Q-learning with Uniformly Bounded Variance

Simons Institute Theory of Reinforcement Learning Workshop

Dec 2, 2020

Adithya M. Devraj

Stanford University University of Florida

Based on joint work with S. Chen and S. Meyn @ UF, and A. Bušić @ Inria

Thanks to ARO, NSF, UFII, and the Simons Institute

$$\mathsf{E}\left[\|\theta_n - \theta^*\|^2\right] \leq \frac{1}{(1-\gamma)^p} \cdot \frac{B}{n}$$

Motivation

$$\mathsf{E}\left[\|\theta_n - \theta^*\|^2\right] \leq \frac{1}{(1-\gamma)^p} \cdot \frac{B}{n}$$

$$\tilde{\mathcal{O}}\left(\frac{SA\ln 1/\delta}{\epsilon^2(1-\gamma)^7}\right)$$

$$\eta V_{\max}\sqrt{\frac{Ld}{\nu_N}}\left(\sqrt{\frac{8\log(2d/\delta)}{N}} + \frac{1}{N}\right) \quad \eta = \frac{\gamma}{1-\gamma}$$

$$\tilde{\mathcal{O}}\left(\frac{SA}{\epsilon^4(1-\gamma)^8}\right)$$

$$\tilde{O}\left(\frac{1}{(1-\gamma)^5}\frac{1}{\epsilon^2}\right) \qquad c\left(\frac{r_{\max}^2}{\epsilon^2}\right)\frac{\log\left(\frac{D}{(1-\gamma)\delta}\right)\log\left(\frac{1}{(1-\gamma)\epsilon}\right)}{(1-\gamma)^4}$$

$$\mathbb{E}\left[\|\theta^* - \theta_T\|_2^2\right] \leq \frac{\nu}{\lambda + T} \quad \textit{where} \quad \nu = \max\left\{\frac{8\sigma^2}{(1-\gamma)^2\omega^2}, \frac{16\|\theta^* - \theta_0\|_2^2}{(1-\gamma)^2\omega}\right\}$$

Motivation

$$\mathsf{E}\left[\|\theta_n - \theta^*\|^2\right] \le \frac{1}{(1-\gamma)^p} \cdot \frac{B}{n}$$

Spoiler alert: The factor $1/(1-\gamma)^p$ is due to estimating a constant

Motivation

Stochastic Optimal Control

MDP Model

$\boldsymbol{X}$ is a stationary controlled Markov chain on X, with input $\boldsymbol{U}$ on U

- |X| and |U| are finite
- For all states x and x' in X,

$$\mathsf{P}\{X_{n+1} = x' \mid X_n = x,\ U_n = u, \text{and prior history}\} = P_u(x, x')$$

- $c\colon \mathsf{X}\times\mathsf{U} \to \mathbb{R}$ denotes the cost function, and $\gamma < 1$ the discount factor

Stochastic Optimal Control

MDP Model

$\boldsymbol{X}$ is a stationary controlled Markov chain on X, with input $\boldsymbol{U}$ on U

- |X| and |U| are finite
- For all states x and x' in X,

$$\mathsf{P}\{X_{n+1} = x' \mid X_n = x,\ U_n = u, \text{and prior history}\} = P_u(x, x')$$

- $c\colon \mathsf{X}\times\mathsf{U} \to \mathbb{R}$ denotes the cost function, and $\gamma < 1$ the discount factor

Q-function:

$$Q_\phi(x,u) := \sum_{n=0}^{\infty} \gamma^n \mathsf{E}[c(X_n, U_n) \mid X_0 = x, U_0 = u;\ \ U_n = \phi(X_n),\ n \ge 1]$$

$$Q^*(x,u) := \min_\phi Q_\phi(x,u)$$

Stochastic Optimal Control

MDP Model

$\boldsymbol{X}$ is a stationary controlled Markov chain on X, with input $\boldsymbol{U}$ on U

- $|\mathsf{X}|$ and $|\mathsf{U}|$ are finite
- For all states x and x' in X,

$$\mathsf{P}\{X_{n+1} = x' \mid X_n = x,\ U_n = u, \text{and prior history}\} = P_u(x, x')$$

