Improving policy optimization: Algorithms and Foundations

Baoxiang Wang

Jun 18, 2020, CUHK

Thesis oral defense in partial fulfilment of the Ph.D. degree

Supervised by Siu On Chan
Ph.D. Candidate, Department of Computer Science and Engineering
The Chinese University of Hong Kong

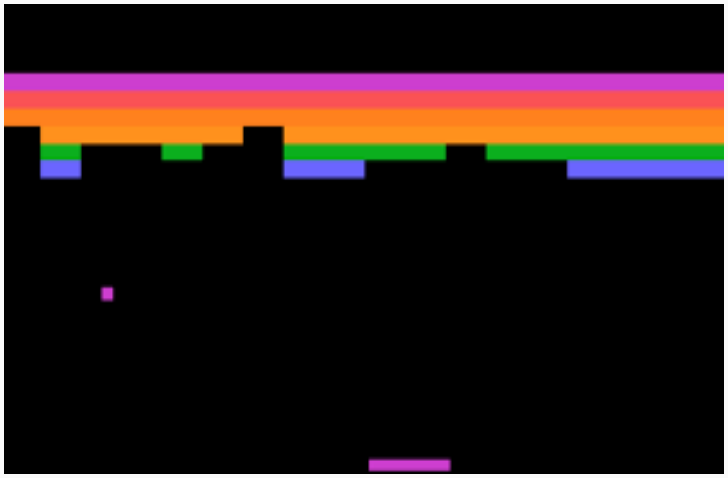
Outline

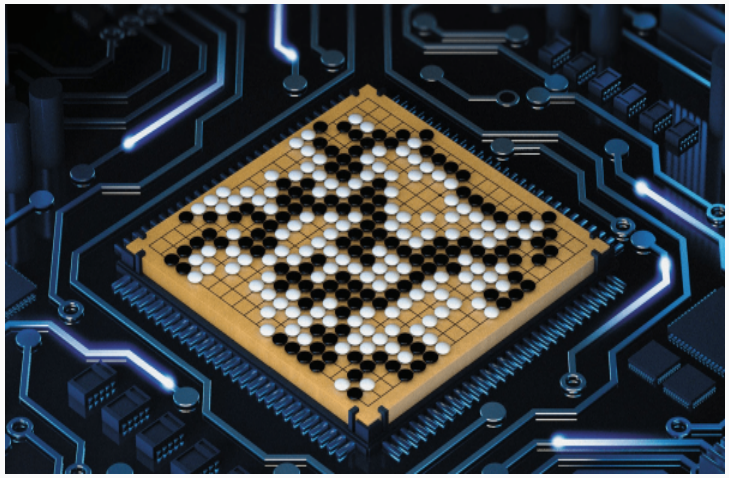
Outline of this thesis defense talk

- Background on sequential decision processes and reinforcement learning, and the position of this thesis
- Improving policy optimization via variance reduction
- In-depth discussions of two underlying fundamental studies on variable partitioning and value function approximation.
- A brief mention of other thesis works

Outline

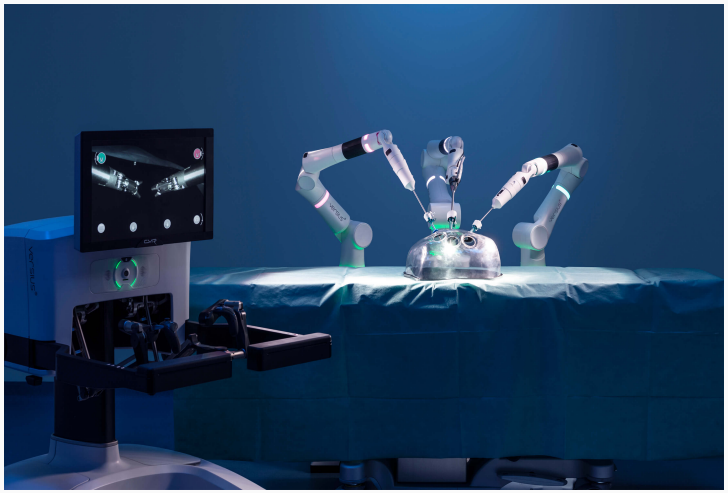
Reinforcement learning and sequential decisions





Milestones, in chronological order: Breakout in Atari 2600, AlphaGo and AlphaZero, Libratus and DeepStack, and AlphaStar

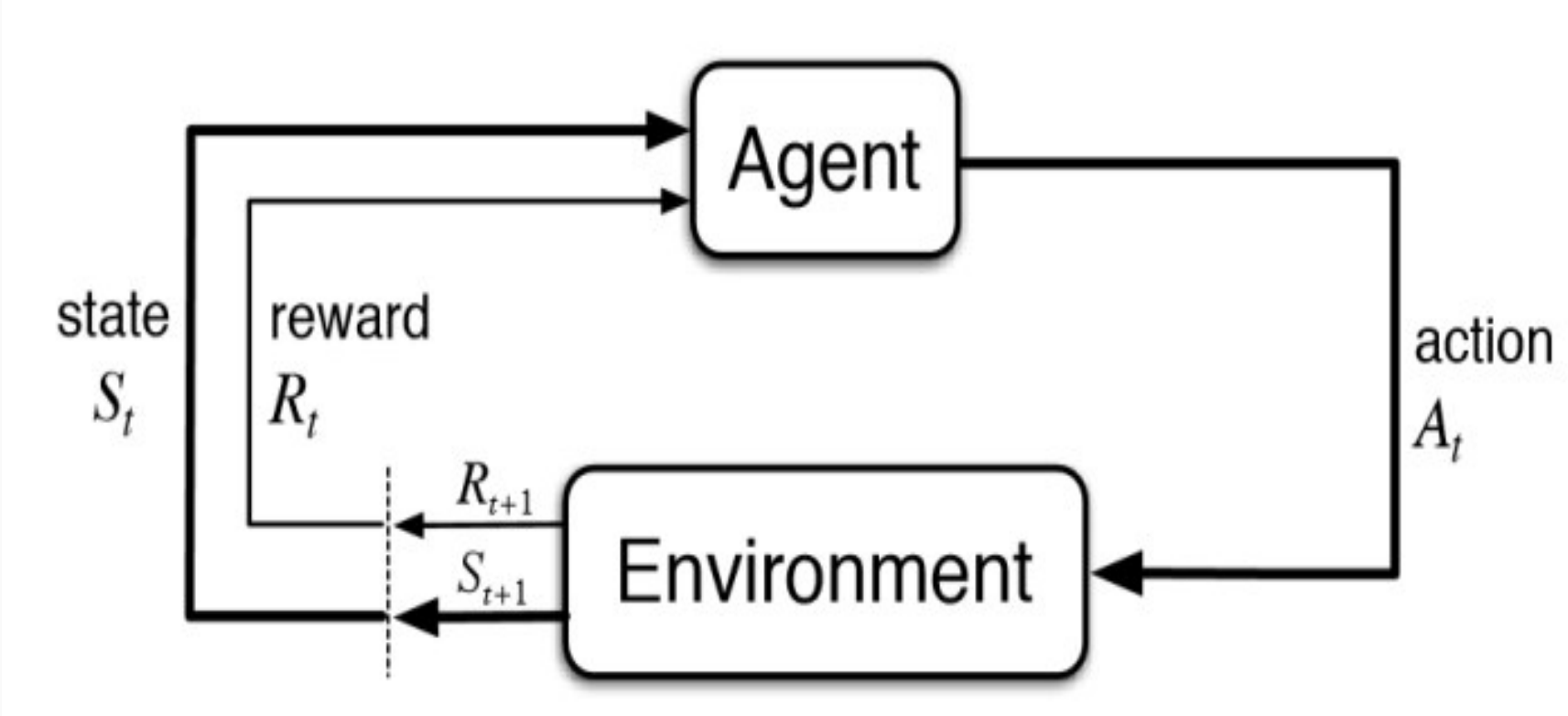
Reinforcement learning and sequential decisions



