Reinforcement Learning Model-based

Odalric-Ambrym Maillard

2021-2022

Inria Scool

Today's lecture

Deep-RL: **Actor-critic** strategies
- Actor-critic, Soft Actor Critic

Robotics: **Entropy**
- REPS

Model-based MDPs: **regret** minimization, **exploration-exploitation**
- Gain, Bias, optimality
- Performance bounds
- Intrinsic Contraction, Value Iteration
- **Optimistic** strategy: UCRL2, UCRL3

Exploration in Deep-learning

Table of contents

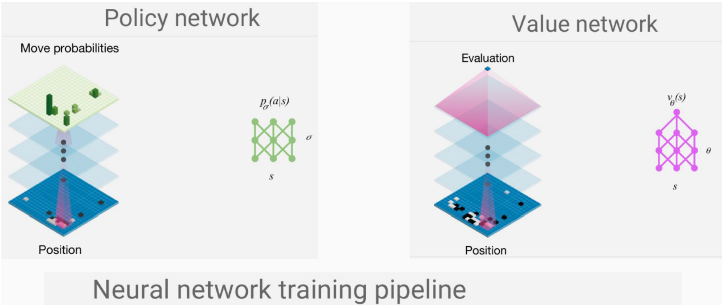
- $\mathcal{S}$ States, $\mathcal{A}$ Actions
- $R : \mathcal{S} \times \mathcal{A} \to \mathcal{P}([0,1])$ **Reward** function (mean m)
- $P : \mathcal{S} \times \mathcal{A} \to \mathcal{P}(\mathcal{S})$ **Transition** function
- $\pi : \mathcal{S} \times \mathcal{A}$ **Policy**, or **class of policies** $\Pi = \{\pi_\theta : \theta \in \Theta\}$
- $V_\pi(s) = \mathbb{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \middle| s_0 = a\right]$ **Value** function
- $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ **Quality** function

- $\mathcal{S}$ States, $\mathcal{A}$ Actions
- $R : \mathcal{S} \times \mathcal{A} \to \mathcal{P}([0,1])$ **Reward** function (mean m)
- $P : \mathcal{S} \times \mathcal{A} \to \mathcal{P}(\mathcal{S})$ **Transition** function
- $\pi : \mathcal{S} \times \mathcal{A}$ **Policy**, or **class of policies** $\Pi = \{\pi_\theta : \theta \in \Theta\}$
- $V_\pi(s) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \middle| s_0 = a \right]$ **Value** function
- $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ **Quality** function

Challenge

How to **represent** R, P, π, Q, V_π when $\mathcal{S}, \mathcal{A}$ are **huge** ? **continuous**?

- **Critic**: Value function oriented Q (SARSA, Q-learning, TD, etc.)
- **Actor**: Policy oriented π (Policy gradient, Reinforce, PPO, TRPO)
- **Actor-critic**: use both (e.g. A3C). e.g. AlphaGo



Take-home critic strategies

Tabular value function:

- **Q-learning**: $Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha(r_t + \gamma \max_b Q_t(s_{t+1}, b) - Q_t(s_t, a_t))$
- **SARSA**: $Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha(r_t + \gamma Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t))$

Parametric Linear Value function:

- **LSTD** for V_θ: $\theta_n = \left(\sum_{i=1}^n \varphi(s_i)(\varphi(s_i) - \gamma\varphi(s_{i+1})^\top)\right)^{-1} \sum_{i=1}^n \varphi(s_i) r_i$
- **LSPI** for Q_θ: $\theta_n = \left(\sum_{i=1}^n \varphi(s_i, a_i)(\varphi(s_i, a_i) - \gamma\varphi(s'_i, \pi(s'_i))^\top)\right)^{-1} \sum_{i=1}^n \varphi(s_i, a_i) r_i$

Parametric Non-Linear Value function:

- **DQN**: $\theta \leftarrow \alpha(r + \gamma Q_{\theta-}(s', \mathrm{argmax}_b\, Q_{\theta-}(s, b)) - Q_\theta(s, a))\nabla_\theta Q_\theta(s, a)$
- **DDQN**: $\theta \leftarrow \alpha(r + \gamma Q_{\theta-}(s', \mathrm{argmax}_b\, Q_\theta(s, b)) - Q_\theta(s, a))\nabla_\theta Q_\theta(s, a)$
  - Exp.Replay: sample $(r, s, a, s') \propto |r + \gamma \max_b Q_{\theta-}(s', b)) - Q_\theta(s, a)|$
  - Multi-step variants
- **SAIL**:

$\theta \leftarrow \alpha(r + \gamma \max_b Q_{\theta-}(s', b) + \beta \max[G_t, Q_{\theta-}(s, a) - \max_a Q_{\theta-}(s, a)])\nabla_\theta Q_\theta(s, a)$

where $G_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}$

Take-home actor strategies

Parameterize policy π_θ :

- **Policy gradient**: $\nabla_\theta J(\theta) = \mathbb{E}_\tau[R(\tau)\nabla_\theta \log P(\tau|\theta)]$
- **REINFORCE**: $\theta \leftarrow \theta + \alpha\Big(\sum_{t=1}^{H} \nabla \log \pi_\theta(s_t, a_t) \sum_{t'=t}^{H} r_t\Big)$
- **PPO**: $\theta \leftarrow \underset{\theta'}{\operatorname{argmax}}\, J_\theta(\theta') - \lambda \mathbb{E} D_{KL}(\theta', \theta)$
  - **TRPO**: $\theta \leftarrow \underset{\theta'}{\operatorname{argmax}}\, J_\theta(\theta')$ s.t. $\mathbb{E} D_{KL}(\theta', \theta) \leqslant \delta$
  - **REPS**: plus entropy constraint $H(\theta') - H(\theta) \leqslant \gamma$ (today).

Table of contents

Back to **Actor** policy:

$$\nabla J_H(\theta) \quad = \quad \mathbb{E}_{\pi_\theta}\left[\sum_{t=1}^{H} \nabla_\theta \log \pi_\theta(a_t|s_t) R_t(\tau)\right], \quad R_t(\tau) = \sum_{t'=t}^{H} r_{t'}$$

▷ Idea 1: $R_t(\tau)$ is reward from trajectory, MC proxy for **value**. Looks like $Q_{\pi_\theta}(s_t, a_t)$.

Can we use representation of Q_{π_θ} as well? $Q_\omega(s, a) = \omega^\top \psi(s, a)$?

Coming back to PG theorem

$$\nabla_\theta J(\theta) = \mathbb{E}[\nabla_\theta \log \pi_\theta(s,a) Q^{\pi_\theta}(s,a)]$$

Approximate Q^{π_θ}

- If $Q^{\pi_\theta}(s,a) \approx Q_\omega(s,a)$
- do we have $\nabla_\theta J(\theta) \approx \mathbb{E}[\nabla_\theta \log \pi_\theta(s,a) Q_\omega(s,a)]$?
- If yes, π_θ is an actor (behaves), Q_ω is a critic (suggests direction to update policy)
- Both can be estimated online: π_θ with PG and Q_ω with SARSA
- It could lead to more stable (less variance) algorithms.

Compatible value function approximation I

Theorem: compatibility of approximations [Sutton et al., 2000]

If the two following conditions are satisfied:

1. The parameters ω minimize the mean square error:

$$\omega^* = \underset{\omega}{\operatorname{argmin}}\, \mathbb{E}_{\pi_\theta}\left[\left(Q^{\pi_\theta}(s,a) - Q_\omega(s,a)\right)^2\right]$$

2. The value and the policy approximation are *compatible*:

$$\nabla_\omega Q_\omega = \nabla_\theta \log \pi_\theta$$

Then the policy gradient is exact:

$$\nabla_\theta J(\theta) = \mathbb{E}[\nabla_\theta \log \pi_\theta(s,a) Q_\omega(s,a)]$$

Compatible value function approximation II

Proof

If mean square error is minimal, than its gradient w.r.t. to ω is zero.

$$\begin{aligned}
\nabla_\omega \mathbb{E}_{\pi_\theta}\left[\left(Q^{\pi_\theta}(s,a) - Q_\omega(s,a)\right)^2\right] &= 0 \\
\mathbb{E}_{\pi_\theta}\left[\left(Q^{\pi_\theta}(s,a) - Q_\omega(s,a)\right)\nabla_\omega Q_\omega(s,a)\right] &= 0 \\
\mathbb{E}_{\pi_\theta}\left[\left(Q^{\pi_\theta}(s,a) - Q_\omega(s,a)\right)\nabla_\theta \log \pi_\theta(s,a)\right] &= 0
\end{aligned}$$

Thus

$$\begin{aligned}
\nabla_\theta J(\theta) &= \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a) Q^{\pi_\theta}(s,a)\right] \\
&= \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a) Q_\omega(s,a)\right]
\end{aligned}$$

Compatible value function approximation III

In practice

- $\nabla_\omega Q_\omega = \nabla_\theta \log \pi_\theta$ only holds for exponential policies (almost never used in practice)
- $\omega^* = \text{argmin}_\omega \, \mathbb{E}_{\pi_\theta} \left[\left(Q^{\pi_\theta}(s,a) - Q_\omega(s,a) \right)^2 \right]$ is generally not true neither as we don't use through gradient descent on residals in online settings and batch methods are not convenient
- Most DeepRL methods for PG do not meet these assumptions, but they work in practice

Actor-Critic Algorithm

Algorithm 2 QAC with linear critic

$Q_\omega(s,a) = \omega^\top \phi(s,a)$
Initialize θ and ω as random
Set α, β
Initialise s
Sample $a \sim \pi_\theta(s,.)$
for all steps **do**
  Sample $r(s,a)$ and $s' \sim p(.|a,s)$
  Sample $a' = \pi_\theta(s',.)$
  $\omega \leftarrow \omega + \beta[r(s,a) + \gamma Q_\omega(s',a') - Q_\omega(s,a)]\phi(s,a)$
  $\theta \leftarrow \theta + \alpha \nabla_\theta \log \pi_\theta(s,a) Q_\omega(s,a)$
  $a \leftarrow a', s \leftarrow s'$
end for
return θ

Back to **Actor** policy:

$$\nabla J_H(\theta) \quad = \quad \mathbb{E}_{\pi_\theta}\left[\sum_{t=1}^{H} \nabla_\theta \log \pi_\theta(a_t|s_t) R_t(\tau)\right], \quad R_t(\tau) = \sum_{t'=t}^{H} r_{t'}$$

- Idea 1: $R_t(\tau)$ is reward from trajectory, MC proxy for **value**. Looks like $Q_{\pi_\theta}(s_t, a_t)$.
- Idea 2: Can shift by any b such that $\mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t) b(s_t)\right] = 0$.

Approximate **Advantage** function $A(s_t, a_t) = R_t(\tau) - b(s_t)$ or $A(s_t, a_t) = Q(s_t, a_t) - V(s_t)$.

Reducing variance with a baseline

Advantage function

- Same intuition as before, we shoud rather compare to average performance than measure absolute performance to compute the gradient.
- Average performance of π_θ starting from state s is $V^{\pi_\theta}(s)$
- Advantage function: $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$

Advantage actor-critic

$$Q^{\pi_\theta}(s,a) \approx Q_\omega(s,a) \quad V^{\pi_\theta}(s) \approx V_\psi(s)$$

$$A_{\omega,\psi}(s,a) = Q_\omega(s,a) - V_\psi(s)$$

$$\nabla J(\theta) \approx \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) A_{\omega,\psi}(s,a)]$$

Estimating the Advantage function

Using the TD error

- TD error: $\delta^{\pi_\theta}(s,a) = r(s,a) + \gamma V^{\pi_\theta}(s') - V^{\pi_\theta}(s)$

$$\begin{aligned}\mathbb{E}_{\pi_\theta}[\delta^{\pi_\theta}|s,a] &= \mathbb{E}_{\pi_\theta}[r(s,a) + \gamma V^{\pi_\theta}(s')|s,a] - V^{\pi_\theta}(s) \\ &= Q^{\pi_\theta}(s,a) - V^{\pi_\theta}(s) \\ &= A^{\pi_\theta}(s,a)\end{aligned}$$

- With approximation: $\delta_\psi(s,a) = r(s,a) + \gamma V_\psi(s') - V_\psi(s)$
- Policy gradient: $\nabla_\theta J(\theta) \approx \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a)\delta_\psi(s,a)]$
- It only depends on θ and ψ parameters (no ω)

1. Sample s_t, a_t from policy π_θ
2. $A_{\pi_\theta}(s_t, a_t) = r_t + V_{\pi_\theta}(s_{t+1}) - V_{\pi_\theta}(s_t)$
3. Use $\nabla J(\theta) \simeq \sum_{t=1}^{H} \nabla_\theta \log \pi_\theta(a_t|s_t) A_{\pi_\theta}(s_t, a_t)$
4. Update $\theta = \theta + \alpha \nabla J(\theta)$

- Uses **progressive** MC using TD instead of cumulative reward (cf. REINFORCE).
- Uses **value network** on top of **policy network**
- Step 3. **does not** take into account estimation error.

- **Soft policy iteration** introduces **regularization** of policy:

$$V_\pi^{(\alpha)}(s_t) = \mathbb{E}_{a_t \sim \pi}[Q_\pi(s_t, a_t) - \alpha \log \pi(\mathbf{a_t}|\mathbf{s_t})]$$

It considers a soft Bellman operator

$$\mathcal{T}_\pi^{(\alpha)}[Q] = r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1}}\Big[\mathbb{E}_{a_t \sim \pi}[Q(s_t, a_t) - \alpha \log \pi(\mathbf{a_t}|\mathbf{s_t})]\Big]$$

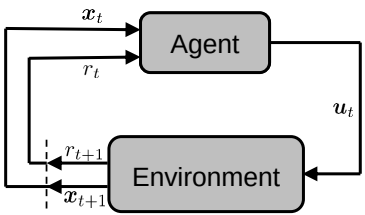
- We can derive actor-critic strategy using such soft operators.
- We can update α using gradient of

$$J(\alpha) = \mathbb{E}_{a_t \sim \pi}[-\alpha \log \pi(a_t|s_t) - \alpha \mathcal{H}\Big]$$

where $\mathcal{H}$ is a given entropy **threshold**.
More details e.g. `https://arxiv.org/pdf/1812.05905.pdf`

Table of contents

Reinforcement Learning (RL)



policy

$\pi(\boldsymbol{u}|\boldsymbol{x}),\ \boldsymbol{u} = \pi(\boldsymbol{x})$

Learning: **Adapting the policy** $\pi(u|x)$ **of the agent**

Objective: Find policy that maximizes expected return

$$J_\pi = \mathbb{E}\left[\sum_{t=0}^{T} r_t \Big| \boldsymbol{u}_t \sim \pi(\cdot|\boldsymbol{x}_t)\right]$$

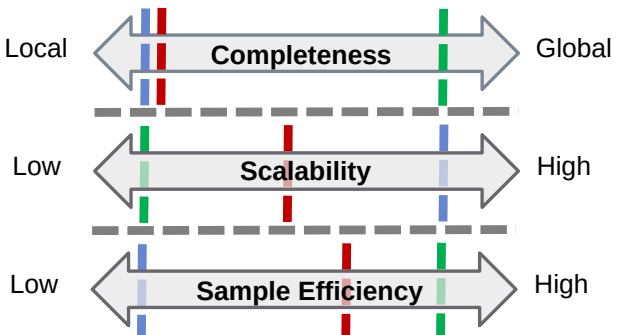
Learning **with movement primitives**

Objective: Find parameters $\boldsymbol{\theta}$ that maximize the expected return

$$R(\boldsymbol{\theta}) = \mathbb{E}\left[\sum_{t=0}^{T} r_t \Big| \boldsymbol{u}_t = \pi_{\boldsymbol{\theta}}(\boldsymbol{x}_t)\right]$$

Coarse Categorization of RL

- Deep Reinforcement Learning
- Bayesian Optimization
- Direct Policy Search / Stochastic Search



Direct Policy Search

Direct Policy Search by Stochastic Search:

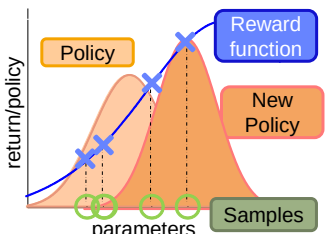
$$\pi^* = \arg\max_{\pi} \int \pi(\boldsymbol{\theta}) R(\boldsymbol{\theta}) d\boldsymbol{\theta}$$

Parameters of Movement Primitive

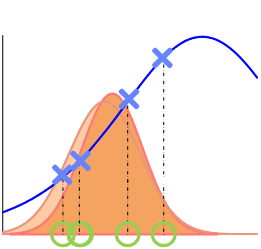
- Find distribution $\pi(\boldsymbol{\theta})$ with maximum expected return $R(\boldsymbol{\theta})$

Three basic steps:

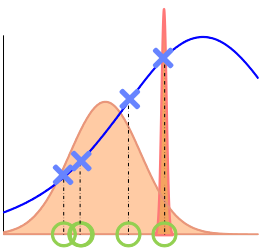
- **Explore:** Generate trajectories $\boldsymbol{\tau}^{[i]}$ following the policy π_k
- **Evaluate:** Assess quality of trajectory or actions
- **Update:** Compute new policy π_{k+1}



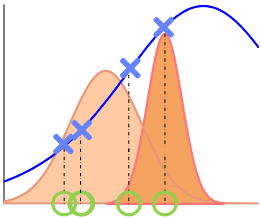
How to find a good policy update?



Too moderate



Too greedy



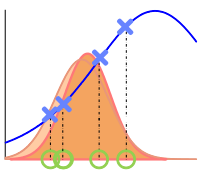
About right

Control exploration-exploitation tradeoff: immediate vs. long-term performance

Information-Theoretic Policy Update

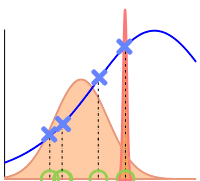
Information-theoretic policy update: incorporate information from new samples

1. Maximize return $\arg\max_{\pi} \int \pi(\boldsymbol{\theta}) R(\boldsymbol{\theta}) d\boldsymbol{\theta}$
2. Bound information gain [Peters 2011] $\text{s.t.} \quad \mathrm{KL}\left(\pi || \pi_{\text{old}}\right) \leq \epsilon$ **Control Greediness**
3. Bound entropy loss [Abdolmaleki 2015] $\underbrace{H(\pi_{\text{old}}) - H(\pi)}_{\text{loss in entropy}} \leq \gamma$ **Control Entropy**



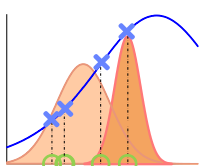
Too moderate

$\epsilon \ll 1$



Too greedy

$\epsilon \gg 1 \quad \gamma \gg 1$



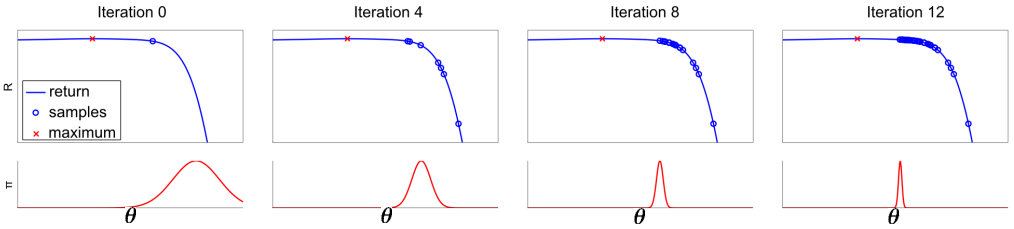
About right

$\epsilon \gg 1 \quad \gamma \ll 1$

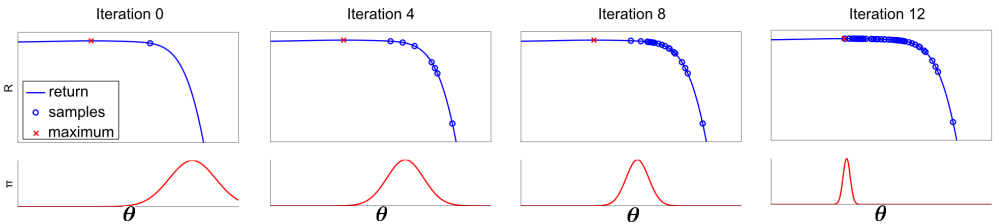
A. Abdolmaleki, …, **G. Neumann**, *Model-Based Relative Entropy Stochastic Search,* NIPS 2015

Illustration: Distribution Update

No entropy loss bound



With bounded entropy loss



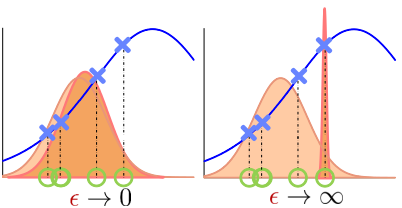
Solution for Search Distribution

Solution for unconstrained distribution: $\pi(\boldsymbol{\theta}) \propto \pi_{\text{old}}(\boldsymbol{\theta})^{\frac{\eta}{\eta+\omega}} \exp\left(\frac{R(\boldsymbol{\theta})}{\eta+\omega}\right)$

- η … Lagrangian multiplier for: $\mathrm{KL}(\pi||\pi_{\text{old}}) \leq \epsilon$

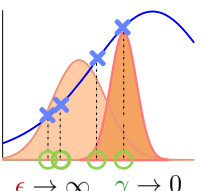
  $\epsilon \to 0$ ➡ $\eta \to \infty$ ➡ $\pi \to \pi_{\text{old}}$

  $\epsilon \to \infty$ ➡ $\eta \to 0$ ➡ $\pi \to$ greedy



- ω … Lagrangian multiplier for: $H(\pi_{\text{old}}) - H(\pi) \leq \gamma$

  $\gamma \to 0$ ➡ $\omega \gg 0$ ➡ $\pi \to$ more uniform



Gaussianity needs to be „enforced“ !