- $c \colon \mathsf{X} \times \mathsf{U} \to \mathbb{R}$ denotes the cost function, and $\gamma < 1$ the discount factor

Bellman equation: $Q^* = TQ^*$

$$\begin{aligned}(TQ^*)(x,u) &:= c(x,u) + \gamma \mathsf{E}\big[\underline{Q}^*(X_{n+1}) \mid X_n = x,\ U_n = u\big] \\ &= c(x,u) + \gamma \sum_{x'} P_u(x,x') \underline{Q}^*(x')\end{aligned}$$

$$\underline{Q}^*(x) := \min_u Q^*(x,u)$$

Q-learning and Galerkin Relaxation

Dynamic programming goal: Find Q^* that satisfies $Q^* = TQ^*$

Q-learning and Galerkin Relaxation

Dynamic programming goal: Find Q^* that satisfies $Q^* = TQ^*$

$$\mathsf{E}\Big[c(X_n, U_n) + \gamma \underline{Q}^*(X_{n+1}) - Q^*(X_n, U_n) \mid \mathcal{F}_n\Big] = 0$$

Q-learning and Galerkin Relaxation

Dynamic programming goal: Find Q^* that satisfies $Q^* = TQ^*$

$$\mathsf{E}\big[c(X_n,U_n) + \gamma \underline{Q}^*(X_{n+1}) - Q^*(X_n,U_n) \mid \mathcal{F}_n\big] = 0$$

Q-learning goal:

Given $\{Q^\theta : \theta \in \mathbb{R}^d\}$, find θ^* that solves the *Projected Bellman equation*:

$$\bar{f}(\theta^*) = \mathsf{E}\Big[\big[c(X_n,U_n) + \gamma \underline{Q}^{\theta^*}(X_{n+1}) - Q^{\theta^*}(X_n,U_n)\big]\zeta_n\Big] = 0$$

The family $\{Q^\theta\}$ and "*eligibility vectors*" $\{\zeta_n\}$, $\zeta_n \in \mathbb{R}^d$ are part of algorithm design.

Q-learning and Galerkin Relaxation

Dynamic programming goal: Find Q^* that satisfies $Q^* = TQ^*$

$$\mathsf{E}\big[c(X_n,U_n) + \gamma \underline{Q}^*(X_{n+1}) - Q^*(X_n,U_n) \mid \mathcal{F}_n\big] = 0$$

Q-learning goal:

Given $\{Q^\theta : \theta \in \mathbb{R}^d\}$, find θ^* that solves the *Projected Bellman equation*:

$$\bar{f}(\theta^*) = \mathsf{E}\Big[\big[c(X_n,U_n) + \gamma \underline{Q}^{\theta^*}(X_{n+1}) - Q^{\theta^*}(X_n,U_n)\big]\zeta_n\Big] = 0$$

The family $\{Q^\theta\}$ and "*eligibility vectors*" $\{\zeta_n\}$, $\zeta_n \in \mathbb{R}^d$ are part of algorithm design. Example: $\zeta_n = \nabla_\theta Q^\theta(X_n,U_n)\big|_{\theta=\theta^*}$

Q-learning and Galerkin Relaxation

Dynamic programming goal: Find Q^* that satisfies $Q^* = TQ^*$

$$\mathsf{E}\big[c(X_n,U_n) + \gamma \underline{Q}^*(X_{n+1}) - Q^*(X_n,U_n) \mid \mathcal{F}_n\big] = 0$$

Watkins' (tabular) Q-learning:

Given $\{Q^\theta : \theta \in \mathbb{R}^d\}$, find θ^* that solves the *Projected Bellman equation*:

$$\bar{f}(\theta^*) = \mathsf{E}\Big[\big[c(X_n,U_n) + \gamma \underline{Q}^{\theta^*}(X_{n+1}) - Q^{\theta^*}(X_n,U_n)\big]\zeta_n\Big] = 0$$

- Linear parameterization: $Q^\theta(x,u) = \theta^\intercal \psi(x,u)$
- $\zeta_n = \psi(X_n,U_n)$
- $d = |\mathsf{X}|\times|\mathsf{U}|$, $\psi_i(x,u) = \mathbb{I}\{x = x^i, u = u^i\}$ (complete basis)

