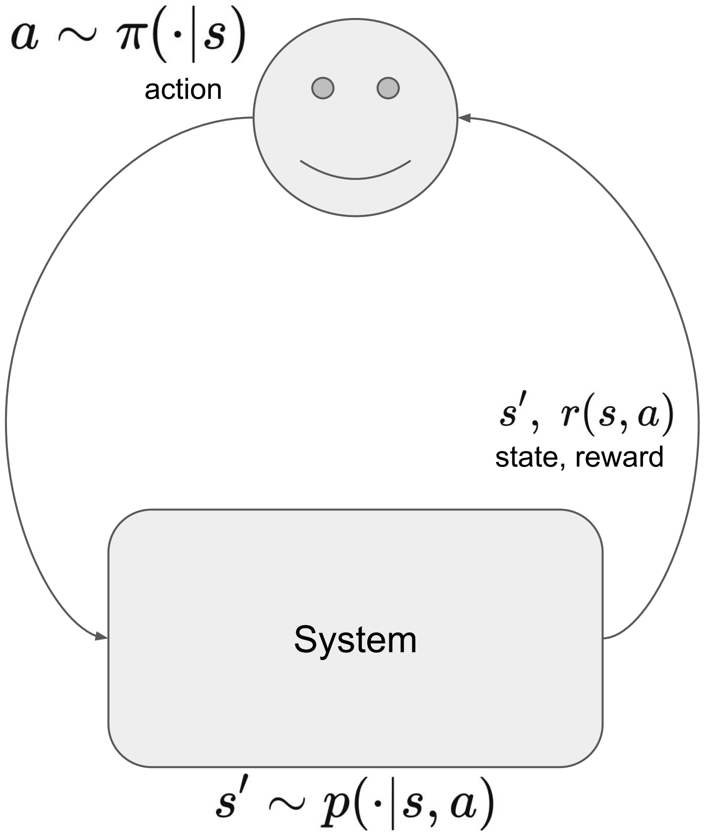
- Closed-loop control
  - the agent observes the state
  - it applies an action
  - the system's state changes
  - the agent is rewarded for the transition
- Agent's goal
  - maximize cumulative rewards
- Control learnt from data
- Formalized with Markov Decision Processes



Google Research

Markov Decision Process

- MDP:
  - $\{\mathcal{S}, \mathcal{A}, p, r, \gamma\}$
- Policy:
  - $\pi : \mathcal{S} \rightarrow \Delta_{\mathcal{A}}$
- Value function:
  - $v_\pi(s) = \mathbb{E}_\pi[\sum_{t\geq 0} \gamma^t r(S_t, A_t) | S_0 = s]$
- Optimal policy
  - $\pi_* \in \underset{\pi}{\operatorname{argmax}}\, v_\pi$
- Computing the optimal policy:
  - Dynamic Programing



q-function and Bellman operator

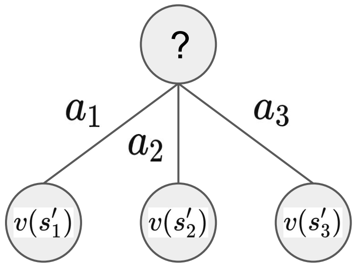
- Q-functions will be convenient: $q_\pi(s, a) = \mathbb{E}_\pi[\sum_{t\geq 0} \gamma^t r(S_t, A_t) | S_0 = s, A_0 = a]$
- Can be simplified:

$$q_\pi(s, a) = \mathbb{E}_\pi[\sum_{t\geq 0} \gamma^t r(S_t, A_t) | S_0 = s, A_0 = a]$$
$$= r(s, a) + \mathbb{E}_\pi[\sum_{t\geq 1} \gamma^t r(S_t, A_t) | S_0 = s, A_0 = a]$$
$$= r(s, a) + \gamma \mathbb{E}_{s' \sim p(\cdot|s,a)} \mathbb{E}_{a' \sim \pi(\cdot|s')} [q_\pi(s', a')]$$

- q_π is the (unique) fixed point of the Bellman operator $T_\pi : \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \to \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$

$$[T_\pi q](s, a) = r(s, a) + \gamma \mathbb{E}_{s' \sim p(\cdot|s,a)} \mathbb{E}_{a' \sim \pi(\cdot|s')} [q(s', a')]$$
$$q_\pi = T_\pi q_\pi$$

Value iteration



- Greediness

$$\pi \in \mathcal{G}(q) \Leftrightarrow \pi(a|s) = \begin{cases} 1 \text{ if } a = \operatorname{argmax} q(s, \cdot) \\ 0 \text{ else} \end{cases}$$

- Value iteration

$$\boxed{\begin{cases} \pi_{k+1} \in \mathcal{G}(q_k) \\ q_{k+1} = T_{\pi_{k+1}} q_k \end{cases}} \quad \begin{cases} \pi_{k+1}(a|s) = \mathbb{1}_{\{a = \operatorname{argmax} q_k(s,\cdot)\}}, \ \forall (s,a) \in \mathcal{S} \times \mathcal{A} \\ q_{k+1}(s,a) = r(s,a) + \gamma \mathbb{E}_{s' \sim p(\cdot|s,a)} \mathbb{E}_{a' \sim \pi_{k+1}(\cdot|s')} [q_k(s',a')], \ \forall (s,a) \in \mathcal{S} \times \mathcal{A} \end{cases}$$

- VI (more classic form)

$$q_{k+1}(s,a) = r(s,a) + \gamma \mathbb{E}_{s' \sim p(\cdot|s,a)} [\max_{a'} q_k(s',a')], \ \forall (s,a) \in \mathcal{S} \times \mathcal{A}$$

Value iteration - toward approximation

- In reinforcement learning:
  - Model unknown (transition kernel, reward)
  - Learning from data
  - State/action spaces too large for representing exactly q-functions.

This can only be approximately represented (eg, neural net)

Not possible to consider all state-action pairs

$$\underbrace{q_{k+1}(s,a)}_{} = r(s,a) + \gamma \underbrace{\mathbb{E}_{s'\sim p(\cdot|s,a)}}_{\text{We can only sample}}[\max_{a'} q_k(s',a')], \ \underbrace{\forall (s,a) \in \mathcal{S}\times\mathcal{A}}_{}$$

We can only sample

Towards DQN

$$q_{k+1}(s,a) = r(s,a) + \gamma \mathbb{E}_{s' \sim p(\cdot|s,a)}[\max_{a'} q_k(s',a')], \ \forall (s,a) \in \mathcal{S} \times \mathcal{A}$$

- Approximate q_{k+1} with a neural net q_θ , let q_k be $q_{\bar{\theta}}$, a copy of the previous network
- Assume we have access to a dataset of transitions, $\mathcal{D} = \{(s_i, a_i, r_i, s_i')_{1 \le i \le n}\}$
- Approximate the q-function by solving a regression problem:

$$\min_\theta \hat{E}_{\mathcal{D}} \left[\left(r_i + \gamma \max_{a'} q_{\bar{\theta}}(s_i', a') - q_\theta(s_i, a_i) \right)^2 \right]$$

Optimize over available samples

Sample instead of expectation

Approximated with a neural network

Towards DQN

- How to fill the dataset?
  - With interaction data

- How to interact with the system?
  - Exploration/exploitation dilemma
  - Simple solution: epsilon-greedy policy, play $\begin{cases} \text{random w.p. } \epsilon \\ \operatorname{argmax} q_\theta(s, \cdot) \text{ w.p. } 1 - \epsilon \end{cases}$

- When to update the target network?
  - Not too often, or will be unstable
  - Often enough, or will be too slow

DQN

Algorithm 1 DQN

Require: $T \in \mathbb{N}^*$ the number of environment steps, $C \in \mathbb{N}^*$ the update period, $F \in \mathbb{N}^*$ the interaction period.

Initialize θ at random
$\mathcal{B} = \{\}$
$\bar{\theta} = \theta$
for $t = 1$ **to** T **do**
  Collect a transition $b = (s_t, a_t, r_t, s_{t+1})$ from $\mathcal{G}_\epsilon(q_\theta)$
  $\mathcal{B} \leftarrow \mathcal{B} \cup \{b\}$
  if $t \mod F == 0$ **then**
    On a random batch of transitions $B_t \subset \mathcal{B}$, update θ with one step of SGD on $\hat{E}_{B_t}[(r_i + \gamma \max_{a'} q_{\bar{\theta}}(s'_i, a') - q_\theta(s_i, a_i))^2]$
  end if
  if $k \mod C == 0$ **then**
    $\bar{\theta} \leftarrow \theta$
  end if
end for
return $\mathcal{G}_0(\theta)$

[1] *V. Mnih et al. Human-level control through deep reinforcement learning. Nature, 2015.*

Theoretical analysis

- DQN is a form of approximate value iteration

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

- Propagation of errors (eg, [1])

$$\underbrace{\|q_* - q_{\pi_k}\|_\infty}_{\text{Distance to optimality}} \leq \underbrace{\frac{2\gamma}{(1-\gamma)^2}}_{\text{Horizon factor}} \underbrace{\left((1-\gamma)\sum_{j=1}^{k} \gamma^{k-j} \|\epsilon_j\|_\infty\right)}_{\text{Error term}} + \underbrace{\frac{2}{1-\gamma}\gamma^k v_{\max}}_{\text{Rate of convergence (without error)}}$$

[1] B. Scherrer et al. Approximate Modified Policy Iteration. JMLR 2015

Regularized (Approximate) Dynamic Programming

Google Research

Why regularization

- What is regularization in RL?
  - See the many next slides!
- Why regularization in RL?
  - Arises when framing RL as probabilistic inference (eg, [1])
  - Favoring exploration (high policy's entropy, eg [2])
  - Smoothing the optimization landscape [3]
  - Trust region for the policy update [4]
  - Theoretical guarantees [5]
  - Works well empirically!
- Here, focus on the viewpoint of regularized ADP [6]
  - Unifying abstraction, allows for theoretical analysis, recovers many/all agents

[1] S. Levine. RL and control as probabilistic inference: tutorial and review. arXiv, 2018
[2] T. Haarnoja et al. Soft Actor-Critic: off-policy maxent deep RL with a stochastic actor. ICML 2018
[3] Z. Ahmed et al. Understanding the impact of entropy on policy optimization. ICML 2019
[4] J. Schulman et al. Trust region policy optimization. ICML 2015
[5] N. Vieillard et al. Leverage the average: an analysis of KL regularization in RL. NeurIPS, 2020
[6] M. Geist et al. A theory of regularized MDPs. ICML 2019

Some notations

Now stochastic policies, $\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}$

$$\langle \pi, q \rangle = \left(\sum_{a \in \mathcal{A}} \pi(a|s) q(s,a) \right)_{s \in \mathcal{S}}$$

$$Pv = \left(\sum_{s'} P(s'|s,a) v(s') \right)_{(s,a) \in \mathcal{S} \times \mathcal{A}}$$

Google Research

Some notations

Now stochastic policies, $\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}$

$$\langle \pi, q \rangle = \left(\sum_{a \in \mathcal{A}} \pi(a|s) q(s,a) \right)_{s \in \mathcal{S}}$$

$$Pv = \left(\sum_{s'} P(s'|s,a) v(s') \right)_{(s,a) \in \mathcal{S} \times \mathcal{A}}$$

Bellman operator:

$$T_\pi q = r + \gamma P \langle \pi, q \rangle$$

Google Research

Some notations

Now stochastic policies, $\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}$

$$\langle \pi, q \rangle = \left(\sum_{a \in \mathcal{A}} \pi(a|s) q(s,a) \right)_{s \in \mathcal{S}}$$

$$Pv = \left(\sum_{s'} P(s'|s,a) v(s') \right)_{(s,a) \in \mathcal{S} \times \mathcal{A}}$$

Bellman operator:

$$T_\pi q = r + \gamma P \langle \pi, q \rangle$$

greedy policy:

$$\underset{a \in \mathcal{A}}{\operatorname{argmax}}\, q(\cdot, a) = \underset{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}}{\operatorname{argmax}} \langle \pi, q \rangle$$