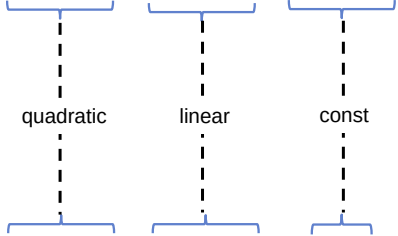
- Fit **new policy** on samples (REPS, [Daniel2012, Kupcsik2014, Neumann2014])
- Fit **return function** using compatible function approximation (MORE, [Abdolmaleki2015])

A. Abdolmaleki, …, **G. Neumann**, *Model-Based Relative Entropy Stochastic Search,* NIPS 2015

Fit Return Function

Use compatible function approximation:

- Gaussian distribution: $\mathcal{N}[\boldsymbol{\theta}|\boldsymbol{m}, \boldsymbol{\Lambda}] \propto \exp\left(-\frac{1}{2}\boldsymbol{\theta}^T\boldsymbol{\Lambda}\boldsymbol{\theta} + \boldsymbol{\theta}^T\boldsymbol{m} + \text{const}\right)$



- Gaussian in cannonical form (log linear)
- **Parameters:** Precision $\boldsymbol{\Lambda}$ and linear part $\boldsymbol{m}$

Match functional form: $\tilde{R}(\boldsymbol{\theta}) = \boldsymbol{\theta}^T\boldsymbol{A}\boldsymbol{\theta} + \boldsymbol{a}^T\boldsymbol{\theta} + a_0 \approx R(\boldsymbol{\theta})$

- Quadratic in $\boldsymbol{\theta}$, but linear in parameters: $\boldsymbol{w} = \{\boldsymbol{A}, \boldsymbol{a}, a_0\}$
- $\boldsymbol{w}$ obtained by **linear regression** on current set of samples
- Fits **local curvature** of the objective function

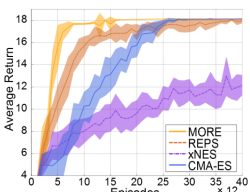
A. Abdolmaleki, …, **G. Neumann**, *Model-Based Relative Entropy Stochastic Search,* NIPS 2015

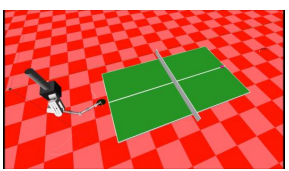
MORE Algorithm

Model-Based Relative Entropy Stochastic Search (MORE) : [Abdolmaleki 2015]

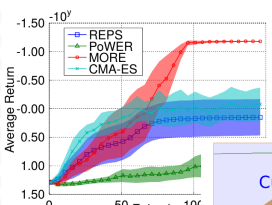
1. **Evaluation:** Fit local surrogate $\tilde{R}(\boldsymbol{\theta}) \approx \boldsymbol{\theta}^T \boldsymbol{A}\boldsymbol{\theta} + \boldsymbol{a}^T\boldsymbol{\theta} + a_0$
2. **Update:** $\pi(\boldsymbol{\theta}) \propto \pi_{\text{old}}(\boldsymbol{\theta})^{\frac{\eta}{\eta+\omega}} \exp\left(\frac{\tilde{R}(\boldsymbol{\theta})}{\eta+\omega}\right) \Rightarrow \pi(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*)$

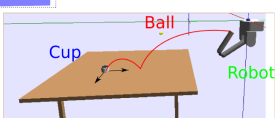
Table-Tennis





Beer-Pong





A. Abdolmaleki, …, **G. Neumann**, *Model-Based Relative Entropy Stochastic Search,* NIPS 2015

Adaptation of Skills

Adapt parameters to context s

- Continuous valued vector
- Characterizes environment / objectives

Learn contextual distribution:

$$\pi(\boldsymbol{\theta}|\boldsymbol{s}) = \mathcal{N}\left(\boldsymbol{\theta}|M\phi(\boldsymbol{s}), \boldsymbol{\Sigma}\right)$$

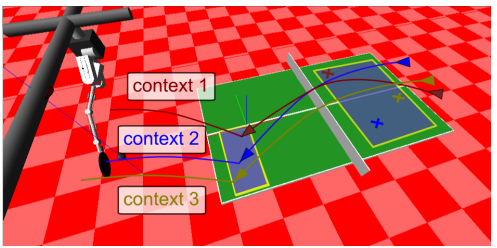
- Parameters are adapted to context

Leads to equivalent formulation:

$$\arg\max_{\pi} \mathbb{E}_{p(\boldsymbol{s})}\left[\int \pi(\boldsymbol{\theta}|\boldsymbol{s}) R(\boldsymbol{s}, \boldsymbol{\theta}) d\boldsymbol{\theta}\right]$$

$$\text{s.t.:} \quad \mathbb{E}_{p(\boldsymbol{s})}\left[\mathrm{KL}\left(\pi(\cdot|\boldsymbol{s})||\pi_{\mathrm{old}}(\cdot|\boldsymbol{s})\right)\right] \leq \epsilon$$

$$H(\pi_{\mathrm{old}}) - H(\pi) \leq \gamma$$



Abdolmaleki, …, **Neumann**, *Model-Based Relative Entropy Stochastic Search*, NIPS 2015

Application to Deep RL

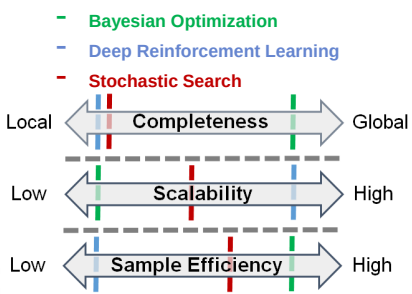
Deep Policies: $\pi(\boldsymbol{a}|\boldsymbol{s}) \propto \exp(\boldsymbol{\phi}_{\boldsymbol{\beta}}(\boldsymbol{s},\boldsymbol{a})^T\boldsymbol{\theta})$

- $\boldsymbol{\phi}_{\boldsymbol{\beta}}(\boldsymbol{s},\boldsymbol{a})$ … neural network with params $\boldsymbol{\beta}$
- $\boldsymbol{\theta}$ … log-linear parameters

Objective:

$$\pi^*_{\text{new}} = \text{argmax}_\pi \mathbb{E}_{\mu_{\pi_{\text{old}}}(\boldsymbol{s})}\left[\int \pi(\boldsymbol{a}|\boldsymbol{s}) A^{\pi_{\text{old}}}(\boldsymbol{s},\boldsymbol{a}) d\boldsymbol{a}\right]$$

- where $A^{\pi_{\text{old}}}(\boldsymbol{s},\boldsymbol{a})$ is the advantage function of the old policy



Some SOTA algorithms use similar trust regions (but use approximations to solve it)

- TRPO[Schulman2015], PPO[Schulman2016], MPO[Abdolmaleki2018]

Some SOTA algorithms use similar entropy regularization (not a bound)

- SAC[Harnoja2017], AC3

Can we unify this using our framework?

Compatible Policy Search

Solution (for log linear case):

$$\pi(\boldsymbol{a}|\boldsymbol{s}) \propto \pi_{\text{old}}(\boldsymbol{a}|\boldsymbol{s})^{\frac{\eta}{\eta+\omega}} \exp\left(\frac{A(\boldsymbol{s},\boldsymbol{a})}{\eta+\omega}\right)$$

1. Move π_{old} into exp:

$$\pi(\boldsymbol{a}|\boldsymbol{s}) \propto \exp\left(\frac{\eta \log \pi_{\text{old}}(\boldsymbol{a}|\boldsymbol{s}) + A(\boldsymbol{s},\boldsymbol{a})}{\eta+\omega}\right)$$

2. Log Linear Model:

$$\pi(\boldsymbol{a}|\boldsymbol{s}) \propto \exp\left(\frac{\eta \boldsymbol{\phi}_{\boldsymbol{\beta}}(\boldsymbol{s},\boldsymbol{a})^T \boldsymbol{\theta}_{\text{old}} + A(\boldsymbol{s},\boldsymbol{a})}{\eta+\omega}\right)$$

3. Compatible approximation:

$$\pi(\boldsymbol{a}|\boldsymbol{s}) \propto \exp\left(\boldsymbol{\phi}_{\boldsymbol{\beta}}(\boldsymbol{s},\boldsymbol{a})^T \frac{\eta \boldsymbol{\theta}_{\text{old}} + \boldsymbol{w}_A}{\eta+\omega}\right)$$

Log-Linear Models:

$$\log \pi_{\text{old}}(\boldsymbol{a}|\boldsymbol{s}) = \boldsymbol{\phi}_{\beta}(\boldsymbol{s},\boldsymbol{a})^T \boldsymbol{\theta}_{\text{old}} - \log Z$$

Compatible Function Approximation

$$A(\boldsymbol{s},\boldsymbol{a}) \approx \boldsymbol{\phi}_{\beta}(\boldsymbol{s},\boldsymbol{a})^T \boldsymbol{w}_A$$

- For **Gaussians:** all **quadratic**, **linear** and **constant** terms

Compatible Policy Update:

$$\boldsymbol{\theta}_{\text{new}} = \frac{\eta \boldsymbol{\theta}_{\text{old}} + \boldsymbol{w}_A}{\eta+\omega}$$

- Can also be obtained for non-linear term using Taylor expansion
- Equivalent to natural gradients if omega = 0 (no entropy bound)

J. Pajarinen, L. Hong Thai, R. Arkour, J. Peters, G. Neumann, *Compatible Natural Gradient Policy Search, Machine Learning, 2019*

Table of contents

Let $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R)$ be an MDP.
For any policy $\pi : \mathcal{S} \to \mathcal{P}(\mathcal{A})$, let

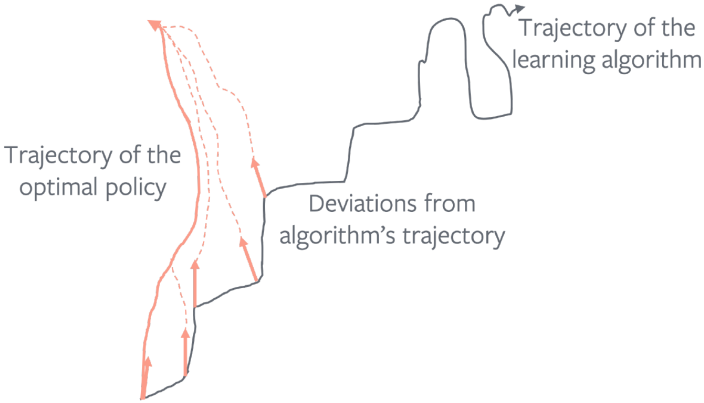
- $\mu_\pi : s \mapsto \mathbb{E}_{A\sim\pi(s)}[\mu(s, A)]$ where $\mu(s, a)$ denotes the mean of distribution $R(s, a)$,
- $P_\pi : f \mapsto \Big(P_\pi f : s \mapsto \sum_{s'\in\mathcal{S}} \mathbb{E}_{A\sim\pi(s)}[P(s'|s, A)] f(s') \Big)$.

Expected cumulative reward

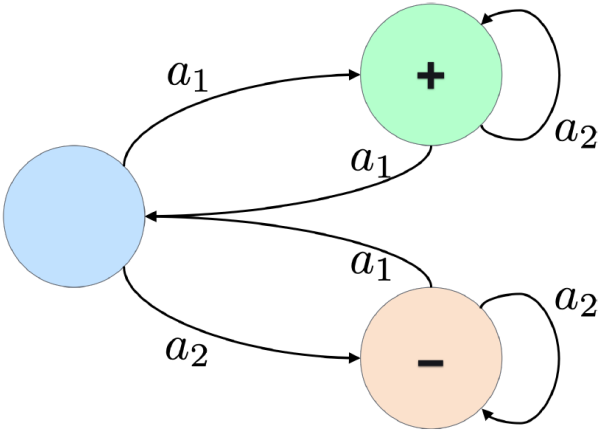
The expected cumulative reward of policy π when run for T steps from state s_1 is

$$R_{\pi,T}(s_1) = \mathbb{E}\Big[\sum_{t=1}^{T} r(s_t, a_t)\Big] = \mu_\pi(s_1) + (P_\pi\mu_\pi)(s_1) + \cdots = \sum_{t=1}^{T} (P_\pi^{t-1}\mu_\pi)(s_1)\,,$$

where $a_t \sim \pi(s_t)$, $s_{t+1} \sim P(\cdot|s_t, a_t)$, and $r(s, a) \sim R(s, a)$ with mean $\mu(s, a)$.

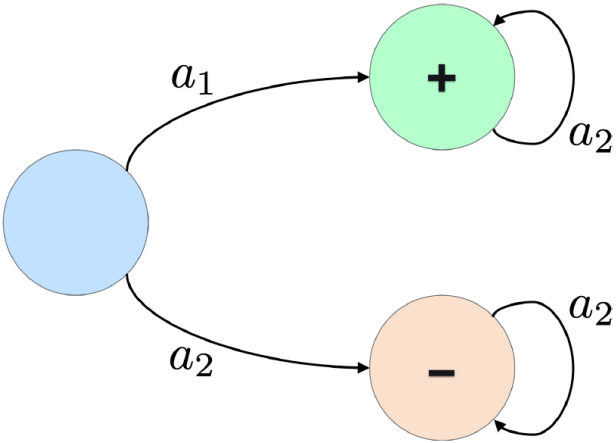


(courtesy: A. Lazaric)



- Sample-complexity: Easy
- Regret minimization: Easy

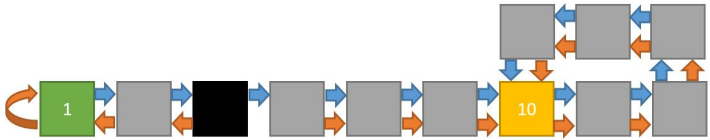
(courtesy: A. Lazaric)



- Sample-complexity: **Trivial**
- Regret minimization: Impossible

(courtesy: A. Lazaric)

- We want **uniformly horizon-optimal** strategies (optimal for each T). Possible ?
- Example of a deterministic MDP with $\mathcal{A} = \{a, b\} = \{ \leftarrow, \rightarrow \}$:



The set of optimal sequences of actions can be made explicit for all horizon T:

T	1	2	3	$4, \dots, 7$	8	9	$10, \dots, 13$	14	15
$\star$	$\mathcal{A}$	a^2	a^3	$b\mathcal{A}^{T-1}$	$a^2b^3\mathcal{A}^{T-5}$	$a^3b^3\mathcal{A}^{T-6}$	$b\mathcal{A}^{T-1}$	$a^2b^3\mathcal{A}^{T-5}$	$a^3b^3\mathcal{A}^{T-6}$
R_T	0	1	2	10	11	12	20	21	22

- **No sequence** (hence policy) can be made optimal simultaneously for all time T, or even for all large enough T.
- Give up on optimality ? Restrict MDPs ? Modify optimality criterion?

Given an MDP M, every policy π induces a Markov chain on $\mathcal{S}$.
An MDP M is **communicating** if every pair of state belongs to at least one communicating class under some policy, that is

$$\forall s, s' \in \mathcal{S}, \exists \pi, \exists t < \infty : P_\pi(s_t = s' | s_1 = s) > 0.$$

An MDP M is **irreducible** if every policy π on M induces an irreducible Markov chain, that is

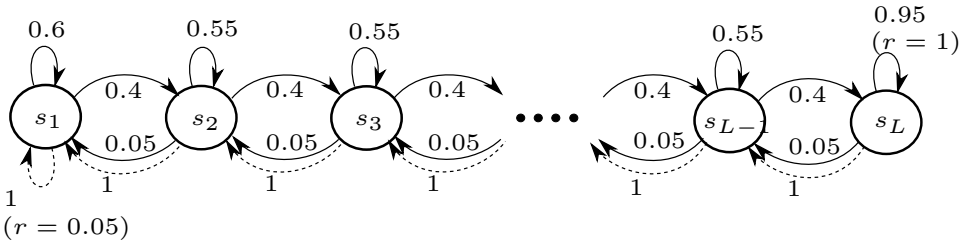
$$\forall \pi, \forall s, s' \in \mathcal{S}, \exists t < \infty : P_\pi(s_t = s' | s_1 = s) > 0.$$

An MDP M is **ergodic** if every policy π on M induces an ergodic Markov chain, that is $\forall \pi$,

$$\begin{aligned}
\text{(irreducibility)} \quad & \forall s, s' \in \mathcal{S}, \exists t < \infty, \ P_\pi^t(s'|s) > 0, \\
\text{(aperiodicity)} \quad & \forall s \in \mathcal{S}, \ \mathrm{GCD}(\{t : P_\pi^t(s,s) > 0\}) = 1, \\
\text{(positive recurrence)} \quad & \forall s \in \mathcal{S}, \ \mathbb{E}(\min\{t > 1 : s_t = s\} | s_1 = s]) < \infty .
\end{aligned}$$

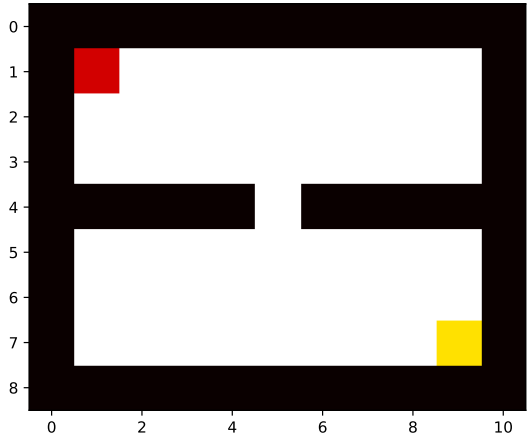
NB: most people use "ergodic" terminology in lieu of "irreducible" MDPs.

▷ $\mathcal{S} = \{1, 2, 3, 4, 5, 6\}$, $\mathcal{A} = \{\text{left,right}\}$, $p = 0.4$ to reach right when go right.



▷ **Communicating**, but not irreducible: policy going left is not irreducible (e.g. not possible to reach state 4 starting from state 3)

▷ $\mathcal{S} = \{$ positions in the maze $\}$, $\mathcal{A} = \{$up,down,left,right$\}$, get reward 1 in yellow state, 0 else. Proba $p = 0.9$ to reach desired state.



▷ **Communicating**, but not irreducible.

- A finite, communicating MDP has finite "**diameter**":

$$D_M = \max_{s,s' \in \mathcal{S}} \min_{\pi} \mathbb{E}[\min\{t > 1 : s_t = s'\} | s_1 = s]$$

that is, the **shortest path** (path with minimal expected length) between any two states is finite.

- In grid-world type MDPs:

$$D \simeq \frac{\text{length of shortest path between maximally distant states}}{\text{probability of reaching desired next state}}.$$

E.g. River-swim $D \simeq L/p$. Two-room MDP: $D \simeq 15/p$

- "Possibility to reach an optimal behavior after taking a **bad** action",

Example with finite horizon, let V_T denote optimal value function with horizon T, then

$$\begin{aligned} \text{(Quality)} \quad Q_T(s,a) &= r(s,a) + \sum_{s' \in \mathcal{S}} P(s'|s,a) V_{T-1}(s') \,. \\ \text{(Advantage)} \quad A_T(s,a) &= V_T(s) - Q_T(s,a) \,. \end{aligned}$$

- An MDP is **recoverable** (alternative def. with discount exist) if

$$\forall s \in \mathcal{S}, \forall a \in \mathcal{A}, \quad \limsup_{T \to \infty} \frac{A_T(s,a)}{T} = 0 \,.$$

- Diameter takes into account P only.
- Recoverability takes into account both P and R.