Q-learning and Galerkin Relaxation

Dynamic programming goal: Find Q^* that satisfies $Q^* = TQ^*$

$$\mathsf{E}\big[c(X_n,U_n) + \gamma \underline{Q}^*(X_{n+1}) - Q^*(X_n,U_n) \mid \mathcal{F}_n\big] = 0$$

Watkins' (tabular) Q-learning:

Given $\{Q^\theta : \theta \in \mathbb{R}^d\}$, find θ^* that solves the *Projected Bellman equation*:

$$\bar{f}(\theta^*) = \mathsf{E}\Big[\big[c(X_n,U_n) + \gamma \underline{Q}^{\theta^*}(X_{n+1}) - Q^{\theta^*}(X_n,U_n)\big]\zeta_n\Big] = 0$$

- Linear parameterization: $Q^\theta(x,u) = \theta^\intercal \psi(x,u)$
- $\zeta_n = \psi(X_n,U_n)$
- $d = |\mathsf{X}|\times|\mathsf{U}|$, $\psi_i(x,u) = \mathbb{I}\{x = x^i, u = u^i\}$ (complete basis)

$$\boxed{\bar{f}(\theta^*) = \Pi\big(TQ^{\theta^*} - Q^{\theta^*}\big)}$$

- $\Pi(i\,,i) = \pi(x^i\,,u^i)$, π is the stationary distribution of $(\boldsymbol{X}\,,\boldsymbol{U})$

Relative Q-learning

Relative Q-learning Goal: Estimate H^* that solves $H^* = \tilde{T}H^*$

$$(\tilde{T}H^*)(x,u) := c(x,u) + \gamma \sum_{x'} P_u(x,x')\underline{H}^*(x') - \delta \cdot \langle \mu, H^* \rangle$$

Relative Q-learning

Relative Q-learning Goal: Estimate H^* that solves $H^* = \tilde{T}H^*$

$$(\tilde{T}H^*)(x,u) := c(x,u) + \gamma \sum_{x'} P_u(x,x')\underline{H}^*(x') - \delta \cdot \langle \mu, H^* \rangle$$

- $\delta > 0$ is a scalar, $\mu : \mathsf{X} \times \mathsf{U} \to [0,1]$ is a pmf, and

$$\langle \mu\,, H^* \rangle := \sum_{(x\,,u)} \mu(x\,,u) H^*(x\,,u)$$

Relative Q-learning

Relative Q-learning Goal: Estimate H^* that solves $H^* = \tilde{T}H^*$

$$(\tilde{T}H^*)(x,u) := c(x,u) + \gamma \sum_{x'} P_u(x,x') \underline{H}^*(x') - \delta \cdot \langle \mu, H^* \rangle$$

- $\delta > 0$ is a scalar, $\mu : \mathsf{X} \times \mathsf{U} \to [0,1]$ is a pmf, and

$$\langle \mu, H^* \rangle := \sum_{(x,u)} \mu(x,u) H^*(x,u)$$

- Q^* from H^*: $\quad Q^*(x,u) = H^*(x,u) + \delta \cdot (1-\gamma)^{-1} \cdot \langle \mu, H^* \rangle$

$$\boxed{\text{But... do we need } Q^*\text{?}}$$

Relative Q-learning

Relative Q-learning Goal: Estimate H^* that solves $H^* = \tilde{T} H^*$

$$(\tilde{T} H^*)(x,u) := c(x,u) + \gamma \sum_{x'} P_u(x,x') \underline{H}^*(x') - \delta \cdot \langle \mu, H^* \rangle$$

- $\delta > 0$ is a scalar, $\mu : \mathsf{X} \times \mathsf{U} \to [0,1]$ is a pmf, and

$$\langle \mu \,, H^* \rangle := \sum_{(x\,,u)} \mu(x\,,u) H^*(x\,,u)$$

- Q^* from H^*: $\qquad Q^*(x,u) = H^*(x,u) + \delta \cdot (1-\gamma)^{-1} \cdot \langle \mu, H^* \rangle$

