Generalization and Exploration via Value Function Randomization

Benjamin Van Roy

Collaborators: Hamid Maei, Ian Osband, Dan Russo, Zheng Wen
Online Optimization

reward

\[ r_t = R(y_t) \]
Online Optimization

reward
\[ r_t = R(y_t) \]

exploration versus exploitation
Exploration Strategies

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Exploration Strategies

<table>
<thead>
<tr>
<th>dithering</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>sometimes exploit</td>
<td></td>
</tr>
<tr>
<td>sometimes randomize</td>
<td></td>
</tr>
</tbody>
</table>

### Probability Distribution

<table>
<thead>
<tr>
<th>Action</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.175</td>
</tr>
<tr>
<td>2</td>
<td>0.175</td>
</tr>
<tr>
<td>3</td>
<td>0.7</td>
</tr>
<tr>
<td>4</td>
<td>0.35</td>
</tr>
</tbody>
</table>

*Statistically inefficient*

*Fails to write off bad actions*
## Exploration Strategies

<table>
<thead>
<tr>
<th>dithering</th>
<th>UCB</th>
</tr>
</thead>
<tbody>
<tr>
<td>sometimes exploit</td>
<td>maximize optimistic estimate</td>
</tr>
<tr>
<td>sometimes randomize</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Probability</th>
<th>Expected Reward</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.175</td>
<td>0</td>
</tr>
<tr>
<td>0.35</td>
<td>7.5</td>
</tr>
<tr>
<td>0.525</td>
<td>15</td>
</tr>
<tr>
<td>0.7</td>
<td>22.5</td>
</tr>
<tr>
<td>0.7</td>
<td>30</td>
</tr>
</tbody>
</table>

- dithering:
  - statistically inefficient
  - fails to write off bad actions

- UCB:
  - near-optimal exploration-exploitation tradeoff?
# Exploration Strategies

<table>
<thead>
<tr>
<th>dithering</th>
<th>UCB</th>
<th>Thompson sampling</th>
</tr>
</thead>
<tbody>
<tr>
<td>sometimes exploit</td>
<td>maximize optimistic estimate</td>
<td>sample ( \sim P(a \text{ is optimal}) )</td>
</tr>
<tr>
<td>sometimes randomize</td>
<td></td>
<td></td>
</tr>
<tr>
<td>probability</td>
<td>expected reward</td>
<td></td>
</tr>
<tr>
<td>0.7</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0.525</td>
<td>7.5</td>
<td></td>
</tr>
<tr>
<td>0.35</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>0.175</td>
<td>22.5</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>action</td>
<td>action</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>

- **statistically inefficient**
- **fails to write off bad actions**
- **near-optimal exploration-exploitation tradeoff?**
- **better than UCB?**
Example: Online Linear Programming
Example: Online Linear Programming

Polytopic action set $a_t \in \mathcal{P}$
Example: Online Linear Programming

Polytopic action set: $a_t \in \mathcal{P}$

Linear bandit feedback: $r_t = \theta^\top a_t + N(0, \sigma^2)$
**Example: Online Linear Programming**

<table>
<thead>
<tr>
<th>Polytopic action set</th>
<th>$a_t \in \mathcal{P}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear bandit feedback</td>
<td>$r_t = \theta^\top a_t + N(0, \sigma^2)$</td>
</tr>
<tr>
<td>Knowledge representation</td>
<td>$\theta_t \sim N(\mu_t, \Sigma_t)$ or $\theta_t \in \Theta_t$</td>
</tr>
</tbody>
</table>
Example: Online Linear Programming

<table>
<thead>
<tr>
<th>Polytopic action set</th>
<th>$a_t \in \mathcal{P}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear bandit feedback</td>
<td>$r_t = \theta^\top a_t + N(0, \sigma^2)$</td>
</tr>
<tr>
<td>Knowledge representation</td>
<td>$\theta_t \sim N(\mu_t, \Sigma_t)$ or $\theta_t \in \Theta_t$</td>
</tr>
</tbody>
</table>
Is TS “better” than UCB?
Is TS “better” than UCB?

Average Instantaneous Regret

Time Period

UCB [Abbasi-Yadkori et al, 2011]

UCB [Srinivas et al, 2012]
Is TS “better” than UCB?

![Chart showing comparison of Average Instantaneous Regret for UCB and TS over time](chart.png)
Is TS “better” than UCB?