▷ The *expected cumulative reward* of policy π when run for T steps from initial state s_1 is defined as

$$R_{\pi,T}(s_1) = \mathbb{E}\left[\sum_{t=1}^{T} r(s_t, a_t)\right] = \mu_\pi(s_1) + (P_\pi \mu_\pi)(s_1) + \ldots = \sum_{t=1}^{T} (P_\pi^{t-1}\mu_\pi)(s_1)\,.$$

where $a_s \sim \pi(s_t)$, $s_{t+1} \sim P(\cdot|s_t, a_t)$, $r(s,a) \sim \nu(s,a)$ with mean $\mu(s,a)$.

▷ Different **optimality** criterion:

$$\text{(Total reward)} \quad \star_T \in \operatorname*{Argmax}_{\pi:\mathcal{S}\to\mathcal{A}} \sum_{t=1}^{T} P_\pi^{t-1}\mu_\pi$$

$$\text{(Discounted reward)} \quad \star_\gamma \in \operatorname*{Argmax}_{\pi:\mathcal{S}\to\mathcal{A}} \sum_{t=1}^{\infty} \gamma^{t-1} P_\pi^{t-1}\mu_\pi$$

$$\text{(Average reward)} \quad \star \in \operatorname*{Argmax}_{\pi:\mathcal{S}\to\mathcal{A}} \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} P_\pi^{t-1}\mu_\pi\,.$$

▷ Corresponding notions of regret: $\mathbf{R}_{\pi,T}, \mathbf{R}_{\pi,\gamma}, \mathbf{R}_\pi$

Definition (Average gain)

The **average transition operator** of policy π is $\overline{P}_\pi = \lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} P_\pi^{t-1}$.
The *average gain* g_π is defined by

$$g_\pi(s_1) \quad = \quad \lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} (P_\pi^{t-1}\mu_\pi)(s_1) = (\overline{P}_\pi \mu_\pi)(s_1)\,.$$

An optimal policy $\star$ satisfies $g_\star = \max_\pi g_\pi$.

- (Invariance) $P_\pi g_\pi = g_\pi$, thanks to the useful key property:

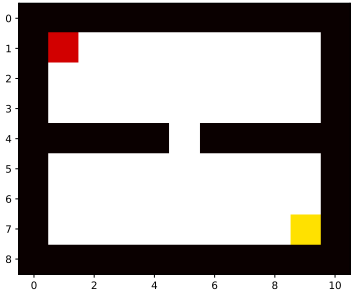
$$\overline{P}_\pi P_\pi = P_\pi \overline{P}_\pi = \overline{P}_\pi \overline{P}_\pi = \overline{P}_\pi\,.$$

- The gain is a **constant** function in a communicating MDP.

We define the Bias function as

$$b_\pi \quad = \quad \sum_{t=1}^{\infty} (P_\pi^{t-1} - \overline{P}_\pi)\mu_\pi \,,$$

▷ Say red state is absorbing and gives reward 1 (all others give 0).



▷ All policies reaching red state satisfy $\overline{P}_\pi \mu_\pi = g_\pi = 1$ and are optimal. Bias function captures how long it takes to reach it.

We define the Bias function as

$$b_\pi = \sum_{t=1}^{\infty} (P_\pi^{t-1} - \overline{P}_\pi)\mu_\pi \,,$$

- (Direct computation) $b_\pi = [I - P_\pi + \overline{P_\pi}]^{-1}[I - \overline{P}_\pi]\mu_\pi$

We define the Bias function as

$$b_\pi = \sum_{t=1}^{\infty} (P_\pi^{t-1} - \overline{P}_\pi)\mu_\pi \,,$$

- (Direct computation) $b_\pi = [I - P_\pi + \overline{P_\pi}]^{-1}[I - \overline{P}_\pi]\mu_\pi$

Lemma (Bias and Gain)

(Poisson equation) $b_\pi + g_\pi = \mu_\pi + P_\pi b_\pi$ ($\simeq$ Bellman equation)

- Almost Bellman equation for b_π, except for additional g_π.
- Unlike Bellman equation, Poisson equation admits **many** solutions:

if b_π then $b_\pi + c\mathbf{1}$ is also solution for all constant c.

Let $A(s,\pi) = \mu_\star(s) + (P_\star b_\star)(s) - \mu_\pi(s) - (P_\pi b_\star)(s)$ be the **advantage function**.

Pseudo-regret

Consider a **communicating** MDP. Let π be a policy run for T time steps. Then,

$$\mathbf{R}_{\pi,T} = \underbrace{\sum_{t=1}^{T}(P_\star^{t-1} - \overline{P}_\star)\mu_\star + (P_\pi^T - I)b_\star}_{(A)} + \underbrace{\sum_{t=1}^{T}(1 - P_\pi^{t-1})(g_\star - g_\pi)}_{(B)} + \sum_{s,a}\mathbb{E}[N_T^\pi(s,a)]A(s,a)\,.$$

- As $T \to \infty$, $(A) \to \overline{P}_\pi b^\star$ is a finite constant.
- If π is such that $g_\star - g_\pi \leqslant \varepsilon$, then $(B) \leqslant \varepsilon T$.

$$\mathbf{R}_{\pi,T} \leqslant \sum_{s,a}\mathbb{E}[N_T^\pi(s,a)]A(s,a) + \varepsilon T + (A)$$

- The pseudo-regret $\sum_{s,a}\mathbb{E}[N_T^\pi(s,a)]A(s,a)$ is reminiscent of **bandits**.

Important to note that:

$$\sum_{t=1}^{T} P_\pi^{t-1} A_\pi = \sum_{s,a} \mathbb{E}[N_T(s,a)] A(s,a)\,,$$

which we naturally call the pseudo-regret, by analogy with the bandit setting. Indeed,

$$\begin{aligned}\sum_{t=1}^{T} P_\pi^{t-1} A_\pi &= \sum_{t=1}^{T} \mathbb{E}_{s_{t-1}}[A_\pi(s_{t-1})] = \sum_{s,a} A(s,a) \sum_{t=1}^{T} \mathbb{E}_{s_{t-1}}[\mathbb{I}\{s_{t-1}=s, \pi(s)=a\}] \\ &= \sum_{s,a} A(s,a) \mathbb{E}[N_T^\pi(s,a)]\,,\end{aligned}$$

How do we compute π such that $g_\star - g_\pi \leqslant \varepsilon$?

How do we compute π such that $g_\star - g_\pi \leqslant \varepsilon$?

▷ **Value iteration** computes a sequence of functions $(u_n)_{n\in\mathbb{N}}$ and policies $(\pi_n)_{n\in\mathbb{N}}$ according to the following equations

$$\forall n \in \mathbb{N} \begin{cases} u_{n+1}(s) = \max_{a\in\mathcal{A}} \mu(s,a) + (P_a u_n)(s)\,, & \text{where } u_0 = 0 \\ \pi_{n+1}(s) \in \operatorname{Argmax}_{a\in\mathcal{A}} \mu(s,a) + (P_a u_n)(s)\,. \end{cases}$$

π_{n+1} is called a u_n-improving policy (Greedy w.r.t. u_n).

$$u_n = \sum_{i=1}^{n} P_{\pi_n,\dots,\pi_{i+1}} \mu_{\pi_i}$$

How do we compute π such that $g_\star - g_\pi \leqslant \varepsilon$?

▷ **Value iteration** computes a sequence of functions $(u_n)_{n\in\mathbb{N}}$ and policies $(\pi_n)_{n\in\mathbb{N}}$ according to the following equations

$$\forall n \in \mathbb{N} \begin{cases} u_{n+1}(s) = \max_{a\in\mathcal{A}} \mu(s,a) + (P_a u_n)(s)\,, & \text{where } u_0 = 0 \\ \pi_{n+1}(s) \in \operatorname{Argmax}_{a\in\mathcal{A}} \mu(s,a) + (P_a u_n)(s)\,. \end{cases}$$

π_{n+1} is called a u_n-improving policy (Greedy w.r.t. u_n).

$$u_n \;=\; \sum_{i=1}^{n} P_{\pi_n,\dots,\pi_{i+1}} \mu_{\pi_i}$$

▷ Average-gain guarantee? When to stop?

▷ **No contraction** of the Bellman operator in the usual sense: Do we have any form of convergence?
The **span** operator is

$$\mathbb{S}(f) = \max_x f(x) - \min_x f(x)$$

▷ It satisfies $\mathbb{S}(f + c\mathbf{1}) = \mathbb{S}(f)$ for any constant c.
▷ It is a semi-norm.
We will soon see that in some cases, applying Bellman operator may yield a contraction of the span (instead of the $\|\cdot\|_\infty$ norm)

Regret and span

Let $\varepsilon > 0$ and n be such that $g_\star - g_{\pi_{n+1}} \leqslant \varepsilon$. Then

$$\begin{aligned}\mathbf{R}_{\pi_{n+1},T}(s_1) &= R_{\star,T}(s_1) - R_{\pi_{n+1},T}(s_1) \\ &\leqslant \sum_{t=1}^{T}((P_\star^{t-1} - \overline{P}_\star)\mu_\star)(s_1) + \varepsilon T + \mathbb{S}(\mathbf{b}_{\pi_{n+1}})\,.\end{aligned}$$

Regret and span

Let $\varepsilon > 0$ and n be such that $g_\star - g_{\pi_{n+1}} \leqslant \varepsilon$. Then

$$\begin{aligned} \mathbf{R}_{\pi_{n+1},T}(s_1) &= R_{\star,T}(s_1) - R_{\pi_{n+1},T}(s_1) \\ &\leqslant \sum_{t=1}^{T} ((P_\star^{t-1} - \overline{P}_\star)\mu_\star)(s_1) + \varepsilon T + \mathbb{S}(\mathbf{b}_{\pi_{n+1}}) \,. \end{aligned}$$

- The first sum depends on the mixing time of the chain induced by the policy $\star$ in the MDP, but not on π_{n+1}.
- **Effective** regret: $\mathbf{R}^{\text{eff}}_{\pi_{n+1},T}(s_1) = Tg_\star - R_{\pi_{n+1},T}(s_1)$

$$\mathbf{R}^{\text{eff}}_{\pi_{n+1},T}(s_1) \leqslant \varepsilon T + \mathbb{S}(b_{\pi_{n+1}}).$$

- Controlling the **span of the bias** is enough to control the regret.

- We have shown that if $g_\star - g_{\pi_{n+1}} \leqslant \varepsilon$, then

$$Tg_\star - R_{\pi_{n+1},T}(s_1) \leqslant \varepsilon T + \mathbb{S}(b_{\pi_{n+1}}) .$$

- We have shown that if $g_\star - g_{\pi_{n+1}} \leqslant \varepsilon$, then

$$Tg_\star - R_{\pi_{n+1},T}(s_1) \leqslant \varepsilon T + \mathbb{S}(b_{\pi_{n+1}}) .$$

  - How to control $\mathbb{S}(b_{\pi_{n+1}})$?
  - How to control $g_\star - g_{\pi_{n+1}}$?

- We have shown that if $g_\star - g_{\pi_{n+1}} \leqslant \varepsilon$, then

$$Tg_\star - R_{\pi_{n+1},T}(s_1) \leqslant \varepsilon T + \mathbb{S}(b_{\pi_{n+1}}) .$$

  - How to control $\mathbb{S}(b_{\pi_{n+1}})$?
  - How to control $g_\star - g_{\pi_{n+1}}$?

Diameter of an MDP

The diameter D of a (discrete) MDP is defined as

$$D = \max_{s_1,s_2 \in \mathcal{S}} \min_{\pi \in \Pi} \mathbb{E}[\tau(s_1, s_2, \pi)] ,$$

where $\tau(s, s', \pi) = \min\{t : s_t = s', \text{ when } s_0 = s \text{ by following } \pi\}$ is the traveling time between s and s'.

Lemma (informal)

For all n (starting from $u_0 = 0$), if rewards are bounded by 1,

$$\mathbb{S}(u_n) \leqslant D, \qquad \mathbb{S}(b_{\pi_n}) \leqslant D\,.$$

Sketch:

Lemma (informal)

For all n (starting from $u_0 = 0$), if rewards are bounded by 1,

$$\mathbb{S}(u_n) \leqslant D, \qquad \mathbb{S}(b_{\pi_n}) \leqslant D\,.$$

Sketch:

- Otherwise $\exists s_1, s_2$ s.t. $u_n(s_1) - u_n(s_2) > D$, that is $u_n(s_2) < u_n(s_1) - D$.

Lemma (informal)

For all n (starting from $u_0 = 0$), if rewards are bounded by 1,

$$\mathbb{S}(u_n) \leqslant D, \qquad \mathbb{S}(b_{\pi_n}) \leqslant D\,.$$

Sketch:

- Otherwise $\exists s_1, s_2$ s.t. $u_n(s_1) - u_n(s_2) > D$, that is $u_n(s_2) < u_n(s_1) - D$.
- However, it takes **at most** D steps to reach s_2 from s_1, by some policy π_{Fast}. The rewards are bounded by 1, thus we loose **at most** D rewards by following π_{Fast} from s_2 to reach s_1 then continue.

Lemma (informal)

For all n (starting from $u_0 = 0$), if rewards are bounded by 1,

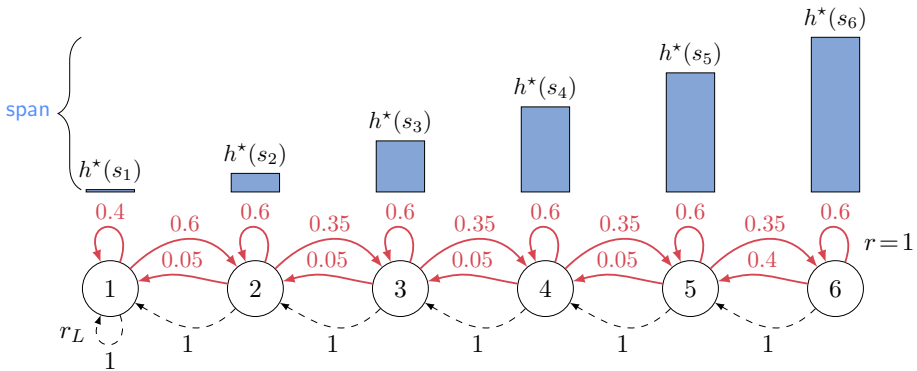
$$\mathbb{S}(u_n) \leqslant D, \qquad \mathbb{S}(b_{\pi_n}) \leqslant D\,.$$

Sketch:

- Otherwise $\exists s_1, s_2$ s.t. $u_n(s_1) - u_n(s_2) > D$, that is $u_n(s_2) < u_n(s_1) - D$.
- However, it takes **at most** D steps to reach s_2 from s_1, by some policy π_{Fast}. The rewards are bounded by 1, thus we loose **at most** D rewards by following π_{Fast} from s_2 to reach s_1 then continue.

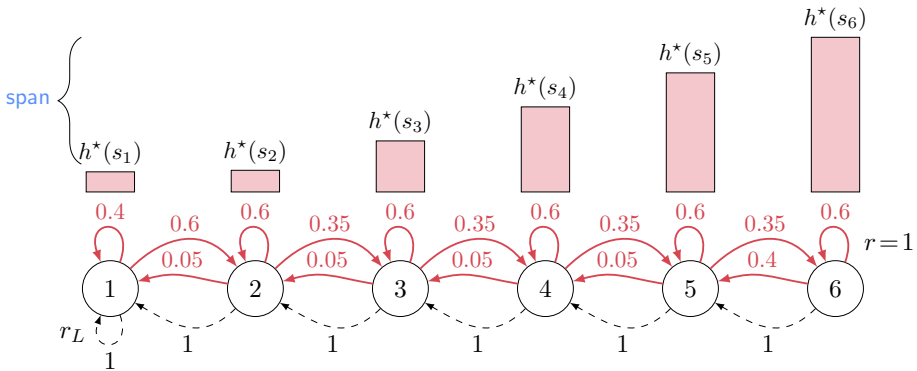
Thus the value of such a combined policy is **at least** $u_n(s_1) - D$, and $u_n(s_2)$ should be **larger** than this quantity by optimality.

River Swim: Optimality



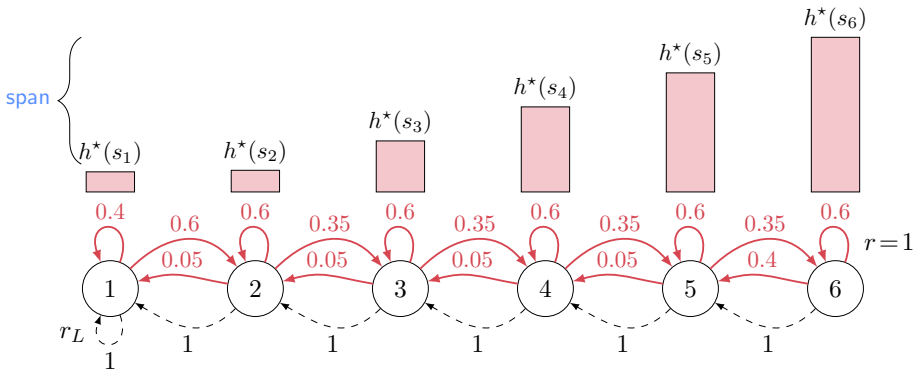
- $\pi^\star = \pi_R$
- If $r_L = 0.01$, $g^\star \approx 0.43$, $\mathsf{sp}(h^\star) \approx 6.4$

River Swim: Optimality



- $\pi^\star = \pi_R$
- If $r_L = 0.01$, $g^\star \approx 0.43$, $\mathsf{sp}(h^\star) \approx 6.4$
- If $r_L = 0.4$, $g^\star \approx 0.43$, $\mathsf{sp}(h^\star) \approx 5.5$

River Swim: Optimality



- $\pi^\star = \pi_R$
- If $r_L = 0.01$, $g^\star \approx 0.43$, $\mathsf{sp}(h^\star) \approx 6.4$
- If $r_L = 0.4$, $g^\star \approx 0.43$, $\mathsf{sp}(h^\star) \approx 5.5$

} D is constant

- **Optimal policy** in MDP M:

$$\star_M \in \operatorname*{argmax}_\pi g_M^\pi,$$

with corresponding optimal **gain** $g_M^\star$ and **bias** $b_M^\star$.

- Set of **optimal actions** in state s:

$$\mathcal{O}_M^\star(s) = \operatorname*{Argmax}_{a \in \mathcal{A}} \left\{ \mu(s,a) + (P_a b_M^\star)(s) \right\}$$

- Set of **confusing** MDPs for M and $(s,a) \in \mathcal{S} \times \mathcal{A}$:

$$\mathcal{C}_M(s,a) = \left\{ M' : p'(\cdot|s',a') = p(\cdot|s',a') \text{ for all } (s',a') \neq (s,a), \right.$$
$$\left. a \notin \mathcal{O}_M^\star(s), a \in \mathcal{O}_{M'(s)}^\star \right\}$$

Theorem [Burnetas and Katehakis, 1997; Ok at al. 2018]

Let alg be a **uniformly good** strategy on the set of MDPs with states $\mathcal{S}$, actions $\mathcal{A}$, and known reward function R. For any **ergodic** MDP, with maximal reward 1,

$$\liminf_{T\to\infty} \frac{\mathbf{R}^{\text{alg}}(T, M)}{\log(T)} \geqslant K_M$$

where $$K_M = \inf_{\eta\geqslant 0} \sum_{s,a} \eta(s,a) A_M(s,a)$$

$$\text{s.t} \sum_{s,a} \eta(s,a) \mathtt{KL}(p(\cdot|s,a), p'(\cdot|s,a)) \geqslant 1 \ \forall M' \in \mathcal{C}_M(s,a).$$

Further it holds $$K_M \leqslant 2SA \frac{(\mathbb{S}(\mathbf{b}^\star_{\mathbf{M}}) + \mathbf{1})^2}{\min_{\mathbf{s},\mathbf{a}} \mathbf{A_M(s,a)}}.$$

- In an ergodic MDP, every policy visits **infinitely often** every state.
- Open question for generic MDPs.

Minimax Lower Bound

Theorem (Thm. 5 Jaksch et al. [2010])

For any communicating MDP $M^\star$ with $r_{\max} = 1$, $S, A \geq 10$, $D \geq 20 \log_A S$, any algorithm $\mathfrak{A}$ at any time $T \geq DSA$ suffers a regret

$$\sup_{M^\star} \overline{R}(T, M^\star, \mathfrak{A}) \geq 0.015\sqrt{DSAT}$$

Minimax Lower Bound

Theorem (Thm. 5 Jaksch et al. [2010])

For any communicating MDP $M^\star$ with $r_{\max} = 1$, $S, A \geq 10$, $D \geq 20 \log_A S$, any algorithm $\mathfrak{A}$ at any time $T \geq DSA$ suffers a regret

$$\sup_{M^\star} \overline{R}(T, M^\star, \mathfrak{A}) \geq 0.015\sqrt{DSAT}$$

In MAB $\Omega(\sqrt{AT})$ since $D = 1$ and $S = 1$.