$\boxed{\text{But... do we need } Q^*\text{?}}$

$\boxed{\text{Advantages of estimating } H^* \text{ instead of } Q^*\text{?}}$

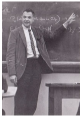
$$\boxed{\mathsf{E}[f(\theta, W)]\Big|_{\theta=\theta^*} \;=\; 0}$$

Stochastic Approximation

Stochastic Approximation

Goal: Find the solution θ^* to $\bar{f}(\theta^*) = 0$, where

$$\bar{f}(\theta) := \mathsf{E}[f(\theta, W_{n+1})]\,, \qquad \theta \in \mathbb{R}^d\,,\ \bar{f} : \mathbb{R}^d \to \mathbb{R}^d$$



Algorithm: $\boxed{\theta_{n+1} = \theta_n + \alpha_{n+1} f(\theta_n, W_{n+1})}$ [Robbins & Monro 1951]

We assume $\alpha_n = g/(n+1)$ with $g > 0$

Analysis: θ^* is the *stationary point* of the ODE

$$\boxed{\frac{d}{dt} x(t) = \bar{f}(x(t))}$$

SA is a noisy Euler discretization:

$$\boxed{\theta_{n+1} = \theta_n + \alpha_{n+1}[\bar{f}(\theta_n) + \Delta_{n+1}]}\,, \qquad \Delta_{n+1} \equiv f(\theta_n, W_{n+1}) - \bar{f}(\theta_n)$$

MDS *for Tabular Q-learning*

Convergence Rates of SA

Goal: Find θ^* such that $\bar{f}(\theta^*) = 0$ Algorithm: $\theta_{n+1} = \theta_n + \alpha_{n+1}[\bar{f}(\theta_n) + \Delta_{n+1}]$

- Error sequence: $\tilde{\theta}_n := \theta_n - \theta^*$

Convergence Rates of SA

Goal: Find θ^* such that $\bar{f}(\theta^*) = 0$ Algorithm: $\theta_{n+1} = \theta_n + \alpha_{n+1}[\bar{f}(\theta_n) + \Delta_{n+1}]$

- Error sequence: $\tilde{\theta}_n := \theta_n - \theta^*$
- Asymptotic covariance: $\Sigma^{\theta}_{\infty} = \lim_{n\to\infty} n\mathsf{E}\big[\tilde{\theta}_n\tilde{\theta}_n^{\intercal}\big]$

Convergence Rates of SA

Goal: Find θ^* such that $\bar{f}(\theta^*) = 0$ Algorithm: $\theta_{n+1} = \theta_n + \alpha_{n+1}[\bar{f}(\theta_n) + \Delta_{n+1}]$

- Error sequence: $\tilde{\theta}_n := \theta_n - \theta^*$
- Asymptotic covariance: $\Sigma_\infty^\theta = \lim_{n\to\infty} n\mathsf{E}\big[\tilde{\theta}_n \tilde{\theta}_n^{\intercal}\big]$

Asymptotic Variance Theory for SA

- Denote $\Sigma_\Delta = \mathsf{E}[\Delta_{n+1}\Delta_{n+1}^{\intercal}]$ and $A = \partial_\theta \bar{f}(\theta)\big|_{\theta=\theta^*}$
- If all $\mathrm{Re}\,\big(\lambda(gA)\big) < -\frac{1}{2}$, Σ_∞^θ solves the Lyapunov equation:

$$\boxed{0 = (gA + \tfrac{1}{2}I)\Sigma_\infty^\theta + \Sigma_\infty^\theta (gA + \tfrac{1}{2}I)^{\intercal} + g^2\Sigma_\Delta}$$

- If $\mathrm{Re}\,\big(\lambda(gA)\big) \geq -\frac{1}{2}$ for some eigenvalue, then Σ_∞^θ is (typically) infinite

Convergence Rates of SA

Goal: Find θ^* such that $\bar{f}(\theta^*) = 0$ Algorithm: $\theta_{n+1} = \theta_n + \alpha_{n+1}[\bar{f}(\theta_n) + \Delta_{n+1}]$

- Error sequence: $\tilde{\theta}_n := \theta_n - \theta^*$
- Asymptotic covariance: $\Sigma_\infty^\theta = \lim_{n\to\infty} n\mathsf{E}\big[\tilde{\theta}_n \tilde{\theta}_n^{\intercal}\big]$

