PC-PG: Policy Cover Directed Exploration for Provable Policy Gradient Learning

Joint work with Alekh Agarwal, Mikael Henaff, and Sham Kakade
Policy Optimization

[AlphaZero, Silver et.al, 17]

[OpenAI Five, 18]

[OpenAI, 19]
Can we design Provably Correct Policy Gradient algorithms?
Infinite Horizon Discounted MDPs

Policy: state to action
\[ \pi(s) \rightarrow a \]

Reward & Next State
\[ r(s, a), s' \sim P(\cdot | s, a) \]

Objective: \[ \max_\pi \mathbb{E}_{\pi, P} \left[ r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots \right] \]
Infinite Horizon Discounted MDPs

Policy: state to action

\[ \pi(s) \rightarrow a \]

Reward & Next State

\[ r(s, a), s' \sim P(\cdot | s, a) \]

Objective:

\[ \max_{\pi} \mathbb{E}_{\pi, P} \left[ r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots \right] \]

Assume \( s_0 \sim \mu_0 \) and we can only reset from \( \mu_0 \)
Policy Gradient Methods

e.g., Reinforce, Natural Policy Gradient, TRPO, PPO:

(Williams 92, Kakade 02, Schulman et al 15, 17)
Policy Gradient Methods

e.g., Reinforce, Natural Policy Gradient, TRPO, PPO:

(Williams 92, Kakade 02, Schulman et al 15, 17)
Policy Gradient Methods

e.g., Reinforce, Natural Policy Gradient, TRPO, PPO:

(Williams 92, Kakade 02, Schulman et al 15, 17)

\[ J(\pi_\theta) = \mathbb{E}_{\pi_\theta} \left[ r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots \right] \]
Policy Gradient Methods

e.g., Reinforce, Natural Policy Gradient, TRPO, PPO:

(Williams 92, Kakade 02, Schulman et al 15, 17)

\[
J(\pi_\theta) = \mathbb{E}_{\pi_\theta} \left[ r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots \right]
\]

\[
\theta = \theta + \eta \nabla_\theta J(\pi_\theta)
\]
Policy Gradient Methods

e.g., Reinforce, Natural Policy Gradient, TRPO, PPO:
(Williams 92, Kakade 02, Schulman et al 15, 17)

\[
J(\pi_\theta) = \mathbb{E}_{\pi_\theta} \left[ r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots \right]
\]

\[
\theta = \theta + \eta \nabla_\theta J(\pi_\theta)
\]

\[
\theta = \theta + \eta F_\theta^{-1} \nabla_\theta J(\pi_\theta)
\]
Policy Gradient Methods

e.g., Reinforce, Natural Policy Gradient, TRPO, PPO:
(Williams 92, Kakade 02, Schulman et al 15, 17)

\begin{align*}
J(\pi_\theta) &= \mathbb{E}_{\pi_\theta} \left[ r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots \right] \\
\theta &= \theta + \eta \nabla_\theta J(\pi_\theta) \\
\theta &= \theta + \eta F_\theta^{-1} \nabla_\theta J(\pi_\theta)
\end{align*}

Preconditioning w/ Fisher Information matrix
(TRPPO and PPO are variants of it)
Advantages of Policy Gradient Methods

Strong Agnostic guarantee:

Compete to the best policy in the given class: \( \tilde{\pi} \in \Pi \)
Advantages of Policy Gradient Methods

Strong Agnostic guarantee:

Compete to the best policy in the given class: $\tilde{\pi} \in \Pi$

under a “wide” reset distribution (e.g., see Agarwal et al 19)
Advantages of Policy Gradient Methods

Strong Agnostic guarantee:

Compete to the best policy in the given class: $\tilde{\pi} \in \Pi$

under a “wide” reset distribution (e.g., see Agarwal et al 19)

$$\sup_s \frac{d\tilde{\pi}(s)}{\mu_0(s)} < \infty$$
Advantages of Policy Gradient Methods

Strong Agnostic guarantee:

Compete to the best policy in the given class: \( \tilde{\pi} \in \Pi \)

under a “wide” reset distribution (e.g., see Agarwal et al 19)

\[
\sup_s \left| \frac{d\tilde{\pi}(s)}{\mu_0(s)} \right| < \infty
\]

State distribution of \( \tilde{\pi} \)

Initial/reset dist

Q-learning, Fitted Q iteration:

Realizability (\& Bellman Complete)

\( Q^* \in Q \)
Successful Story of PG:

Robot hand manipulation (OpenAI, 19)
Successful Story of PG:

Robot hand manipulation (OpenAI, 19)
Successful Story of PG:

Robot hand manipulation (OpenAI, 19)
Successful Story of PG:

Robot hand manipulation (OpenAI, 19)