Google Research

Some notations

Now stochastic policies, $\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}$

$$\langle \pi, q \rangle = \left(\sum_{a \in \mathcal{A}} \pi(a|s) q(s,a) \right)_{s \in \mathcal{S}}$$

$$Pv = \left(\sum_{s'} P(s'|s,a) v(s') \right)_{(s,a) \in \mathcal{S} \times \mathcal{A}}$$

Bellman operator:

$$T_\pi q = r + \gamma P \langle \pi, q \rangle$$

greedy policy:

$$\underset{a \in \mathcal{A}}{\operatorname{argmax}}\, q(\cdot, a) = \underset{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}}{\operatorname{argmax}} \langle \pi, q \rangle$$

Entropy and KL divergence:

$$\mathcal{H}(\pi) = -\langle \pi, \ln \pi \rangle$$

$$\mathrm{KL}(\pi_1 || \pi_2) = \langle \pi_1, \ln \pi_1 - \ln \pi_2 \rangle$$

Google Research

Some notations

Now stochastic policies, $\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}$

$$\langle \pi, q \rangle = \left(\sum_{a \in \mathcal{A}} \pi(a|s) q(s,a) \right)_{s \in \mathcal{S}}$$

$$Pv = \left(\sum_{s'} P(s'|s,a) v(s') \right)_{(s,a) \in \mathcal{S} \times \mathcal{A}}$$

AVI

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \langle \pi, q_k \rangle \\ q_{k+1} = r + \gamma P \langle \pi_{k+1}, q_k \rangle + \epsilon_{k+1} \end{cases}$$

Bellman operator:

$$T_\pi q = r + \gamma P \langle \pi, q \rangle$$

greedy policy:

$$\operatorname*{argmax}_{a \in \mathcal{A}} q(\cdot, a) = \operatorname*{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \langle \pi, q \rangle$$

Entropy and KL divergence:

$$\mathcal{H}(\pi) = -\langle \pi, \ln \pi \rangle$$

$$\mathrm{KL}(\pi_1 || \pi_2) = \langle \pi_1, \ln \pi_1 - \ln \pi_2 \rangle$$

Google Research

Regularizing the greedy step

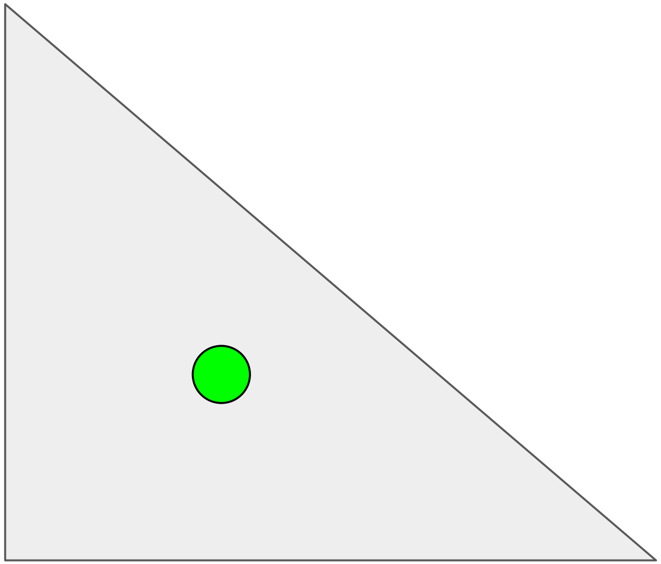
$$\begin{cases} \boxed{\pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{S}} \langle \pi, q_k \rangle} \\ q_{k+1} = r + \gamma P \langle \pi_{k+1}, q_k \rangle + \epsilon_{k+1} \end{cases}$$

No regularization

$$\pi_{k+1} = \underset{\pi \in \Delta_{\mathcal{A}}^{S}}{\operatorname{argmax}} \left(\langle \pi, q_k \rangle \qquad\qquad\qquad \right)$$

Classic greedy policy

Ok if there is no error in the q-values

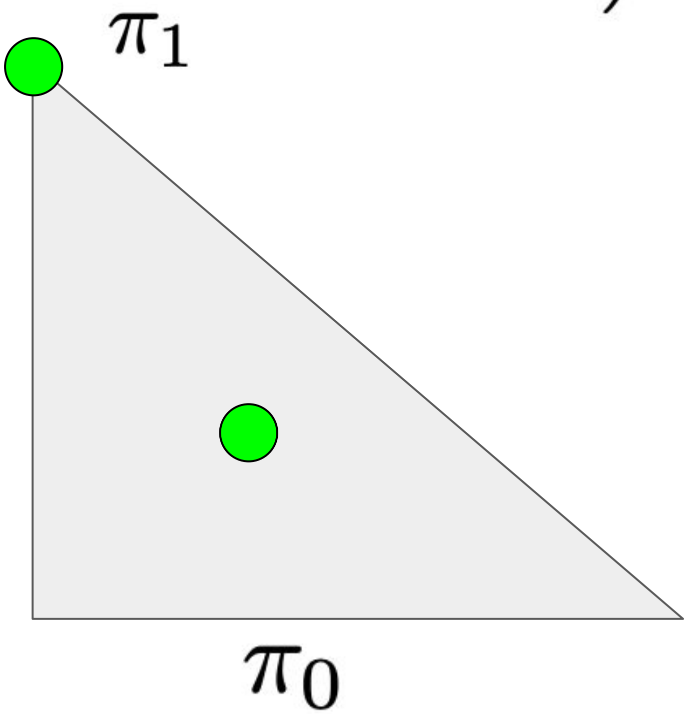


π_0

Google Research

No regularization

$$\pi_{k+1} = \underset{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}}{\operatorname{argmax}} \left(\langle \pi, q_k \rangle \qquad\qquad\qquad\qquad \right)$$



Classic greedy policy

Ok if there is no error in the q-values

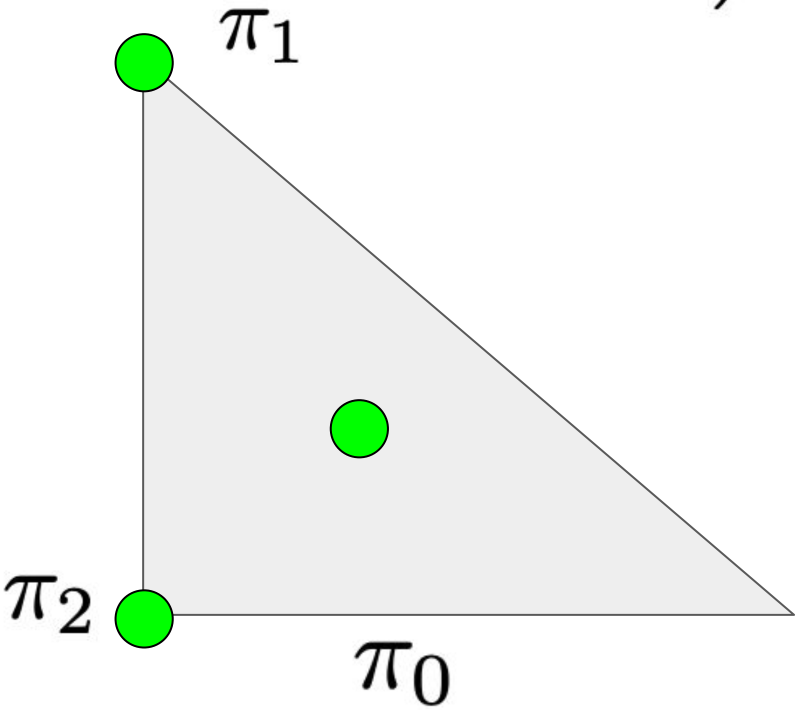
Google Research

No regularization

$$\pi_{k+1} = \underset{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}}{\operatorname{argmax}} \left(\langle \pi, q_k \rangle \qquad\qquad \right)$$

Classic greedy policy

Ok if there is no error in the q-values

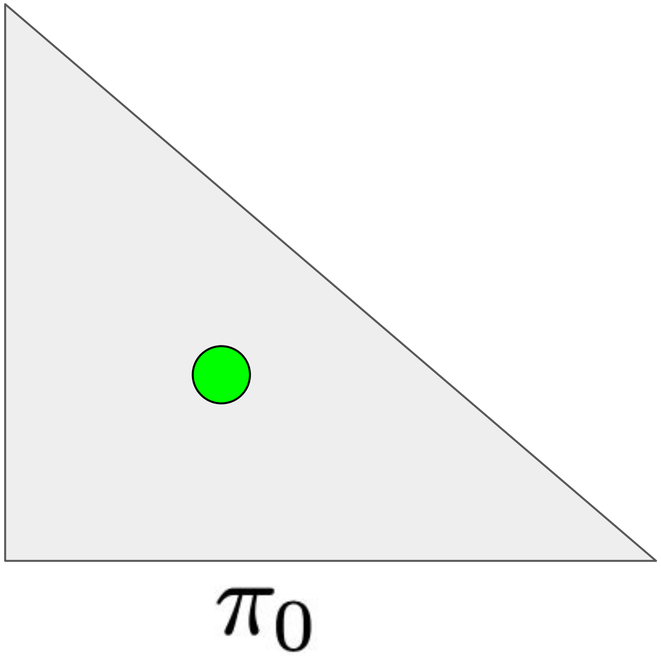


Google Research

Regularization with entropy

$$\pi_{k+1} = \operatorname*{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \left(\langle \pi, q_k \rangle + \tau \mathcal{H}(\pi) \right)$$

Penalized for going too far from the uniform policy

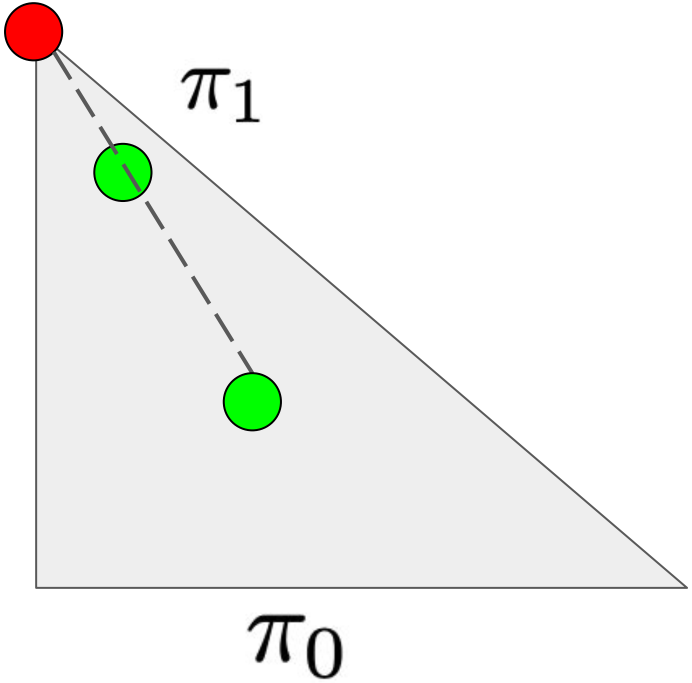


Google Research

Regularization with entropy

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

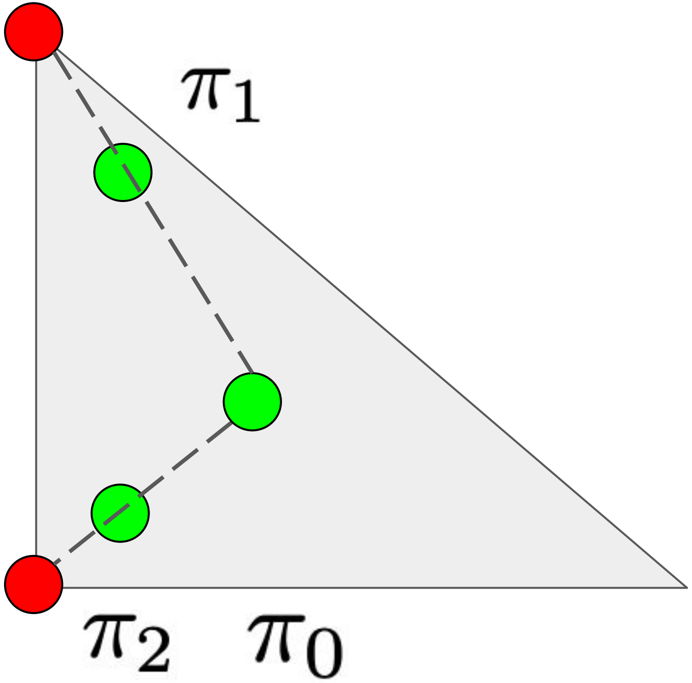
Penalized for going too far from the uniform policy