Applications: humanoid simulation, robot surgeon, robotics, and autonomous driving

Reinforcement learning and sequential decisions

Reinforcement learning: To model and learn sequential agent-environment interaction from reinforces



Sutton and Barto. Reinforcement learning: An introduction. 2018.

Connections to other areas

Cognitive science

- RL discusses the interaction between action and perception of an agent, while cognitive science studies that of humans.
- Cognitive science concepts are heavily adopted

Optimal control

- RL targets mostly model-free learning. To learn only from the reward signals *tabula rasa* without knowing the environment
- Optimal control is based on the model instead

Online learning

- RL is contextual multi-arm bandit with an additional feedback: The action will in turn impact the environment.

Outline

Position of this thesis

- Reinforcement learning
  - Cognitive science considerations (perspective, theory of mind etc.)
  - Computing considerations
    - Model-based (dynamic programming, optimal control)
    - Model-free trial-and-error learning

      $\Downarrow$
    - Bandit algorithms
    - Learning via value iteration (Q-learning)
    - Learning via policy iteration, known as policy optimization.

      This thesis aim to improve such methods

Outline

Markov decision processes (MDP) - Formulation

- RL is formulated as MDP - tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \rho_0, \gamma)$
  - $\mathcal{S} \in \mathbb{R}^m$ state space, $\mathcal{A} \in \mathbb{R}^n$ action space, $\rho_0 \in \Delta(\mathcal{S})$ the initial state distribution[1]
  - $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ environment transition probability function
  - $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathbb{R})$ reward function
  - $\gamma \in [0, 1]$ unnormalized discount factor
- The MDP follows

  $a_t \sim \pi(a|s_t), r_t \sim \mathcal{R}(s_t, a_t), s_{t+1} \sim \mathcal{T}(s_t, a_t), t = 0, 1, 2, \ldots$
- The objective is to learn the policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$
- To maximize the expected return:

$$R_T = \sum_{t \leq T} \gamma^t r_t, \quad J = \mathbb{E}[R_\infty] = \mathbb{E}\Big[\sum_{t \geq 0} \gamma^t r_t \Big| s_0 \sim \rho_0, \pi\Big]$$

[1] $\Delta(\cdot)$ denotes the set of all random variables over the input space

Markov decision processes - temporal-difference methods

- Action-value function $Q(s, a)$: $\mathbb{E}[R_\infty]$ condition on initial s, a

$$Q(s,a) = \mathbb{E}[R_\infty] = \mathbb{E}\Big[\sum_{t\geq 0} \gamma^t r_t \Big| s_0 = s, a_0 = a, \pi\Big]$$

- Many temporal-difference (TD) methods can learn $Q(s, a)$:
  - TD(1): Use Monte-Carlo samples of $R_T|_{s_0=s,a_0=a}$ to estimate $Q(s, a)$; e.g. REINFORCE, A3C [Wil92, SMSM00, MBM+16]
  - TD(0): Use $\mathbb{E}[r_t + \gamma Q(s_{t+1}, \pi(s_{t+1}))]$ to estimate $Q(s_t, a_t)$; e.g. Q-learning [WD92, MKS+15]
  - TD(λ): in between; e.g. Eligibility Traces [SS96], TRPO and PPO [SLA+15, SWD+17]
  - Assume $Q(s, a)$ is learned by TD(λ) and neural network approximation for the rest of the discussion

Markov decision processes - Policy gradient methods

- Let π be parameterized by θ (e.g. neural network, θ is the set of network parameters), then by [Wil92]

$$\nabla_\theta \mathbb{E}[R_T|\theta] = \mathbb{E}_{\pi(a|s)}[\nabla_\theta \log \pi(a|s) R_T] \tag{1}$$

- Several landmark works
  [Wil92, SMSM00, MBM+16, DWS12, SLA+15]

$$R_T \Longrightarrow R_T - \hat{R}_T \Longrightarrow Q(s,a) \Longrightarrow Q(s,a) - \mathbb{E}_a[Q(s,a)]$$

- Let the advantage function $A = Q - \mathbb{E}_a[Q]$, the policy gradient is

$$\nabla_\theta \mathbb{E}[R_T|\theta] = \mathbb{E}_{\pi(a|s)}[\nabla_\theta \log \pi(a|s) A(s,a)] \tag{2}$$

- Despite unbiased, both (1) and (2) are notoriously high in variance and unstable during optimization

Outline

Connections to variance reduction

- Revisit the landmarks
  $R_T \Longrightarrow R_T - \hat{R}_T \Longrightarrow Q(s,a) \Longrightarrow A(s,a)$ in

  $$\nabla_\theta \mathbb{E}[R_T|\theta] = \mathbb{E}_{\pi(a|s)}[\nabla_\theta \log \pi(a|s) A(s,a)].$$

  From a statistical variance reduction perspective, these improvements are all flavors of Control Variates.
- More works on Control Variates in recent years [GLG+16, LFM+18, WRD+18, TBG+18]
- Is this method good enough for reinforcement learning?

High-dimensional reinforcement learning control

Is this method good enough for RL? No.

- From statistics: Control Variates (CV) is known to be inefficient in high-dimension Monte-Carlo sampling
  - Almost all practical problems are high-dimensional (e.g. robotics, games, multi-agent control)
- From RL: Policy gradient is notoriously unstable due to high variance in high action-dimensional tasks $\mathcal{A} \subseteq \mathbb{R}^m$
  - Sample complexity scales exponentially with respect to number m of dimensions. CV's reduction scales poly(m)
- The problem arises: How can policy gradient methods be efficient also on high-dimensional tasks?

Idea: Divide and conquer

- A motivating example
  - We control k robots. No interaction between them.
  - Action space is $k \cdot m$-dimensional
  - Reward is the sum of the individual rewards (say, speed minus energy cost)
  - We do not know the above a priori
- Divide and conquer: If action space $\mathcal{A}$ is $\mathcal{A}_1 \times \mathcal{A}_2$ for some $\mathcal{A}_1$ and $\mathcal{A}_2$, can we solve each of $\mathcal{A}_i$ "separately"?
  - Then each $\mathcal{A}_i$ becomes lower-dimensional
- Can we keep it unbiased? Can we guarantee the variance reduction? Can we extend to k-partitions? (Yes for all!)