A notable technique

Domain Randomization:
Successful Story of PG:

Robot hand manipulation (OpenAI, 19)

A notable technique
Domain Randomization:
Successful Story of PG:

Robot hand manipulation (OpenAI, 19)

A notable technique

Domain Randomization:

\[
\sup_s \left| \frac{d\tilde{\pi}(s)}{\mu_0(s)} \right| < \infty
\]

State dist of the best \( \tilde{\pi} \)

Initial/reset dist

Make \( \mu_0 \) as “wide” as possible!!
But PG Fails if initial condition does not hold, provably

Initialization: $S_0 + \text{random walk}$
But PG Fails if initial condition does not hold, provably

Initialization: $S_0 + \text{random walk}$

Extremely flatten gradient:
magnitude of gradient is exponentially small (even in higher order): $2^{-H}$

(e.g., see CPI from Kadade & Langford 02, and Agarwal et al 19)
The Optimization Landscape

Supervised Learning
The Optimization Landscape

Supervised Learning

• Gradient descent tends to just work
• Not sensitive to initialization
• Saddle point is not a problem
The Optimization Landscape

Supervised Learning

- Gradient descent tends to just work
- Not sensitive to initialization
- Saddle point is not a problem

Reinforcement Learning:

The Optimization Landscape

Supervised Learning

- Gradient descent tends to just work
- Not sensitive to initialization
- Saddle point is not a problem

Reinforcement Learning:

- Extremely flatten region even at initialization
- Due to lack of exploration
Bidirectional Combination Lock

$s_0$

chain 1

chain 2

optimal reward

suboptimal reward
Bidirectional Combination Lock

Initial state $S_0$

chain 1

chain 2

optimal reward

suboptimal reward
Bidirectional Combination Lock

Initial state

\( S_0 \)

\( r = 5 \)

optimal reward

suboptimal reward
Bidirectional Combination Lock

Initial state

\[ r = 5 \]  
optimal reward

\[ r = 2 \]  
suboptimal reward
Bidirectional Combination Lock

“survived” state (white): 9 out of 10 actions go to bad state (black)
**Bidirectional Combination Lock**

“survived” state (white): 9 out of 10 actions go to bad state (black)

Bad state (cannot recover) has Anti-shaped reward:

\[ r = \frac{1}{H} \]
Bidirectional Combination Lock

“survived” state (white): 9 out of 10 actions go to bad state (black)

Bad state (cannot recover) has Anti-shaped reward:

\[ r = \frac{1}{H} \]

Initial state

1. Local minima
2. Forgetting (policy becomes deterministic)
Bidirectional Combination Lock

“survived” state (white): 9 out of 10 actions go to bad state (black)

Bad state (cannot recover) has Anti-shaped reward:
\[ r = 1/H \]

1. Local minima
2. Forgetting (policy becomes deterministic)
Experiments on Bi-directional Comb Lock

Success Rate (visit the better chain):

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Horizon</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td>PPO</td>
<td>1.0</td>
</tr>
<tr>
<td>PPO+RND</td>
<td>0.75</td>
</tr>
<tr>
<td>PC-PG</td>
<td>?</td>
</tr>
</tbody>
</table>

PPO+RND: Random Network Distillation [Burda et.al, 19]
# Experiments on Bi-directional Comb Lock

Success Rate (visit the better chain):

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>2</th>
<th>5</th>
<th>10</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPO</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>PPO+RND</td>
<td>0.75</td>
<td>0.40</td>
<td>0.50</td>
<td>0.55</td>
</tr>
<tr>
<td>PC-PG</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>

PPO+RND: Random Network Distillation [Burda et.al, 19]
### Experiments on Bi-directional Comb Lock

Success Rate (visit the better chain):

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Horizon</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>PPO</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>PPO+RND</td>
<td>0.75</td>
<td>0.40</td>
<td>0.50</td>
<td>0.55</td>
</tr>
<tr>
<td>PC-PG</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>

Forgets to visit the other chain!

PPO+RND: Random Network Distillation [Burda et.al, 19]
Experiments on Bi-directional Comb Lock

RND traces during training

chain 1

chain 2

ep 500 ep 3000 ep 3500 ep 5000

optimal chain

Sub-optimal chain
Experiments on Bi-directional Comb Lock

RND traces during training

Chain 1
- ep 500
- ep 3000
- ep 3500
- ep 5000
- ep 5500
- ep 6000
- ep 6500
- ep 8000

Chain 2
- ep 500
- ep 3000
- ep 3500
- ep 5000
- ep 5500
- ep 6000
- ep 6500
- ep 8000

Optimal chain

Sub-optimal chain
Experiments on Bi-directional Comb Lock

RND traces during training

Policy quickly becomes too deterministic and forgets to explore the other (better!) chain
Summary of PG methods’ common issues