Regularization with entropy

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

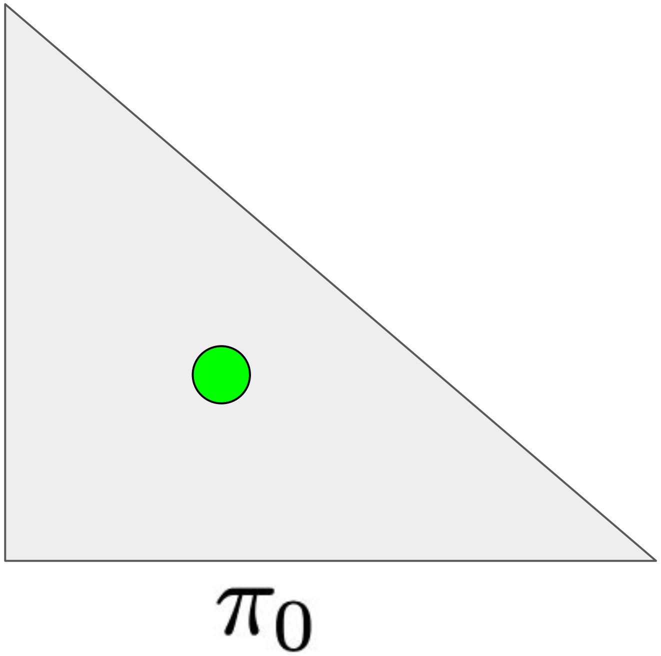
Penalized for going too far from the uniform policy



Regularization with Kullback-Leibler

$$\pi_{k+1} = \operatorname*{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \Big(\langle \pi, q_k \rangle - \lambda \, \mathrm{KL}(\pi || \pi_k) \qquad \Big)$$

Penalized for going too far from the previous policy

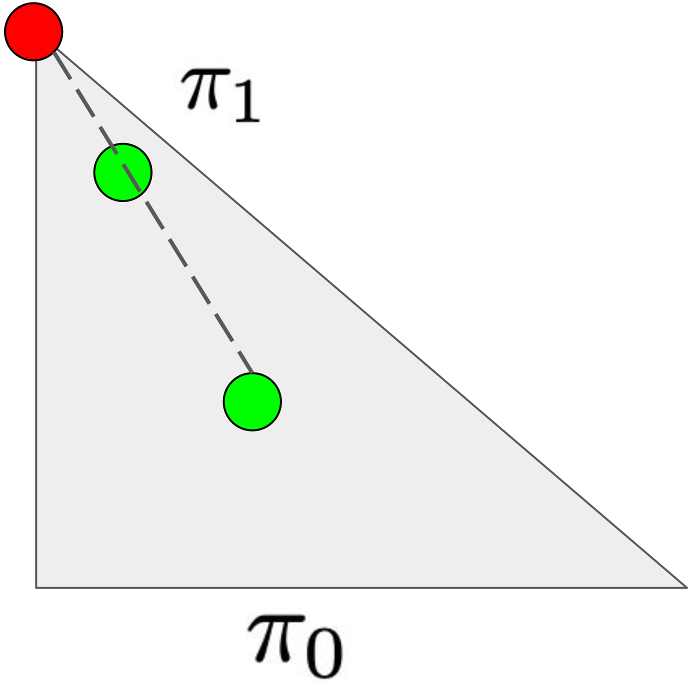


Google Research

Regularization with Kullback-Leibler

$$\pi_{k+1} = \underset{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}}{\operatorname{argmax}} \left(\langle \pi, q_k \rangle - \lambda \, \mathrm{KL}(\pi || \pi_k) \qquad \right)$$

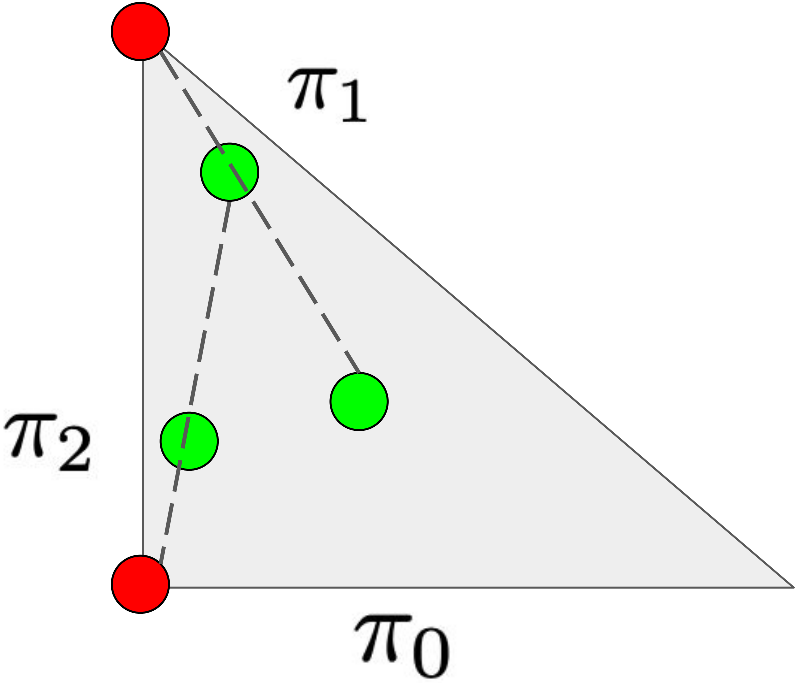
Penalized for going too far from the previous policy



Regularization with Kullback-Leibler

$$\pi_{k+1} = \underset{\pi \in \Delta_{\mathcal{A}}^{S}}{\operatorname{argmax}} \Big(\langle \pi, q_k \rangle - \lambda \, \mathrm{KL}(\pi || \pi_k) \Big)$$

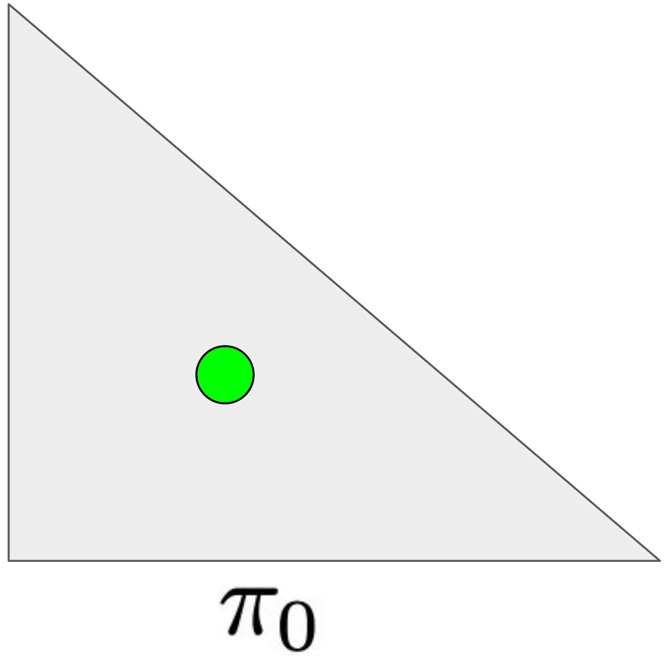
Penalized for going too far from the previous policy



Regularization with both

$$\pi_{k+1} = \underset{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}}{\mathrm{argmax}} \Big(\langle \pi, q_k \rangle - \lambda \mathrm{KL}(\pi || \pi_k) + \tau \mathcal{H}(\pi) \Big)$$

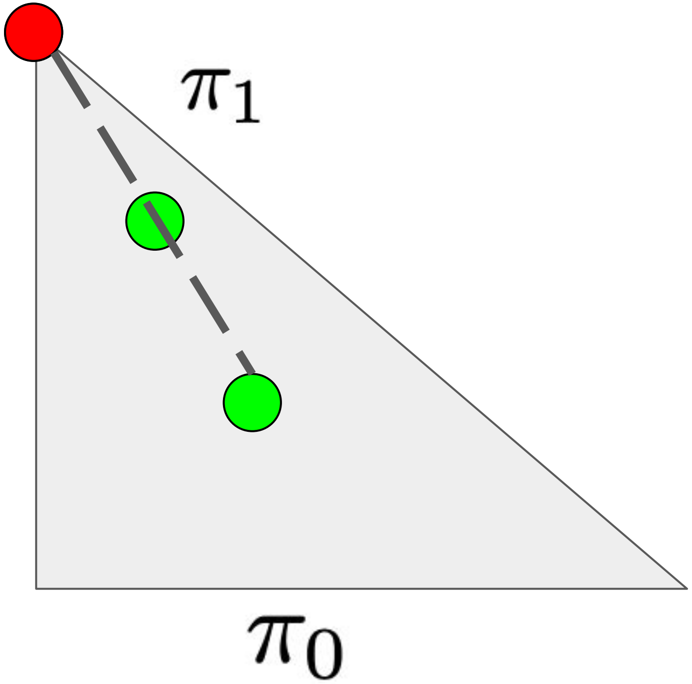
Penalized for going too far from the previous policy and for going too far from the uniform policy



Regularization with both

$$\pi_{k+1} = \underset{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}}{\operatorname{argmax}} \Big(\langle \pi, q_k \rangle - \lambda \, \mathrm{KL}(\pi || \pi_k) + \tau \mathcal{H}(\pi) \Big)$$

Penalized for going too far from the previous policy and for going too far from the uniform policy

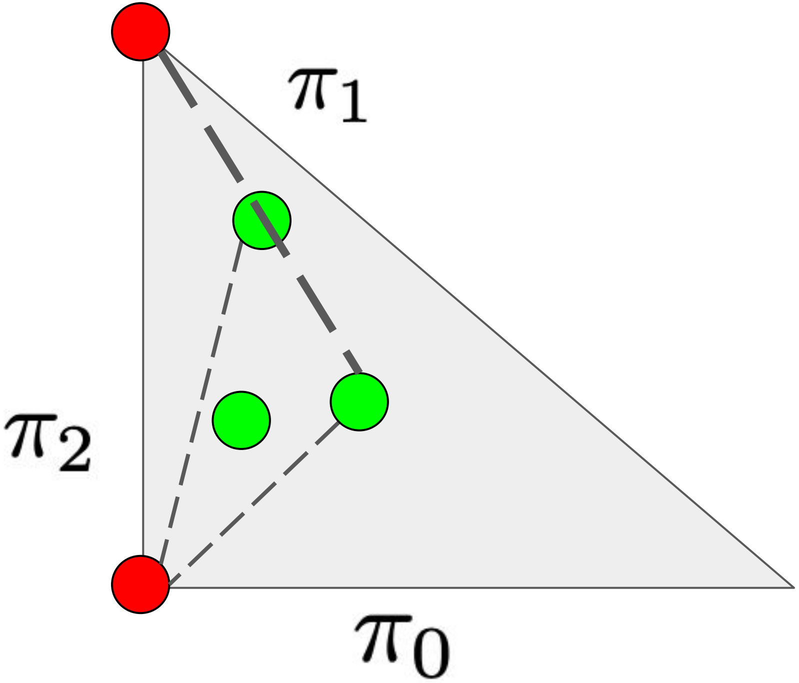


Google Research

Regularization with both

$$\pi_{k+1} = \underset{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}}{\operatorname{argmax}} \Big(\langle \pi, q_k \rangle - \lambda \, \mathrm{KL}(\pi || \pi_k) + \tau \mathcal{H}(\pi) \Big)$$