Open Questions

C could be arbitrarily large ($C = \infty$ for non ergodic)

1. *Asymptotic* regime and *ergodicity* assumption

$$\mathbb{P}_M^{\pi}\big[N_T(s) \geq \rho T\big] \geq 1 - \; C \; \exp(-\rho T/2)$$ [Prop.2 Burnetas and Katehakis [1997]]

2. *Span vs. diameter*

$D = 2\mathsf{sp}(h^\star)$ in the proof

$$\overline{R}(T, M^\star, \mathfrak{A}) \geq 0.015\sqrt{D \; SAT}$$

3. *Number of states vs branching factor* $\Gamma = \max_{s,a} |\mathsf{supp}(p(\cdot|s,a))|$

$$\overline{R}(T, M^\star, \mathfrak{A}) \geq 0.015\sqrt{D \; S \; AT}$$

$\Gamma = 2$ in the proof

▷ We considered **value iteration** and **regret** guarantees....
But under average-gain, there is no discount ($\gamma = 1.$)

- Where is the **contraction** property?
- Is value iteration **converging** at all (no discount) ?
- Where is contraction of Bellman operator $T_\pi[f] = \mu_\pi + P_\pi f$?

▷ We considered **value iteration** and **regret** guarantees....
But under average-gain, there is no discount ($\gamma = 1.$)

- Where is the **contraction** property?
- Is value iteration **converging** at all (no discount) ?
- Where is contraction of Bellman operator $T_\pi[f] = \mu_\pi + P_\pi f$?

We now show a **contraction** property w.r.t. to the Span operator (semi-norm), instead of the $\|\cdot\|_\infty$ norm.

- We use the span semi-norm $\mathbb{S}(f) = \max_x f(x) - \min_x f(x)$.

Lemma [Contraction in span semi-norm]

The Bellman operator has the following contraction property:

$$\forall f, g \in \mathbb{R}^S, \quad \mathbb{S}(T_\pi[f] - T_\pi[g]) \leqslant \frac{1}{2}\|P_\pi(\cdot|\bar{s}) - P_\pi(\cdot|\underline{s})\|_1 \mathbb{S}(f-g)$$

where $\bar{s} = \operatorname{argmax}_{s\in\mathcal{S}} P_\pi(f-g)(s)$, $\underline{s} = \operatorname{argmin}_{s\in\mathcal{S}} P_\pi(f-g)(s)$.

- **MDP-dependent** contraction.
- Possibly contracting even if no discount!

▷ $\frac{1}{2}\|P_\pi(\cdot|\overline{s}) - P_\pi(\cdot|\underline{s})\|_1 = 1 - \sum_{s' \in \mathcal{S}} \min(P_\pi(s'|\overline{s}), P_\pi(s'|\underline{s}))$ offers interpretation: The higher the probability of reaching same states from $\overline{s}$ and $\underline{s}$, the higher $\sum_{s' \in \mathcal{S}} \min(P_\pi(s'|\overline{s}), P_\pi(s'|\underline{s}))$, and the more contraction there is.

The higher the **probability of coalescing** from two different states, the more **contraction**.

Definition [Policy contraction coefficient]

We define the one step and multi-step **contraction coefficients** as:

$$\gamma_\pi = \max_{s_1,s_2} \frac{1}{2}\|P_\pi(\cdot|s_1) - P_\pi(\cdot|s_2)\|_1 = 1 - \min_{s_1,s_2} \sum_{s' \in \mathcal{S}} \min(P_\pi(s'|s_1), P_\pi(s'|s_2))$$

$$\forall k, \gamma_{\pi,k} = \max_{s_1,s_2} \frac{1}{2}\|P^k_\pi(\cdot|s_1) - P^k_\pi(\cdot|s_2)\|_1 = 1 - \min_{s_1,s_2} \sum_{s' \in \mathcal{S}} \min(P^k_\pi(s'|s_1), P^k_\pi(s'|s_2))$$

- In particular

$$\mathbb{S}(T^k_\pi[f] - T^k_\pi[g]) \leqslant \min_{\ell \in \{1,\dots,k\}} \gamma^{k/\ell}_{\pi,\ell} \mathbb{S}(f - g)$$

Value Iteration convergence

How to ensure $g_\star - g_{\pi_{n+1}}$ is controlled?

Lemma [Sandwich bounds]

$$\forall n \in \mathbb{N}, \quad \overline{P}_{\pi_{n+1}}[u_{n+1} - u_n] \leqslant g_{\pi_{n+1}} \leqslant g_\star \leqslant \overline{P}_\star[u_{n+1} - u_n] \,.$$

How to ensure $g_\star - g_{\pi_{n+1}}$ is controlled?

Lemma [Sandwich bounds]

$$\forall n \in \mathbb{N}, \quad \overline{P}_{\pi_{n+1}}[u_{n+1} - u_n] \leqslant g_{\pi_{n+1}} \leqslant g_\star \leqslant \overline{P}_\star[u_{n+1} - u_n]\,.$$

Corollary [Value and gain]

For each n, it holds $\quad g_\star - g_{\pi_{n+1}} \leqslant \mathbb{S}(u_{n+1} - u_n)$,

How to ensure $g_\star - g_{\pi_{n+1}}$ is controlled?

Lemma [Sandwich bounds]

$$\forall n \in \mathbb{N}, \quad \overline{P}_{\pi_{n+1}}[u_{n+1} - u_n] \leqslant g_{\pi_{n+1}} \leqslant g_\star \leqslant \overline{P}_\star[u_{n+1} - u_n]\,.$$

Corollary [Value and gain]

For each n, it holds $\quad g_\star - g_{\pi_{n+1}} \leqslant \mathbb{S}(u_{n+1} - u_n)$,

- This gives a criterion to **stop** Value Iteration:

$$\text{Stop when } \mathbb{S}(u_{n+1} - u_n) \leqslant \varepsilon$$

- Is value iteration converging?

Let $\Delta_k(s) = u_{k+1}(s) - u_k(s)$. It can be shown that,

$$\forall k, P_{\pi_{k+2}}\Delta_k \geqslant \Delta_{k+1} \geqslant P_{\pi_{k+1}}\Delta_k.$$

Lemma [k-step contraction of VI]

For all k, it holds

$$\mathbb{S}(\Delta_n) \leqslant \left(1 - \sum_{s' \in \mathcal{S}} \min\left\{P_{\pi_{n+1},\dots,\pi_{n+2-k}}(s'|\overline{s}), P_{\pi_n,\dots,\pi_{n+1-k}}(s'|\underline{s})\right\}\right)\mathbb{S}(\Delta_{n-k}).$$

with $\overline{s} = \operatorname*{argmax}_{s \in \mathcal{S}} \left(P_{\pi_{n+1},\dots,\pi_{n+2-k}}\Delta_{n-k}\right)(s)$, $\underline{s} = \operatorname*{argmin}_{s \in \mathcal{S}} \left(P_{\pi_n,\dots,\pi_{n-k}}\Delta_{n-k}\right)(s)$.

- Is value iteration converging?

Let $\Delta_k(s) = u_{k+1}(s) - u_k(s)$. It can be shown that,

$$\forall k, P_{\pi_{k+2}}\Delta_k \geqslant \Delta_{k+1} \geqslant P_{\pi_{k+1}}\Delta_k.$$

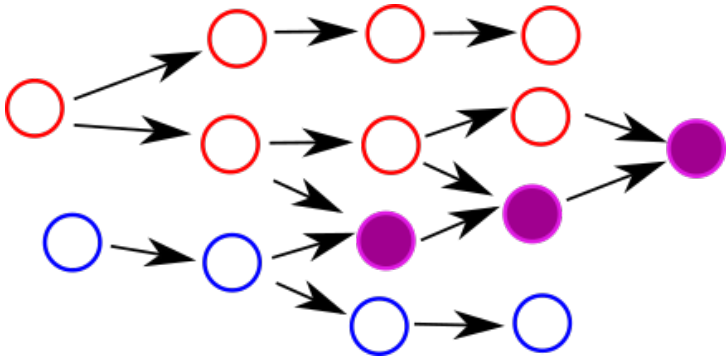
Lemma [k-step contraction of VI]

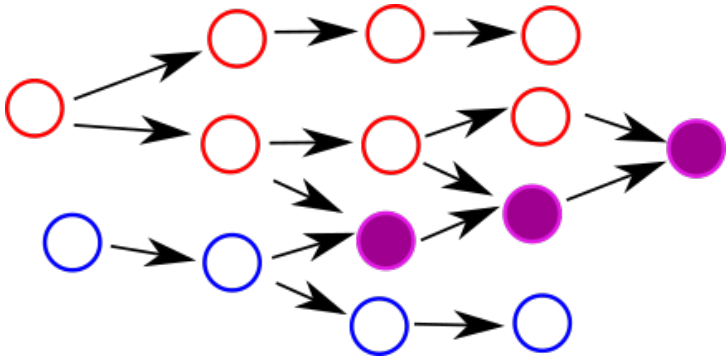
For all k, it holds

$$\mathbb{S}(\Delta_n) \leqslant \left(1 - \sum_{s' \in \mathcal{S}} \min\left\{P_{\pi_{n+1},\dots,\pi_{n+2-k}}(s'|\overline{s}), P_{\pi_n,\dots,\pi_{n+1-k}}(s'|\underline{s})\right\}\right)\mathbb{S}(\Delta_{n-k}).$$

with $\overline{s} = \operatorname*{argmax}_{s \in \mathcal{S}} \left(P_{\pi_{n+1},\dots,\pi_{n+2-k}}\Delta_{n-k}\right)(s)$, $\underline{s} = \operatorname*{argmin}_{s \in \mathcal{S}} \left(P_{\pi_n,\dots,\pi_{n-k}}\Delta_{n-k}\right)(s)$.

- Multi-step transitions $P_{\pi_{n+1},\dots,\pi_{n+2-k}}$
- When is contraction guaranteed? **Coalescence** of two policies from two different states.





Remark: **In discounted MDP**, $1-\gamma$ can be interpreted as probability to "exit MDP", hence to reach same external state $\perp$ from any state: **all policies coalesce**.

Table of contents

- We now have a proper **value iteration** procedure, working when $\gamma = 1$, provided that **intrinsic** contraction happens.
- Still requires full knowledge of the MDP.

What to do when P, R are **unknown**?

Markov Decision Process

A discrete-time finite Markov decision process (MDP) is a tuple $M = \langle \mathcal{S}, \mathcal{A}, r, p \rangle$

- State space $\mathcal{S}$, $|\mathcal{S}| = S < \infty$
- Action space $\mathcal{A}$, $|\mathcal{A}| = A < \infty$

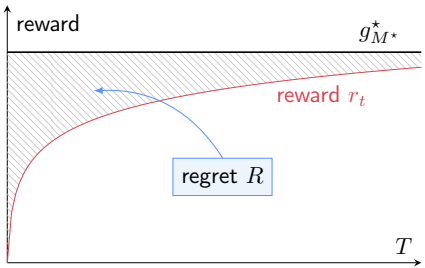
} finite

- Transition distribution $p(\cdot|s,a) \in \Delta(\mathcal{S})$ } Markov
- Reward distribution with expectation $r(s,a) \in [0, r_{\max}]$

☞ The process generates history $H_t = (s_1, a_1, \ldots, s_{t-1}, a_{t-1}, s_t)$, with $s_{t+1} \sim p(\cdot|s_t, a_t)$

In (contextual) bandit, actions do not influence the evolution of states

Regret Minimization



unknown true MDP $M^\star = \langle \mathcal{S}, \mathcal{A}, r, p \rangle$

algorithm $\mathfrak{A} = \{\pi_t\}$

$$R(T,\ M^\star,\ \mathfrak{A}) = T g^\star_{M^\star} - \sum_{t=1}^{T} r_t$$

reward obtained by $\mathfrak{A}$

A no-regret algorithm satisfies $\mathbb{E}\big[R(T, M^\star, \mathfrak{A})\big] = o(T)$

What is Wrong with Q-learning with ϵ-greedy?

- ϵ-greedy strategy

$$a_t = \begin{cases} \arg\max_a Q_{\theta_t}(s_t, a) & \text{w.p. } 1-\epsilon \\ \mathcal{U}(\mathcal{A}) & \text{otherwise} \end{cases}$$

- Q-learning update

$$\theta_{t+1} = (1-\alpha_t)\theta_t + \alpha_t\big(r_t + \gamma \max_{a'} Q_{\theta_t}(s_{t+1}, a') - Q_{\theta_t}(s_t, a)\big)\nabla_\theta Q_{\theta_t}(s_t, a)$$

What is Wrong with Q-learning with ϵ-greedy?

- ϵ-greedy strategy

$$a_t = \begin{cases} \arg\max_a Q_{\theta_t}(s_t, a) & \text{w.p. } 1-\epsilon \\ \mathcal{U}(\mathcal{A}) & \text{otherwise} \end{cases}$$

- Q-learning update

$$\theta_{t+1} = (1-\alpha_t)\theta_t + \alpha_t\big(r_t + \gamma \max_{a'} Q_{\theta_t}(s_{t+1}, a') - Q_{\theta_t}(s_t, a)\big)\nabla_\theta Q_{\theta_t}(s_t, a)$$

👎 The exploration strategy relies on **biased** estimates Q_{θ_t}

What is Wrong with Q-learning with ϵ-greedy?

- ϵ-greedy strategy

$$a_t = \begin{cases} \arg\max_a Q_{\theta_t}(s_t, a) & \text{w.p. } 1-\epsilon \\ \mathcal{U}(\mathcal{A}) & \text{otherwise} \end{cases}$$

- Q-learning update

$$\theta_{t+1} = (1-\alpha_t)\theta_t + \alpha_t\big(r_t + \gamma \max_{a'} Q_{\theta_t}(s_{t+1}, a') - Q_{\theta_t}(s_t, a)\big)\nabla_\theta Q_{\theta_t}(s_t, a)$$

  - 👎 The exploration strategy relies on **biased** estimates Q_{θ_t}
  - 👎 Samples are used **once**

What is Wrong with Q-learning with ϵ-greedy?

- ϵ-greedy strategy

$$a_t = \begin{cases} \arg\max_a Q_{\theta_t}(s_t, a) & \text{w.p. } 1-\epsilon \\ \mathcal{U}(\mathcal{A}) & \text{otherwise} \end{cases}$$

- Q-learning update

$$\theta_{t+1} = (1-\alpha_t)\theta_t + \alpha_t\big(r_t + \gamma \max_{a'} Q_{\theta_t}(s_{t+1}, a') - Q_{\theta_t}(s_t, a)\big)\nabla_\theta Q_{\theta_t}(s_t, a)$$

- The exploration strategy relies on **biased** estimates Q_{θ_t}
- Samples are used **once**
- **Dithering effect:** exploration is not effective in covering the state space

What is Wrong with Q-learning with ϵ-greedy?

- ϵ-greedy strategy

$$a_t = \begin{cases} \arg\max_a Q_{\theta_t}(s_t, a) & \text{w.p. } 1-\epsilon \\ \mathcal{U}(\mathcal{A}) & \text{otherwise} \end{cases}$$

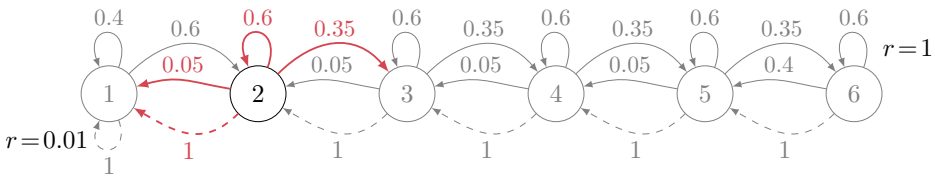
- Q-learning update

$$\theta_{t+1} = (1-\alpha_t)\theta_t + \alpha_t\big(r_t + \gamma \max_{a'} Q_{\theta_t}(s_{t+1}, a') - Q_{\theta_t}(s_t, a)\big)\nabla_\theta Q_{\theta_t}(s_t, a)$$

- The exploration strategy relies on **biased** estimates Q_{θ_t}
- Samples are used **once**
- **Dithering effect:** exploration is not effective in covering the state space
- **Policy shift:** the policy changes at each step

River Swim: Markov Decision Processes

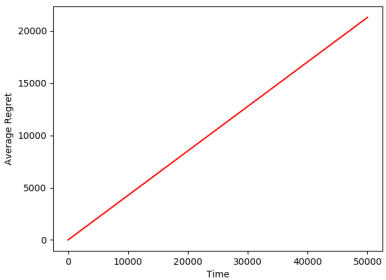
Strehl and Littman [2008]



- $\mathcal{S} = \{1, 2, 3, 4, 5, 6\}$, $\mathcal{A} = \{L, R\}$
- $\pi_L(s) = L$, $\pi_R(s) = R$

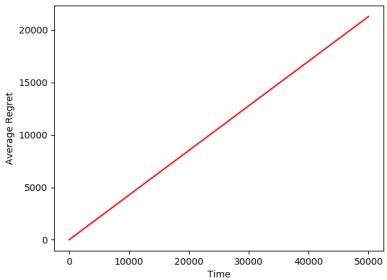
River Swim: Q-learning w\ ϵ-greedy Exploration

- $\epsilon_t = 1.0$



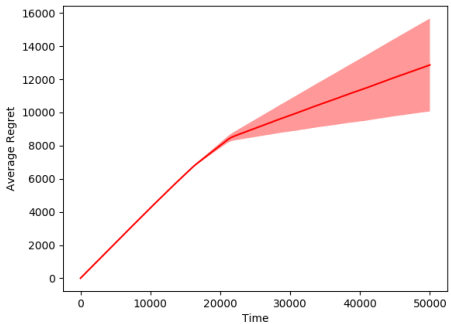
River Swim: Q-learning w\ ϵ-greedy Exploration

- $\epsilon_t = 1.0$
- $\epsilon_t = 0.5$



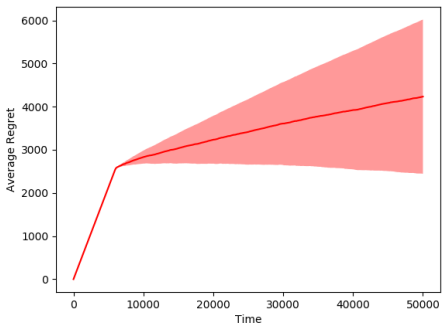
River Swim: Q-learning w\ ϵ-greedy Exploration

- $\epsilon_t = 1.0$
- $\epsilon_t = 0.5$
- $\epsilon_t = \frac{\epsilon_0}{(N(s_t) - 1000)^{2/3}}$



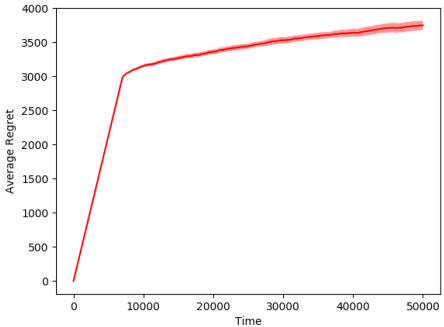
River Swim: Q-learning w\ ϵ-greedy Exploration

- $\epsilon_t = 1.0$
- $\epsilon_t = 0.5$
- $\epsilon_t = \frac{\epsilon_0}{(N(s_t) - 1000)^{2/3}}$
- $\epsilon_t = \begin{cases} 1.0 & t < 6000 \\ \frac{\epsilon_0}{N(s_t)^{1/2}} & \text{otherwise} \end{cases}$



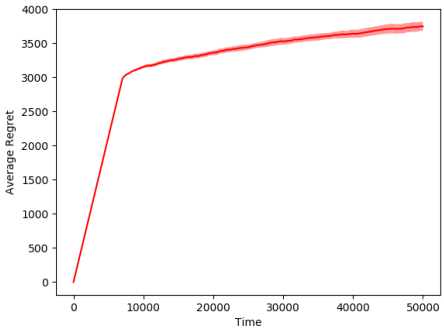
River Swim: Q-learning w\ ϵ-greedy Exploration

- $\epsilon_t = 1.0$
- $\epsilon_t = 0.5$
- $\epsilon_t = \frac{\epsilon_0}{(N(s_t) - 1000)^{2/3}}$
- $\epsilon_t = \begin{cases} 1.0 & t < 6000 \\ \frac{\epsilon_0}{N(s_t)^{1/2}} & \text{otherwise} \end{cases}$
- $\epsilon_t = \begin{cases} 1.0 & t < 7000 \\ \frac{\epsilon_0}{N(s_t)^{1/2}} & \text{otherwise} \end{cases}$



River Swim: Q-learning w\ ϵ-greedy Exploration

- $\epsilon_t = 1.0$
- $\epsilon_t = 0.5$
- $\epsilon_t = \frac{\epsilon_0}{(N(s_t) - 1000)^{2/3}}$
- $\epsilon_t = \begin{cases} 1.0 & t < 6000 \\ \frac{\epsilon_0}{N(s_t)^{1/2}} & \text{otherwise} \end{cases}$
- $\epsilon_t = \begin{cases} 1.0 & t < 7000 \\ \frac{\epsilon_0}{N(s_t)^{1/2}} & \text{otherwise} \end{cases}$

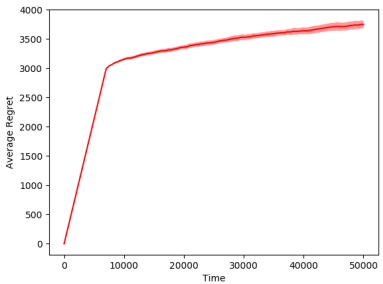


Tuning the ϵ schedule is **difficult and problem dependent**

River Swim: Q-learning w\ ϵ-greedy Exploration

Main drawbacks of Q-learning with ϵ-greedy*

- Q-learning is *model-free*
  - Inefficient *use* of samples
- ϵ-greedy performs *undirected* exploration
  - *Non-informative* samples

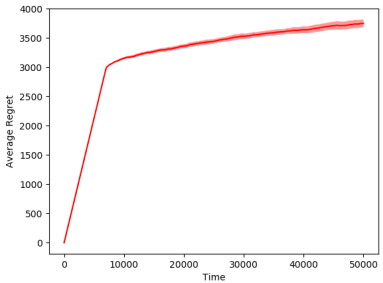


*All of this can be said for large majority for model-free undirected exploration methods

River Swim: Q-learning w\ ϵ-greedy Exploration

Main drawbacks of Q-learning with ϵ-greedy*

- Q-learning is *model-free*
  - Inefficient *use* of samples
- ϵ-greedy performs *undirected* exploration
  - *Non-informative* samples



Model-based uncertainty-driven exploration-exploitation

*All of this can be said for large majority for model-free undirected exploration methods

The Optimism Principle: Intuition

OPTIMISM
It's the best way to see life.