1. Lack ability to explore

1. Catastrophic forgetting (even w/ reward bonus)
Summary of PG methods’ common issues

1. Lack ability to explore

1. Catastrophic forgetting (even w/ reward bonus)

Next:

Our Solution: Policy Cover Policy Gradient (PC-PG)

Policy Ensemble + Reward Bonus
Notations

Policy: state to action

$$\pi(s) \rightarrow a$$

Reward & Next State

$$r(s, a), s' \sim P(\cdot | s, a)$$
**Notations**

**Policy: state to action**

\[
\pi(s) \rightarrow a
\]

**Value and Q function**

\[
V^\pi(s) = \mathbb{E} \left[ r(s_0, a_0) + \gamma r(s_1, a_1) + \ldots | s_0 = s, a_h \sim \pi(s_h) \right]
\]

\[
Q^\pi(s, a) = r(s, a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s, a)} V^\pi(s')
\]

**Reward & Next State**

\[
r(s, a), s' \sim P(\cdot | s, a)
\]
Notations

Policy: state to action
\[ \pi(s) \rightarrow a \]

Reward & Next State
\[ r(s, a), s' \sim P(\cdot|s, a) \]

Value and Q function
\[ V^\pi(s) = \mathbb{E} \left[ r(s_0, a_0) + \gamma r(s_1, a_1) + \ldots | s_0 = s, a_h \sim \pi(s_h) \right] \]
\[ Q^\pi(s, a) = r(s, a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s, a)} V^\pi(s') \]

Policy's state-action distribution
\[ d^\pi(s, a) = (1 - \gamma) \sum_{h=0}^{\infty} \gamma^h \mathbb{P}^\pi ((s_h, a_h) = (s, a)) \]
PC-PG: Policy Cover - Policy Gradient

1. Form Cover:

\[ \{ \pi_1, \pi_2, \ldots, \pi_n \} \]
1. Form Cover:

\[
\left\{ \pi_1, \pi_2, \ldots, \pi_n \right\}
\]

\[
\rho_n := \sum_{i=1}^{n} d^{\pi_i} / n
\]
PC-PG: Policy Cover - Policy Gradient

1. Form Cover:

\[ \{\pi_1, \pi_2, \ldots, \pi_n\} \]

\[ \rho_n := \sum_{i=1}^{n} d^{\pi_i} / n \]

2. Bonus

Intrinsic Reward \( b^n(s, a) \)
PC-PG: Policy Cover - Policy Gradient

1. Form Cover:

\[ \{\pi_1, \pi_2, \ldots, \pi_n\} \]

\[ \rho_n := \sum_{i=1}^{n} d^{\pi_i} / n \]

2. Bonus

Intrinsic Reward \( b^n(s, a) \)
PC-PG: Policy Cover - Policy Gradient

1. Form Cover:

\[ \{\pi_1, \pi_2, \ldots, \pi_n\} \]

\[ \rho_n := \sum_{i=1}^{n} d^{\pi_i} / n \]

2. Bonus

Intrinsic Reward \( b^n(s, a) \)
PC-PG: Policy Cover - Policy Gradient

1. Form Cover:

\[ \{ \pi_1, \pi_2, \ldots, \pi_n \} \]

\[ \rho_n := \sum_{i=1}^{n} d^{\pi_i} / n \]

2. Bonus

Intrinsic Reward \( b^n(s, a) \)
PC-PG: Policy Cover - Policy Gradient

1. Form Cover:
\[ \{ \pi_1, \pi_2, \ldots, \pi_n \} \]

\[ \rho_n := \sum_{i=1}^{n} d^{\pi_i} / n \]

2. Bonus

Large
\[ b^n(s, a) \]

Intrinsic Reward
\[ Q^\pi_{r+b^n}(s, a) \]

\[ \{(s_i, a_i), Q^\pi_{r+b^n}(s_i, a_i)\} \]
PC-PG: Policy Cover - Policy Gradient

1. Form Cover:

\[ \{ \pi_1, \pi_2, \ldots, \pi_n \} \]

\[ \rho_n := \sum_{i=1}^{n} d^{\pi_i} / n \]

2. Bonus

\[ b^n(s, a) \]

3. On-Policy Critic Fit (least square)

\[ \hat{f} = \max_{f \in F} \sum_i \left( f(s_i, a_i) - Q^\pi_{r+b^n}(s_i, a_i) \right)^2 \]

\[ \{ (s_i, a_i), Q^\pi_{r+b^n}(s_i, a_i) \} \]
PC-PG: Policy Cover - Policy Gradient

1. Form Cover:
\[ \{ \pi_1, \pi_2, \ldots, \pi_n \} \]
\[ \rho_n := \sum_{i=1}^{n} d^{\pi_i} / n \]
\[ \pi(s, a) \leftarrow \pi(s, a) \exp(\eta \hat{f}(s, a)) \]