Penalized for going too far from the previous policy and for going too far from the uniform policy



Google Research

Regularization with q-values

$$\pi_{k+1} = \operatorname*{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \left\langle \pi, \sum_{j=0}^{k} q_j \right\rangle$$

Greedy (so policy in a corner of the simplex), but w.r.t. the sum of all q-values

Google Research

Regularization with q-values

$$\pi_{k+1} = \underset{\pi \in \Delta_{\mathcal{A}}^{S}}{\operatorname{argmax}} \left\langle \pi, \sum_{j=0}^{k} q_j \right\rangle$$

Greedy (so policy in a corner of the simplex), but w.r.t. the sum of all q-values

Rational: assume $q_k = q_* + \epsilon_k$ with the errors being i.i.d., classical greediness would not converge, while this regularized greediness would provide asymptotically the optimal policy

Regularizing the evaluation step?

$$\begin{cases} \pi_{k+1} = \text{argmax}_{\pi \in \Delta_A^S} \langle \pi, q_k \rangle \\ \boxed{q_{k+1} = r + \gamma P \langle \pi_{k+1}, q_k \rangle + \epsilon_{k+1}} \end{cases}$$

Naive approach

- For a general greedy step

$$\pi_{k+1} = \operatorname*{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \Big(\langle \pi, q_k \rangle - \Omega(\pi || \pi_k) \Big)$$

- Just consider the usual evaluation step

$$q_{k+1} = r + \gamma P \langle \pi_{k+1}, q_k \rangle + \epsilon_{k+1}$$

- This is the usual approach in the litterature (called **<u>type 2</u>** in my own nomenclature)

Google Research

A principled approach

- For a general greedy step

$$\pi_{k+1} = \operatorname*{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \Big(\langle \pi, q_k \rangle - \Omega(\pi || \pi_k) \Big)$$

- Regularize the same way the evaluation step

$$q_{k+1} = r + \gamma P \Big(\langle \pi_{k+1}, q_k \rangle - \Omega(\pi_{k+1} || \pi_k) \Big) + \epsilon_{k+1}$$

- This is much less usual in the litterature
  (called **type 1** in my own nomenclature)

Summary

- Mirror-Descent VI, type 1 [1]

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{S}} \left(\langle \pi, q_k \rangle - \Omega(\pi || \pi_k) \right) \\ q_{k+1} = r + \gamma P\Big(\langle \pi_{k+1}, q_k \rangle - \Omega(\pi_{k+1} || \pi_k) \Big) + \epsilon_{k+1} \end{cases}$$

- Mirror-Descent VI, type 2 [1]

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{S}} \left(\langle \pi, q_k \rangle - \Omega(\pi || \pi_k) \right) \\ q_{k+1} = r + \gamma P \langle \pi_{k+1}, q_k \rangle + \epsilon_{k+1} \end{cases}$$

[1] M. Geist et al. A theory of regularized MDPs. ICML 2019.

Case study

Entropy regularization

Google Research

Objective

- Regularized DP Scheme

$$\begin{cases} \pi_{k+1} = \text{argmax}(\langle \pi, q_k \rangle + \tau\mathcal{H}(\pi)) \\ q_{k+1} = r + \gamma P(\langle \pi_{k+1}, q_k \rangle + \tau\mathcal{H}(\pi)) + \epsilon_{k+1} \end{cases}$$

- What practical algorithms can be derived from this?
    - Same approach as AVI->DQN
    - Will consider also continuous actions

- What theoretical guarantees?

A look at the (regularized) greedy step

- Greedy step, $\pi_{k+1} = \operatorname{argmax}(\langle \pi, q_k \rangle + \tau \mathcal{H}(\pi))$
- The negative entropy $-\mathcal{H}(\pi)$ is convex, unique solution
- This is indeed a Legendre-Fenchel transform (convex conjugate)

$$\Omega^*(q) = \max_\pi \langle \pi, q \rangle - \Omega(\pi)$$
$$\nabla \Omega^*(q) = \operatorname*{argmax}_\pi \langle \pi, q \rangle - \Omega(\pi)$$

- With the negative entropy, the convex conjugate is the log-sum-exp and the maximizer is the softmax

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

Soft-DQN

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}(\langle \pi, q_k \rangle + \tau \mathcal{H}(\pi)) \\ q_{k+1} = r + \gamma P(\langle \pi_{k+1}, q_k \rangle + \tau \mathcal{H}(\pi)) + \epsilon_{k+1} \end{cases}$$

$$\pi_{k+1} = \operatorname{softmax}(\frac{q_k}{\tau})$$

- Same approach as for DQN

$$\hat{E}_{B_t}\left[\left(r_i + \sum_{a'} \pi_{k+1}(a'|s')\big(q_{\bar{\theta}}(s'_i, a') - \tau \ln \pi_{k+1}(a'|s')\big) - q_\theta(s_i, a_i)\right)^2\right] \text{ with } \pi_{k+1} = \operatorname{softmax}(\frac{q_{\bar{\theta}}}{\tau})$$

- We get DQN back as $\tau \rightarrow 0$

Soft-DQN (bis)

- By Legendre-Fenchel, we have

$$\tau \ln\langle 1, \exp \frac{q_k}{\tau}\rangle = \max_\pi \langle \pi, q_k\rangle + \tau\mathcal{H}(\pi)$$

- Same approach as DQN (equivalent to before)

$$\hat{E}_{B_t}\left[\left(r_i + \gamma\tau \ln\left(\sum_{a'} \exp \frac{q_{\bar\theta}}{\tau}\right) - q_\theta(s_i, a_i)^2\right)^2\right]$$

- Again, we retrieve DQN as $\tau \to 0$

Google Research

With continuous actions?

$$\pi_{k+1} = \text{argmax}(\langle \pi, q_k \rangle + \tau \mathcal{H}(\pi))$$

- The policy can no longer be computed (softmax over continuous actions)
  - Learn it! (add an actor --the policy-- to the critic --the q-function--)
- First solution, direct approach

$$J(w) = \hat{\mathbb{E}}_{s_i}[\mathbb{E}_{a \sim \pi_w(\cdot|s_i)}[q_{\bar{\theta}}(s_i, a) - \tau \ln \pi_w(a|s_i)]]$$

$$= \hat{\mathbb{E}}_{s_i}[\mathbb{E}_{a \sim \pi_{\bar{w}}(\cdot|s_i)}[\frac{\pi_w(a|s_i)}{\pi_{\bar{w}}(a|s_i)}(q_{\bar{\theta}}(s_i, a) - \tau \ln \pi_w(a|s_i))]] \approx \hat{\mathbb{E}}_{s_i} \frac{1}{N} \sum_{j=1}^{N} \frac{\pi_w(a_{ij}|s_i)}{\pi_{\bar{w}}(a_{ij}|s_i)}(q_{\bar{\theta}}(s_{ij}, a) - \tau \ln \pi_w(a_{ij}|s_i))$$

- (alternative to importance sampling, reparameterization trick)
- Second solution, indirect approach (equivalent). We know analytically $\pi_{k+1} = \text{softmax}(\frac{q_k}{\tau})$

$$J(w) = \hat{\mathbb{E}}_{s_i}[\text{KL}(\pi_w(\cdot|s_i)||\frac{\exp \frac{q_{\bar{\theta}}(s_i, \cdot)}{\tau}}{Z_{\bar{\theta}}(s_i)})]$$

$$= \hat{\mathbb{E}}_{s_i}[\mathbb{E}_{a \sim \pi_w(\cdot|s_i)}[\ln \pi_w(a|s_i) - \frac{1}{\tau} q_{\bar{\theta}}(s_i, a)]] + \text{cst}$$

- Evaluation: $\hat{E}_{B_t}\left[\left(r_i + \mathbb{E}_{a' \sim \pi_w(\cdot|s_i')}(q_{\bar{\theta}}(s_i', a') - \tau \ln \pi_w(a'|s')) - q_\theta(s_i, a_i)\right)^2\right]$

Google Research

Theoretical analysis (exact greedy step)

- Regularized DP scheme

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}(\langle \pi, q_k \rangle + \tau \mathcal{H}(\pi)) \\ q_{k+1} = r + \gamma P(\langle \pi_{k+1}, q_k \rangle + \tau \mathcal{H}(\pi)) + \epsilon_{k+1} \end{cases}$$

- Propagation of errors [1]

$$\underbrace{\|q_*^\tau - q_{\pi_k}^\tau\|_\infty}_{\text{Biased solution } (q_*^\tau \neq q_*)} \leq \underbrace{\frac{2\gamma}{(1-\gamma)^2}\left((1-\gamma)\sum_{j=1}^{k} \gamma^{k-j}\|\epsilon_j\|_\infty\right) + \frac{2}{1-\gamma}\gamma^k v_{\max}}_{\text{Same bound as DQN}}$$

- No advantage regarding propagation of errors, but other arguments:
  - Exploration (eg [2]), optimization landscape [3], smoothness (eg [4])...

[1] M. Geist et al. A theory of regularized MDPs. ICML 2019

[2] T. Haarnoja et al. Soft Actor-Critic: off-policy maxent deep RL with a stochastic actor. ICML 2018

[3] Z. Ahmed et al. Understanding the impact of entropy on policy optimization. ICML 2019

[4] L. Shani et al. Adaptive Trust Region Policy Optimization: Global Convergence and Faster Rates for Regularized MDPs. AAAI 2020

Case study

KL regularization

Google Research

Objective

- Regularized DP Scheme

$$\begin{cases}\pi_{k+1} = \operatorname{argmax}(\langle \pi, q_k \rangle - \lambda \,\mathrm{KL}(\pi || \pi_k)) \\ q_{k+1} = r + \gamma P(\langle \pi_{k+1}, q_k \rangle - \lambda \,\mathrm{KL}(\pi_{k+1} || \pi_k)) + \epsilon_{k+1}\end{cases}$$

- What practical algorithms can be derived from this?
  - Same approach as AVI->DQN
  - Will consider also continuous actions

- What theoretical guarantees?

A look at the (regularized) greedy step

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} (\langle \pi, q_k \rangle - \lambda \mathrm{KL}(\pi || \pi_k)) \\ q_{k+1} = r + \gamma P(\langle \pi_{k+1}, q_k \rangle - \lambda \mathrm{KL}(\pi_{k+1} || \pi_k)) + \epsilon_{k+1} \end{cases}$$

- The greedy step is a Legendre-Fenchel transform: $\pi_{k+1} \propto \pi_k \exp \frac{q_k}{\lambda}$
- With a direct induction argument:

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

- Equivalent AVI scheme, Dual Averaging viewpoint

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} (\langle \pi, h_k \rangle + \frac{\lambda}{k+1} \mathcal{H}(\pi)) \\ q_{k+1} = r + \gamma P(\langle \pi_{k+1}, q_k \rangle - \lambda \mathrm{KL}(\pi_{k+1} || \pi_k)) + \epsilon_{k+1} \\ h_{k+1} = \frac{k+1}{k+2} h_k + \frac{1}{k+2} q_{k+1} \end{cases}$$

Practical algorithm(s)

$$\pi_{k+1} = \text{argmax}(\langle \pi, q_k \rangle - \lambda \, \text{KL}(\pi || \pi_k))$$

- Now, even with discrete actions, the policy should be learnt
- Direct/indirect approach provide the same loss, works for continuous actions

$$J(w) = \hat{\mathbb{E}}_{s_i}[\mathbb{E}_{a \sim \pi_w(\cdot|s_i)}[q_{\bar{\theta}}(s_i, a) - \lambda(\ln \pi_w(a|s_i) - \ln \pi_{\bar{w}}(a|s_i))]]$$