The Optimism Principle: Intuition

Exploration vs. Exploitation

The Optimism Principle: Intuition

Exploration vs. Exploitation

Optimism in Face of Uncertainty

When you are uncertain, consider the **best possible world (reward-wise)**

The Optimism Principle: Intuition

Exploration vs. Exploitation

Optimism in Face of Uncertainty

When you are uncertain, consider the **best possible world (reward-wise)**

If the best possible world is **correct**

$\Longrightarrow$ **no regret**

Exploitation

If the best possible world is **wrong**

$\Longrightarrow$ **learn useful information**

Exploration

The Optimism Principle: Intuition

Exploration vs. Exploitation

Optimism in **gain**

Optimism in Face of Uncertainty

When you are uncertain, consider the **best possible world (reward-wise)**

If the best possible world is **correct**

$\Longrightarrow$ **no regret**

Exploitation

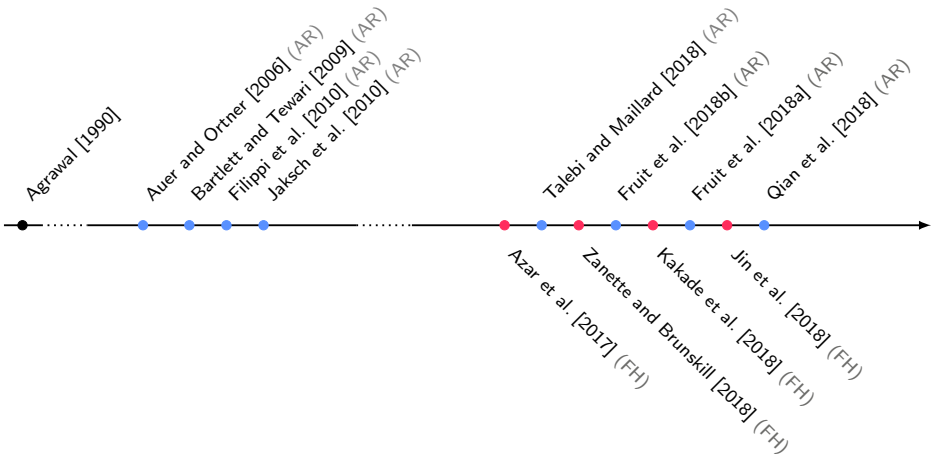
If the best possible world is **wrong**

$\Longrightarrow$ **learn useful information**

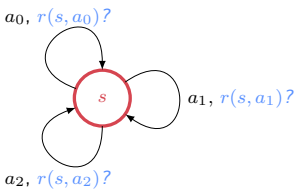
Exploration

History: OFU for Regret Minimization in RL

FH: finite-horizon
AR: average reward

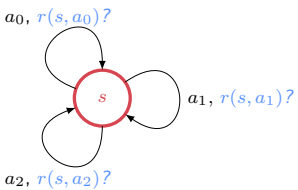


Gain Optimism: Example



- Deterministic *policies*:
  - $\pi_0(s) = a_0$
  - $\pi_1(s) = a_1$
  - $\pi_2(s) = a_2$

Gain Optimism: Example

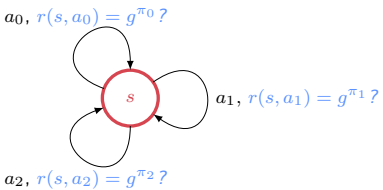


- Deterministic *policies*:
  - $\pi_0(s) = a_0$
  - $\pi_1(s) = a_1$
  - $\pi_2(s) = a_2$

- Optimism

$$\widetilde{\pi} = \arg\max_{\pi_i} \mathsf{UCB}(g^{\pi_i})$$

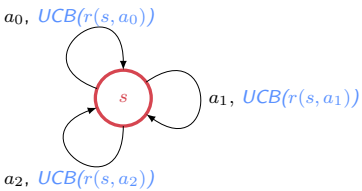
Gain Optimism: Example



- Deterministic *policies*:
  - $\pi_0(s) = a_0$
  - $\pi_1(s) = a_1$
  - $\pi_2(s) = a_2$
- Reward $r(s, a_i) =$ *gain* g^{π_i}
- Optimism

$$\widetilde{\pi} = \arg\max_{\pi_i} \mathsf{UCB}(g^{\pi_i})$$

Gain Optimism: Example



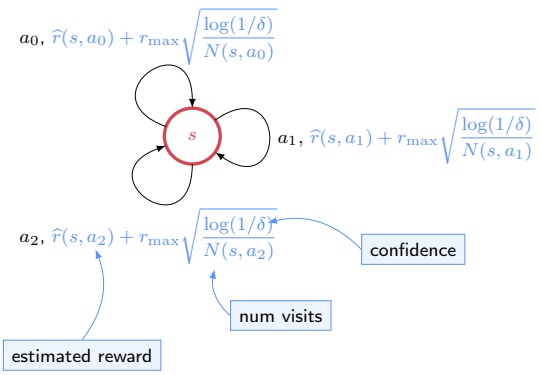
- Deterministic *policies*:
  - $\pi_0(s) = a_0$
  - $\pi_1(s) = a_1$
  - $\pi_2(s) = a_2$
- Reward $r(s, a_i) =$ *gain* g^{π_i}
- Upper confidence bound

$$\text{UCB}(g^{\pi_i}) = \text{UCB}(r(s, a_i))$$

- Optimism

$$\widetilde{\pi} = \arg\max_{\pi_i} \text{UCB}(g^{\pi_i})$$

Gain Optimism: Example



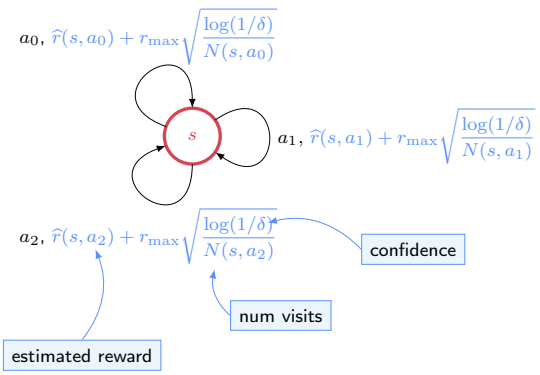
- Deterministic *policies*:
  - $\pi_0(s) = a_0$
  - $\pi_1(s) = a_1$
  - $\pi_2(s) = a_2$
- Reward $r(s, a_i) =$ *gain* g^{π_i}
- Upper confidence bound

$$\text{UCB}(g^{\pi_i}) = \text{UCB}(r(s, a_i))$$

- Optimism

$$\widetilde{\pi} = \arg\max_{\pi_i} \text{UCB}(g^{\pi_i})$$

Gain Optimism: Example



- Deterministic *policies*:
  - $\pi_0(s) = a_0$
  - $\pi_1(s) = a_1$
  - $\pi_2(s) = a_2$
- Reward $r(s, a_i) =$ *gain* g^{π_i}
- Upper confidence bound

$$\text{UCB}(g^{\pi_i}) = \text{UCB}(r(s, a_i))$$

- Optimism

$$\widetilde{\pi} = \arg\max_{\pi_i} \text{UCB}(g^{\pi_i})$$

👉 UCB algorithm (Bandit)

Gain Optimism: Implementation

Tentative algorithm

Observe s_1
for $t = 1, 2, \ldots$ **do**
  Compute $\pi_t \leftarrow \arg\max_{\pi} UCB_t(g^\pi)$
  Take action $a_t = \pi_t(s_t)$
  Observe reward r_t and next state s_{t+1}
  Compute $\text{UCB}_{t+1}(g^\pi)$ for all π based on $\text{UCB}_t(g^\pi)$ and $\langle s_t, a_t, r_t, s_{t+1} \rangle$
end

Gain Optimism: Implementation

Tentative algorithm

Observe s_1
for $t = 1, 2, \ldots$ **do**
  Compute $\pi_t \leftarrow \arg\max_{\pi} UCB_t(g^\pi)$
  Take action $a_t = \pi_t(s_t)$
  Observe reward r_t and next state s_{t+1}
  Compute $\text{UCB}_{t+1}(g^\pi)$ for all π based on $\text{UCB}_t(g^\pi)$ and $\langle s_t, a_t, r_t, s_{t+1} \rangle$
end

⚠ *3 major issues:*

- *Upper confidence bounds*: construct $\text{UCB}_t(g^\pi)$ with unknown dynamics
- *Computational complexity*: exponential number of policies
- *Frequent policy update*: inefficient exploration

Gain Optimism: Implementation

Tentative algorithm

Observe s_1
for $t = 1, 2, \ldots$ **do**
  Compute $\pi_t \leftarrow \arg\max_{\pi} UCB_t(g^\pi)$
  Take action $a_t = \pi_t(s_t)$
  Observe reward r_t and next state s_{t+1}
  Compute $\text{UCB}_{t+1}(g^\pi)$ for all π based on $\text{UCB}_t(g^\pi)$ and $\langle s_t, a_t, r_t, s_{t+1} \rangle$
end

⚠ *3 major issues:*

- *Upper confidence bounds*: construct $\text{UCB}_t(g^\pi)$ with unknown dynamics
- *Computational complexity*: exponential number of policies
- *Frequent policy update*: inefficient exploration

Bounded Parameter MDP: Definition

Bounded parameter MDP [Strehl and Littman, 2008]

$$\mathcal{M}_t = \Big\{ \langle \mathcal{S}, \mathcal{A}, r, p \rangle : \; r(s,a) \in B_t^r(s,a), \; p(\cdot|s,a) \in B_t^p(s,a), \forall (s,a) \in \mathcal{S} \times \mathcal{A} \Big\}$$

Compact *confidence sets*

$$B_t^r(s,a) := \Big[\widehat{r}_t(s,a) - \beta_t^r(s,a), \; \widehat{r}_t(s,a) + \beta_t^r(s,a) \Big]$$

$$B_t^p(s,a) := \Big\{ p(\cdot|s,a) \in \Delta(\mathcal{S}) : \; \| p(\cdot|s,a) - \widehat{p}_t(\cdot|s,a) \|_1 \leq \; \beta_t^p(s,a) \Big\}$$

Bounded Parameter MDP: Definition

Bounded parameter MDP [Strehl and Littman, 2008]

$$\mathcal{M}_t = \left\{ \langle \mathcal{S}, \mathcal{A}, r, p \rangle : \; r(s,a) \in B_t^r(s,a), \; p(\cdot|s,a) \in B_t^p(s,a), \forall (s,a) \in \mathcal{S} \times \mathcal{A} \right\}$$

Compact *confidence sets*

$$B_t^r(s,a) := \left[\widehat{r}_t(s,a) - \beta_t^r(s,a), \; \widehat{r}_t(s,a) + \beta_t^r(s,a) \right]$$

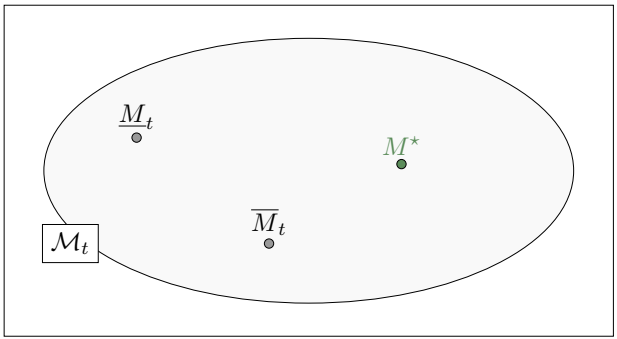
$$B_t^p(s,a) := \left\{ p(\cdot|s,a) \in \Delta(\mathcal{S}) : \; \|p(\cdot|s,a) - \widehat{p}_t(\cdot|s,a)\|_1 \leq \; \beta_t^p(s,a) \right\}$$

Confidence bounds based on [Hoeffding, 1963] and [Weissman et al., 2003]

$$\beta_t^r(s,a) \propto \sqrt{\frac{\log(N_t(s,a)/\delta)}{N_t(s,a)}}$$

$$\beta_t^p(s,a) \propto \sqrt{\frac{S \log(N_t(s,a)/\delta)}{N_t(s,a)}}$$

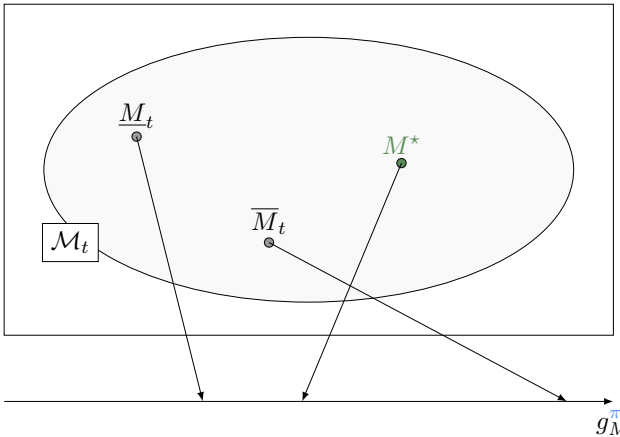
Bounded Parameter MDP: Optimism



g_M^π

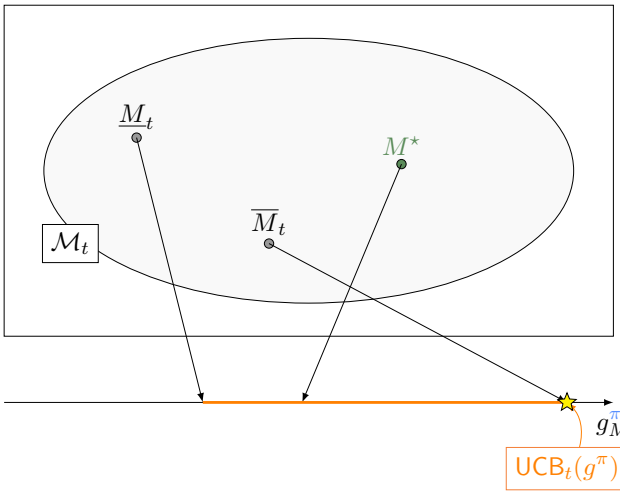
Fix a *policy* π

Bounded Parameter MDP: Optimism



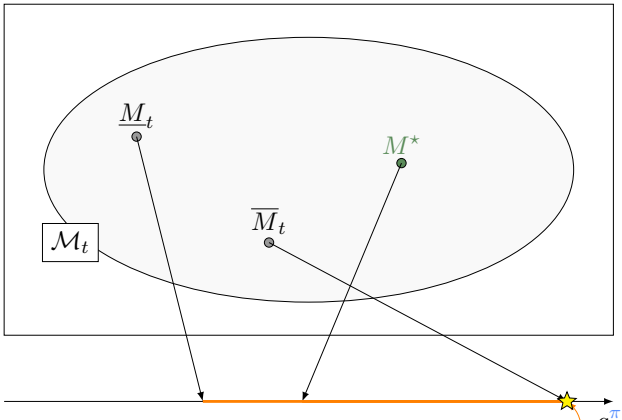
Fix a *policy* π

Bounded Parameter MDP: Optimism



Fix a *policy* π

Bounded Parameter MDP: Optimism



Optimism: $\text{UCB}_t(g^\pi) = \max_{M \in \mathcal{M}_t} g^\pi_M \geq g^\pi_{M^\star}$

$\text{UCB}_t(g^\pi)$

Fix a *policy* π

Gain Optimism: Implementation

Tentative algorithm

Observe state s_1
for $t = 1, 2, \ldots$ **do**
  Compute $\pi_t \leftarrow \arg\max_{\pi} UCB_t(g^\pi)$
  Take action $a_t = \pi_t(s_t)$
  Observe reward r_t and next state s_{t+1}
  Compute $\text{UCB}_{t+1}(g^\pi)$ for all π based on $\text{UCB}_t(g^\pi)$ and $\langle s_t, a_t, r_t, s_{t+1} \rangle$
end

⚠ *3 major issues:*

- *Upper confidence bounds*: construct $\text{UCB}_t(g^\pi)$ with unknown dynamics? ✔
- *Computational complexity*: exponential number of policies
- *Frequent policy update*: inefficient exploration

Gain Optimism: Implementation

Tentative algorithm

Observe state s_1
for $t = 1, 2, \ldots$ **do**
  Compute $\pi_t \leftarrow \arg\max_{\pi} UCB_t(g^\pi)$
  Take action $a_t = \pi_t(s_t)$
  Observe reward r_t and next state s_{t+1}
  Compute $\text{UCB}_{t+1}(g^\pi)$ for all π based on $\text{UCB}_t(g^\pi)$ and $\langle s_t, a_t, r_t, s_{t+1} \rangle$
end

⚠ *3 major issues:*

- *Upper confidence bounds*: construct $\text{UCB}_t(g^\pi)$ with unknown dynamics? ✔
- *Computational complexity*: exponential number of policies
- *Frequent policy update*: inefficient exploration

Gain Optimism: Implementation

Tentative algorithm

Observe state s_1
for $t = 1, 2, \ldots$ **do**
  Compute $\pi_t \leftarrow \arg\max_{\pi} \left\{ \max_{M \in \mathcal{M}_t} g_M^{\pi} \right\}$
  Take action $a_t = \pi_t(s_t)$
  Observe reward r_t and next state s_{t+1}
  Compute $\text{UCB}_{t+1}(g^{\pi})$ for all π based on $\text{UCB}_t(g^{\pi})$ and $\langle s_t, a_t, r_t, s_{t+1} \rangle$
end

⚠ *3 major issues:*

- *Upper confidence bounds*: construct $\text{UCB}_t(g^{\pi})$ with unknown dynamics? ✔
- *Computational complexity*: exponential number of policies
- *Frequent policy update*: inefficient exploration

Gain Optimism: Implementation

Tentative algorithm

Observe state s_1
for $t = 1, 2, \ldots$ **do**
  Compute $\pi_t \leftarrow \arg\max_{\pi} \left\{ \max_{M \in \mathcal{M}_t} g_M^{\pi} \right\}$
  Take action $a_t = \pi_t(s_t)$
  Observe reward r_t and next state s_{t+1}
  Compute $\text{UCB}_{t+1}(g^{\pi})$ for all π based on $\text{UCB}_t(g^{\pi})$ and $\langle s_t, a_t, r_t, s_{t+1} \rangle$
end

⚠ *3 major issues:*

- *Upper confidence bounds*: construct $\text{UCB}_t(g^{\pi})$ with unknown dynamics? ✔
  - How to efficiently *compute* $\max_{M \in \mathcal{M}_t} g_M^{\pi}$ for every π?
- *Computational complexity*: exponential number of policies
- *Frequent policy update*: inefficient exploration

Gain Optimism: Implementation

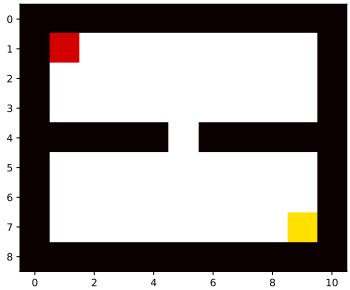
Tentative algorithm

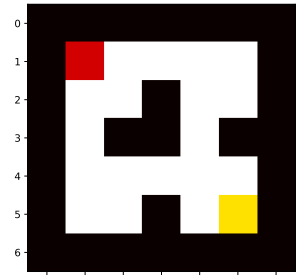
Observe state s_1
for $t = 1, 2, \ldots$ **do**
  Compute $\pi_t \leftarrow \arg\max_{\pi} \left\{ \max_{M \in \mathcal{M}_t} g_M^{\pi} \right\}$
  Take action $a_t = \pi_t(s_t)$
  Observe reward r_t and next state s_{t+1}
  Compute $\text{UCB}_{t+1}(g^{\pi})$ for all π based on $\text{UCB}_t(g^{\pi})$ and $\langle s_t, a_t, r_t, s_{t+1} \rangle$
end

⚠ *3 major issues:*

- *Upper confidence bounds*: construct $\text{UCB}_t(g^{\pi})$ with unknown dynamics? ✔
  - How to efficiently *compute* $\max_{M \in \mathcal{M}_t} g_M^{\pi}$ for every π?
- *Computational complexity*: exponential number of policies
- *Frequent policy update*: inefficient exploration

Example MDPs





- Frozen-lake type mazes.