2. Bonus
\[ b^n(s, a) \]

3. On-Policy Critic Fit (least square)
\[ \hat{f} = \max_{f \in \mathcal{F}} \sum_i \left( f(s_i, a_i) - Q^\pi_{r+b^n}(s_i, a_i) \right)^2 \]

4. Actor NPG update (mirror descent)
\[ \left\{ (s_i, a_i), Q^\pi_{r+b^n}(s_i, a_i) \right\} \]
PC-PG: Policy Cover - Policy Gradient

1. Form Cover:
\[ \{ \pi_1, \pi_2, \ldots, \pi_n \} \]
\[ \rho_n := \sum_{i=1}^{n} d^{\pi_i} / n \]

2. Bonus
\[ b^n(s, a) \]

3. On-Policy Critic Fit (least square)
\[ \hat{f} = \max_{f \in \mathcal{F}} \sum_{i} (f(s_i, a_i) - Q^\pi_{r+b^n}(s_i, a_i))^2 \]

4. Actor NPG update (mirror descent)
\[ \pi(s, a) \leftarrow \pi(s, a) \exp(\eta \hat{f}(s, a)) \]
PC-PG: Policy Cover - Policy Gradient

1. Form Cover:
\[ \{ \pi_1, \pi_2, \ldots, \pi_n \} \]
\[ \rho_n := \sum_{i=1}^{n} d^{\pi_i} / n \]

2. Bonus
\[ b^n(s, a) \]

3. On-Policy Critic Fit (least square)
\[ \hat{f} = \max_{f \in \mathcal{F}} \sum_i (f(s_i, a_i) - Q^\pi_{r+b^n}(s_i, a_i))^2 \]

4. Actor NPG update (mirror descent)
\[ \pi(s, a) \leftarrow \pi(s, a) \exp(\eta \hat{f}(s, a)) \]

5. Append
\[ \{ (s_i, a_i), Q^\pi_{r+b^n}(s_i, a_i) \} \]
What objective function PC-PG is trying to optimize?

At episode $n$: Natural PG is optimizing

$$
\mathbb{E}_{s_0, a_0 \sim \rho_n} \left[ \sum_{t=0}^{\infty} \gamma^t (r(s_t, a_t) + b^n(s_t, a_t)) \right]
$$
What objective function PC-PG is trying to optimize?

At episode \( n \): Natural PG is optimizing

\[
\mathbb{E}_{s_0, a_0 \sim \rho_n} \left[ \sum_{t=0}^{\infty} \gamma^t (r(s_t, a_t) + b^n(s_t, a_t)) \right]
\]

Use the cover and roll-in via policies in the cover

(note we do not start at \( \mu_0 \))
What objective function PC-PG is trying to optimize?

At episode $n$: Natural PG is optimizing

$$\mathbb{E}_{s_0, a_0 \sim \rho_n} \left[ \sum_{t=0}^{\infty} \gamma^t (r(s_t, a_t) + b^n(s_t, a_t)) \right]$$

Use the cover and roll-in via policies in the cover

(Bonus based on the cover)

(Note we do not start at $\mu_0$)
What objective function PC-PG is trying to optimize?

At episode $n$: Natural PG is optimizing

$$
\mathbb{E}_{s_0, a_0 \sim \rho_n} \left[ \sum_{t=0}^{\infty} \gamma^t (r(s_t, a_t) + b^n(s_t, a_t)) \right]
$$

Use the cover and roll-in via policies in the cover

(Bonus based on the cover)

(Note we do not start at $\mu_0$)

No more forgetting!
What objective function PC-PG is trying to optimize?

At episode \( n \): Natural PG is optimizing

\[
\mathbb{E}_{s_0, a_0 \sim \rho_n} \left[ \sum_{t=0}^{\infty} \gamma^t (r(s_t, a_t) + b^n(s_t, a_t)) \right]
\]

Use the cover and roll-in via policies in the cover

(\textbf{note we do not start at } \mu_0 \text{ )}

Bonus based on the cover

No more sparse reward!