![Graph comparing average instantaneous regret of TS and UCB over time]

- UCB [Abbasi-Yadkori et al, 2011]
- UCB [Srinivas et al, 2012]
- UCB-tuned
- TS
Is TS “better” than UCB?

Confidence set selection drives theory and practical performance of UCB.
UCB is Often Computationally Intractable
UCB is Often Computationally Intractable

- Consider online linear programming
UCB is Often Computationally Intractable

- Consider online linear programming

- Thompson sampling

  sample: \( \hat{\theta}_t \sim N(\mu_t, \Sigma_t) \)

  optimize: \( \max_{a_t \in \mathcal{P}} \hat{\theta}_t^\top a_t \)
UCB is Often Computationally Intractable

- Consider online linear programming

- Thompson sampling
  
  sample: $\hat{\theta}_t \sim N(\mu_t, \Sigma_t)$

  optimize: $\max_{a_t \in P} \hat{\theta}_t^\top a_t$

- UCB

  $\max_{a_t \in P} \max_{\hat{\theta} \in \Theta_t} \hat{\theta}^\top a_t$
UCB is Often Computationally Intractable

- Consider online linear programming

- Thompson sampling

\[
\text{sample: } \hat{\theta}_t \sim N(\mu_t, \Sigma_t) \\
\text{optimize: } \max_{a_t \in \mathcal{P}} \hat{\theta}_t^\top a_t
\]

- UCB

\[
\max_{a_t \in \mathcal{P}} \max_{\hat{\theta} \in \Theta_t} \hat{\theta}^\top a_t
\]

NP-hard
UCB is Often Computationally Intractable

- Consider online linear programming

- Thompson sampling

  sample: $\hat{\theta}_t \sim N(\mu_t, \Sigma_t)$

  optimize: $\max_{a_t \in \mathcal{P}} \hat{\theta}_t^\top a_t$

- UCB

  \[
  \max_{a_t \in \mathcal{P}} \max_{\hat{\theta} \in \Theta_t} \hat{\theta}^\top a_t
  \]

  NP-hard

  tractable for small # of vertices
UCB is Often Computationally Intractable

- Consider online linear programming

- Thompson sampling

  sample: $\hat{\theta}_t \sim N(\mu_t, \Sigma_t)$

  optimize: $\max_{a_t \in \mathcal{P}} \hat{\theta}_t^\top a_t$

- UCB

  $\max_{a_t \in \mathcal{P}} \max_{\hat{\theta} \in \Theta_t} \hat{\theta}^\top a_t$

- Computationally tractable version of UCB

  - Regret scaled by a factor of $d$  [Dani-Hayes-Kakade, 2008]
Summary on TS versus UCB
Summary on TS versus UCB

• Main points
  • TS outperforms UCB designed for analysis
  • TS slightly underperforms well-tuned UCB
  • TS often tractable when UCB is not
  • TS outperforms UCB designed for tractability
Summary on TS versus UCB

- **Main points**
  - TS outperforms UCB designed for analysis
  - TS slightly underperforms well-tuned UCB
  - TS often tractable when UCB is not
  - TS outperforms UCB designed for tractability

  - UCB/TS: TS $\cong$ randomized approximation of UCB
  - UCB results $\rightarrow$ TS results
  - problem-specific UCB results
  - bandit feedback, general reward models (dependencies among actions)
  - contextual, cautious, adversarial, etc.
Summary on TS versus UCB

- **Main points**
  - TS outperforms UCB designed for analysis
  - TS slightly underperforms well-tuned UCB
  - TS often tractable when UCB is not
  - TS outperforms UCB designed for tractability

  - UCB/TS: TS $\cong$ randomized approximation of UCB
  - UCB results $\rightarrow$ TS results
  - problem-specific UCB results
  - bandit feedback, general reward models (dependencies among actions)
  - contextual, cautious, adversarial, etc.