UCRL (Jaksch et al., 2010): A **model-based** algorithm for undiscounted RL implementing the principle of **optimism in the face of uncertainty**.

- Maintains a set of **plausible MDPs (models)** by defining high-probability **confidence sets** for μ and p
- Chooses an optimistic model (among models) and an optimistic policy leading to the **highest average-reward**.

- UCRL2 maintains a set of **plausible MDPs** ($\widetilde{\mu}$ denotes the mean of $\widetilde{\nu}$)

$$\mathcal{M}_{t,\delta} = \left\{ \widetilde{M} = (\mathcal{S}, \mathcal{A}, \widetilde{p}, \widetilde{\nu}) : \right.$$
$$\left. \forall (s,a) \in \mathcal{S} \times \mathcal{A},\ \widetilde{p}(\cdot|s,a) \in \mathcal{C}_{t,\delta}^{\texttt{UCRL2}}(s,a) \text{ and } \widetilde{\mu}(s,a) \in c_{t,\delta}^{\texttt{UCRL2}}(s,a) \right\}.$$

- Vanilla UCRL2 uses $(s,a) \in \mathcal{S} \times \mathcal{A}$,

$$c_{t,\delta}^{\text{UCRL2}}(s,a) = \left\{ \mu' \in [0,1] : |\widehat{\mu}_t(s,a) - \mu'| \leq \sqrt{\frac{3.5 \log(\frac{2SAt}{\delta})}{N_t(s,a)}} \right\},$$

$$C_{t,\delta}^{\text{UCRL2}}(s,a) = \left\{ p' \in \Delta_{\mathcal{S}} : \|\widehat{p}_t(\cdot|s,a) - p'\|_1 \leqslant \sqrt{\frac{14S \log(\frac{2At}{\delta})}{N_t(s,a)}} \right\}.$$

  - Empirical estimates of transition probabilities and rewards:

$$\widehat{\mu}_t(s,a) = \frac{\sum_{t'=0}^{t-1} r_{t'} \mathbb{I}\{s_{t'} = s, a_{t'} = a\}}{\max\{N_t(s,a), 1\}}$$

$$\widehat{p}_t(s'|s,a) = \frac{N_t(s,a,s')}{\max\{N_t(s,a), 1\}}$$

  - $N_t(s,a)$: number of visits, up to time t, to (s,a).
  - $N_t(s,a,s')$: number of visits, up to time t, to (s,a) followed by a visit to s'.
- Hoeffding plus many union bounds (improved in new version UCRL3)

- UCRL2 maintains a set of **plausible MDPs** ($\widetilde{\mu}$ denotes the mean of $\widetilde{\nu}$)

$$\mathcal{M}_{t,\delta} = \left\{ \widetilde{M} = (\mathcal{S}, \mathcal{A}, \widetilde{p}, \widetilde{\nu}) : \right.$$
$$\left. \forall (s,a) \in \mathcal{S} \times \mathcal{A},\ \widetilde{p}(\cdot|s,a) \in \mathcal{C}_{t,\delta}^{\text{UCRL2}}(s,a) \text{ and } \widetilde{\mu}(s,a) \in c_{t,\delta}^{\text{UCRL2}}(s,a) \right\}.$$

- **Optimistic principle**: Try to compute

$$\overline{\pi}_t^+ = \operatorname*{argmax}_{\pi:\mathcal{S}\to\mathcal{A}} \max\{g_\pi^M : M \in \mathcal{M}_{t,\delta}\},$$

where g_π^M is gain of π in MDP M.

- Use Extended Value Iteration (EVI): builds a **near-optimal** policy π_t^+ and MDP $\widetilde{M}_t$ such that $g_{\pi_t^+}^{\widetilde{M}_t} \geqslant \max_\pi \max\{g_\pi^M : M \in \mathcal{M}_{t,\delta}\} - \varepsilon_t$ with $\varepsilon_t = \frac{1}{\sqrt{t}}$.

Input: ε_t

Let $u_0 \equiv 0, u_{-1} \equiv -\infty$, $n = 0$

while $\mathbb{S}(u_n - u_{n-1}) > \varepsilon_t$ **do**

Compute $\begin{cases} \mu^+ : s, a \mapsto \max\{\mu' : \mu' \in c_{t,\delta}^{\texttt{UCRL2}}(s,a)\} \\ p_n^+ : s, a \mapsto \operatorname{argmax}\{P' u_n : p' \in \mathcal{C}_{t,\delta}^{\texttt{UCRL2}}(s,a)\} \end{cases}$

Update $\begin{cases} u_{n+1}(s) = \max\{\mu^+(s,a) + (P_n^+ u_n)(s,a) \colon a \in \mathcal{A}\} \\ \pi_{n+1}^+(s) \in \operatorname{Argmax}\{\mu^+(s,a) + (P_n^+ u_n)(s,a) \colon a \in \mathcal{A}\} \end{cases}$

$n = n + 1$

end while

- Value iteration for **unknown** r, P:

$$
\begin{aligned}
u_{n+1}(s) &= \max_{(a,r,P)\in\mathcal{A}\times c_{t,\delta}^{\texttt{UCRL2}}(s,a)\times\mathcal{C}_{t,\delta}^{\texttt{UCRL2}}(s,a)} \left\{ r + Pu_n \right\} && \textbf{(Extended actions)} \\
&= \max_{a\in\mathcal{A}} \left\{ \max_{r\in c_{t,\delta}^{\texttt{UCRL2}}(s,a)} r + \max_{P\in\mathcal{C}_{t,\delta}^{\texttt{UCRL2}}(s,a)} Pu_n \right\} && \textbf{(Optimistic model)} \\
\pi_{n+1} &= \text{Greedy with respect to } v_n.
\end{aligned}
$$

Extended MDP

[Strehl and Littman, 2008, Jaksch et al., 2010]

Theorem (Bounded parameter MDP $\iff$ Extended MDP)

Let $\mathcal{M}_t^+ := \langle \mathcal{S}, \mathcal{A}_t^+, r^+, p^+ \rangle$ *be an extended MDP such that*

$$\mathcal{A}_t^+(s) = \mathcal{A}(s) \times B_t^r(s,a) \times B_t^p(s,a)$$

with $a^+ = (a, r, p) \in \mathcal{A}_t^+(s),\ r^+(s,a^+) = r,\ p^+(\cdot|s,a^+) = p.$

Continuous **compact** action space

Then the optimal gain of $\mathcal{M}_t^+$ *satisfies*

$$g^*_{\mathcal{M}_t^+} := \max_\pi \left\{ \max_{M \in \mathcal{M}_t} g^\pi_M \right\}$$

Let $\pi_t^+ = \arg\max_\pi g^\pi_{\mathcal{M}_t^+}$, *then*

$$\pi_t = \arg\max_\pi \left\{ \max_{M \in \mathcal{M}_t} g^\pi_M \right\} \text{ s.t. } \pi_t(s) = \pi_t^+(s)[a]$$

Extended MDP

[Strehl and Littman, 2008, Jaksch et al., 2010]

Theorem (Bounded parameter MDP $\iff$ Extended MDP)

Let $\mathcal{M}_t^+ := \langle \mathcal{S}, \mathcal{A}_t^+, r^+, p^+ \rangle$ *be an extended MDP such that*

$$\mathcal{A}_t^+(s) = \mathcal{A}(s) \times B_t^r(s,a) \times B_t^p(s,a)$$

with $a^+ =$ *Abuse of notation*: $\mathcal{M}_t$ denotes the extended MDP

ompact
pace

Then the optimal gain of $\mathcal{M}_t^+$ *satisfies*

$$g^*_{\mathcal{M}_t^+} := \max_\pi \left\{ \max_{M \in \mathcal{M}_t} g^\pi_M \right\}$$

Let $\pi_t^+ = \arg\max_\pi g^\pi_{\mathcal{M}_t^+}$, *then*

$$\pi_t = \arg\max_\pi \left\{ \max_{M \in \mathcal{M}_t} g^\pi_M \right\} \text{ s.t. } \pi_t(s) = \pi_t^+(s)[a]$$

- UCRL2 maintains a set of **plausible MDPs** ($\widetilde{\mu}$ denotes the mean of $\widetilde{\nu}$)

$$\mathcal{M}_{t,\delta} = \left\{ \widetilde{M} = (\mathcal{S}, \mathcal{A}, \widetilde{p}, \widetilde{\nu}) : \right.$$

$$\left. \forall (s,a) \in \mathcal{S} \times \mathcal{A},\ \widetilde{p}(\cdot|s,a) \in \mathcal{C}_{t,\delta}^{\texttt{UCRL2}}(s,a) \text{ and } \widetilde{\mu}(s,a) \in c_{t,\delta}^{\texttt{UCRL2}}(s,a) \right\}.$$

- **Optimistic principle**: Try to compute

$$\overline{\pi}_t^+ = \operatorname*{argmax}_{\pi:\mathcal{S}\to\mathcal{A}} \max\{g_\pi^M : M \in \mathcal{M}_{t,\delta}\},$$

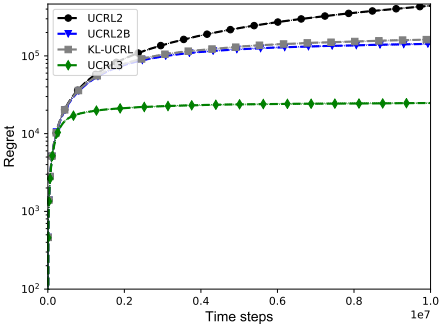
where g_π^M is gain of π in MDP M. Use EVI.

- **Episodes**: Only recompute policies at time $(t_k)_k$ (hence $\pi_k^+ := \pi_{t_k}^+$) s.t. $t_1 = 1$, and

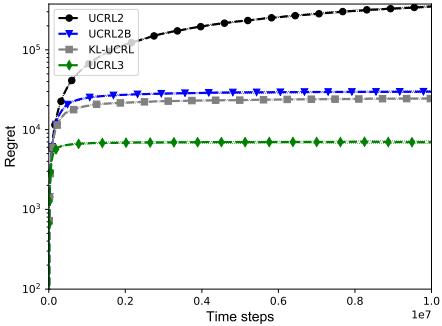
$$\forall k > 1,\ t_k = \min \left\{ t > t_{k-1} : \max_{s,a} \frac{n_{t_{k-1}:t}(s,a)}{N_{t_{k-1}}(s,a)} \geqslant 1 \right\},$$

where $n_{t_1:t_2}(s,a)$ is the number of observations of (s,a) between time t_1 and t_2.

Results for UCRL and variants



2-room MDP



4-room MDP

UCRL (Jaksch et al., 2010): A model-based algorithm for undiscounted RL implementing the principle of optimism in the face of uncertainty.

- For any communicating MDP with S states, A actions, and diameter D, UCRL satisfies

$$\mathbf{R}(T) \leq 34 r_{\max} DS \sqrt{AT \log(T/\delta)} \quad \text{w.p. at least } 1-\delta.$$

- Minimax lower bound (Jaksch et al., 2010): $\Omega(\sqrt{DSAT})$

UCRL (Jaksch et al., 2010): A model-based algorithm for undiscounted RL implementing the principle of optimism in the face of uncertainty.

- For any communicating MDP with S states, A actions, and diameter D, UCRL satisfies

$$\mathbf{R}(T) \leq 34 r_{\max} DS \sqrt{AT \log(T/\delta)} \quad \text{w.p. at least } 1-\delta.$$

- Minimax lower bound (Jaksch et al., 2010): $\Omega(\sqrt{DSAT})$

UCRL and its variants do not perform empirically great despite their strong regret guarantees.

Refined Confidence Bounds

- UCRL2 with *Bernstein bounds* (instead of Hoeffding/Weissman):

$$R(T, M^\star, \text{UCRL2B}) = \mathcal{O}\left(\sqrt{D\Gamma SAT \log\left(\frac{T}{\delta}\right) \log(T)}\right)$$

- 👎 Still not matching the lower bound!
- 👍 For most MPDs: $\Gamma \ll S$

Refined Confidence Bounds

- UCRL2 with *Bernstein bounds* (instead of Hoeffding/Weissman):

$$R(T, M^\star, \text{UCRL2B}) = \mathcal{O}\left(\sqrt{D\Gamma SAT \log\left(\frac{T}{\delta}\right) \log(T)}\right)$$

  - 👎 Still not matching the lower bound!
  - 👍 For most MPDs: $\Gamma \ll S$

- *Kullback-Leibler* UCRL [Filippi et al., 2010, Talebi and Maillard, 2018]:

$$R(T, M^\star, \text{UCRL-KL}) = \mathcal{O}\left(\sqrt{\underbrace{\sum_{s,a} \mathbb{V}_{X\sim p^\star(\cdot|s,a)}\left(h^\star_{M^\star}(X)\right)}_{\leq D^2 SA} ST \log\left(\frac{T}{\delta}\right)} + D\sqrt{T}\right)$$

  - 👎 Only for ergodic MDPs!

Infinite Diameter (weakly communicating MDPs)

- *Known* bound on the optimal bias span $C \geq \mathsf{sp}(h^\star_{M^\star})$
  [Bartlett and Tewari, 2009, Fruit et al., 2018b]

$$R(T, M^\star, \mathsf{SCAL}) = \mathcal{O}\left(\sqrt{C\Gamma SAT \log\left(\frac{T}{\delta}\right)\log(T)}\right)$$

👎 Requires prior knowledge!

Infinite Diameter (weakly communicating MDPs)

- *Known* bound on the optimal bias span $C \geq \mathrm{sp}(h^\star_{M^\star})$
  [Bartlett and Tewari, 2009, Fruit et al., 2018b]

$$R(T, M^\star, \mathsf{SCAL}) = \mathcal{O}\left(\sqrt{C\Gamma SAT \log\left(\frac{T}{\delta}\right)\log(T)}\right)$$

  👎 Requires prior knowledge!

- No prior knowledge: TUCRL [Fruit et al., 2018a]:

$$R(T, M^\star, \mathsf{SCAL}) = \mathcal{O}\left(\sqrt{D_{\mathsf{com}} S_{\mathsf{com}} \Gamma AT \log\left(\frac{T}{\delta}\right)\log(T)}\right)$$

  👎 Never achieves *logarithmic* regret! Intrinsic limitation of the setting!

- We can refine the conf. bounds of UCRL, using time-uniform concentration bounds.
- We can refine conf. bounds replacing $\|.\|_1$ bounds on distributions p , with KL balls, and also with intervals on each $p(s)$ instead.
- We can suggest a less conservative alternative stopping criterion
- We can revisit the way sparse transitions are handled (important in practice).
- ...

- **Hoefding inequality** (with Laplace method) for $[0,1]$-bounded observations:

$$\mathbb{P}\left(|\widehat{\mu}_\tau - \mu| \geqslant \sqrt{\frac{(1+\frac{1}{\tau})\log(2\sqrt{\tau+1}/\delta)}{2\tau}}\right) \leqslant \delta\,.$$

- **Weissman inequality** for probability distribution on discrete set $\mathcal{S}$:

$$\mathbb{P}\left(\|\widehat{p}_\tau - p\|_1 \geqslant \sqrt{\frac{2(1+\frac{1}{\tau})\log\left(\sqrt{\tau+1}\,\frac{2^{|\mathcal{S}|}-2}{\delta}\right)}{\tau}}\right) \leqslant \delta\,.$$

- Proof using method of types:

For discrete measures, $\mathbb{P}\left(\|\widehat{p}_\tau - p\|_1 \geqslant \varepsilon\right) \leqslant \sum_{B \subset \mathcal{S}} \mathbb{P}\left(p_\tau(B) - p(B) \geqslant \frac{1}{2}\varepsilon\right)$.

  - $2^{|\mathcal{S}|} - 2$ non trivial sets.
  - Each term: Bernoulli concentration, handled e.g. by Hoeffding.

UCRL3

Our main contribution is UCRL3, a new algorithm for average-reward RL.

UCRL3 is a variant of UCRL2, combining the following key elements:

- Tight and element-wise confidence intervals for transition function p
  - Intersection of time-uniform Bernstein and sub-Gaussian Bernoulli concentration for each $p(s'|s,a)$
- A modified planning algorithm, called EVI-NOSS, to compute a near-optimistic policy.

To simplify the presentation, we assume that μ is known.

UCRL3: Confidence Set for p

For each pair (s, a), define

$$\mathcal{C}_{t,\delta}(s,a) := \left\{ q \in \Delta_{\mathcal{S}} : q(s') \in \underbrace{C^1_{t,\delta}(s,a,s')}_{\text{Bernstein}} \cap \underbrace{C^2_{t,\delta}(s,a,s')}_{\text{sub-Gaussian}} \text{ for all } s' \right\}$$

UCRL3: Confidence Set for p

For each pair (s, a), define

$$\mathcal{C}_{t,\delta}(s,a) := \left\{ q \in \Delta_{\mathcal{S}} : q(s') \in \underbrace{C^1_{t,\delta}(s,a,s')}_{\text{Bernstein}} \cap \underbrace{C^2_{t,\delta}(s,a,s')}_{\text{sub-Gaussian}} \text{ for all } s' \right\}$$

- $C^1_{t,\delta}(s,a,s')$ is defined using Bernstein's concentration modified using **a peeling technique**.
- $C^2_{t,\delta}(s,a,s')$ is obtained by leveraging sub-Gaussianity of Bernoulli distributions combined with **the method of mixtures**.

UCRL3: Set of Models

At time t, UCRL3 considers the set $\mathcal{M}_{t,\delta}$ of plausible MDPs:

$$\mathcal{M}_{t,\delta} = \Big\{ M' = (\mathcal{S}, \mathcal{A}, p', \mu) : p'(\cdot|s,a) \in \mathcal{C}_{t,\delta}(s,a) \text{ for all } (s,a) \Big\}$$

Lemma (Time-uniform confidence bounds)

For any MDP M with transition function p, for all $\delta \in (0,1)$, it holds

$$\mathbb{P}\big(\exists t \in \mathbb{N}, M \notin \mathcal{M}_{t,\delta}\big) \leq \delta.$$

UCRL3: Revisiting EVI

- To compute an optimistic policy (i.e., planning) in UCRL2 is done by EVI as a subroutine, which involves solving

$$\max \left\{ \sum_{x \in \mathcal{S}} p'(x) u_n(x) : p' \in \mathcal{C}_{t,\delta}(s,a) \right\}$$

  where u_n is a value function (at iteration n of EVI)
- EVI outputs a *conservative* policy (hence introducing unnecessary exploration), in particular when transition function p has a sparse support.
- UCRL3 remedies this issue by combining EVI with an adaptive support selection procedure.

UCRL3: Revisiting EVI

More specifically, at each iteration n of EVI:

- We first compute $\widetilde{S}_{s,a} \subset \mathcal{S}$, an approximation of the support of $p(\cdot|s,a)$, using NOSS (Algorithm 2 in the paper).
- Then, we solve

$$\max \Big\{ \sum_{x \in \mathcal{S}} p'(x) u_n(x) : p' \in \mathcal{C}_{t,\delta}(s,a) \text{ and } \mathrm{supp}(p') = \widetilde{S}_{s,a} \Big\}$$

This combined algorithm is called EVI-NOSS and outputs a near-optimistic policy.