Outline

Divide-and-conquer algorithm [LW18, BW20]

- Action subspace dependent ($a_{(j)} \in \mathcal{A}_j$, assume we have $\mathcal{A}_j$) gradient estimator

$$\nabla_\theta \mathbb{E}[R_T|\theta] = \sum_{j=1}^{k} \mathbb{E}_{\pi(a_{(j)}|s)}[\nabla_\theta \log \pi(a_{(j)}|s)(A(s, a_{(j)})]. \quad (3)$$

- Alternatively, combining other techniques [GLG+16, LFM+18, GCW+18], a more subtle estimator

$$\begin{aligned}\nabla_\theta \mathbb{E}[R_T|\theta] = \sum_{j=1}^{k} \mathbb{E}_{\pi(a_{(j)}|s)}[\nabla_\theta \log \pi(a_{(j)}|s)(A(s, a_{(j)}) \\ - c(s, (a_{(j)}, \tilde{a}_{(-j)}))) - \nabla_\theta f_j(\theta, s, \xi) \nabla_{a_{(j)}} c_k(s, a_{(j)})].\end{aligned}$$

- We present Equation (3) for simplicity

Unbiased estimator

- The unbiasedness holds when at least one variable partitioning can be found on the advantage function:

Theorem 1

When

$$A(s,a) = A_1(s, a_{(1)}) + \cdots + A_k(s, a_{(k)})$$

for some functions $A_1, \ldots, A_k$, *estimator* (3) *is unbiased.*

Combining with the statement of variance reduction (the next), it suffices to find a partition with this additive equality.

Variance reduction

- By conditional probabilities (known as the Rao-Blackwell theorem [CR96, RGB14, TBG$^+$18]), any partitioning $\{\mathcal{A}_i\}_{1\leq i\leq k}$ guarantees strict variance reduction for (3).

Proposition 2

For the two estimators $g(X,Y) = \nabla \log \mathbb{P}(X,Y) f(X,Y)$ *and* $h(X,Y) = \nabla_\theta \log \mathbb{P}(X) f_X + \nabla \log \mathbb{P}(Y) f_Y$,

$$\mathrm{Var}[\frac{1}{n_s}\sum_{i=1}^{n_s} h(X_i,Y_i)] < \mathrm{Var}[\frac{1}{n_s}\sum_{i=1}^{n_s} g(X_i,Y_i)],$$

where n_s *is the number of Monte-Carlo samples and* X_i, Y_i *are the samples.*

We defer algorithms and experiments to the next section.

Outline

Variable partitioning problem

- It is a more general problem to find $a_{(1)}, \ldots, a_{(k)}$ such that $A(s, a) = A_1(s, a_{(1)}) + \cdots + A_k(s, a_{(k)})$.
- In general, let $\boldsymbol{V} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ and F be a function. We desire a partition $\boldsymbol{X}_1, \ldots \boldsymbol{X}_k \subset \boldsymbol{V}$, $\boldsymbol{X}_i \cap \boldsymbol{X}_j = \emptyset$, such that there exists $F_1, \ldots, F_k$,

$$F(\boldsymbol{V}) = F_1(\boldsymbol{X}_1) + \cdots + F_k(\boldsymbol{X}_k).$$

- The problem is useful for a wider audience:
  - When F is log-probability density function (log-likelihood) this finds independent subsets of variables
  - When F is quadratic (polynomial) this finds (hyper)graph minimum cut
  - When $k = 2$ and $F_2 = 0$ this is feature selection (juntas)

Variable partitioning problem

- Let $F(\boldsymbol{V}) : \Sigma^n \to G$ be a function and given oracle access to F. When the decomposition is imperfect, the decomposition error is measured by

$$\delta(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k) = \min_{F_1,\ldots,F_k} \|F(X_1, \ldots, X_k) - F_1(X_1) - \cdots - F_k(X_k)\|.$$

- Our objective is to minimize over $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k$

$$\delta_k(F) = \min_{\boldsymbol{X}_1,\ldots,\boldsymbol{X}_k} \delta(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k).$$

- Σ is any set. G is any Abelian group (equivalently, as long as "+" is defined). $\|\cdot\|$ is any norm.

Outline

Ideas and techniques

- We propose the dependence score

$$D_F(\boldsymbol{X}, \boldsymbol{Y}) = F(X, Y) - F(X', Y) - F(X, Y') + F(X', Y').$$

  D_F is a random variable by randomly sampling $X, X' \sim \boldsymbol{X}$, $Y, Y' \sim \boldsymbol{Y}$.

- When the decomposition is perfect ($\delta(\boldsymbol{X}, \boldsymbol{Y}) = 0$) D_F is 0
- We are interested in if the converse holds approximately (Yes!)

Proposition 3

For partition $\boldsymbol{X}, \boldsymbol{Y}$ and arbitrary $\epsilon > 0$, D can be estimated in polynomial time such that

$$\delta(\boldsymbol{X}, \boldsymbol{Y}) \leq \|D_F(\boldsymbol{X}, \boldsymbol{Y})\| \leq 4\delta(\boldsymbol{X}, \boldsymbol{Y}) + \varepsilon,$$

where $\delta(\boldsymbol{X}, \boldsymbol{Y})$ is the true partition error.

When G is $\mathbb{R}$, this approximation is exact

$$\|D_F(\boldsymbol{X}, \boldsymbol{Y})\|_{\mathbb{R},2} = 2 \cdot \delta_{\mathbb{R},2}(\boldsymbol{X}, \boldsymbol{Y}).$$

Thus, $\|D_F\|$ is a good estimate of our objective δ.

Variable partitioning over general groups

Algorithm 1 Approximate partitioning via pairwise estimates

1: **Input:** number of partitions k
2: **Output:** partition $\mathcal{P}$
3: For every pair of variables $\boldsymbol{x}, \boldsymbol{y} \in \boldsymbol{V}$, find estimate $\hat{e}(\boldsymbol{x}, \boldsymbol{y})$ for $e(\boldsymbol{x}, \boldsymbol{y}) = \|D_F(\{\boldsymbol{x}\}, \{\boldsymbol{y}\})\|$;
4: Create a weighted graph with vertices $\boldsymbol{V} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ and weights $\hat{e}(\boldsymbol{x}, \boldsymbol{y})$;
5: Order the edges in increasing weight;
6: **repeat**
7: Remove the edge with the smallest weight;
8: **until** The graph has exactly k connected components

Variable partitioning over general groups

- For general groups, the estimation error is at most $O(kn^2)$ times the optimal error

Proposition 4

Assuming $e(\boldsymbol{x}, \boldsymbol{y}) \leq \hat{e}(\boldsymbol{x}, \boldsymbol{y}) \leq e(\boldsymbol{x}, \boldsymbol{y}) + \varepsilon$ *for all* $\boldsymbol{x}$ *and* $\boldsymbol{y}$*,*

$$\delta(\mathcal{P}) \leq (8k - 10)n^2(4\delta_k(F) + \varepsilon). \tag{4}$$

Partitioning real-valued functions

Theorem 5

For real F, $\mathbb{E}[D_F(\boldsymbol{X}, \overline{\boldsymbol{X}})^2]$ is a symmetric submodular function.

Minimizing submodular function is easy, thus we find almost optimal bipartitions (up to an arbitrarily small sampling error ε).