- **Russo-VR (2014): IT Analysis of Thompson Sampling**
  - simple analysis based on information theory
  - handles general feedback information structures
Troubling Example: Sparse Linear Bandit
Troubling Example: Sparse Linear Bandit

- A 1-sparse case

\[ r_t = \theta^\top a_t \]

\[ \theta \in \{0, 1\}^N \quad \|\theta\|_0 = 1 \]

uniform prior

\[ a_t = \text{“average over subset of components”} \]
Troubling Example: Sparse Linear Bandit

- A 1-sparse case

\[ r_t = \theta^\top a_t \]

- UCB/TS require $\Omega(d)$ samples to identify
  - Rule out one action per period

\[ \theta \in \{0, 1\}^N \quad \|\theta\|_0 = 1 \]

uniform prior

\[ a_t = \text{“average over subset of components”} \]
Troubling Example: Sparse Linear Bandit

• A 1-sparse case

\[ r_t = \theta^\top a_t \]

\[ \theta \in \{0, 1\}^N, \quad \|\theta\|_0 = 1 \]

uniform prior

\[ a_t = \text{“average over subset of components”} \]

• UCB/TS require \( \Omega(d) \) samples to identify
  • Rule out one action per period

• Easy to design algorithms for which \( \log_2(d) \) suffice
  • Binary search
Troubling Example: Assortment Optimization

$N$ customer types

many products, each geared toward a type
Troubling Example: Assortment Optimization

N customer types

customer of unknown type

many products, each geared toward a type
Troubling Example: Assortment Optimization

$N$ customer types

customer of unknown type

assortment of $M$ products

many products, each geared toward a type

learn customer type through sequence of interactions
Troubling Example: Assortment Optimization

$N$ customer types

customer of unknown type

learn customer type through sequence of interactions

assortment of $M$ products

many products, each geared toward a type

$N$ customer types

$M$ products

UCB/TS select single-type assortments
Troubling Example: Assortment Optimization

$N$ customer types

$M$ products, each geared toward a type

learn customer type through sequence of interactions

UCB/TS select single-type assortments

diversity can accelerate learning by a factor of $M$
Information-Directed Sampling (IDS)

\[
\min_{a_t} \left( \mathbb{E}_t [r^* - r_t] \right)^2 \frac{1}{I_t(a^*, y_t)} = \min_{a_t} \text{mutual information}
\]

\[
\text{squared expected regret}
\]
Information-Directed Sampling (IDS)

\[
\min_{a_t} \frac{(\mathbb{E}_t [r^* - r_t])^2}{I_t(a^*, y_t)} = \min_{a_t} \frac{\text{squared expected regret}}{\text{mutual information}}
\]

- Kills UCB/TS in aforementioned troubling examples
- Slight improvement in cases where UCB/TS work well
- Strong regret bounds
Information-Directed Sampling (IDS)

\[ \min_{a_t} \frac{(E_t[r^* - r_t])^2}{I_t(a^*, y_t)} = \min_{a_t} \text{squared expected regret} \]

- Kills UCB/TS in aforementioned troubling examples
- Slight improvement in cases where UCB/TS work well
- Strong regret bounds
- “Tractable” but more practical algorithmic work needed
- Is this the “right” information measure?
Information-Directed Sampling (IDS)

$$\min_{a_t} \frac{(\mathbb{E}_t[r^* - r_t])^2}{I_t(a^*, y_t)} = \min_{a_t} \frac{\text{squared expected regret}}{\text{mutual information}}$$

- Kills UCB/TS in aforementioned troubling examples
- Slight improvement in cases where UCB/TS work well
- Strong regret bounds

- “Tractable” but more practical algorithmic work needed
- Is this the “right” information measure?

- Russo-VR (2014): Learning to Optimize via IDS
reinforcement learning

$$r_t = R(y_t)$$

action $a_t$

outcome $y_t$

environment

Learning to Optimize

RLDM 2015
Reinforcement Learning

reward

\[ r_t = R(y_t) \]

action

\[ a_t \]

environment

outcome

\[ y_t \]

delayed consequences add complexity
Deep Exploration
Deep Exploration

\[ a_0 = 0 \quad r_0 = 0 \]

\[ a_0 = 1 \quad r_0 = 1 \]
Deep Exploration

\[ a_0 = 0 \]
\[ r_0 = 0 \]
\[ a_0 = 1 \]
\[ r_0 = 1 \]
Deep Exploration

only uncertain about this branch
Deep Exploration

invest now to learn downstream

only uncertain about this branch
Deep Exploration

Only uncertain about this branch

Invest now to learn downstream

Delayed consequences call for deep exploration
“Efficient RL” Literature
“Efficient RL” Literature

• Deep exploration enables polynomial time RL
  [Kearns-Singh, 2002]
“Efficient RL” Literature

- Deep exploration enables polynomial time RL
  [Kearns-Singh, 2002]