For the complete pseudo-code of UCRL3, we refer to the paper.

UCRL3: Local Diameter

Definition (Local Diameter of State s)

Consider state $s \in \mathcal{S}$. For $s_1, s_2 \in \cup_{a \in \mathcal{A}} \text{supp}\big(p(\cdot|s,a)\big)$ with $s_1 \neq s_2$, let $T^\pi(s_1, s_2)$ denote the number of steps it takes to get to s_2 starting from s_1 and following policy π. Then, the local diameter of MDP M for s is defined as

$$D_s := \max_{s_1, s_2 \in \cup_a \text{supp}(p(\cdot|s,a))} \min_\pi \mathbb{E}[T^\pi(s_1, s_2)].$$

- D_s refines the (global) diameter (Jaksch et al., 2010).
- For all s, $D_s \leq D$, and for some states $D_s \ll D$.

UCRL3: Regret

Theorem (Regret of UCRL3)

With probability higher than $1-\delta$, uniformly over all $T \geq 3$, the regret under UCRL3 satisfies:

$$\mathfrak{R}(T) \leq \mathcal{O}\Big(\Big[\sqrt{\sum\nolimits_{s,a} \max(D_s^2 L_{s,a}, 1)} + D\Big]\sqrt{T\log(T/\delta)}\Big),$$

where $L_{s,a} := \big(\sum_{x\in\mathcal{S}} \sqrt{p(x|s,a)\big(1-p(x|s,a)\big)}\big)^2$ *denotes the local effective support of* (s,a).

UCRL3: Regret

Theorem (Regret of UCRL3)

With probability higher than $1-\delta$, uniformly over all $T \geq 3$, the regret under UCRL3 satisfies:

$$\mathfrak{R}(T) \leq \mathcal{O}\Big(\Big[\sqrt{\sum_{s,a} \max(D_s^2 L_{s,a}, 1)} + D\Big]\sqrt{T \log(T/\delta)}\Big),$$

where $L_{s,a} := \big(\sum_{x \in \mathcal{S}} \sqrt{p(x|s,a)\big(1-p(x|s,a)\big)}\big)^2$ denotes the local effective support of (s,a).

Note that $L_{s,a} \leq K_{s,a} - 1$ (with $K_{s,a} := |\mathrm{supp}\big(p(\cdot|s,a)\big)|$). Hence,

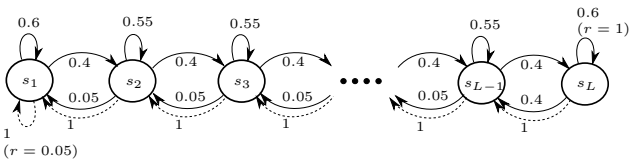
$$\mathfrak{R}(T) \leq \widetilde{\mathcal{O}}\Big(\Big[\sqrt{\sum_{s,a} \max(D_s^2 K_{s,a}, 1)} + D\Big]\sqrt{T}\Big).$$

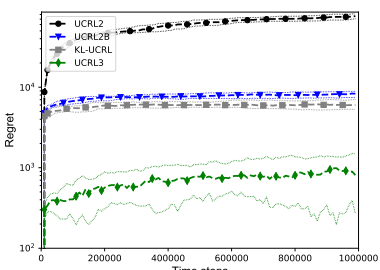
State-of-the-Art Regret Bounds

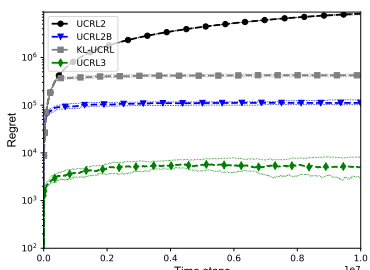
Algorithm	Regret bound
UCRL2 (Jaksch et al., 2010)	$\mathcal{O}\Big(DS\sqrt{AT\log(T/\delta)}\Big)$
KL-UCRL (Filippi et al., 2010)	$\mathcal{O}\Big(DS\sqrt{AT\log(\log(T)/\delta)}\Big)$
KL-UCRL (Talebi et al., 2018)	$\mathcal{O}\Big(\Big[D+\sqrt{S\sum_{s,a}\max(\mathbb{V}_{s,a},1)}\Big]\sqrt{T\log(\log(T)/\delta)}\Big)$
SCAL$^+$ (Qian et al., 2019)	$\mathcal{O}\Big(D\sqrt{\sum_{s,a}K_{s,a}T\log(T/\delta)}\Big)$
UCRL2B (Fruit et al., 2019)	$\mathcal{O}\Big(\sqrt{D\sum_{s,a}K_{s,a}T\log(T)\log(T/\delta)}\Big)$
UCRL3 **(This Paper)**	$\mathcal{O}\Big(\big(D+\sqrt{\sum_{s,a}\max(D_s^2L_{s,a},1)}\big)\sqrt{T\log(T/\delta)}\Big)$
Lower Bound (Jaksch et al., 2010)	$\Omega\big(\sqrt{DSAT}\big)$

Numerical Experiments

UCRL3 vs. existing algorithms in RiverSwim: $L=6$ (left) vs. $L=25$ (right)

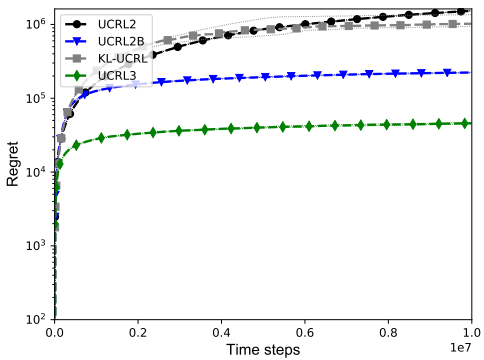






Numerical Experiments

UCRL3 vs. existing algorithms in a 100-state randomly generated MDP using Garnet (Bhatnagar et al., 2009)



- We used a VI approach: what about PI, MPI instead?
- What about using structured bandit algorithms ?
- What are generic lower bounds ?
- What if MDP transitions are not arbitrary but structured?
- What about function approximation? large/continuous states space?
  - Chowdhury, Gopalan, Maillard. Reinforcement Learning in Parametric MDPs with Exponential Families, AISTATS, 2021.

▷ From observations, build **posterior** on **reward** function (using e.g. Beta, Gaussian) $\pi^{\mathbf{r}}$ and on **transitions** $\pi^{\mathbf{p}}$ using Dirichlet.

$$\textbf{Dirichlet} \quad \pi^{\mathbf{p}} = D(n_t(s,a,s_1), \dots, n_t(s,1,s_{|S|}))$$

▷ **Sample** an MDP $\tilde{M}$:

$$\forall s, a, \tilde{r}(s,a) \sim \pi^{\mathbf{r}}(s,a), \tilde{p}(\cdot|s,a) \sim \pi^{\mathbf{p}}(s,a)$$

▷ Compute best policy for $\tilde{M}$ using e.g. Value-Iteration.
▷ Execute it until stopping criterion is met : e.g. nb of observations on one (s,a) as doubled.

▷ From observations, build **posterior** on **reward** function (using e.g. Beta, Gaussian) $\pi^{\mathbf{r}}$ and on **transitions** $\pi^{\mathbf{p}}$ using Dirichlet.

$$\textbf{Dirichlet} \quad \pi^{\mathbf{p}} = D(n_t(s,a,s_1), \dots, n_t(s,1,s_{|S|}))$$

▷ **Sample** an MDP $\tilde{M}$:

$$\forall s, a, \tilde{r}(s,a) \sim \pi^{\mathbf{r}}(s,a), \tilde{p}(\cdot|s,a) \sim \pi^{\mathbf{p}}(s,a)$$

▷ Compute best policy for $\tilde{M}$ using e.g. Value-Iteration.
▷ Execute it until stopping criterion is met : e.g. nb of observations on one (s,a) as doubled.

Careful: Initial analysis wrong, due to lack of optimism. Requires correction.
(Osband, von Roy 2016) `https://arxiv.org/pdf/1608.02731.pdf`
(Agrawal, Jia 2017) `https://dl.acm.org/doi/abs/10.5555/3294771.3294884`
If $N_t(s,a) < \eta$, $\tilde{p}(\cdot|s,a)$ chosen optimistically.

Table of contents

facebook
Artificial Intelligence Research

Exploration-Exploitation in Reinforcement Learning (Part2)

Alessandro Lazaric
Facebook AI Research (on leave from Inria Lille)

The Three Ingredients Recipe

1. Build accurate estimators
2. Evaluate the uncertainty of the prediction
3. Define a mechanism to combine estimation and uncertainty

The Three Ingredients Recipe

Optimism in face of uncertainty

1. Build accurate estimators

$$\widehat{M}_t \Rightarrow g^{\pi}_{\widehat{M}_t}$$

2. Evaluate the uncertainty of the estimators

$$B^r_t(s,a) := \left[\widehat{r}_t(s,a) - \beta^r_t(s,a),\ \widehat{r}_t(s,a) + \beta^r_t(s,a)\right]$$

$$B^p_t(s,a) := \left\{p(\cdot|s,a) \in \Delta(\mathcal{S}) :\ \|p(\cdot|s,a) - \widehat{p}_t(\cdot|s,a)\|_1 \leq\ \beta^p_t(s,a)\right\}$$

3. Define a mechanism to combine estimation and uncertainty

$$\pi_t = \arg\max_{\pi} \max_{M \in \mathcal{M}_t} g^{\pi}_M$$

The Three Ingredients Recipe

Posterior Sampling

1. Build accurate estimators
2. Evaluate the uncertainty of the estimators

$$\forall \Theta, \quad \mathbb{P}(M^* \in \Theta | H_t, \mu_1) = \mu_t(\Theta) \qquad \mu_t \text{ updated using Bayes' rule}$$

3. Define a mechanism to combine estimation and uncertainty

$$\pi_t = \arg\max_{\pi} g^{\pi}_{\widetilde{M}_t}, \qquad \widetilde{M}_t \sim \mu_t$$

"Practical" Limitations

Optimism in face of uncertainty

- Confidence intervals

$$\beta_t^r(s,a) \propto \sqrt{\frac{\log(N_t(s,a)/\delta)}{N_t(s,a)}} \qquad \beta_t^p(s,a) \propto \sqrt{\frac{S\log(N_t(s,a)/\delta)}{N_t(s,a)}}$$

- Solving

$$\pi_t = \arg\max_{\pi} \max_{M \in \mathcal{M}_t} g_M^\pi$$

Posterior sampling

- Posterior (dynamics for any state-action pair)

$$\text{Dirichlet}\big(N_t(s_1'|s,a), N_t(s_2'|s,a), \ldots, N_t(s_S'|s,a)\big)$$

- Update/sample from a unstructured/non-conjugate posteriors

Count-based Exploration

General Scheme

1. Estimate a "proxy" for the number of visits $\widetilde{N}(s_t)$
2. Add an exploration bonus to the rewards

$$\widetilde{r}_t^+ = r_t + c\sqrt{\frac{1}{\widetilde{N}(s_t)}}$$

3. Run any DeepRL algorithm on $\left\{(s_t, a_t, \widetilde{r}_t^+, s_{t+1})\right\}$

Count-based Exploration

General Scheme

1. Estimate a "proxy" for the number of visits $\widetilde{N}(s_t)$
2. Add an exploration bonus to the rewards

$$\widetilde{r}_t^+ = r_t + c\sqrt{\frac{1}{\widetilde{N}(s_t)}}$$

3. Run any DeepRL algorithm on $\left\{(s_t, a_t, \widetilde{r}_t^+, s_{t+1})\right\}$

⚠ What happened to optimism in the dynamics?

Extended Value Iteration

$$
\begin{aligned}
v_{n+1}(s) = \mathcal{L}_t v_n(s) &= \max_{(a,r,p)\in\mathcal{A}(s)\times B_t^r(s,a)\times B_t^p(s,a)} \left\{ r + p^\mathsf{T} v_n \right\} \\
&= \max_{a\in\mathcal{A}(s)} \left\{ \max_{r\in B_t^r(s,a)} r + \max_{p\in B_t^p(s,a)} p^\mathsf{T} v_n \right\} \\
&= \max_{a\in\mathcal{A}(s)} \left\{ \widehat{r}_t(s,a) + \beta_t^r(s,a) + \max_{p\in B_t^p(s,a)} p^\mathsf{T} v_n \right\} \\
&\leq \max_{a\in\mathcal{A}(s)} \left\{ \widehat{r}_t(s,a) + \beta_t^r(s,a) + \|p - p(\cdot|s,a)\|_1 \|v_n\|_\infty + \widehat{p}_t^\mathsf{T} v_n \right\} \\
&\leq \max_{a\in\mathcal{A}(s)} \left\{ \widehat{r}_t(s,a) + \beta_t^r(s,a) + C\sqrt{S}\beta_t^r(s,a) + \widehat{p}_t^\mathsf{T} v_n \right\}
\end{aligned}
$$

Extended Value Iteration

$$
\begin{aligned}
v_{n+1}(s) = \mathcal{L}_t v_n(s) &= \max_{(a,r,p)\in\mathcal{A}(s)\times B_t^r(s,a)\times B_t^p(s,a)} \left\{ r + p^\mathsf{T} v_n \right\} \\
&= \max_{a\in\mathcal{A}(s)} \left\{ \max_{r\in B_t^r(s,a)} r + \max_{p\in B_t^p(s,a)} p^\mathsf{T} v_n \right\} \\
&= \max_{a\in\mathcal{A}(s)} \left\{ \widehat{r}_t(s,a) + \beta_t^r(s,a) + \max_{p\in B_t^p(s,a)} p^\mathsf{T} v_n \right\} \\
&\leq \max_{a\in\mathcal{A}(s)} \left\{ \widehat{r}_t(s,a) + \beta_t^r(s,a) + \|p - p(\cdot|s,a)\|_1 \|v_n\|_\infty + \widehat{p}_t^\mathsf{T} v_n \right\} \\
&\leq \max_{a\in\mathcal{A}(s)} \left\{ \widehat{r}_t(s,a) + \beta_t^r(s,a) + C\sqrt{S}\beta_t^r(s,a) + \widehat{p}_t^\mathsf{T} v_n \right\}
\end{aligned}
$$

👍 Exploration bonus $(1 + C\sqrt{S})\beta_t^r(s,a)$ for the reward

Count-based Exploration

Tang et al. [2017]

Algorithm 1: Count-based exploration through static hashing, using SimHash

Define state preprocessor $g : \mathcal{S} \to \mathbb{R}^D$

(In case of SimHash) Initialize $A \in \mathbb{R}^{k \times D}$ with entries drawn i.i.d. from the standard Gaussian distribution $\mathcal{N}(0, 1)$

Initialize a hash table with values $n(\cdot) \equiv 0$

for each iteration j **do**

  Collect a set of state-action samples $\{(s_m, a_m)\}_{m=0}^{M}$ with policy π

  Compute hash codes through any LSH method, e.g., for SimHash, $\phi(s_m) = \text{sgn}(Ag(s_m))$

  Update the hash table counts $\forall m : 0 \le m \le M$ as $n(\phi(s_m)) \leftarrow n(\phi(s_m)) + 1$

  Update the policy π using rewards $\left\{ r(s_m, a_m) + \frac{\beta}{\sqrt{n(\phi(s_m))}} \right\}_{m=0}^{M}$ with any RL algorithm

- Use locality-sensitive hashing to discretize the input
  - Encode the state into a k-dim vector by random project
  - Use the sign to discretize
- Count on discrete hashed-states

Count-based Exploration

Tang et al. [2017]

Algorithm 1: Count-based exploration through static hashing, using SimHash

Define state preprocessor $g : \mathcal{S} \to \mathbb{R}^D$

(In case of SimHash) Initialize $A \in \mathbb{R}^{k \times D}$ with entries drawn i.i.d. from the standard Gaussian distribution $\mathcal{N}(0, 1)$

Initialize a hash table with values $n(\cdot) \equiv 0$

for each iteration j **do**

  Collect a set of state-action samples $\{(s_m, a_m)\}_{m=0}^{M}$ with policy π

  Compute hash codes through any LSH method, e.g., for SimHash, $\phi(s_m) = \text{sgn}(Ag(s_m))$

  Update the hash table counts $\forall m : 0 \leq m \leq M$ as $n(\phi(s_m)) \leftarrow n(\phi(s_m)) + 1$

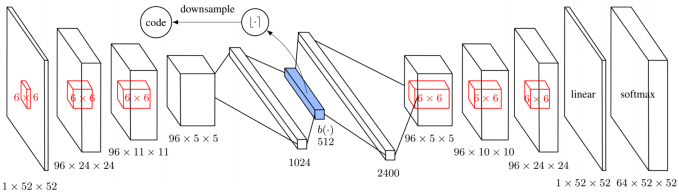
  Update the policy π using rewards $\left\{ r(s_m, a_m) + \frac{\beta}{\sqrt{n(\phi(s_m))}} \right\}_{m=0}^{M}$ with any RL algorithm

- Use locality-sensitive hashing to discretize the input
  - Encode the state into a k-dim vector by random project
  - Use the sign to discretize
- Count on discrete hashed-states

👎 Difficult to define a good hashing function

Count-based Exploration

Tang et al. [2017]



$$L\left(\{s_n\}_{n=1}^N\right) = -\frac{1}{N}\sum_{n=1}^{N}\left[\log p(s_n) - \frac{\lambda}{K}\sum_{i=1}^{D}\min\left\{(1-b_i(s_n))^2, b_i(s_n)^2\right\}\right]$$

- Entropy loss for the auto-encoder
- "Binarization" loss for the "projection"

Count-based Exploration

Tang et al. [2017]

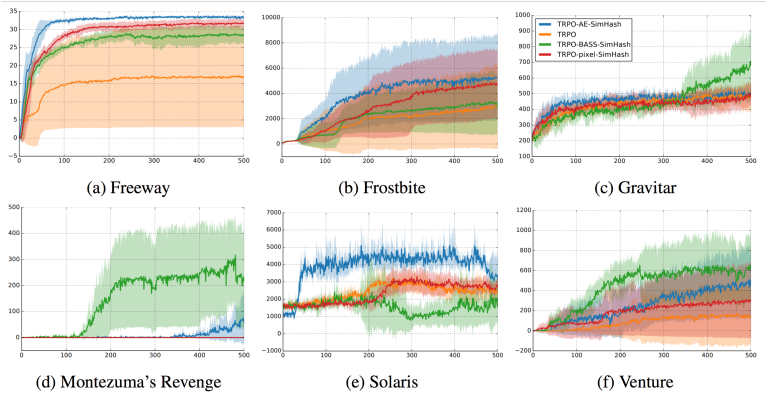
Algorithm 2: Count-based exploration using learned hash codes

Define state preprocessor $g : \mathcal{S} \to \{0,1\}^D$ as the binary code resulting from the autoencoder (AE)
Initialize $A \in \mathbb{R}^{k \times D}$ with entries drawn i.i.d. from the standard Gaussian distribution $\mathcal{N}(0,1)$
Initialize a hash table with values $n(\cdot) \equiv 0$
for each iteration j **do**
  Collect a set of state-action samples $\{(s_m, a_m)\}_{m=0}^{M}$ with policy π
  Add the state samples $\{s_m\}_{m=0}^{M}$ to a FIFO replay pool $\mathcal{R}$
  if j mod $j_{\text{update}} = 0$ **then**
    Update the AE loss function in Eq. (3) using samples drawn from the replay pool $\{s_n\}_{n=1}^{N} \sim \mathcal{R}$, for example using stochastic gradient descent
  Compute $g(s_m) = \lfloor b(s_m) \rceil$, the D-dim rounded hash code for s_m learned by the AE
  Project $g(s_m)$ to a lower dimension k via SimHash as $\phi(s_m) = \text{sgn}(Ag(s_m))$
  Update the hash table counts $\forall m : 0 \le m \le M$ as $n(\phi(s_m)) \leftarrow n(\phi(s_m)) + 1$
  Update the policy π using rewards $\left\{ r(s_m, a_m) + \frac{\beta}{\sqrt{n(\phi(s_m))}} \right\}_{m=0}^{M}$ with any RL algorithm

- Use all past history to update the AE
- AE should not be updated too often

Count-based Exploration

Tang et al. [2017]



(a) Freeway (b) Frostbite (c) Gravitar

(d) Montezuma's Revenge (e) Solaris (f) Venture

Count-based Exploration

Bellemare et al. [2016], Ostrovski et al. [2017]

- Density estimation over a countable set $\mathcal{X}$

$$\rho_n(x) = \rho(x|x_1, \ldots, x_n) \approx \mathbb{P}\big[X = x|x_1, \ldots, x_n\big]$$

- Recording probability

$$\rho'_n(x) = \rho(x|x_1, \ldots, x_n, x) \approx \mathbb{P}\big[X = x|x_1, \ldots, x_n, X_{n+1} = x\big]$$

- Pseudo "local" and "total" counts $\widetilde{N}_n(x)$ and $\widetilde{N}_n(x)$ s.t.

$$\frac{\widetilde{N}_n(x)}{\widetilde{n}} = \rho_n(x); \qquad \frac{\widetilde{N}_n(x)+1}{\widetilde{n}+1} = \rho'_n(x) \;\; \Rightarrow \widetilde{N}_n(x) = \frac{\rho_n(x)(1-\rho'_n(x))}{\rho'_n(x) - \rho_n(x)} = \widetilde{n}\rho_n(x)$$

Count-based Exploration

Bellemare et al. [2016], Ostrovski et al. [2017]

- Density estimation over a countable set $\mathcal{X}$

$$\rho_n(x) = \rho(x|x_1, \ldots, x_n) \approx \mathbb{P}\big[X = x|x_1, \ldots, x_n\big]$$

- Recording probability

$$\rho'_n(x) = \rho(x|x_1, \ldots, x_n, x) \approx \mathbb{P}\big[X = x|x_1, \ldots, x_n, X_{n+1} = x\big]$$

- Pseudo "local" and "total" counts $\widetilde{N}_n(x)$ and $\widetilde{N}_n(x)$ s.t.