- One could also approximate the mean of q-values by a moving average

$$J(w) = \hat{\mathbb{E}}_{s_i, a_i}\left[((1-\alpha) h_{\bar{w}}(s_i, a_i) + \alpha q_\theta(s_i, a_i) - h_w(s_i, a_i))^2\right]$$

$$\pi_{\bar{w}} \propto \exp h_{\bar{w}}$$

- Evaluation step

$$\hat{E}_{B_t}\left[\left(r_i + \mathbb{E}_{a' \sim \pi_w(\cdot|s_i')}(q_{\bar{\theta}}(s_i', a') - \tau(\ln \pi_w(a'|s') - \ln \pi_{\bar{w}}(a'|s'))) - q_\theta(s_i, a_i)\right)^2\right]$$

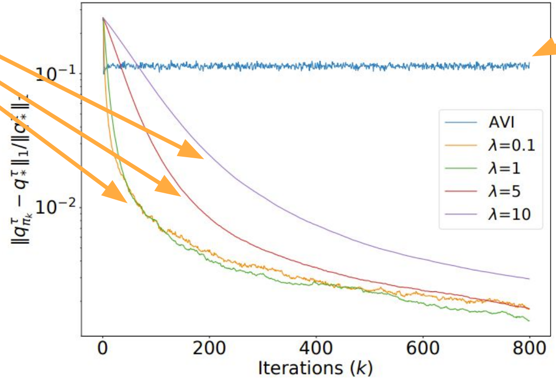
Theoretical analysis (exact greedy step)

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}(\langle \pi, q_k \rangle - \lambda \, \mathrm{KL}(\pi || \pi_k)) \\ q_{k+1} = r + \gamma P(\langle \pi_{k+1}, q_k \rangle - \lambda \, \mathrm{KL}(\pi_{k+1} || \pi_k)) + \epsilon_{k+1} \end{cases}$$

KL-regularized AVI vs AVI

$$\|q_* - q_{\pi_k}\|_\infty \leq \frac{2}{1-\gamma} \left\| \frac{1}{k} \sum_{j=1}^{k} \epsilon_j \right\|_\infty + \frac{4}{(1-\gamma)^2} \frac{r_{\max} + \tau \ln |\mathcal{A}|}{k}$$

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



The many (?) ways to do regularization

A quick overview

Google Research

Encompassed algorithms

With either the (equivalent) Mirror Descent or Dual Averaging viewpoints

	Only entropy	Only KL	Both
Reg. evaluation	Soft Q-learning [1,2], SAC [3], Mellowmax [4]	DPP [6], SQL [7]	CVI [12], AL [13,14], Munchausen-RL [15]
Unreg. evaluation	softmax DQN [5]	TRPO [8], MPO [9], Politex [10], MoVI [11]	Softened LSPI [16], MoDQN [11]

[1] Fox, R., Pakman, A., and Tishby, N. Taming the noise in reinforcement learning via soft updates. In UAI, 2016.
[2] Haarnoja, T., Tang, H., Abbeel, P., and Levine, S. Reinforcement learning with deep energy-based policies. In ICML, 2017.
[3] Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic. In ICML, 2018.
[4] Asadi, K. and Littman, M. L. An alternative softmax operator for reinforcement learning. In ICML, 2017.
[5] Song, Z., Parr, R., and Carin, L. Revisiting the softmax bellman operator: New benefits and new perspective. In ICML, 2019.
[6] Azar, M. G., Gómez, V., and Kappen, H. J. Dynamic policy programming. JMLR, 2012.
[7] Azar, M. G., Munos, R., Ghavamzadeh, M., and Kappen, H. J. Speedy q-learning. In NeurIPS, 2011.
[8] Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. Trust region policy optimization. In ICML, 2015.
[9] Abdolmaleki, A., Springenberg, J. T., Tassa, Y., Munos, R., Heess, N., and Riedmiller, M. Maximum a posteriori policy optimisation. In ICLR, 2018.
[10] Abbasi-Yadkori, Y., Bartlett, P., Bhatia, K., Lazic, N., Szepesvári, C., and Weisz, G. Politex: Regret bounds for policy iteration using expert prediction. In ICML, 2019.
*[11] Vieillard, N., Scherrer, B., Pietquin, O., and **Geist, M.** Momentum in reinforcement learning. In AISTATS, 2020.*
[12] Kozuno, T., Uchibe, E., and Doya, K. Theoretical analysis of efficiency and robustness of softmax and gap-increasing operators in RL. In AISTATS, 2019.
[13] Baird III, L. C. Reinforcement Learning Through Gradient Descent. PhD thesis, US Air Force Academy, US, 1999.
[14] Bellemare, M. G., Ostrovski, G., Guez, A., Thomas, P. S., and Munos, R. Increasing the action gap: New operators for reinforcement learning. In AAAI, 2016.
*[15] Vieillard, N., Pietquin, O., and **Geist, M.** Munchausen Reinforcement Learning. In NeurIPS, 2020.*
*[16] Pérolat, J., Piot, B., **Geist, M.**, Scherrer, B., and Pietquin, O. Softened approximate policy iteration for markov games. In ICML, 2016.*

Google Research

Entropy, type 2

- DP scheme

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{S}} \left(\langle \pi, q_k \rangle + \tau \mathcal{H}(\pi) \right) \\ q_{k+1} = r + \gamma P \langle \pi_{k+1}, q_k \rangle + \epsilon_{k+1} \end{cases}$$

- Equivalent to applying the softmax Bellman operator (**softmax DQN** [1])

$$q_{k+1} = r + \gamma P \Big\langle \operatorname{softmax}\left(\frac{q_k}{\tau}\right), q_k \Big\rangle + \epsilon_{k+1}$$

- Even without error, this might not be convergent (multiple fixed points)
- **Regularizing the evaluation step is important!**
- The **mellowmax policy** [2] is indeed a complicated way to do so [3]

[1] Z. Song et al. Revisiting the softmax bellman operator: New benefits and new perspective. ICML 2019.
[2] K. Asadi et al. An alternative softmax operator for reinforcement learning. ICML 2017.
[3] M. Geist et al. A theory of regularized MDPs. ICML 2019.

Entropy, type 1

- DP scheme

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \left(\langle \pi, q_k \rangle + \tau \mathcal{H}(\pi) \right) \\ q_{k+1} = r + \gamma P \left(\langle \pi_{k+1}, q_k \rangle + \tau \mathcal{H}(\pi_{k+1}) \right) + \epsilon_{k+1} \end{cases}$$

- **SAC** [1] and **soft Q-learning** [2,3] can be derived from this DP scheme

[1] T. Haarnoja et al. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. ICML 2018.
[2] T. Haarnoja et al. Reinforcement learning with deep energy-based policies. ICML 2017.
[3] R. Fox et al. Taming the noise in reinforcement learning via soft updates. UAI 2016.
[4] M. Geist et al. A theory of regularized MDPs. ICML 2019.

Google Research

Entropy, type 1

- DP scheme

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \left(\langle \pi, q_k \rangle + \tau \mathcal{H}(\pi) \right) \\ q_{k+1} = r + \gamma P \left(\langle \pi_{k+1}, q_k \rangle + \tau \mathcal{H}(\pi_{k+1}) \right) + \epsilon_{k+1} \end{cases}$$

- **SAC** [1] and **soft Q-learning** [2,3] can be derived from this DP scheme
- Analysis [4] (VI vs reg. entropy, type 1)

$$\|q_* - q_{\pi_k}\|_\infty \leq \mathcal{O}\left(\frac{1}{1-\gamma} \sum_{j=0}^{k} \gamma^{k-j} \|\epsilon_j\|_\infty + \frac{1}{1-\gamma} \gamma^k v_{\max} \right) \quad \Bigg\| \quad \|q_*^\tau - q_{\pi_k}^\tau\|_\infty \leq \mathcal{O}\left(\frac{1}{1-\gamma} \sum_{j=0}^{k} \gamma^{k-j} \|\epsilon_j\|_\infty + \frac{1}{1-\gamma} \gamma^k v_{\max} \right)$$

[1] T. Haarnoja et al. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. ICML 2018.
[2] T. Haarnoja et al. Reinforcement learning with deep energy-based policies. ICML 2017.
[3] R. Fox et al. Taming the noise in reinforcement learning via soft updates. UAI 2016.
[4] M. Geist et al. A theory of regularized MDPs. ICML 2019.

Google Research

KL, type 1

- DP scheme

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{S}} \left(\langle \pi, q_k \rangle - \lambda \, \mathrm{KL}(\pi || \pi_k) \right) \\ q_{k+1} = r + \gamma P \left(\langle \pi_{k+1}, q_k \rangle - \lambda \, \mathrm{KL}(\pi_{k+1} || \pi_k) \right) + \epsilon_{k+1} \end{cases}$$

- **DPP** [1] can be derived from this DP scheme

[1] M. Azar et al. Dynamic Policy Programming. JMLR 2012.
[2] M. Azar et al. Speedy Q-learning. NeurIPS 2011.
[3] N. Vieillard et al. Leverage the average: an analysis of KL regularization in RL. NeurIPS 2020.

Google Research

KL, type 1

- DP scheme

$$\begin{cases}\pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \left(\langle \pi, q_k \rangle - \lambda \mathrm{KL}(\pi || \pi_k) \right) \\ q_{k+1} = r + \gamma P \left(\langle \pi_{k+1}, q_k \rangle - \lambda \mathrm{KL}(\pi_{k+1} || \pi_k) \right) + \epsilon_{k+1} \end{cases} \Leftrightarrow \begin{cases}\pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \left(\langle \pi, h_k \rangle + \frac{\lambda}{k+1} \mathcal{H}(\pi) \right) \\ q_{k+1} = r + \gamma P \left(\langle \pi_{k+1}, q_k \rangle - \lambda \mathrm{KL}(\pi_{k+1} || \pi_k) \right) + \epsilon_{k+1} \\ h_{k+1} = \frac{k+1}{k+2} h_k + \frac{1}{k+2} q_{k+1} \end{cases}$$

- **DPP** [1] can be derived from this DP scheme
- It is a generalization of **Speedy Q-learning** [2] (*+link to reg. with q-values*)

[1] M. Azar et al. Dynamic Policy Programming. JMLR 2012.
[2] M. Azar et al. Speedy Q-learning. NeurIPS 2011.
[3] N. Vieillard et al. Leverage the average: an analysis of KL regularization in RL. NeurIPS 2020.

Google Research

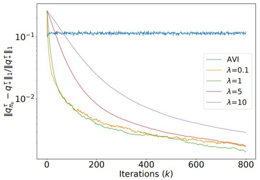
KL, type 1

- DP scheme

$$\begin{cases}\pi_{k+1} = \operatorname{argmax}_{\pi\in\Delta_{\mathcal{A}}^{\mathcal{S}}} \left(\langle \pi, q_k\rangle - \lambda \operatorname{KL}(\pi\|\pi_k)\right)\\ q_{k+1} = r + \gamma P\left(\langle \pi_{k+1}, q_k\rangle - \lambda \operatorname{KL}(\pi_{k+1}\|\pi_k)\right) + \epsilon_{k+1}\end{cases} \Longleftrightarrow \begin{cases}\pi_{k+1} = \operatorname{argmax}_{\pi\in\Delta_{\mathcal{A}}^{\mathcal{S}}} \left(\langle \pi, h_k\rangle + \frac{\lambda}{k+1}\mathcal{H}(\pi)\right)\\ q_{k+1} = r + \gamma P\left(\langle \pi_{k+1}, q_k\rangle - \lambda \operatorname{KL}(\pi_{k+1}\|\pi_k)\right) + \epsilon_{k+1}\\ h_{k+1} = \frac{k+1}{k+2} h_k + \frac{1}{k+2} q_{k+1}\end{cases}$$

- **DPP** [1] can be derived from this DP scheme
- It is a generalization of **Speedy Q-learning** [2] (*+link to reg. with q-values*)
- Analysis [3] (VI vs MD-VI with KL, type 1)

$$\|q_* - q_{\pi_k}\|_\infty \leq \mathcal{O}\left(\frac{1}{1-\gamma}\sum_{j=0}^{k}\gamma^{k-j}\|\epsilon_j\|_\infty + \frac{1}{1-\gamma}\gamma^k v_{\max}\right) \quad \Bigg\| \quad \|q_* - q_{\pi_k}\|_\infty \leq \mathcal{O}\left(\frac{1}{1-\gamma}\left\|\frac{1}{k}\sum_{j=1}^{k}\epsilon_j\right\|_\infty + \frac{1}{1-\gamma}\frac{v_{\max}^{\lambda}}{k}\right)$$



[1] M. Azar et al. Dynamic Policy Programming. JMLR 2012.
[2] M. Azar et al. Speedy Q-learning. NeurIPS 2011.
[3] N. Vieillard et al. Leverage the average: an analysis of KL regularization in RL. NeurIPS 2020.