Theorem 6

Let $F\colon \Sigma^n \to \mathbb{R}$ be a function with $\|F\|_{\mathbb{R},4} \leq 1$. There is an algorithm that given inputs n, ε, β, and oracle access to F, runs in time $O(n^5 \log(n/\beta)/\varepsilon^2)$ and outputs a ε-optimal bipartition $(\boldsymbol{X}, \overline{\boldsymbol{X}})$ with probability at least $1 - \beta$.

When $k > 2$, partitioning real-valued functions is factor-2 optimal.

Outline

Approximate partitioning via submodular minimization

Algorithm 2 Approximately partitioned policy optimization

Input: Total number of samples T, batch size B, partition frequency M_p, number of value iterations M_w, initial policy parameter θ, initial value and advantage parameters w and μ
Output: Optimized policy π_θ
for each iteration j in $[T/B]$ **do**
    Collect a batch of trajectory data $\{s_t^{(i)}, a_t^{(i)}, r_t^{(i)}\}_{i=1}^{B}$;
    for M_θ iterations **do**
        Update θ by one SGD step using PPO with the gradient estimator (3);
    end for
    for M_w iterations **do**
        Update w and μ by minimizing $||V^w(s_t) - R_t||_2^2$ and $\|\hat{A} - A^\mu\|_2^2$ in one SGD step;
    end for
    Estimate $\hat{A}(s_t, a_t)$ using $V^w(s_t)$ by GAE;
    if $j \equiv 0 \pmod{M_p}$ **then**
        Define random function $\xi(\boldsymbol{X})$ to be an estimation of $\mathbb{E}[D_A(\boldsymbol{X}, \overline{\boldsymbol{X}})^2]$;
        Run submodular minimization over $\boldsymbol{X}$ on $\xi(\boldsymbol{X})$;
        Assign $\boldsymbol{X}$ and $\bar{\boldsymbol{X}}$ to $a_{(1)}$ and $a_{(-1)}$ in (3), respectively;
    end if
end for

Comparisons of different algorithms

PG estimator	Variance reduction	Heuristics	Partitioning	Guarantees	Limits
A2C [MBM+16]	CV	-	-	-	-
Wu et al. [WRD+18]	CV & RB[2]	yes	fully	no	$k = n$
POSA [LW18]	CV & RB	yes	greedy	no	**no**
PE[3] [BW20]	CV & RB	**no**	greedy	factor-$O(kn^2)$	**no**
SM[4] [BW20]	CV & RB	**no**	**optimal**	**almost opt**	**no**

Table 1: Comparisons of recent policy optimization algorithms

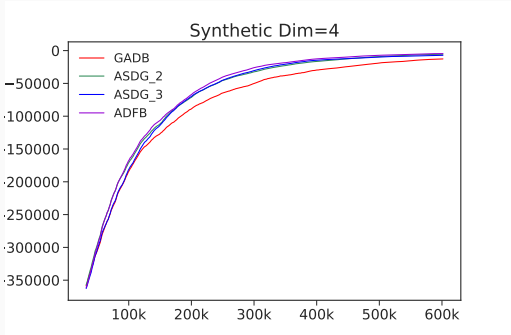
[2] CV: Control Variates; RB: Rao-Blackwellization and divide-and-conquer

[3] Approximate partitioning via pairwise estimates

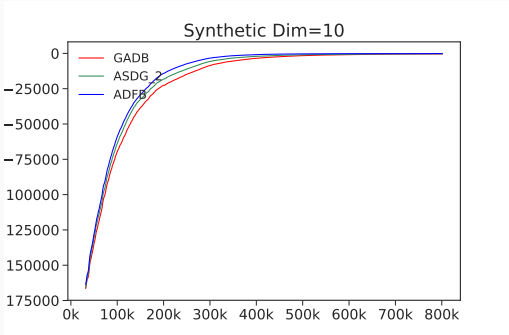
[4] Approximate partitioning via submodular minimization

Experiments

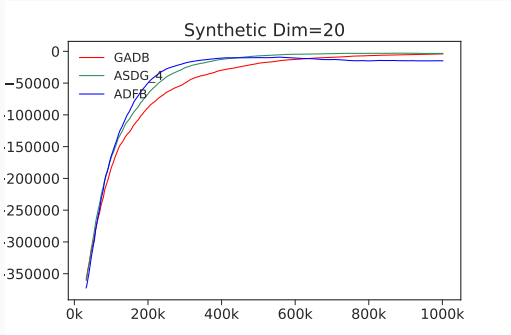
- Synthetic high-dimensional environments [LW18]



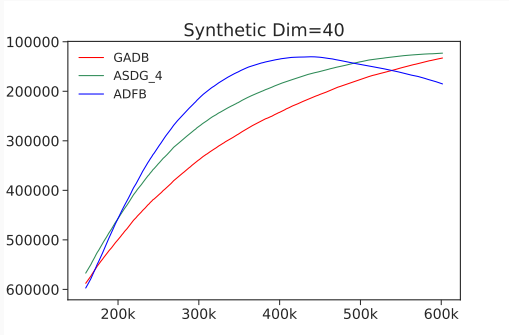
(a) Dim=4, K=2



(b) Dim=10, K=2



(c) Dim=20, K=4

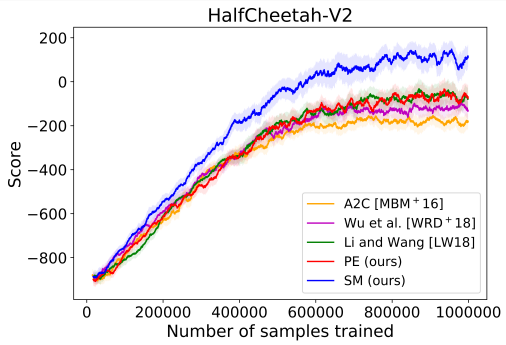


(d) Dim=40, K=4

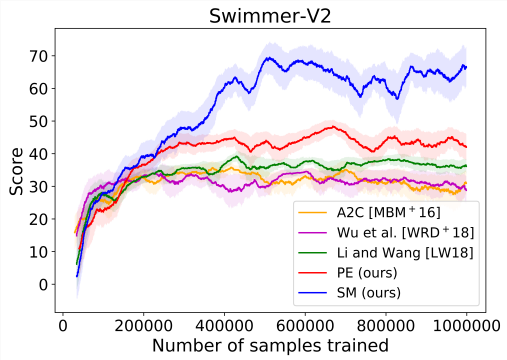
Figure 1: ASDG_k (green): our approach. GADB (k =1, [MBM+16, LFM+18, GCW+18]), ADFB (k =Dim, [WRD+18, Kos18]): baselines.