- Improving understanding, algorithms, regret bounds
  [Brafman-Tennenholtz, 2002; Kakade, 2003; Strehl et al, 2006; Szita-Szepesvari, 2008; Jaksch et al, 2010; Li-Littman, 2010; Osband et al, 2014]
“Efficient RL” Literature

- Deep exploration enables polynomial time RL
  [Kearns-Singh, 2002]

- Improving understanding, algorithms, regret bounds
  [Brafman-Tennenholtz, 2002; Kakade, 2003; Strehl et al, 2006; Szita-Szepesvari, 2008; Jaksch et al, 2010; Li-Littman, 2010; Osband et al, 2014]

- Focus has been on case of *tabula rasa* MDPs
“Efficient RL” Literature

- Deep exploration enables polynomial time RL
  [Kearns-Singh, 2002]

- Improving understanding, algorithms, regret bounds
  [Brafman-Tennenholtz, 2002; Kakade, 2003; Strehl et al, 2006; Szita-Szepesvari, 2008; Jaksch et al, 2010; Li-Littman, 2010; Osband et al, 2014]

- Focus has been on case of *tabula rasa* MDPs

- Some find this line of work practically useless
“Efficient RL” Literature

• Deep exploration enables polynomial time RL
  [Kearns-Singh, 2002]

• Improving understanding, algorithms, regret bounds
  [Brafman-Tennenholtz, 2002; Kakade, 2003; Strehl et al, 2006; Szita-Szepesvari, 2008; Jaksch et al, 2010; Li-Littman, 2010; Osband et al, 2014]

• Focus has been on case of tabula rasa MDPs

• Some find this line of work practically useless
  • Realistic problems require generalization
  • Sometimes they also require deep exploration
Two Cultures?
Two Cultures?
Two Cultures?

agenda: design practical RL algorithms that combine deep exploration and generalization
Toward Deep Exploration + Generalization
Toward Deep Exploration + Generalization


- Specialized and computationally intractable
Toward Deep Exploration + Generalization

- Specialized and computationally intractable

- Deep exploration + interpolative value function generalization [Pazis-Parr, 2013]
- Extrapolation is important in high-dimensional spaces
Toward Deep Exploration + Generalization

  • Specialized and computationally intractable

• Deep exploration + interpolative value function generalization [Pazis-Parr, 2013]
  • Extrapolation is important in high-dimensional spaces

• Deep exploration + value function generalization for deterministic systems [Wen-VR, 2013]
  • Brittle and does not accommodate stochasticity
Toward Deep Exploration + Generalization

- **Model-based approaches** [Kearns-Koller, 1999; Abbasi-Yadkori, 2011; Ibrahimi et al, 2012; Osband-VR, 2014]
  - Specialized and computationally intractable

- Deep exploration + interpolative value function generalization [Pazis-Parr, 2013]
  - Extrapolation is important in high-dimensional spaces

- Deep exploration + value function generalization for deterministic systems [Wen-VR, 2013]
  - Brittle and does not accommodate stochasticity

**new approach: value function randomization** [Dearden et al, 1998]
Episodic RL Framework
Episodic RL Framework

- Episodic learning in a finite-horizon MDP
Episodic RL Framework

- Episodic learning in a finite-horizon MDP

- Reinforcement learning algorithm
Episodic RL Framework

- Episodic learning in a finite-horizon MDP

- Reinforcement learning algorithm
  - Given observations made through episode $\ell - 1$
Episodic RL Framework

- Episodic learning in a finite-horizon MDP

\[ s_0, a_0, r_0 \rightarrow s_1, a_1, r_1 \rightarrow s_2, a_2, r_2 \rightarrow \ldots \rightarrow s_{H-1}, a_{H-1}, r_{H-1} \rightarrow s_H \]

- Reinforcement learning algorithm
  - Given observations made through episode \( \ell - 1 \)
  - Select policy \( \pi^\ell = (\pi_0^\ell, \ldots, \pi_{H-1}^\ell) \)
Episodic RL Framework

- Episodic learning in a finite-horizon MDP

\begin{align*}
\pi^\ell &= (\pi^\ell_0, \ldots, \pi^\ell_{H-1}) \\
a^\ell_h &= \pi^\ell_h(s^\ell_h)
\end{align*}
Episodic RL Framework