KL, type 2

- DP scheme

$$\begin{cases}\pi_{k+1} = \mathrm{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \left(\langle \pi, q_k \rangle - \lambda \mathrm{KL}(\pi || \pi_k)\right) \\ q_{k+1} = r + \gamma P \left(\langle \pi_{k+1}, q_k \rangle - \lambda \cancel{\mathrm{KL}(\pi_{k+1} || \pi_k)}\right) + \epsilon_{k+1}\end{cases}$$

- **TRPO** [1] and even more **MPO** [2] are close to this scheme

[1] J. Schulman et al. Trust region policy optimization. ICML 2015.
[2] A. Abdolmaleki et al. Maximum a posteriori policy optimisation. ICLR 2018.
[3] N. Vieillard et al. Momentum in reinforcement learning. AISTATS 2020.
[4] Y. Abbasi-Yadkori et al. Politex: Regret bounds for policy iteration using expert prediction. ICML 2019.
[5] N. Vieillard et al. Leverage the average: an analysis of regularization in RL. arXiv 2020.

KL, type 2

- DP scheme

$$\begin{cases}\pi_{k+1} = \mathrm{argmax}_{\pi\in\Delta_{\mathcal{A}}^{\mathcal{S}}}\left(\langle \pi, q_k\rangle - \lambda \mathrm{KL}(\pi\|\pi_k)\right)\\ q_{k+1} = r + \gamma P\left(\langle \pi_{k+1}, q_k\rangle - \lambda \cancel{\mathrm{KL}(\pi_{k+1}\|\pi_k)}\right) + \epsilon_{k+1}\end{cases} \Longleftrightarrow \begin{cases}\pi_{k+1} = \mathrm{argmax}_{\pi\in\Delta_{\mathcal{A}}^{\mathcal{S}}}\left(\langle \pi, h_k\rangle + \frac{\lambda}{k+1}\mathcal{H}(\pi)\right)\\ q_{k+1} = r + \gamma P\left(\langle \pi_{k+1}, q_k\rangle - \lambda \cancel{\mathrm{KL}(\pi_{k+1}\|\pi_k)}\right) + \epsilon_{k+1}\\ h_{k+1} = \frac{k+1}{k+2}h_k + \frac{1}{k+2}q_{k+1}\end{cases}$$

- **TRPO** [1] and even more **MPO** [2] are close to this scheme
- Generalizes **Momentum-VI** [3], VI-based variation of **Politex** [4]

[1] J. Schulman et al. Trust region policy optimization. ICML 2015.
[2] A. Abdolmaleki et al. Maximum a posteriori policy optimisation. ICLR 2018.
[3] N. Vieillard et al. Momentum in reinforcement learning. AISTATS 2020.
[4] Y. Abbasi-Yadkori et al. Politex: Regret bounds for policy iteration using expert prediction. ICML 2019.
[5] N. Vieillard et al. Leverage the average: an analysis of regularization in RL. arXiv 2020.

Google Research

KL, type 2

- DP scheme

$$\begin{cases}\pi_{k+1} = \operatorname{argmax}_{\pi\in\Delta_{\mathcal{A}}^{\mathcal{S}}}\left(\langle \pi, q_k\rangle - \lambda\,\mathrm{KL}(\pi\|\pi_k)\right)\\ q_{k+1} = r + \gamma P\left(\langle \pi_{k+1}, q_k\rangle - \cancel{\lambda\,\mathrm{KL}(\pi_{k+1}\|\pi_k)}\right) + \epsilon_{k+1}\end{cases} \Longleftrightarrow \begin{cases}\pi_{k+1} = \operatorname{argmax}_{\pi\in\Delta_{\mathcal{A}}^{\mathcal{S}}}\left(\langle \pi, h_k\rangle + \frac{\lambda}{k+1}\mathcal{H}(\pi)\right)\\ q_{k+1} = r + \gamma P\left(\langle \pi_{k+1}, q_k\rangle - \cancel{\lambda\,\mathrm{KL}(\pi_{k+1}\|\pi_k)}\right) + \epsilon_{k+1}\\ h_{k+1} = \frac{k+1}{k+2}h_k + \frac{1}{k+2}q_{k+1}\end{cases}$$

- **TRPO** [1] and even more **MPO** [2] are close to this scheme
- Generalizes **Momentum-VI** [3], VI-based variation of **Politex** [4]
- Analysis [5] (VI vs MD-VI with KL, type 2)

$$\|q_* - q_{\pi_k}\|_\infty \le \mathcal{O}\left(\frac{1}{1-\gamma}\sum_{j=0}^{k}\gamma^{k-j}\|\epsilon_j\|_\infty + \frac{1}{1-\gamma}\gamma^k v_{\max}\right) \qquad \|q_* - q_{\pi_k}\|_\infty \le \mathcal{O}\left(\frac{\|E_k\|_\infty + \frac{\max_{j\le k-1}\|\mathcal{E}_{j,k-1}\|_\infty}{1-\gamma} + v^{\lambda}_{\max}}{(1-\gamma)k}\right)$$

[1] J. Schulman et al. Trust region policy optimization. ICML 2015.

[2] A. Abdolmaleki et al. Maximum a posteriori policy optimisation. ICLR 2018.

[3] N. Vieillard et al. Momentum in reinforcement learning. AISTATS 2020.

[4] Y. Abbasi-Yadkori et al. Politex: Regret bounds for policy iteration using expert prediction. ICML 2019.

[5] N. Vieillard et al. Leverage the average: an analysis of regularization in RL. arXiv 2020.

Mixed entropy and KL, type 1

- DP scheme

$$\begin{cases}\pi_{k+1} = \text{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \left(\langle \pi, q_k \rangle - \lambda \, \text{KL}(\pi || \pi_k) + \tau \mathcal{H}(\pi) \right) \\ q_{k+1} = r + \gamma P \left(\langle \pi_{k+1}, q_k \rangle - \lambda \, \text{KL}(\pi_{k+1} || \pi_k) + \tau \mathcal{H}(\pi_{k+1}) \right) + \epsilon_{k+1}\end{cases}$$

- **CVI** [1] can be derived from this scheme (thus, **advantage learning** [2] too)

[1] T. Kozuno et al. Theoretical analysis of efficiency and robustness of softmax and gap-increasing operators in reinforcement learning. AISTATS 2019
[2] M. Bellemare at al. Increasing the action gap: New operators for reinforcement learning. AAAI 2019.
[3] N. Vieillard, B. Scherrer, O. Pietquin and M. Geist. Momentum in reinforcement learning. AISTATS 2020.
[4] N. Vieillard et al. Leverage the average: an analysis of KL regularization in RL. NeurIPS 2020.

Google Research

Mixed entropy and KL, type 1

- DP scheme

$$\begin{cases}\pi_{k+1} = \text{argmax}_{\pi\in\Delta_{\mathcal{A}}^{S}} \left(\langle \pi, q_k\rangle - \lambda \,\text{KL}(\pi||\pi_k) + \tau\mathcal{H}(\pi)\right) \\ q_{k+1} = r + \gamma P\left(\langle \pi_{k+1}, q_k\rangle - \lambda\,\text{KL}(\pi_{k+1}||\pi_k) + \tau\mathcal{H}(\pi_{k+1})\right) + \epsilon_{k+1}\end{cases} \Leftrightarrow \begin{cases}\pi_{k+1} = \text{argmax}_{\pi\in\Delta_{\mathcal{A}}^{S}} \left(\langle \pi, h_k\rangle + \tau\mathcal{H}(\pi)\right) \\ q_{k+1} = r + \gamma P\left(\langle \pi_{k+1}, q_k\rangle - \lambda\,\text{KL}(\pi_{k+1}||\pi_k) + \tau\mathcal{H}(\pi_{k+1})\right) + \epsilon_{k+1} \\ h_{k+1} = \beta h_k + (1-\beta) q_{k+1} \text{ with } \beta = \frac{\lambda}{\lambda+\tau}\end{cases}$$

- **CVI** [1] can be derived from this scheme (thus, **advantage learning** [2] too)
- Generalizes **Momentum-DQN** [3]

[1] T. Kozuno et al. Theoretical analysis of efficiency and robustness of softmax and gap-increasing operators in reinforcement learning. AISTATS 2019

[2] M. Bellemare at al. Increasing the action gap: New operators for reinforcement learning. AAAI 2019.

[3] N. Vieillard, B. Scherrer, O. Pietquin and M. Geist. Momentum in reinforcement learning. AISTATS 2020.

[4] N. Vieillard et al. Leverage the average: an analysis of KL regularization in RL. NeurIPS 2020.

Google Research

Mixed entropy and KL, type 1

- DP scheme

$$\begin{cases}\pi_{k+1} = \operatorname{argmax}_{\pi\in\Delta_{\mathcal{A}}^{\mathcal{S}}} \left(\langle \pi, q_k\rangle - \lambda\,\mathrm{KL}(\pi||\pi_k) + \tau\mathcal{H}(\pi)\right)\\ q_{k+1} = r + \gamma P\left(\langle \pi_{k+1}, q_k\rangle - \lambda\,\mathrm{KL}(\pi_{k+1}||\pi_k) + \tau\mathcal{H}(\pi_{k+1})\right) + \epsilon_{k+1}\end{cases} \Leftrightarrow \begin{cases}\pi_{k+1} = \operatorname{argmax}_{\pi\in\Delta_{\mathcal{A}}^{\mathcal{S}}} \left(\langle \pi, h_k\rangle + \tau\mathcal{H}(\pi)\right)\\ q_{k+1} = r + \gamma P\left(\langle \pi_{k+1}, q_k\rangle - \lambda\,\mathrm{KL}(\pi_{k+1}||\pi_k) + \tau\mathcal{H}(\pi_{k+1})\right) + \epsilon_{k+1}\\ h_{k+1} = \beta h_k + (1-\beta) q_{k+1} \text{ with } \beta = \frac{\lambda}{\lambda+\tau}\end{cases}$$

- **CVI** [1] can be derived from this scheme (thus, **advantage learning** [2] too)
- Generalizes **Momentum-DQN** [3]
- Analysis [4] (VI vs MD-VI with KL+entropy, type 1)

$$\|q_* - q_{\pi_k}\|_\infty \le \mathcal{O}\left(\frac{1}{1-\gamma}\sum_{j=0}^{k}\gamma^{k-j}\|\epsilon_j\|_\infty + \frac{1}{1-\gamma}\gamma^k v_{\max}\right) \quad \Big\| \quad \|q_*^\tau - q_{\pi_{k+1}}^\tau\|_\infty \le \mathcal{O}\left(\frac{1}{1-\gamma}\sum_{j=1}^{k}\gamma^{k-j}\|E_j^\beta\|_\infty + g^2(k)\right)$$

[1] T. Kozuno et al. Theoretical analysis of efficiency and robustness of softmax and gap-increasing operators in reinforcement learning. AISTATS 2019

[2] M. Bellemare at al. Increasing the action gap: New operators for reinforcement learning. AAAI 2019.

[3] N. Vieillard, B. Scherrer, O. Pietquin and M. Geist. Momentum in reinforcement learning. AISTATS 2020.

[4] N. Vieillard et al. Leverage the average: an analysis of KL regularization in RL. NeurIPS 2020.