No more forgetting!
PC-PG Specialized to Linear Function Approximation

Use linear function $\theta \cdot \phi(s, a)$ to approximate $Q^\pi$

1. Form Cover:

$$\{\pi_1, \pi_2, \ldots, \pi_n\}$$

$$\sum_{\pi_i} = \mathbb{E}_{s, a \sim d_{\pi_i}} \phi(s, a)\phi(s, a)^T$$

$$\Sigma_n = \sum_{i=1}^{n} \sum_{\pi_i} + \lambda I$$
PC-PG Specialized to Linear Function Approximation

Use linear function $\theta \cdot \phi(s, a)$ to approximate $Q^\pi$

1. Form Cover:

\[
\{ \pi_1, \pi_2, \ldots, \pi_n \}
\]

\[
\Sigma_{\pi_i} = \mathbb{E}_{s,a \sim d_{\pi_i}} \phi(s, a)\phi(s, a)^\top
\]

\[
\Sigma_n = \sum_{i=1}^n \Sigma_{\pi_i} + \lambda I
\]

2. Bonus

\[
b^n(s, a) = 1 \left\{ \phi(s, a)^\top \Sigma_n^{-1} \phi(s, a) \geq \beta \right\} / (1 - \gamma)
\]
Use linear function $\theta \cdot \phi(s, a)$ to approximate $Q^\pi$

1. Form Cover:
   \[
   \{ \pi_1, \pi_2, \ldots, \pi_n \}
   \]
   \[
   \Sigma_{\pi_i} = \mathbb{E}_{s,a \sim d_{\pi_i}} \phi(s, a)\phi(s, a)^\top
   \]
   \[
   \Sigma_n = \sum_{i=1}^n \Sigma_{\pi_i} + \lambda I
   \]

2. Bonus
   \[
   b^n(s, a) = 1 \left\{ \phi(s, a)^\top \Sigma_n^{-1} \phi(s, a) \geq \beta \right\} / (1 - \gamma)
   \]
   Rewarding $(s, a)$ whose feature $\phi(s, a)$ aligns with small eigenvectors
PC-PG Specialized to Linear Function Approximation

Use linear function $\theta \cdot \phi(s, a)$ to approximate $Q^\pi$

1. Form Cover:

$$\left\{ \pi_1, \pi_2, \ldots, \pi_n \right\}$$

$$\Sigma_{\pi_i} = \mathbb{E}_{s,a \sim d_{\pi_i}} \phi(s, a)\phi(s, a)^\top$$

$$\Sigma_n = \sum_{i=1}^{n} \Sigma_{\pi_i} + \lambda I$$

2. Bonus

$$b^n(s, a) = 1 \left\{ \phi(s, a)^\top \Sigma_n^{-1} \phi(s, a) \geq \beta \right\} / (1 - \gamma)$$

Rewarding $(s, a)$ whose feature $\phi(s, a)$ aligns with small eigenvectors

3. Natural PG:

$$Q^\pi_{r+b^n}(s, a) \approx \theta \cdot \phi(s, a) + b^n(s, a)$$

$$\pi \leftarrow \pi \exp \left( \eta \left( b^n + \theta \cdot \phi \right) \right)$$
PC-PG Specialized to Linear Function Approximation

Use linear function $\theta \cdot \phi(s, a)$ to approximate $Q^\pi$

1. Form Cover:

$$\{ \pi_1, \pi_2, \ldots, \pi_n \}$$

$$\Sigma_{\pi_i} = \mathbb{E}_{s, a \sim d_{\pi_i}} \phi(s, a)\phi(s, a)^\top$$

$$\Sigma_n = \sum_{i=1}^{n} \Sigma_{\pi_i} + \lambda I$$

2. Bonus

$$b^n(s, a) = 1 \left\{ \phi(s, a)^\top \Sigma_n^{-1} \phi(s, a) \geq \beta \right\} / (1 - \gamma)$$

Rewarding $(s, a)$ whose feature $\phi(s, a)$ aligns with small eigenvectors

3. Natural PG:

$$Q^\pi_{r + b^n}(s, a) \approx \theta \cdot \phi(s, a) + b^n(s, a)$$

$$\pi \leftarrow \pi \exp \left( \eta \left( b^n + \theta \cdot \phi \right) \right)$$
PC-PG Specialized to Tabular MDPs

One-hot vector: \( \phi(s, a) \in \mathbb{R}^{SA} \)

\[
\rho_n = \sum_{i=1}^{n} \frac{d^{\pi_i}}{n}
\]

\[
\Sigma_n = \text{diag} \left( \ldots, \rho_n(s, a), \ldots, \right)
\]
PC-PG Specialized to Tabular MDPs

One-hot vector: \( \phi(s, a) \in \mathbb{R}^{SA} \)

\[
\rho_n = \sum_{i=1}^{n} d^{\pi_i} / n
\]

\[
\Sigma_n = \text{diag} \left( \ldots, \rho_n(s, a), \ldots, \right)
\]

Rewarding state that has low probability of being covered

\[
\phi(s, a)^\top \Sigma_n^{-1} \phi(s, a) \leq \frac{1}{\rho_n(s, a)}
\]
Well-Specified Setting: Linear MDPs (and Tabular MDPs)

Reward and transition in RKHS:  
\[ r(s, a) = \theta \cdot \phi(s, a), \quad P(\cdot | s, a) = \mu \phi(s, a) \]