Experiments

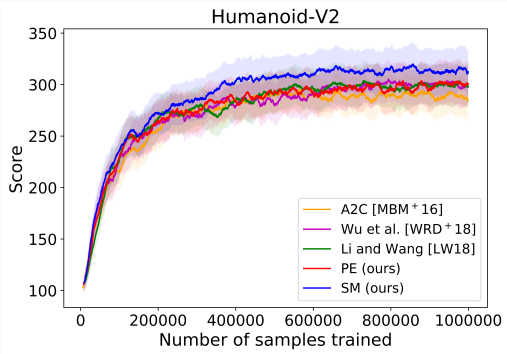
- Experiments on High-dimensional MuJoCo tasks [BW20]



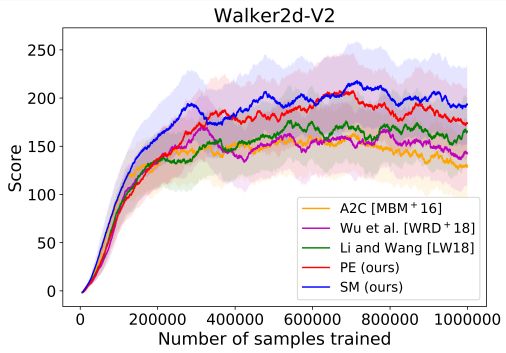
(a) HalfCheetah-v2



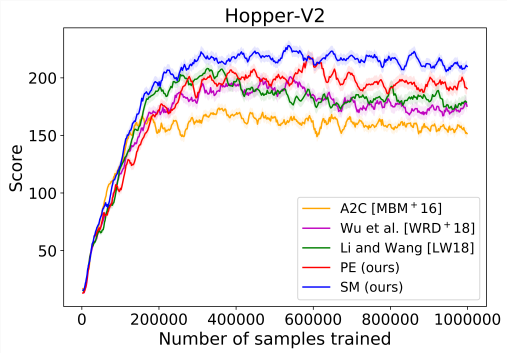
(b) Swimmer-v2



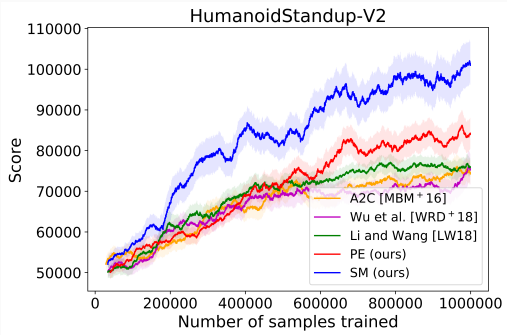
(c) Humanoid-v2



(d) Walker-v2



(e) Hopper-v2



(f) HumanoidSU-v2

Figure 2: SM (blue): Our approach. Each curve is averaged over 10 random seeds. Shades for 1 standard deviation.

Value function approximation

- We possess an unbiased, low variance estimator

$$\nabla_\theta \mathbb{E}[R_T|\theta] = \sum_{j=1}^{k} \mathbb{E}_{\pi(a_{(j)}|s)}[\nabla_\theta \log \pi(a_{(j)}|s)(A(s, a_{(j)})]$$

- We also have almost optimal variable partitions

$$A(s, a) = A_1(s, a_{(1)}) + \cdots + A_k(s, a_{(k)})$$

- Now it amounts to have a good input - a good value function approximation of A.
  This, in fact, is a difficult task studied for long by many researchers. What if we start with a simple case for some insights?

Outline

The Gambler's problem

The Gambler's problem is an early example in the RL textbook by Sutton and Barto [SB18, SB98]

- The gambler starts with $s \leq 1$ capital (state)
- At each round bets a, $0 < a \leq s$ (action) and

$$\text{receives} \begin{cases} 0 & \text{with probability constant } p > 0.5, \\ 2a & \text{with probability } 1 - p. \end{cases}$$

- Target capital is 1. Game terminates at $s = 1$ or $s = 0$

What is the probability of reaching the target, under the best a (the optimal state-value function $v(s)$)?

The problem description

Some additional notes on the problem

1. The problem looks very simple (but deceptively!). It's in fact the most simple RL setting in the book apart from bandits.
2. The original problem starts with capital n, bets only integers, and targets capital N. We solve both the original and the continuous versions.
3. Numerically estimated in the book by the value iteration algorithm. Strange patterns have been observed.

The optimal value function

Theorem 7. $v(s) = \sum_{i=1}^{\infty}(1-p)\gamma^i b_i \prod_{j=1}^{i-1}((1-p)+(2p-1)b_j)$ is the optimal state-value function for any $0 \leq \gamma \leq 1$ and $p > 0.5$, where $s = 0.b_1 b_2 \ldots b_\ell \ldots_{(2)}$ is the binary representation of the state $0 \leq s < 1$.

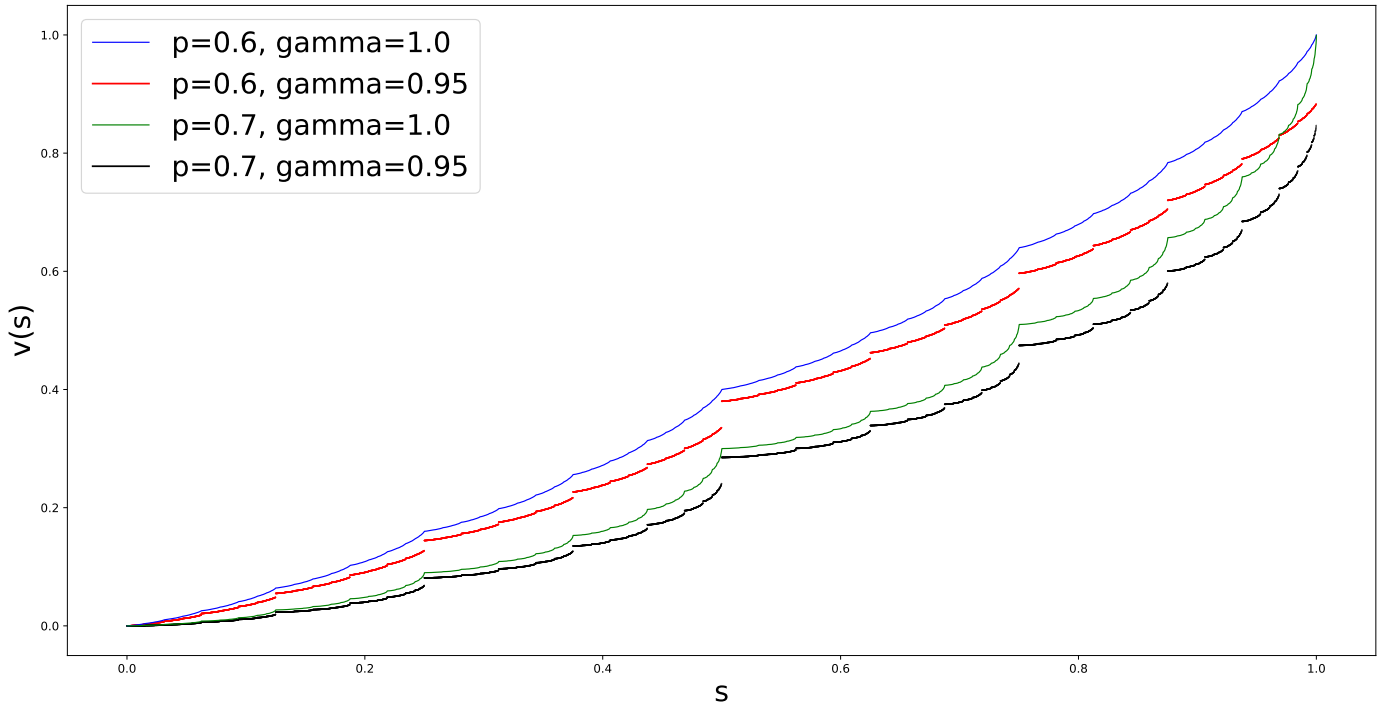
(p: probability of losing a bet. γ: constant discount factor)