Mixed entropy and KL, type 2

- DP scheme

$$\begin{cases} \pi_{k+1} = \mathrm{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \left(\langle \pi, q_k \rangle - \lambda \mathrm{KL}(\pi || \pi_k) + \tau \mathcal{H}(\pi) \right) \\ q_{k+1} = r + \gamma P \left(\langle \pi_{k+1}, q_k \rangle - \lambda \cancel{\mathrm{KL}(\cdots \mathcal{H}(\pi_{k+1})} \right) + \epsilon_{k+1} \end{cases} \Longleftrightarrow \begin{cases} \pi_{k+1} = \mathrm{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \left(\langle \pi, h_k \rangle + \tau \mathcal{H}(\pi) \right) \\ q_{k+1} = r + \gamma P \left(\langle \pi_{k+1}, q_k \rangle - \lambda \cancel{\mathrm{KL}(\cdots \mathcal{H}(\pi_{k+1})} \right) + \epsilon_{k+1} \\ h_{k+1} = \beta h_k + (1 - \beta) q_{k+1} \text{ with } \beta = \frac{\lambda}{\lambda + \tau} \end{cases}$$

- Existing algorithm doing this?

Google Research

Mixed entropy and KL, type 2

- DP scheme

$$\begin{cases}\pi_{k+1} = \operatorname{argmax}_{\pi\in\Delta_{\mathcal{A}}^{\mathcal{S}}}\left(\langle \pi, q_k\rangle - \lambda\,\mathrm{KL}(\pi\|\pi_k) + \tau\mathcal{H}(\pi)\right)\\ q_{k+1} = r + \gamma P\left(\langle \pi_{k+1}, q_k\rangle - \cancel{\lambda\,\mathrm{KL}(\pi_{k+1}\|\pi_k) + \tau\mathcal{H}(\pi_{k+1})}\right) + \epsilon_{k+1}\end{cases} \Longleftrightarrow \begin{cases}\pi_{k+1} = \operatorname{argmax}_{\pi\in\Delta_{\mathcal{A}}^{\mathcal{S}}}\left(\langle \pi, h_k\rangle + \tau\mathcal{H}(\pi)\right)\\ q_{k+1} = r + \gamma P\left(\langle \pi_{k+1}, q_k\rangle - \cancel{\lambda\,\mathrm{KL}(\pi_{k+1}\|\pi_k) + \tau\mathcal{H}(\pi_{k+1})}\right) + \epsilon_{k+1}\\ h_{k+1} = \beta h_k + (1-\beta) q_{k+1} \text{ with } \beta = \frac{\lambda}{\lambda+\tau}\end{cases}$$

- Existing algorithm doing this?
- Analysis: same issue as entropy/type 2
- Solution (for the analysis): introduce a **type 3** [1]

$$\begin{cases}\pi_{k+1} = \operatorname{argmax}_{\pi\in\Delta_{\mathcal{A}}^{\mathcal{S}}}\left(\langle \pi, q_k\rangle - \lambda\,\mathrm{KL}(\pi\|\pi_k) + \tau\mathcal{H}(\pi)\right)\\ q_{k+1} = r + \gamma P\left(\langle \pi_{k+1}, q_k\rangle - \cancel{\lambda\,\mathrm{KL}(\pi_{k+1}\|\pi_k)} + \tau\mathcal{H}(\pi_{k+1})\right) + \epsilon_{k+1}\end{cases}$$

- Mixes moving average of the errors with bounding of the biased quantity and kernel preconditioning... but not that interesting

[1] N. Vieillard et al. Leverage the average: an analysis of regularization in RL. arXiv 2020.

The issue with (KL) regularization

Hint: it's in the greedy step

Google Research

OK for linear param., but...

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \left(\langle \pi, q_k \rangle - \lambda \operatorname{KL}(\pi || \pi_k) \right) \\ q_{k+1} = r + \gamma P \left(\langle \pi_{k+1}, q_k \rangle - \lambda \operatorname{KL}(\pi_{k+1} || \pi_k) \right) + \epsilon_{k+1} \end{cases}$$

$$\Updownarrow$$

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \left(\langle \pi, h_k \rangle - \frac{\lambda}{k+1} \mathcal{H}(\pi) \right) \\ q_{k+1} = r + \gamma P \left(\langle \pi_{k+1}, q_k \rangle - \lambda \operatorname{KL}(\pi_{k+1} || \pi_k) \right) + \epsilon_{k+1} \\ h_{k+1} = \frac{k+1}{k+2} h_k + \frac{1}{k+2} q_{k+1} \end{cases}$$

OK for linear param., but...

We do errors here (with deep nets)

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{S}} \left(\langle \pi, q_k \rangle - \lambda \, \mathrm{KL}(\pi || \pi_k) \right) \\ q_{k+1} = r + \gamma P \left(\langle \pi_{k+1}, q_k \rangle - \lambda \, \mathrm{KL}(\pi_{k+1} || \pi_k) \right) + \epsilon_{k+1} \end{cases}$$

$$\Updownarrow$$

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{S}} \left(\langle \pi, h_k \rangle - \frac{\lambda}{k+1} \mathcal{H}(\pi) \right) \\ q_{k+1} = r + \gamma P \left(\langle \pi_{k+1}, q_k \rangle - \lambda \, \mathrm{KL}(\pi_{k+1} || \pi_k) \right) + \epsilon_{k+1} \\ h_{k+1} = \frac{k+1}{k+2} h_k + \frac{1}{k+2} q_{k+1} \end{cases}$$

Google Research

OK for linear param., but...

We do errors here (with deep nets)

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{S}} \left(\langle \pi, q_k \rangle - \lambda \, \mathrm{KL}(\pi \| \pi_k) \right) \\ q_{k+1} = r + \gamma P \left(\langle \pi_{k+1}, q_k \rangle - \lambda \, \mathrm{KL}(\pi_{k+1} \| \pi_k) \right) + \epsilon_{k+1} \end{cases}$$

$$\Updownarrow$$

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{S}} \left(\langle \pi, h_k \rangle - \frac{\lambda}{k+1} \mathcal{H}(\pi) \right) \\ q_{k+1} = r + \gamma P \left(\langle \pi_{k+1}, q_k \rangle - \lambda \, \mathrm{KL}(\pi_{k+1} \| \pi_k) \right) + \epsilon_{k+1} \\ h_{k+1} = \frac{k+1}{k+2} h_k + \frac{1}{k+2} q_{k+1} \end{cases}$$

We do errors here (with deep nets)

OK for linear param., but...

We do errors here (with deep nets)

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{S}} \left(\langle \pi, q_k \rangle - \lambda \, \mathrm{KL}(\pi \| \pi_k) \right) \\ q_{k+1} = r + \gamma P \left(\langle \pi_{k+1}, q_k \rangle - \lambda \, \mathrm{KL}(\pi_{k+1} \| \pi_k) \right) + \epsilon_{k+1} \end{cases}$$

$$\not\Updownarrow$$

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{S}} \left(\langle \pi, h_k \rangle - \frac{\lambda}{k+1} \mathcal{H}(\pi) \right) \\ q_{k+1} = r + \gamma P \left(\langle \pi_{k+1}, q_k \rangle - \lambda \, \mathrm{KL}(\pi_{k+1} \| \pi_k) \right) + \epsilon_{k+1} \\ h_{k+1} = \frac{k+1}{k+2} h_k + \frac{1}{k+2} q_{k+1} \end{cases}$$

We do errors here (with deep nets)

Google Research

OK for linear param., but...

We do errors here (with deep nets)

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{S}} \left(\langle \pi, q_k \rangle - \lambda \operatorname{KL}(\pi || \pi_k) \right) \\ q_{k+1} = r + \gamma P \left(\langle \pi_{k+1}, q_k \rangle - \lambda \operatorname{KL}(\pi_{k+1} || \pi_k) \right) + \epsilon_{k+1} \end{cases}$$

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{S}} \left(\langle \pi, h_k \rangle - \frac{\lambda}{k+1} \mathcal{H}(\pi) \right) \\ q_{k+1} = r + \gamma P \left(\langle \pi_{k+1}, q_k \rangle - \lambda \operatorname{KL}(\pi_{k+1} || \pi_k) \right) + \epsilon_{k+1} \\ h_{k+1} = \frac{k+1}{k+2} h_k + \frac{1}{k+2} q_{k+1} \end{cases}$$

We do errors here (with deep nets)

Not really compatible with stochastic approx. (but ok if moving average)

A remedy

Munchausen Reinforcement Learning

Google Research

A reparameterization trick

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \langle \pi, q'_k \rangle - \alpha\tau \, \mathrm{KL}(\pi || \pi_k) + (1-\alpha)\tau \mathcal{H}(\pi) \\ q'_{k+1} = r + \gamma P(\langle \pi_{k+1}, q'_k \rangle - \alpha\tau \, \mathrm{KL}(\pi_{k+1} || \pi_k) + (1-\alpha)\tau \mathcal{H}(\pi_{k+1})) + \epsilon_{k+1} \end{cases}$$

Google Research

A reparameterization trick

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \langle \pi, q'_k \rangle - \alpha\tau\,\mathrm{KL}(\pi || \pi_k) + (1-\alpha)\tau\mathcal{H}(\pi) \\ q'_{k+1} = r + \gamma P(\langle \pi_{k+1}, q'_k \rangle - \alpha\tau\,\mathrm{KL}(\pi_{k+1} || \pi_k) + (1-\alpha)\tau\mathcal{H}(\pi_{k+1})) + \epsilon_{k+1} \end{cases}$$

$$\Updownarrow \quad (q_k = q'_k + \alpha\tau \ln \pi_k)$$

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \langle \pi, q_k \rangle + \tau\mathcal{H}(\pi) \\ q_{k+1} = r + \alpha\tau \ln \pi_{k+1} + \gamma P\langle \pi_{k+1}, q_k - \tau \ln \pi_{k+1} \rangle + \epsilon_{k+1}. \end{cases}$$

A reparameterization trick

$$\begin{cases} \pi_{k+1} = \mathrm{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \langle \pi, q'_k \rangle - \alpha\tau \, \mathrm{KL}(\pi || \pi_k) + (1-\alpha)\tau\mathcal{H}(\pi) \\ q'_{k+1} = r + \gamma P(\langle \pi_{k+1}, q'_k \rangle - \alpha\tau \, \mathrm{KL}(\pi_{k+1} || \pi_k) + (1-\alpha)\tau\mathcal{H}(\pi_{k+1})) + \epsilon_{k+1} \end{cases}$$

$$\Updownarrow \quad \left(q_k = q'_k + \alpha\tau \ln \pi_k\right)$$

$$\begin{cases} \pi_{k+1} = \mathrm{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \langle \pi, q_k \rangle + \tau\mathcal{H}(\pi) \\ q_{k+1} = r + \underbrace{\alpha\tau \ln \pi_{k+1}}_{\text{Munchausen term [1]}} + \gamma P\langle \pi_{k+1}, q_k - \tau \ln \pi_{k+1} \rangle + \epsilon_{k+1}. \end{cases}$$

[1] N. Vieillard et al. Munchausen Reinforcement Learning. NeurIPS 2020

Google Research

A reparameterization trick

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \langle \pi, q'_k \rangle - \alpha\tau \, \mathrm{KL}(\pi || \pi_k) + (1-\alpha)\tau\mathcal{H}(\pi) \\ q'_{k+1} = r + \gamma P(\langle \pi_{k+1}, q'_k \rangle - \alpha\tau \, \mathrm{KL}(\pi_{k+1} || \pi_k) + (1-\alpha)\tau\mathcal{H}(\pi_{k+1})) + \epsilon_{k+1} \end{cases}$$

$$\Updownarrow \quad (q_k = q'_k + \alpha\tau \ln \pi_k)$$

$$\begin{cases} \pi_{k+1} = \operatorname{argmax}_{\pi \in \Delta_{\mathcal{A}}^{\mathcal{S}}} \langle \pi, q_k \rangle + \tau\mathcal{H}(\pi) \\ q_{k+1} = r + \underbrace{\alpha\tau \ln \pi_{k+1}}_{\text{Munchausen term [1]}} + \gamma P\langle \pi_{k+1}, q_k - \tau \ln \pi_{k+1} \rangle + \epsilon_{k+1}. \end{cases}$$