[RKHS version of the Linear mdp model from Jin et al, 19]
Well-Specified Setting: Linear MDPs (and Tabular MDPs)

Reward and transition in RKHS:  
\[
r(s, a) = \theta \cdot \phi(s, a), \quad P(\cdot | s, a) = \mu \phi(s, a)
\]

A bellman backup of any \(f(s)\) will be linear in \(\phi(s, a)\):  
\[
r(s, a) + \mathbb{E}_{s' \sim P(\cdot | s,a)} f(s') = w^\top \phi(s, a)
\]
Well-Specified Setting: Linear MDPs (and Tabular MDPs)

Reward and transition in RKHS:

\[ r(s, a) = \theta \cdot \phi(s, a), \quad P(\cdot | s, a) = \mu \phi(s, a) \]

A bellman backup of any \( f(s) \) will be linear in \( \phi(s, a) \):

\[ r(s, a) + \mathbb{E}_{s' \sim P(\cdot | s, a)} f(s') = w^\top \phi(s, a) \]

\[ V^{\hat{\pi}} \geq V^* - \epsilon \]

with # of samples

\[ \text{poly}(d, \log(A), 1/(1 - \gamma), 1/\epsilon) \]
Well-Specified Setting: Linear MDPs (and Tabular MDPs)

Reward and transition in RKHS:

\[ r(s, a) = \theta \cdot \phi(s, a), \quad P(\cdot | s, a) = \mu \phi(s, a) \]

A bellman backup of any \( f(s) \) will be linear in \( \phi(s, a) \):

\[ r(s, a) + \mathbb{E}_{s' \sim P(\cdot | s, a)} f(s') = w^T \phi(s, a) \]

\[ \hat{V}^\pi \geq V^* - \epsilon \]

with \# of samples

\[ \text{poly}(d, \log(A), 1/(1 - \gamma), 1/\epsilon) \]

Dim of feature

(extendable to RKHS w/ Information Gain)
Comparison to vanilla Natural PG in Linear MDPs

A wide initial distribution:

\[ \kappa = 1/\sigma_{\min} \left( \mathbb{E}_{s,a \sim \mu_0} \phi(s, a)\phi(s, a)^T \right) < \infty \]
Comparison to vanilla Natural PG in Linear MDPs

[Agarwal et al 19]

A wide initial distribution:

$$\kappa = \frac{1}{\sigma_{\min}} \left( \mathbb{E}_{s,a \sim \mu_0} \phi(s, a) \phi(s, a)^\top \right) < \infty$$

$$V^{\hat{\pi}} \geq V^* - \epsilon$$

w/ # of samples $\text{poly}(1/\epsilon, 1/(1 - \gamma), \ln(A), \kappa)$
Comparison to vanilla Natural PG in Linear MDPs

[Agarwal et al 19]

A wide initial distribution:

$$\kappa = 1/\sigma_{\text{min}} \left( \mathbb{E}_{s,a \sim \mu_0} \phi(s,a)\phi(s,a)^T \right) < \infty$$

$$V^\hat{\pi} \geq V^* - \epsilon$$

w/ # of samples $\text{poly}(1/\epsilon, 1/(1 - \gamma), \ln(A), \kappa)$

Condition number could be exponential !!
Comparison to vanilla Natural PG in Linear MDPs

[Agarwal et al 19]

A wide initial distribution:

$$\kappa = 1/\sigma_{\text{min}} \left( \mathbb{E}_{s,a \sim \mu_0} \phi(s,a)\phi(s,a)^\top \right) < \infty$$

$$V_{\hat{\pi}} \geq V^* - \epsilon$$

w/ # of samples $\text{poly}(1/\epsilon, 1/(1 - \gamma), \ln(A), \kappa)$

Condition number could be exponential !!

PC-PG eliminates the condition number by actively 
exploring and building policy cover
Robustness to Model-Misspecification

Average VS $\ell_\infty$
Model-Misspecification from State-Abstraction

$\phi : S \rightarrow Z$

Group “similar” states (s)
into an abstracted state (z)
Model-Misspecification from State-Abstraction

\[ \phi : S \rightarrow Z \]

Group "similar" states (s) into an abstracted state (z)

\[ |Z| \ll |S| \]

\[ \text{poly}(|Z|) \]
Model-Misspecification from State-Abstraction

Group “similar” states \( s \) into an abstracted state \( z \)

\[ \phi : S \rightarrow Z \]

\(|Z| \ll |S|\)

\[ \text{poly}(|Z|) \]

Model-misspecification:

\[ s, s', \text{ s.t. } \phi(s) = \phi(s') \]

\[ \| P(\cdot | s, a) - P(\cdot | s', a) \|_1 \leq \epsilon_{z,a} \]
Model-Misspecification from State-Abstraction