Asymptotic Variance Theory for SA

- Denote $\Sigma_\Delta = \mathsf{E}[\Delta_{n+1}\Delta_{n+1}^{\intercal}]$ and $A = \partial_\theta \bar{f}(\theta)\big|_{\theta=\theta^*}$
- If all $\text{Re}\,\big(\lambda(gA)\big) < -\frac{1}{2}$, Σ_∞^θ solves the Lyapunov equation:

$$\boxed{0 = (gA + \tfrac{1}{2}I)\Sigma_\infty^\theta + \Sigma_\infty^\theta(gA + \tfrac{1}{2}I)^{\intercal} + g^2\Sigma_\Delta}$$

- If $\text{Re}\,\big(\lambda(gA)\big) \geq -\frac{1}{2}$ for some eigenvalue, then Σ_∞^θ is (typically) infinite

- Asymptotically Optimal SA Algorithms: $A^{-1}\Sigma_\Delta(A^{-1})^{\intercal}$

Convergence Rates of SA

Goal: Find θ^* such that $\bar{f}(\theta^*) = 0$ Algorithm: $\theta_{n+1} = \theta_n + \alpha_{n+1}[\bar{f}(\theta_n) + \Delta_{n+1}]$

- Error sequence: $\tilde{\theta}_n := \theta_n - \theta^*$
- Asymptotic covariance: $\Sigma^\theta_\infty = \lim_{n\to\infty} n\mathsf{E}\big[\tilde{\theta}_n \tilde{\theta}_n^{\mathsf{T}}\big]$

Asymptotic Variance Theory for SA

- Denote $\Sigma_\Delta = \mathsf{E}[\Delta_{n+1}\Delta_{n+1}^{\mathsf{T}}]$ and $A = \partial_\theta \bar{f}(\theta)\big|_{\theta=\theta^*}$
- If all Re $\big(\lambda(gA)\big) < -\frac{1}{2}$, Σ^θ_∞ solves the Lyapunov equation:

$$\boxed{0 = (gA + \tfrac{1}{2}I)\Sigma^\theta_\infty + \Sigma^\theta_\infty(gA + \tfrac{1}{2}I)^{\mathsf{T}} + g^2\Sigma_\Delta}$$

- If Re $\big(\lambda(gA)\big) \geq -\frac{1}{2}$ for some eigenvalue, then Σ^θ_∞ is (typically) infinite

- Asymptotically Optimal SA Algorithms: $A^{-1}\Sigma_\Delta(A^{-1})^{\mathsf{T}}$

  Examples: LSTD(λ), Ruppert's Stochastic Newton Raphson, Polyak-Ruppert Averaging Technique, Zap Q-learning
  [D. & Meyn, 2017], [D., 2019]

$$\boxed{\mathsf{E}[f(\theta, W)]\Big|_{\theta=\theta^*} = 0}$$

$$\bar{f}(\theta^*) = \Pi\big(TQ^{\theta^*} - Q^{\theta^*}\big)$$

Stochastic Approximation → Q-learning

Watkins' Q-learning

Q-learning is SA:

$$Q_{n+1}(X_n, U_n) = Q_n(X_n, U_n) + \alpha_{n+1}\Big(c(X_n, U_n) + \gamma \underline{Q}_n(X_{n+1}) - Q_n(X_n, U_n)\Big)$$

Watkins' Q-learning

Q-learning is SA:

$$Q_{n+1}(X_n, U_n) = Q_n(X_n, U_n) + \alpha_{n+1}\Big(c(X_n, U_n) + \gamma \underline{Q}_n(X_{n+1}) - Q_n(X_n, U_n)\Big)$$

Case 1: $\alpha_n = 1/n$

Linearization Matrix: $A = -\Pi[I - \gamma P_{\phi^*}]$

Watkins' Q-learning

Q-learning is SA:

$$Q_{n+1}(X_n, U_n) = Q_n(X_n, U_n) + \alpha_{n+1}\big(c(X_n, U_n) + \gamma \underline{Q}_n(X_{n+1}) - Q_n(X_n, U_n)\big)$$