- The answer is surprisingly complicated despite the problem being simple
- Describing $v(s)$ using elementary functions is not possible

Plots and characterizations

x-axis: Initial capital (state); y-axis: Probability of winning (value function)

Characterizations: Fractal; self-similar; derivative is either zero or infinity; not written as elementary functions



The solution of optimal value function

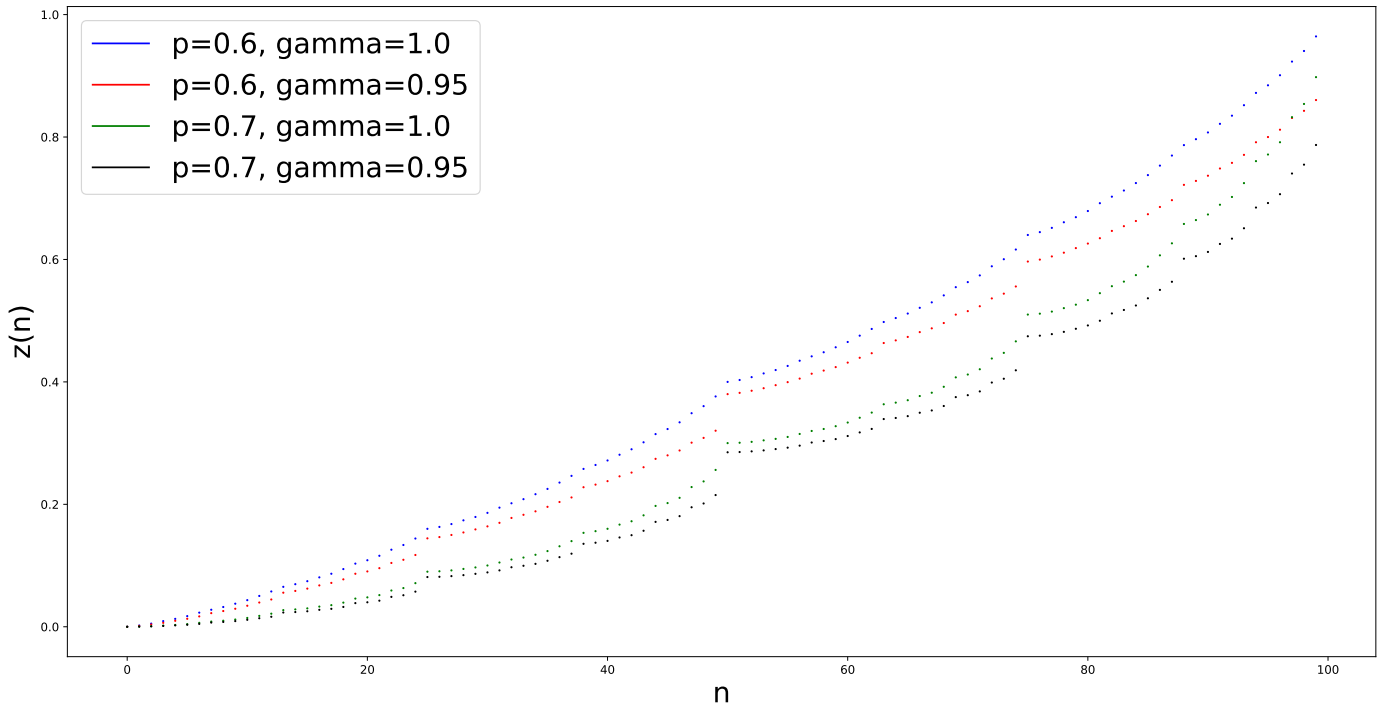
Proposition 8. The optimal value function $z(n)$ is $v(n/N)$ in the discrete setting of the Gambler's problem, where $v(\cdot)$ is the optimal value function under the continuous case defined in Theorem 7.

Corollary 9. The policy $\pi(s) = \min(s, 1-s)$ is (Blackwell) optimal in both the discrete and the continuous cases.

Discrete plots

Discrete problem value function is exactly the continuous problem value function evaluated at discrete points.

This is the "strange pattern" in Sutton and Barto's book.



Bellman equation TD(0)

The Bellman equation of the Gambler's problem is $f(0) = 0$, $f(1) = 1$,

$$f(s) = \max_{0<a\leq \min\{s,1-s\}} (1-p)\gamma f(s+a) + p\gamma f(s-a)$$

for some real function $f : [0, 1] \to \mathbb{R}$.

Theorem 10. Let $\gamma = 1$, $p > 0.5$. $f(s)$ solves the Bellman equation if and only if either

- $f(s)$ is $v(s)$ defined in Theorem 7, or
- $f(0) = 0$, $f(1) = 1$, and $f(s) = C$ for all $0 < s < 1$, for some constant $C \geq 1$.

The mathematical complexity of reinforcement learning

In a difficult case, this problem explores the most fundamental arguments in probabilities and math - the belief of axioms.