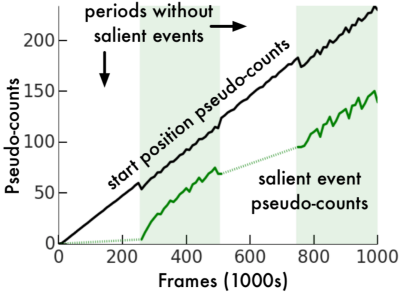
$$\frac{\widetilde{N}_n(x)}{\widetilde{n}} = \rho_n(x); \qquad \frac{\widetilde{N}_n(x)+1}{\widetilde{n}+1} = \rho'_n(x) \;\; \Rightarrow \widetilde{N}_n(x) = \frac{\rho_n(x)(1-\rho'_n(x))}{\rho'_n(x)-\rho_n(x)} = \widetilde{n}\rho_n(x)$$

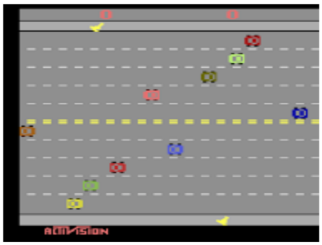
👍 Any density estimation algorithm (accurate for images)

👎 Density estimation in continuous spaces is hard

Count-based Exploration

Bellemare et al. [2016], Ostrovski et al. [2017]





Count-based Exploration

Bellemare et al. [2016], Ostrovski et al. [2017]

Montezuma!

Prediction-based Exploration

Burda et al. [2018]

Sources of prediction errors

1. Amount of data 👍
2. Stochasticity (e.g., noisy-TV) 👎
3. Model misspecification 👎
4. Learning dynamics 👎

Prediction-based Exploration

Burda et al. [2018]

- Randomly initialize two instances of the same NN (target θ_* and prediction θ_0)

$$f_{\theta_*} : \mathcal{S} \to \mathbb{R}; \qquad f_\theta : \mathcal{S} \to \mathbb{R}$$

- Train the prediction network minimizing loss w.r.t. the target network

$$\theta_n = \arg\min_\theta \sum_{t=1}^{n} \Big(f_\theta(s_t) - f_{\theta_*}(s_t) \Big)^2$$

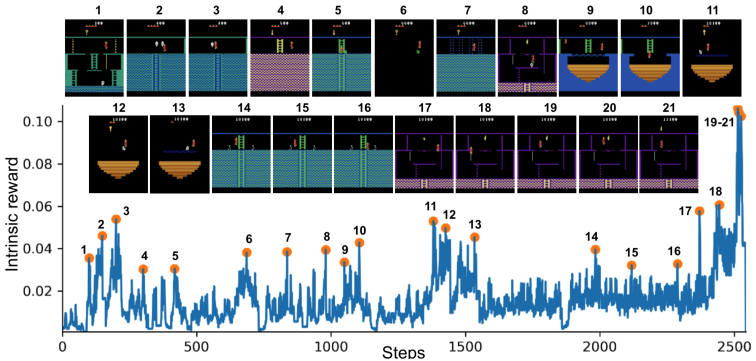
- Build "intrinsic" reward

$$r_t^E = \Big| f_\theta(s_t) - f_{\theta_*}(s_t) \Big|$$

- 👍 No influence from stochastic transitions
- 👍 No model misspecification (f_θ can exactly predict f_{θ_*})
- 👍 Influence of learning dynamics can be reduced

Prediction-based Exploration

Burda et al. [2018]



Prediction-based Exploration

Burda et al. [2018]

General architecture

- Separate extrinsic r_t^I and intrinsic reward r_t^E
- PPO with two heads to estimate V^I and V^E
- Greedy policy w.r.t. $V^I + cV^E$

"Tricks"

- Rewards should be in the same range
- Use different discount factors for intrinsic and extrinsic rewards

Prediction-based Exploration

Burda et al. [2018]

How to do exploration in Deep RL?

- Bootstrap: use multiple versions.
- Add noise to parameters
- Use Bayesian updates

Randomized Exploration

General Scheme

1. Estimate the parameters θ for either policy or value function
2. Add randomness to the parameters $\widetilde{\theta} = \theta + \text{noise}$
3. Run the corresponding (greedy) policy

Randomized Exploration

General Scheme

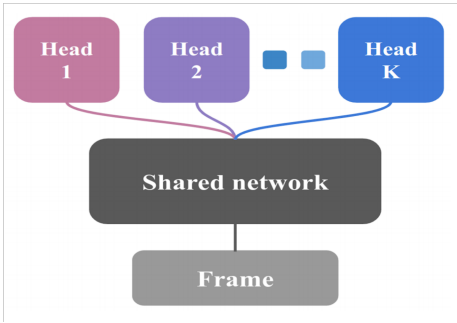
1. Estimate the parameters θ for either policy or value function
2. Add randomness to the parameters $\widetilde{\theta} = \theta + \text{noise}$
3. Run the corresponding (greedy) policy

⚠ The randomness needs to represent "uncertainty"

Bootstrap DQN

Osband et al. [2016]

- Define multiple value functions Q_k
- Update functions with different datasets
- Share part of the architecture



Bootstrap DQN

Osband et al. [2016]

Algorithm 1 Bootstrapped DQN

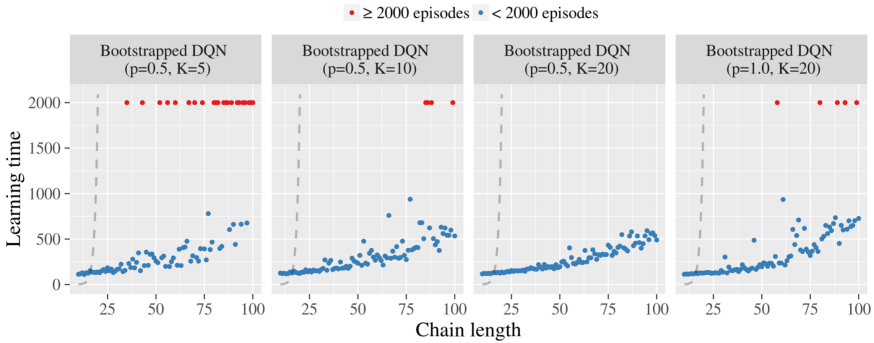
Input: Value function networks Q with K outputs $\{Q_k\}_{k=1}^{K}$. Masking distribution M.
Let B be a replay buffer storing experience for training.
for each episode **do**
    Obtain initial state from environment s_0
    Pick a value function to act using $k \sim \text{Uniform}\{1, \ldots, K\}$
    for step $t = 1, \ldots$ until end of episode **do**
        Pick an action according to $a_t \in \arg\max_a Q_k(s_t, a)$
        Receive state s_{t+1} and reward r_t from environment, having taking action a_t
        Sample bootstrap mask $m_t \sim M$
        Add $(s_t, a_t, r_{t+1}, s_{t+1}, m_t)$ to replay buffer B
    end for
end for

- M_t determines the type of bootstrapping strategy

$$g_t^k = m_t^k \big(y_t^Q - Q_k(s_t, a_t; \theta)\big) \nabla_\theta Q_k(s_t, a_t, ; \theta)$$

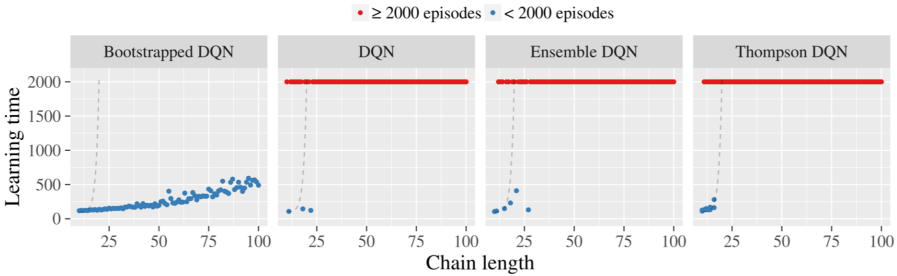
Bootstrap DQN

Osband et al. [2016]



Bootstrap DQN

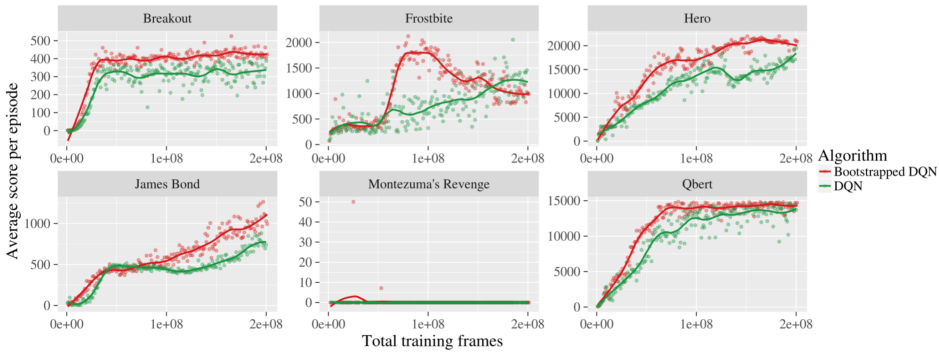
Osband et al. [2016]



- Ensemble DQN: ensemble policy?
- Thompson DQN: resample at each step

Bootstrap DQN

Osband et al. [2016]



Noisy Networks

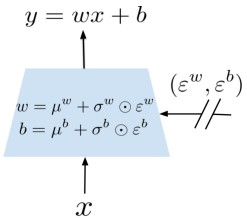
Fortunato et al. [2017]

- Normal NN layer $y = wx + b$
- Double the parameters with mean and variance $w \to \mu^w, \sigma^w$ and $b \to \mu^b, \sigma^b$
- Whenever a layer is evaluated draw $\varepsilon^w, \varepsilon^b \sim \mathcal{D}$
- Evaluate the "random" layer as $y = (\mu^w + \sigma^w \odot \varepsilon^w) + \mu^b + \sigma^b \odot \varepsilon^b$
- Let $\zeta = (\mu^w, \sigma^w, \mu^b, \sigma^b)$, define the expected loss

$$\overline{L}(\zeta) = \mathbb{E}_\varepsilon\big[L(\zeta, \varepsilon)\big]$$

- Gradient estimation

$$\nabla_\zeta \overline{L}(\zeta) = \mathbb{E}_\varepsilon\big[\nabla_\zeta L(\zeta, \varepsilon)\big] \approx \frac{1}{n}\sum_{i=1}^{n} \nabla_\zeta L(\zeta, \varepsilon_i)$$



Noisy Networks

Fortunato et al. [2017]

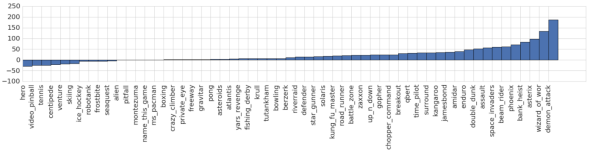
Noise models

- Independent noise $\varepsilon_{i,j}$ for each weight i at layer j
- Factorized noise $\varepsilon_{i,j} = f(\varepsilon_i)f(\varepsilon_j)$ (e.g., $f(x) = \text{sgn}(x)\sqrt{x}$)
- Independent noise for target and online networks

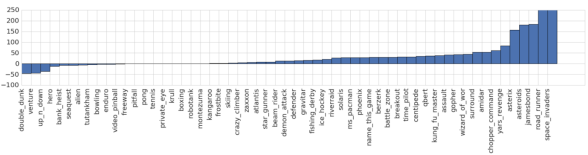
$$y_t = r_t + \max_{a'} Q(s'_t, a'; \varepsilon', \zeta^-); \qquad L_t(\zeta, \varepsilon) = \left(y_t - Q(s_t, a_t; \varepsilon, \zeta)\right)^2$$

Noisy Networks

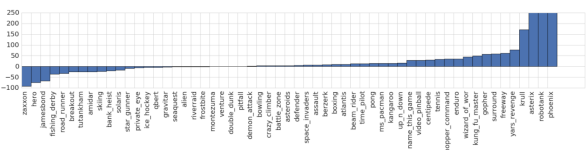
Fortunato et al. [2017]



(a) Improvement in percentage of NoisyNet-DQN over DQN (Mnih et al., 2015)



(b) Improvement in percentage of NoisyNet-Dueling over Dueling (Wang et al., 2016)



Bayesian DQN

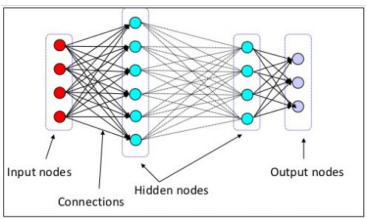
Azizzadenesheli et al. [2018]

⚠ Same tools as in linear bandit

1. Bayesian linear regression with given feature $\phi(s) \in \mathbb{R}^d$ and given target vector for each action y_a

$$\mu_a = (\Phi_a^\mathsf{T}\Phi_a)^{-1}\Phi_a^\mathsf{T} y_a \qquad \Sigma_a = \Phi_a^\mathsf{T}\Phi_a$$

2. Draw a weight vector at random $w_a \sim \mathcal{N}(\mu_a, \Sigma_a^{-1})$
3. Run the corresponding (greedy) policy $a_t = \arg\max_a w_a^\mathsf{T}\phi(s_t)$
4. Train ϕ with standard NN



Bayesian DQN

Azizzadenesheli et al. [2018]

Game	BDQN	DDQN	DDQN$^+$	Bootstrap	NoisyNet	CTS	Pixel	Reactor	Human	SC	SC^+	Step
Amidar	**5.52k**	0.99k	0.7k	1.27k	1.5k	1.03k	0.62k	1.18k	1.7k	22.9M	4.4M	100M
Alien	3k	2.9k	2.9k	2.44k	2.9k	1.9k	1.7k	**3.5k**	6.9k	-	36.27M	100M
Assault	**8.84k**	2.23k	5.02k	8.05k	3.1k	2.88k	1.25k	3.5k	1.5k	1.6M	24.3M	100M
Asteroids	**14.1k**	0.56k	0.93k	1.03k	2.1k	3.95k	0.9k	1.75k	13.1k	58.2M	9.7M	100M
Asterix	**58.4k**	11k	15.15k	19.7k	11.0	9.55k	1.4k	6.2k	8.5k	3.6M	5.7M	100M
BeamRider	8.7k	4.2k	7.6k	**23.4k**	14.7k	7.0k	3k	3.8k	5.8k	4.0M	8.1M	70M
BattleZone	**65.2k**	23.2k	24.7k	36.7k	11.9k	7.97k	10k	45k	38k	25.1M	14.9M	50M
Atlantis	3.24M	39.7k	64.76k	99.4k	7.9k	1.8M	40k	**9.5M**	29k	3.3M	5.1M	40M
DemonAttack	11.1k	3.8k	9.7k	82.6k	26.7k	**39.3k**	1.3k	7k	3.4k	2.0M	19.9M	40M
Centipede	**7.3k**	6.4k	4.1k	4.55k	3.35k	5.4k	1.8k	3.5k	12k	-	4.2M	40M
BankHeist	0.72k	0.34k	0.72k	**1.21k**	0.64k	1.3k	0.42k	1.1k	0.72k	2.1M	10.1M	40M
CrazyClimber	124k	84k	102k	**138k**	121k	112.9k	75k	119k	35.4k	0.12M	2.1M	40M
ChopperCmd	**72.5k**	0.5k	4.6k	4.1k	5.3k	5.1k	2.5k	4.8k	9.9k	4.4M	2.2M	40M
Enduro	1.12k	0.38k	0.32k	1.59k	0.91k	0.69k	0.19k	**2.49k**	0.31k	0.82M	0.8M	30M
Pong	**21**	18.82	**21**	20.9	**21**	20.8	17	20	9.3	1.2M	2.4M	5M

Randomized Prior

Osband et al. [2018]

Computational generation of posterior samples for linear Bayesian regression

- Let $f_\theta(x) = x^\mathsf{T}\theta$ and $y_i = x_i^\mathsf{T}\theta + \epsilon_i$
- Generate a sample $\theta|\mathcal{D}_n$ from the linear Bayesian posterior
  1. Compute $\widetilde{y}_i \sim \mathcal{N}(y_i, \sigma^2)$, $\widetilde{\theta} \sim \mathcal{N}(\theta_0, \Sigma_0)$ (prior)
  2. Compute

$$\arg\min_\theta \sum_{i=1}^n \|\widetilde{y}_i - f_\theta(x_i)\|_2^2 + \frac{\sigma^2}{\lambda}\|\widetilde{\theta} - \theta\|_2$$

Randomized Prior

Osband et al. [2018]

Algorithm 1 Randomized prior functions for ensemble posterior.

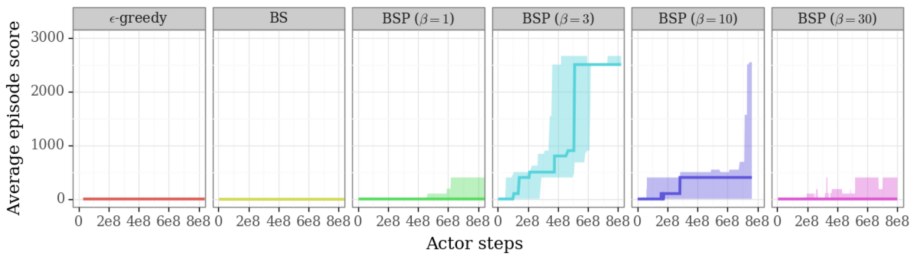
Require: Data $\mathcal{D} \subseteq \{(x,y)|x \in \mathcal{X}, y \in \mathcal{Y}\}$, loss function $\mathcal{L}$, neural model $f_\theta : \mathcal{X} \to \mathcal{Y}$,
Ensemble size $K \in \mathbb{N}$, noise procedure `data_noise`, distribution over priors $\mathcal{P} \subseteq \{\mathbb{P}(p)|p : \mathcal{X} \to \mathcal{Y}\}$.

1: **for** $k = 1, .., K$ **do**
2: initialize $\theta_k \sim$ Glorot initialization [23].
3: form $\mathcal{D}_k =$ `data_noise`$(\mathcal{D})$ (e.g. Gaussian noise or bootstrap sampling [50]).
4: sample prior function $p_k \sim \mathcal{P}$.
5: optimize $\nabla_{\theta|\theta=\theta_k} \mathcal{L}(f_\theta + p_k; \mathcal{D}_k)$ via ADAM [28].
6: **return** ensemble $\{f_{\theta_k} + p_k\}_{k=1}^K$.

$$\mathcal{L}_\gamma(\theta; \theta^-, p, \mathcal{D}) := \sum_{t \in \mathcal{D}} \left(r_t + \gamma \max_{a' \in \mathcal{A}} \overbrace{(f_{\theta^-} + p)}^{\text{target } Q}(s'_t, a') - \overbrace{(f_\theta + p)}^{\text{online } Q}(s_t, a_t) \right)^2$$

Randomized Prior

Osband et al. [2018]



Take home message

Regret-minimization in MDPs:

- Target **gain**, no discount is ok (intrinsic contraction, coalescence).
- Use **optimistic** principle inspired from UCB: **Extended VI** (UCRL2)

requires some care regarding transitions (UCRL3).

- Q: What about KL-UCRL, TS, IMED, etc.? PSRL is an attempt with caveats.
- **Exploration** in Deep RL: how to count?

"*The more applied you go, the stronger theory you need*"

Merci

odalricambrym.maillard@inria.fr
odalricambrymmaillard.wordpress.com