• Episodic learning in a finite-horizon MDP

\[ \begin{align*}
S_0 & \xrightarrow{a_0} S_1 \xrightarrow{a_1} S_2 \xrightarrow{a_2} \ldots \xrightarrow{a_{H-1}} S_H \\
S_0 & \xrightarrow{r_0} S_1 \xrightarrow{r_1} S_2 \xrightarrow{r_2} \ldots \xrightarrow{r_{H-1}} S_H
\end{align*} \]

• Reinforcement learning algorithm
  • Given observations made through episode \( \ell - 1 \)
  • Select policy \( \pi^\ell = (\pi_0^\ell, \ldots, \pi_{H-1}^\ell) \)
  • Apply actions \( a^\ell_h = \pi^\ell_h (s^\ell_h) \)

• Regret

\[
\text{Regret}(T) = \frac{T}{H} \sum_{\ell=1}^{T/H} \left( V_0^*(s_0) - V_0^{\pi^\ell}(s_0) \right)
\]
Value Function Randomization
Value Function Randomization

- Generalize via value functions parameterized by $\theta$
Value Function Randomization

• Generalize via value functions parameterized by $\theta$

• To select $\pi^\ell$
  • Sample statistically plausible parameters $\theta$
  • Use greedy policy
Value Function Randomization

• Generalize via value functions parameterized by $\theta$

• To select $\pi^\ell$
  • Sample statistically plausible parameters $\theta$
  • Use greedy policy

• How does this accomplish deep exploration?
Value Function Randomization

- Generalize via value functions parameterized by $\theta$

- To select $\pi^\ell$
  - Sample statistically plausible parameters $\theta$
  - Use greedy policy

- How does this accomplish deep exploration?
  - All downstream uncertainty is reflected in value variance
Value Function Randomization

• Generalize via value functions parameterized by $\theta$

• To select $\pi^\ell$
  • Sample statistically plausible parameters $\theta$
  • Use greedy policy

• How does this accomplish deep exploration?
  • All downstream uncertainty is reflected in value variance

• How to sample?
Randomized Least-Squares Value Iteration (RLSVI)
Randomized Least-Squares Value Iteration (RLSVI)

- Linearly parameterized value function

\[ \tilde{Q}_{\theta_h}^h(s, a) = \sum_{k=1}^{K} \theta_{hk} \phi_{hk}(s, a) \]
Randomized Least-Squares Value Iteration (RLSVI)

- Linearly parameterized value function

\[
\tilde{Q}^{\theta_h}(s, a) = \sum_{k=1}^{K} \theta_{hk} \phi_{hk}(s, a)
\]
Randomized Least-Squares Value Iteration (RLSVI)

- Linearly parameterized value function

\[ \tilde{Q}^{\theta_h}(s, a) = \sum_{k=1}^{K} \theta_{hk} \phi_{hk}(s, a) \]

- Least-squares value iteration
  - Typically coupled with Boltzmann or \( \varepsilon \)-greedy exploration

“basis functions”
Randomized Least-Squares Value Iteration (RLSVI)

- Linearly parameterized value function

\[ \tilde{Q}^{\theta_h}(s, a) = \sum_{k=1}^{K} \theta_{hk} \phi_{hk}(s, a) \]

- Least-squares value iteration
  - Typically coupled with Boltzmann or \( \varepsilon \)-greedy exploration

- Randomized least-squares value iteration
  - Adds Gaussian noise to regression coefficients
  - Noise drawn based on error covariance matrices
  - Applies greedy policy
RLSVI
RLSVI

• Least-squares value iteration

$$\min_{\hat{\theta}_h} \left( \frac{1}{\sigma^2} \sum_{\ell=1}^{L} \left( \tilde{Q}_{h}^{\hat{\theta}_h}(s_{h}^{\ell}, a_{h}^{\ell}) - \left( r_{h}^{\ell} + \max_{\alpha} \tilde{Q}_{h+1}^{\hat{\theta}_h+1}(s_{h+1}^{\ell}, \alpha) \right) \right)^2 + \lambda \| \hat{\theta}_h \|_2^2 \right)$$
RLSVI

• Least-squares value iteration

\[ \hat{\theta}_h \leftarrow \frac{1}{\sigma^2} \left( \frac{1}{\sigma^2} A^\top A + \lambda I \right)^{-1} A^\top b \]

\[ \Sigma_h \leftarrow \left( \frac{1}{\sigma^2} A^\top A + \lambda I \right)^{-1} \]
RLSVI