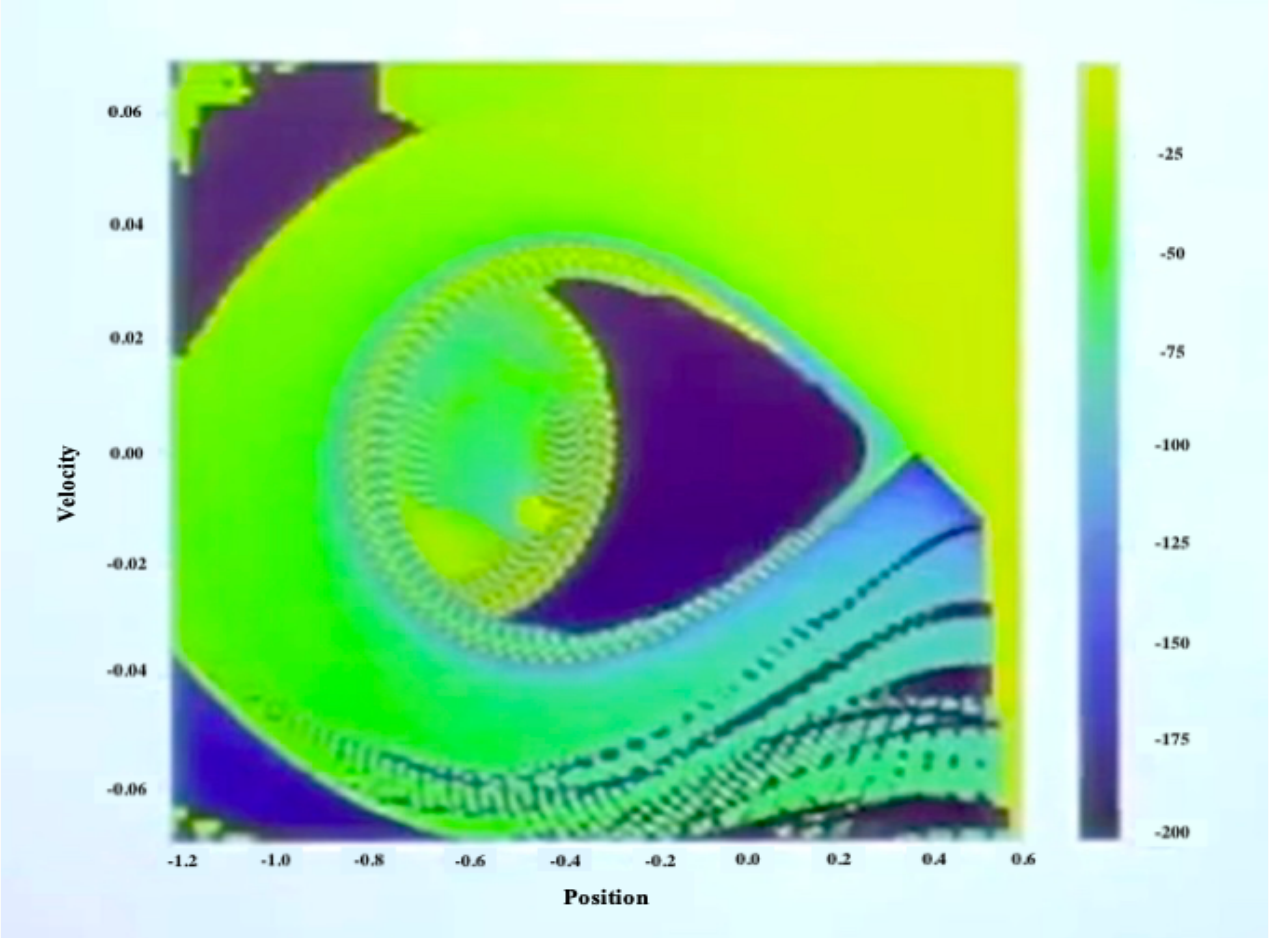
Theorem 11. Let $\gamma = 1$ and $p = 0.5$. A real function $f(s)$ satisfies the Bellman equation if and only if either

- $f(s) = C's + B'$ on $s \in (0, 1)$, for some constants $C' + B' \geq 1$, or
- $f(s)$ is some non-constructive, not Lebesgue measurable function under Axiom of Choice.

Outline

Implications (1) - Generalization

- Similar observations of chaos in other RL problems (e.g. Mountain Car, as below)
- Results and characterizations apply to RL in general



Implications (2) - Fractal and self-similarity

2. The value function is non-smooth on any interval
   - Modern deep reinforcement learning (incorrectly) assume the value function to be smooth to use neural networks.
   - **Proposition 4 and 5.** Using N-bin discretization incurs at least $O(1/N)$ approximation error. Using L-Lipschitz function has at least $O(1/L)$ error.
   - Revisit state and value representation

   The state-of-the-art algorithm, soft actor-critic [HZAL18, HTAL17], learns a smooth surrogate instead of the optimal function. It achieves the state of the art by unintentionally avoiding the optimality.

Implications (3) - Singularity

3. Singularity means a function's derivation takes either zero or infinity, on its entire interval $(0, 1)$.
   - Remark: The curve still goes from $(0, 0)$ to $(1, 1)$, counter-intuitively
   - Algorithmically this denies the access to $\partial v(s)/\partial s$ and $\partial Q(s, a)/\partial a$
     [LHP+15, GLT+17, HWS+15, FA12, Fai08, PYFW19, LJL+18], including famous DDPG and Dyna

Their are many more algorithms than what I can enumerate. The code will always return a *gradient* when called but it will depend on the discrete gradient rather than what the algorithm expect.

Implications (4) - Q-learning

4. The Q-learning algorithm minimizes the Bellman equation. We do not know which point it will converge to.

Optimization and approximation algorithms might prefer a large constant function than the desired optimal value function.

In fact, original Q-learning rarely works in continuous spaces and people did not know why. DeepMind made it work by combination of tricks while biasing the objective.

Conclusion - Improving policy optimization

- Divide-and-conquer algorithms significantly improve policy optimization by variance reduction
- Dividing the advantage function can be achieved by learning variable partitions
  - Partitioning over real or any general groups
  - Bipartitioning or k-partitioning, with guarantees
- Approximating the advantage function can be difficult
  - Fractal and self-similarity for the simplest sequential decision problem
  - Implied hardness of Q-learning and future research directions

Outline

Improving sample quality

- Previously, we discussed how sample efficiency can be improved via variance reduction
- This chapter improves policy optimization by obtaining samples with a higher quality
- The topic is in general rooted in specific applications
  - Methods e.g. reward shaping, hierarchical learning, learning from demonstration, curriculum learning, and meta-learning
- We study reward shaping methods in recurrent attention models

Proposed reward shaping [Wan19]

- For each step t, the step prediction $\hat{y}_t \in [0, 1]$

$$\hat{y}_t = \tanh(W_{ys} s_t).$$

- Assembling $\hat{y}_t$ of all T steps. The final prediction is by time-discounted k-maximum aggregation

$$\hat{\mathcal{Y}} = \frac{1}{Z} \sum_{t \in K} (1 - \gamma^t) \hat{y}_t,$$

where $K = k\text{-argmax}_{t_0 \le t \le T}\{\hat{y}_t\}$ is the set of the indexes of the top k-largest predictions.

Outline

Improving algorithm sensibility [WH19]

- Reinforcement learning algorithms is known to be prone to malicious attacks
- Method in inverse reinforcement learning infers the reward function by observing the policy or the actions
    - For example, a recommendation system may use the reward signals simulated by users' historical records
- We reduce the sensibility of reinforcement learning and guarantee differential privacy, which stringently disables such attacks

Theorem 7

Our Q-learning algorithm is $(\epsilon, \delta + J\exp(-(2k - 8.68\sqrt{\beta}\sigma)^2/2))$*-differentially private with respect to two neighboring reward functions* $\|\mathcal{R} - \mathcal{R}'\|_\infty \leq 1$*, provided that* $2k > 8.68\sqrt{\beta}\sigma$*, and*

$$\sigma \geq \sqrt{2(T/B)\ln(e + \epsilon/\delta)}C(\alpha, k, L, B)/\epsilon.$$

Proposition 8

Let v' *and* v^* *be the value function learned by our algorithm and the optimal value function, respectively. In the case* $J = 1$*,* $|S| = n < \infty$*, and* $\gamma < 1$*, the utility loss of the algorithm satisfies*

$$\mathbb{E}[\frac{1}{n}\|v' - v^*\|_1] \leq \frac{2\sqrt{2}\sigma}{\sqrt{n\pi}(1-\gamma)}.$$

Publications

Thesis Chapter 3:

[LW18] Policy optimization with second-order advantage information. Jiajin Li and Baoxiang Wang. Alphabetical author list. IJCAI 2018.

[BW20] Learning and testing variable partitions. Andrej Bogdanov and Baoxiang Wang. Alphabetical author list. ITCS 2020.

Thesis Chapter 4:

[W19] Recurrent existence determination through policy optimization. Baoxiang Wang. IJCAI 2019.