Bonus: it also increases the action-gap, $\lim_{k \to \infty} \mathrm{gap}_k^{\alpha,\tau}(s) = \frac{1+\alpha}{1-\alpha} \mathrm{gap}_*^{(1-\alpha)\tau}(s)$

[1] N. Vieillard et al. Munchausen Reinforcement Learning. NeurIPS 2020

Google Research

Case study: DQN

- Let's modify DQN with the Munchausen term to get Munchausen-DQN
- We'll only modify the regression target of DQN:

$$\hat{q}_{\text{dqn}}(r_t, s_{t+1}) = r_t + \gamma \sum_{a' \in \mathcal{A}} \pi_{\bar{\theta}}(a'|s_{t+1}) q_{\bar{\theta}}(s_{t+1}, a') \text{ with } \pi_{\bar{\theta}} \in \mathcal{G}(q_{\bar{\theta}})$$

- We need a stochastic policy, so just add some entropy regularization:

$$\hat{q}_{\text{s-dqn}}(r_t, s_{t+1}) = r_t + \gamma \sum_{a' \in \mathcal{A}} \pi_{\bar{\theta}}(a'|s_{t+1}) \Big(q_{\bar{\theta}}(s_{t+1}, a') - \tau \ln \pi_{\bar{\theta}}(a'|s_{t+1}) \Big) \text{ with } \pi_{\bar{\theta}} = \text{softmax}(\frac{q_{\bar{\theta}}}{\tau})$$

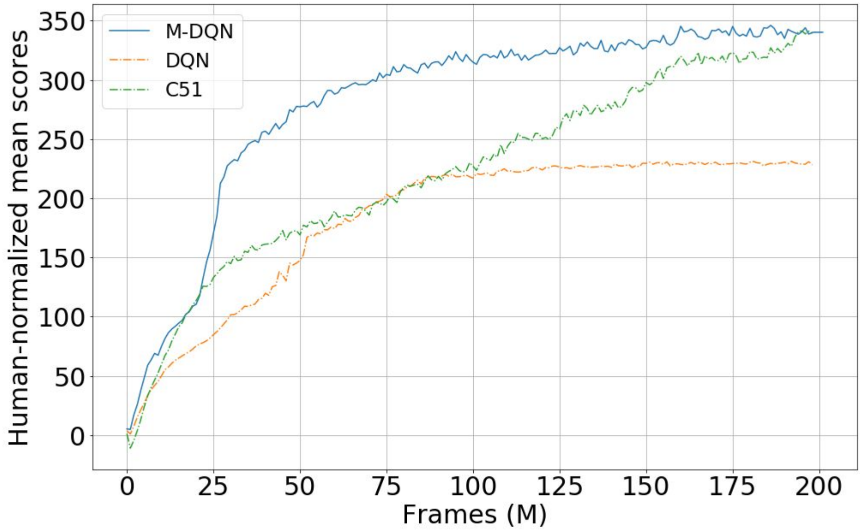
- Then, we just have to add the Munchausen term ($\pi_{\bar{\theta}}$ as above):

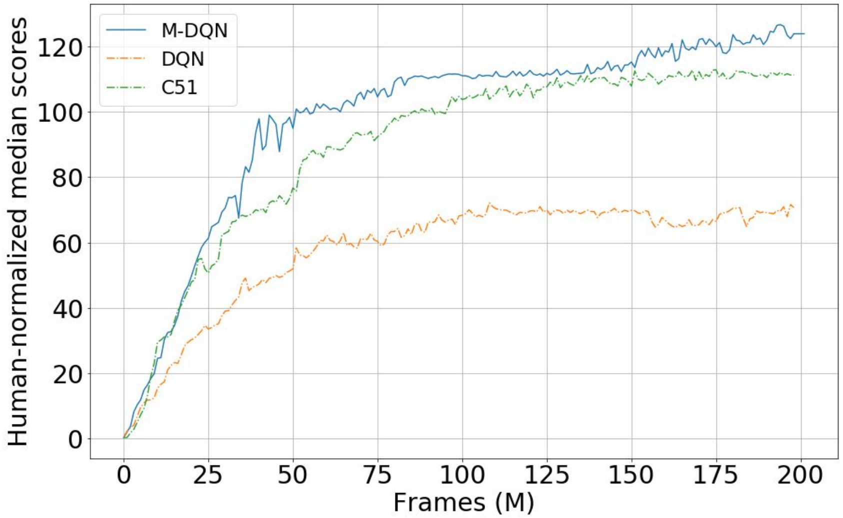
$$\hat{q}_{\text{m-dqn}}(r_t, s_{t+1}) = r_t + \alpha \tau \ln \pi_{\bar{\theta}}(a_t|s_t) + \gamma \sum_{a' \in \mathcal{A}} \pi_{\bar{\theta}}(a'|s_{t+1}) \Big(q_{\bar{\theta}}(s_{t+1}, a') - \tau \ln \pi_{\bar{\theta}}(a'|s_{t+1}) \Big)$$

- (notice that the log-policy terms have different signs)
- That's it!

Case study: DQN

- How good is Munchausen-DQN compared to DQN?
  - Aggregated results on the 60 Atari games of ALE, with also C51

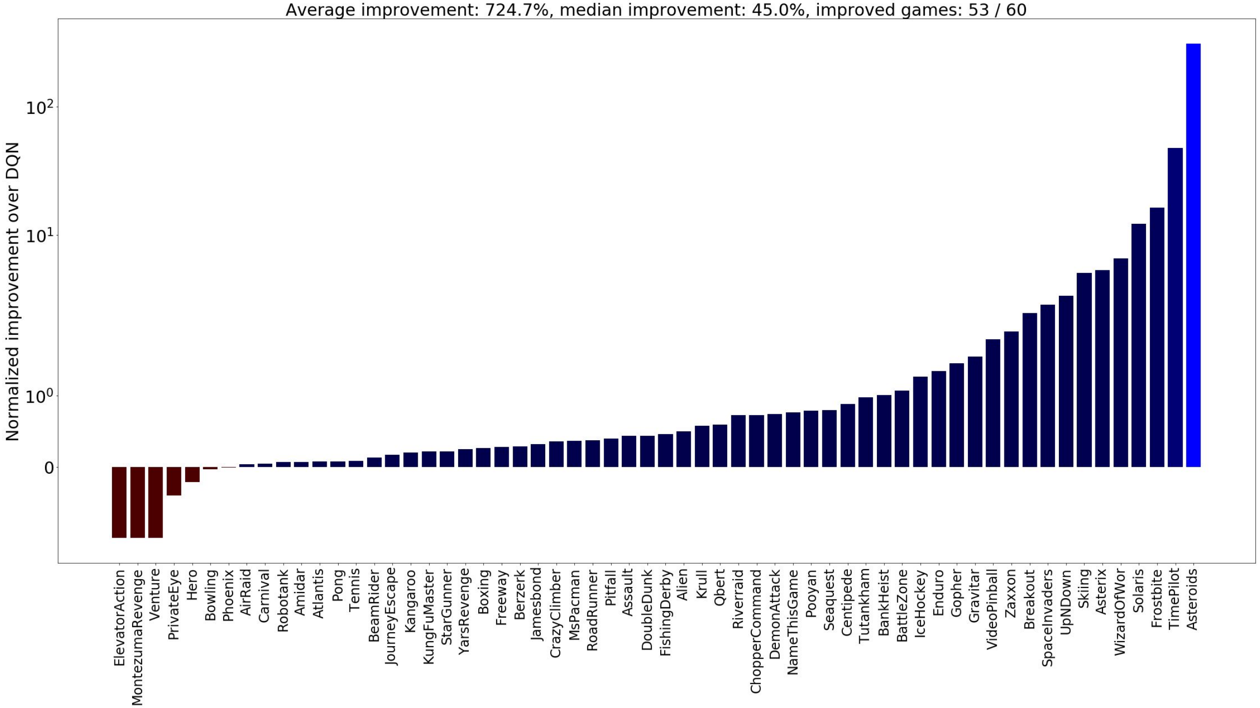




Google Research

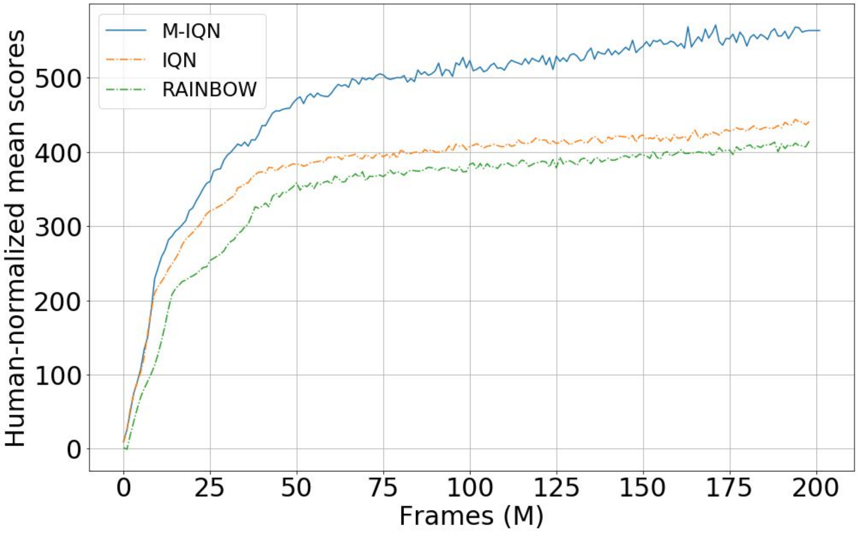
Case study: DQN

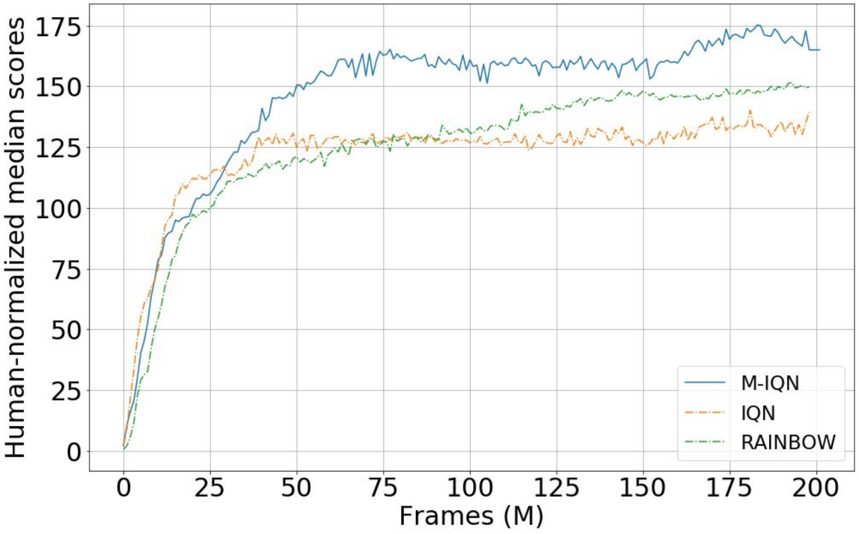
- How good is Munchausen-DQN compared to DQN?
  - Per game improvement



Case study: IQN

- This is a general approach. As an example, we apply it to IQN [1]
- Munchausen-IQN vs IQN, aggregated results over 60 games





[1] W. Dabney et al. Implicit Quantile Networks for Distributional Reinforcement Learning, ICML 2018.

Conclusion

This talk

Google Research

- Overview of regularized approximate dynamic programming
  - Connections to convex optimization/online learning
  - Allows recovering (variations of) (many) regularized RL agents
  - Allows for a theoretical analysis
  - Bring new agents, simple, theoretically grounded and very efficient (Munchausen RL)
- Many other possible outcomes
  - Imitation learning
  - Inverse RL
  - Offline RL
  - Multi-agent RL and game theory
  - ...

Google Research

Thanks!