Model-misspecification:

\[ s, s', \text{ s.t. } \phi(s) = \phi(s') \]

\[ \|P(\cdot | s, a) - P(\cdot | s', a)\|_1 \leq \epsilon_{z,a} \]

\[ \phi : S \rightarrow Z \]

Group “similar” states \( (s) \)
into an abstracted state \( (z) \)

\[ |Z| \ll |S| \]

\[ \text{poly}(|Z|) \]
Model-Misspecification from State-Abstraction

Model-misspecification:
\[ s, s', \text{ s.t. } \phi(s) = \phi(s') \]
\[ \|P(\cdot | s, a) - P(\cdot | s', a)\|_1 \leq \epsilon_{z,a} \]

Bellman-backup based (e.g., Q-learning [Dong et al, 19])
\[ \text{poly}\left(\frac{1}{(1 - \gamma)}\right) \left(\max_{z,a} \epsilon_{z,a}\right) \]

\( \phi : S \rightarrow Z \)
Group “similar” states (s) into an abstracted state (z)

|Z| \ll |S|

poly(|Z|)
Model-Misspecification from State-Abstraction

Model-misspecification:
\[ s, s', \text{ s.t. } \phi(s) = \phi(s') \]
\[ \|P(\cdot | s, a) - P(\cdot | s', a)\|_1 \leq \epsilon_{z,a} \]

Bellman-backup based (e.g., Q-learning [Dong et al, 19])

\[ \text{poly} \left( \frac{1}{1 - \gamma} \right) \left( \max_{z,a} \epsilon_{z,a} \right) \]
Model-Misspecification from State-Abstraction

\[ \phi : S \rightarrow Z \]
Group “similar” states \((s)\) into an abstracted state \((z)\)

\[ |Z| \ll |S| \]
\[ \text{poly}(|Z|) \]

Model-misspecification:
\[ s, s', \text{ s.t. } \phi(s) = \phi(s') \]
\[ \|P( \cdot | s, a) - P( \cdot | s', a)\|_1 \leq \epsilon_{z,a} \]

Bellman-backup based (e.g., Q-learning [Dong et al, 19])

\[ \text{poly} \left( \frac{1}{1-\gamma} \right) \left( \max_{z,a} \epsilon_{z,a} \right) \]

\[ \ell_\infty \]

PC-PG

\[ \text{poly}(1/(1 - \gamma)) \left( \mathbb{E}_{z,a \sim \hat{d} \pi} \left[ \epsilon_{z,a} \right] \right) \]
Model-Misspecification from State-Abstraction

Model-misspecification:

\[ s, s', \text{ s.t. } \phi(s) = \phi(s') \]

\[ \|P(\cdot | s, a) - P(\cdot | s', a)\|_1 \leq \epsilon_{z,a} \]

Bellman-backup based (e.g., Q-learning [Dong et al, 19])

\[ \text{poly} \left( \frac{1}{(1 - \gamma)} \right) \left( \max_{z,a} \epsilon_{z,a} \right) \]

\[ \ell_\infty \]

PC-PG

\[ \text{poly}(1/(1 - \gamma)) \left( \mathbb{E}_{z,a \sim \hat{d}\pi} [\epsilon_{z,a}] \right) \]
Model-Misspecification from State-Abstraction

\[ \phi : S \rightarrow Z \]
Group “similar” states (s) into an abstracted state (z)

\[ |Z| \ll |S| \]
\[ \text{poly}(|Z|) \]

Model-misspecification:
\[ s, s', \text{ s.t. } \phi(s) = \phi(s') \]
\[ \| P(\cdot | s, a) - P(\cdot | s', a) \|_1 \leq \epsilon_{z,a} \]

Bellman-backup based (e.g., Q-learning [Dong et al, 19])

\[ \text{poly} \left( \frac{1}{1 - \gamma} \right) \left( \max_{z,a} \epsilon_{z,a} \right) \]

PC-PG

\[ \text{Average over comparator’s distribution} \]
Additional Example w/ linear approximation

\[ r(s_0, L) = 1/2 \]
Additional Example w/ linear approximation

\[ r(s_0, L) = 1/2 \]

Path with reward \( 1/2 \)
Additional Example w/ linear approximation

\[ r(s_0, L) = \frac{1}{2} \]

\[ \phi(s_0, L) = e_1, \]

Path with reward \(1/2\)
$r(s_0, L) = 1/2$

$\phi(s_0, L) = e_1,$

$\phi(s_0, R) = e_2,$

Path with reward $1/2$
Additional Example w/ linear approximation

\[ r(s_0, L) = \frac{1}{2} \]
\[ \phi(s_0, L) = e_1, \]
\[ \phi(s_0, R) = e_2, \]
\[ \phi(s_1, a) = e_3. \]
Additional Example w/ linear approximation

\[
\begin{align*}
\rho(s_0, L) &= 1/2 \\
\phi(s_0, L) &= e_1, \\
\phi(s_0, R) &= e_2, \\
\phi(s_1, a) &= e_3.
\end{align*}
\]
Additional Example w/ linear approximation