- Least-squares value iteration

\[
\hat{\theta}_h \leftarrow \frac{1}{\sigma^2} \left( \frac{1}{\sigma^2} A^\top A + \lambda I \right)^{-1} A^\top b
\]

\[
\Sigma_h \leftarrow \left( \frac{1}{\sigma^2} A^\top A + \lambda I \right)^{-1}
\]

- Randomized least-squares value iteration

\[
\bar{\theta}_h \leftarrow \frac{1}{\sigma^2} \left( \frac{1}{\sigma^2} A^\top A + \lambda I \right)^{-1} A^\top b
\]

\[
\hat{\theta}_h \sim N(\bar{\theta}_h, \Sigma_h)
\]
Regret Analysis
Regret Analysis

• Preliminary analysis of \textit{tabula rasa} case [Osband et al, 2015]
  • Assumes particular prior over episodic MDPs
Regret Analysis

- Preliminary analysis of *tabula rasa* case [Osband et al, 2015]
  - Assumes particular prior over episodic MDPs

- Regret bound

\[ \mathbb{E} [\text{Regret}(T)] = \tilde{O} \left( \sqrt{SATH} \right) \]
Regret Analysis

- Preliminary analysis of *tabula rasa* case [Osband et al, 2015]
  - Assumes particular prior over episodic MDPs

- Regret bound

\[
\mathbb{E} \left[ \text{Regret}(T) \right] = \tilde{O} \left( \sqrt{SATH} \right)
\]

- Compare against

\[
\mathbb{E} \left[ \text{Regret}(T) \right] = \tilde{O} \left( S\sqrt{ATH} \right)
\]
Regret Analysis

• Preliminary analysis of *tabula rasa case* [Osband et al, 2015]
  • Assumes particular prior over episodic MDPs

• Regret bound

\[ \mathbb{E} [\text{Regret}(T)] = \tilde{O} \left( \sqrt{SATH} \right) \]

• Compare against

\[ \mathbb{E} [\text{Regret}(T)] = \tilde{O} \left( S\sqrt{ATH} \right) \]

• Bound implies deep exploration
Regret Analysis

• Preliminary analysis of *tabula rasa* case [Osband et al, 2015]
  • Assumes particular prior over episodic MDPs

• Regret bound

\[ \mathbb{E} [\text{Regret}(T)] = \tilde{O} \left( \sqrt{SATH} \right) \]

• Compare against

\[ \mathbb{E} [\text{Regret}(T)] = \tilde{O} \left( S\sqrt{ATH} \right) \]

• Bound implies deep exploration

• Regret analysis with generalization remains open
LSVI-Boltzmann vs. RLSVI
LSVI-Boltzmann vs. RLSVI

![Graph showing the comparison between LSVI-Boltzmann and RLSVI](image-url)
LSVI-Boltzmann vs. RLSVI

![Graph comparing Per-Episode Reward for LSVI-Boltzmann and RLSVI over different episodes. The x-axis represents episode number, and the y-axis represents the per-episode reward. The graph shows a consistent trend for both methods with LSVI-Boltzmann slightly outperforming RLSVI in terms of reward.](image-url)
Varying the Number of Basis Functions

The graph shows the per-episode reward for different values of $K$. The x-axis represents the episode number, ranging from 0 to 400, and the y-axis represents the per-episode reward, ranging from 0 to 1.

- $K=2$, represented by a red line.
- $K=5$, represented by a blue dashed line.
- $K=10$, represented by a green dotted line.
- $K=20$, represented by a black line.

As $K$ increases, the per-episode reward generally increases, indicating a faster learning rate.
Agnostic Learning

Degree of Mis-Specification

Cumulative Regret

- Cumulative Regret
- Upper Bound

Expected Normalized Distance

0
0.05
0.1
0.15

0
50
100
150
200
250
300
350
400

Cumulative Regret

0
50
100
150
200
250
300
350
400

Degree of Mis-Specification
Deeper Reinforcement Learning
Can we apply value function randomization with nonlinear parameterizations?
Deeper Reinforcement Learning

Can we apply value function randomization with nonlinear parameterizations?

deep exploration + deep learning
Can we apply value function randomization with nonlinear parameterizations?

Deep exploration + Deep learning

Randomize via bootstrap
Can we apply value function randomization with nonlinear parameterizations?

- Deep exploration + Deep learning
- Randomize via bootstrap
- Scattered experience replay