Thesis Chapter 5:

[WLLC20] The Gambler's problem and beyond. Baoxiang Wang, Shuai Li, Jiajin Li, and Siu On Chan. ICLR 2020.

Publications

[WH19] Privacy-preserving Q-Learning with functional noise in continuous spaces. Baoxiang Wang and Nidhi Hegde. NeurIPS 2019.

[WSZ19] Beyond winning and losing: Modeling human motivations and behaviors using vector-valued inverse reinforcement learning. Baoxiang Wang, Tongfang Sun, and Xianjun Zheng. AAAI-AIIDE 2019.

[LWZC16] Contextual combinatorial cascading bandits. Shuai Li, Baoxiang Wang, Shengyu Zhang, and Wei Chen. ICML 2016.

[YWT19] Metatrace actor-critic: Online step-size tuning by meta-gradient descent for reinforcement learning control. Kenny Young, Baoxiang Wang, and Matthew E. Taylor. IJCAI 2019.

[GWH+19] PAID: Prioritizing app issues for developers by tracking user reviews over versions. Cuiyun Gao, Baoxiang Wang, Pinjia He, Jieming Zhu, Yangfan Zhou, and Michael R. Lyu. ISSRE 2015.

Thank you

Baoxiang Wang

Ph.D. candidate, The Chinese University of Hong Kong

June 18, 2020. CUHK.

References i

[BW20] Andrej Bogdanov and Baoxiang Wang.
Learning and testing variable partitions.
In *Innovations in Theoretical Computer Science*, 2020.

[CR96] George Casella and Christian P Robert.
Rao-blackwellisation of sampling schemes.
Biometrika, 83(1):81–94, 1996.

[DWS12] Thomas Degris, Martha White, and Richard S Sutton.
Off-policy actor-critic.
arXiv preprint arXiv:1205.4839, 2012.

References ii

[FA12] Michael Fairbank and Eduardo Alonso.
Value-gradient learning.
In *The 2012 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2012.

[Fai08] Michael Fairbank.
Reinforcement learning by value gradients.
arXiv preprint arXiv:0803.3539, 2008.

[GCW+18] Will Grathwohl, Dami Choi, Yuhuai Wu, Geoff Roeder, and David Duvenaud.
Backpropagation through the void: Optimizing control variates for black-box gradient estimation.

In *International Conference on Learning Representations*, 2018.

[GLG+16] Shixiang Gu, Timothy Lillicrap, Zoubin Ghahramani, Richard E Turner, and Sergey Levine.
Q-prop: Sample-efficient policy gradient with an off-policy critic.
arXiv preprint arXiv:1611.02247, 2016.

[GLT+17] Shixiang Shane Gu, Timothy Lillicrap, Richard E Turner, Zoubin Ghahramani, Bernhard Schölkopf, and Sergey Levine.

Interpolated policy gradient: Merging on-policy and off-policy gradient estimation for deep reinforcement learning.
In *Advances in neural information processing systems*, pages 3846–3855, 2017.

[HTAL17] Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine.
Reinforcement learning with deep energy-based policies.
In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1352–1361. JMLR. org, 2017.

References v

[HWS+15] Nicolas Heess, Gregory Wayne, David Silver, Timothy Lillicrap, Tom Erez, and Yuval Tassa.
Learning continuous control policies by stochastic value gradients.
In *Advances in Neural Information Processing Systems*, pages 2944–2952, 2015.

[HZAL18] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine.
Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor.
arXiv preprint arXiv:1801.01290, 2018.

[Kos18] Ilya Kostrikov.
Pytorch implementations of reinforcement learning algorithms.
https://github.com/ikostrikov/pytorch-a2c-ppo-acktr, 2018.

[LFM+18] Hao Liu, Yihao Feng, Yi Mao, Dengyong Zhou, Jian Peng, and Qiang Liu.
Action-dependent control variates for policy optimization via stein identity.
In *International Conference on Learning Representations*, 2018.

[LHP+15] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra.
Continuous control with deep reinforcement learning.
arXiv preprint arXiv:1509.02971, 2015.

[LJL+18] Sungsu Lim, Ajin Joseph, Lei Le, Yangchen Pan, and Martha White.
Actor-expert: A framework for using action-value methods in continuous action spaces.
arXiv preprint arXiv:1810.09103, 2018.

[LW18] Jiajin Li and Baoxiang Wang.
Policy optimization with second-order advantage information.
arXiv preprint arXiv:1805.03586, 2018.

[MBM+16] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy P Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu.
Asynchronous methods for deep reinforcement learning.
In *International Conference on Machine Learning*, 2016.

[MKS+15] Volodymyr Mnih, KoPieterray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al.
Human-level control through deep reinforcement learning.
Nature, 518(7540):529, 2015.

[PYFW19] Yangchen Pan, Hengshuai Yao, Amir-massoud Farahmand, and Martha White.
Hill climbing on value estimates for search-control in dyna.
arXiv preprint arXiv:1906.07791, 2019.

References x

[RGB14] Rajesh Ranganath, Sean Gerrish, and David Blei.
Black box variational inference.
In *Artificial Intelligence and Statistics*, pages 814–822, 2014.

[SB98] Richard S Sutton and Andrew G Barto.
Reinforcement learning: An introduction.
MIT press Cambridge, 1998.

[SB18] Richard S Sutton and Andrew G Barto.
Reinforcement learning: An introduction.
MIT press, 2018.

[SLA+15] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz.
Trust region policy optimization.
In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 1889–1897, 2015.

[SMSM00] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour.
Policy gradient methods for reinforcement learning with function approximation.
In *Advances in neural information processing systems*, pages 1057–1063, 2000.

[SS96] Satinder P Singh and Richard S Sutton.
Reinforcement learning with replacing eligibility traces.
Machine learning, 22(1-3):123–158, 1996.

[SWD+17] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov.
Proximal policy optimization algorithms.
arXiv preprint arXiv:1707.06347, 2017.

[TBG+18] George Tucker, Surya Bhupatiraju, Shixiang Gu, Richard E Turner, Zoubin Ghahramani, and Sergey Levine.
The mirage of action-dependent baselines in reinforcement learning.
arXiv preprint arXiv:1802.10031, 2018.

[Wan19] Baoxiang Wang.
Recurrent existence determination through policy optimization.
arXiv preprint arXiv:1905.13551, 2019.

[WD92] Christopher JCH Watkins and Peter Dayan.
Q-learning.
Machine learning, 8(3-4):279–292, 1992.

[WH19] Baoxiang Wang and Nidhi Hegde.
Privacy-preserving q-learning with functional noise in continuous spaces.
In *Advances in Neural Information Processing Systems*, pages 11323–11333, 2019.

[Wil92] Ronald J Williams.
Simple statistical gradient-following algorithms for connectionist reinforcement learning.
Machine learning, 8(3-4):229–256, 1992.

[WRD+18] Cathy Wu, Aravind Rajeswaran, Yan Duan, Vikash Kumar, Alexandre M.Bayen, Sham Kakade, Igor Mordatch, and Pieter Abbeel.
Variance reduction for policy gradient with action-dependent factorized baselines.
In *International Conference on Learning Representations*, 2018.