Path with reward $1/2$

$$r(s_0, L) = 1/2$$
$$\phi(s_0, L) = e_1,$$
$$\phi(s_0, R) = e_2,$$
$$\phi(s_1, a) = e_3.$$

Features is arbitrary in the tree, i.e., model-misspecification can be very serious.
Features is arbitrary in the tree, i.e., model-misspecification can be very serious.

Additional Example w/ linear approximation

\[ r(s_0, L) = 1/2 \]
\[ \phi(s_0, L) = e_1, \]
\[ \phi(s_0, R) = e_2, \]
\[ \phi(s_1, a) = e_3. \]

Path with reward 1/2

PC-PG is at least as good as the green trajectory

\[ \forall s' \in \text{sub-tree}, \phi(s', a) \perp \text{span}(e_1, e_2, e_3). \]
Experiments

Bidirectional Combination Lock

Good state (white): 9 out of 10 actions go to bad state (black)

Bad state has Anti-shaped reward: $r = 1/H$

1. Local minima
2. Forgetting
Experiments on Bi-directional Comb Lock

Feature vector: binary vector indicating state-action and time step

Policy Opt procedure: PPO w/ NN policy

Bonus: $\phi^T \sum_n^{-1} \phi$
Experiments on Bi-directional Comb Lock

Feature vector: binary vector indicating state-action and time step

Policy Opt procedure: PPO w/ NN policy

Bonus: $\phi^\top \sum_n -1 \phi$

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Horizon 2</th>
<th>Horizon 5</th>
<th>Horizon 10</th>
<th>Horizon 15</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPO</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>PPO+RND</td>
<td>0.75</td>
<td>0.40</td>
<td>0.50</td>
<td>0.55</td>
</tr>
<tr>
<td>PC-PG</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
</tbody>
</table>
Experiments on Bi-directional Comb Lock

Due to the policy cover PC-PG maintains..

State visitations for top weighted policies in mixture
Experiments on Bi-directional Comb Lock

Due to the policy cover PC-PG maintains..

State visitations for top weighted policies in mixture

visitations for policy mixture
Experiments on Bi-directional Comb Lock

Due to the policy cover PC-PG maintains..

State visitations for top weighted policies in mixture

visitations for policy mixture

Cover $\rho_n$ near uniformly cover both chains
Experiments on Bi-directional Comb Lock

Due to the policy cover PC-PG maintains..

Cover $\rho_n$ near uniformly cover both chains
PG with $\rho_n$ will succeed!
Reward-Free Explore in Maze

\[ r(s, a) = 0 \]

\[ \phi(s, a) : \text{Random initialized CovNet} \]

Policy Opt procedure: PPO w/ CovNet-based policy
Reward-Free Explore in Maze

Traces of policies in the policy cover:

Policy 1  Policy 4  Policy 5  Policy 11  Policy 12

Agent start location

Higher Better

% states explored

steps

- PPO
- PPO+RND
- PC-PG
Continuous Control w/ sparse reward

Sparse reward: large reward at the goal;

Anti-shaped reward: penalize control inputs
Continuous Control w/ sparse reward

Sparse reward: large reward at the goal;
Anti-shaped reward: penalize control inputs
Continuous Control w/ sparse reward

Sparse reward: large reward at the goal;
Anti-shaped reward: penalize control inputs
Conclusion
Conclusion

Strong agnostic results

Average model-misspecification VS $\ell_\infty$
Conclusion

Strong agnostic results

Average model-misspecification VS $\ell_\infty$

Polynomial Sample Complexity in well-specified case:

Linear MDPs (RKHS) & Tabular MDPs
Conclusion

Strong agnostic results

Average model-misspecification VS $\ell_\infty$

Polynomial Sample Complexity in well-specified case:

- Linear MDPs (RKHS)
- Tabular MDPs

Policy Cover/bonus solve the issue of flatten gradient & Forgetting

- Treat policy cover’s distribution as the reset distribution for PG
Conclusion

Strong agnostic results
Average model-misspecification VS $\ell_\infty$

Polynomial Sample Complexity in well-specified case:
Linear MDPs (RKHS) & Tabular MDPs

Policy Cover/bonus solve the issue of flatten gradient & Forgetting
Treat policy cover’s distribution as the reset distribution for PG

Flexibility to leverage existing deep learning/RL tools
Vanilla implementation explores 4 to 5 rooms in M-Revenge