Case 1: $\alpha_n = 1/n$

Linearization Matrix: $A = -\Pi[I - \gamma P_{\phi^*}]$

$$\max\big\{\mathrm{Re}\big(\lambda(A)\big)\big\} \geq -(1-\gamma)\max_{x,u} \pi(x,u)$$

Watkins' Q-learning

Q-learning is SA:

$$Q_{n+1}(X_n,U_n)=Q_n(X_n,U_n)+\alpha_{n+1}\Big(c(X_n,U_n)+\gamma\underline{Q}_n(X_{n+1})-Q_n(X_n,U_n)\Big)$$

Case 1: $\alpha_n = 1/n$

Linearization Matrix: $A = -\Pi[I-\gamma P_{\phi^*}]$

$$\max\Big\{\mathrm{Re}\big(\lambda(A)\big)\Big\} \geq -(1-\gamma)\max_{x,u}\pi(x,u)$$

$$\boxed{\|\Sigma^\theta_\infty\| = \infty \text{ if } \gamma > \tfrac{1}{2}} \qquad \max\Big\{\mathrm{Re}\big(\lambda(A)\big)\Big\} > -\tfrac{1}{2}$$

Watkins' Q-learning

Q-learning is SA:

$$Q_{n+1}(X_n,U_n)=Q_n(X_n,U_n)+\alpha_{n+1}\big(c(X_n,U_n)+\gamma \underline{Q}_n(X_{n+1})-Q_n(X_n,U_n)\big)$$

Case 1: $\alpha_n = 1/n$

Linearization Matrix: $A = -\Pi[I - \gamma P_{\phi^*}]$

$$\max\big\{\mathrm{Re}\big(\lambda(A)\big)\big\} \geq -(1-\gamma)\max_{x,u}\pi(x,u)$$

$$\boxed{\|\Sigma_\infty^\theta\| = \infty \text{ if } \gamma > \tfrac{1}{2}} \qquad \max\big\{\mathrm{Re}\big(\lambda(A)\big)\big\} > -\tfrac{1}{2}$$

"Asymptotic" MSE convergence rate is slower than $1/n^{2(1-\gamma)}$ if $\gamma > \frac{1}{2}$

Watkins' Q-learning

Q-learning is SA:

$$Q_{n+1}(X_n,U_n)=Q_n(X_n,U_n)+\alpha_{n+1}\Big(c(X_n,U_n)+\gamma\underline{Q}_n(X_{n+1})-Q_n(X_n,U_n)\Big)$$

Case 2: $\alpha_n(x,u)=\big[n(x,u)\big]^{-1}$

Watkins' Q-learning

Q-learning is SA:

$$Q_{n+1}(X_n,U_n) = Q_n(X_n,U_n) + \alpha_{n+1}\Big(c(X_n,U_n) + \gamma \underline{Q}_n(X_{n+1}) - Q_n(X_n,U_n)\Big)$$

Case 2: $\alpha_n(x,u) = \big[n(x,u)\big]^{-1}$

Linearization Matrix: $\quad A = -[I - \gamma P_{\phi^*}]$

$$\lambda_{\max}(A) = -(1-\gamma)\,, \quad \text{with right eigenvector } \mathbb{1}$$

Watkins' Q-learning

Q-learning is SA:

$$Q_{n+1}(X_n,U_n) = Q_n(X_n,U_n) + \alpha_{n+1}\big(c(X_n,U_n) + \gamma \underline{Q}_n(X_{n+1}) - Q_n(X_n,U_n)\big)$$

Case 2: $\alpha_n(x,u) = \big[n(x,u)\big]^{-1}$

Linearization Matrix: $A = -[I - \gamma P_{\phi^*}]$

$\lambda_{\max}(A) = -(1-\gamma)\,,$ with right eigenvector $\mathbb{1}$

$$\boxed{\|\Sigma^\theta_\infty\| = \infty \text{ if } \gamma > \tfrac{1}{2}}$$

Watkins' Q-learning

Q-learning is SA:

$$Q_{n+1}(X_n,U_n)=Q_n(X_n,U_n)+\alpha_{n+1}\Big(c(X_n,U_n)+\gamma\underline{Q}_n(X_{n+1})-Q_n(X_n,U_n)\Big)$$

Case 2: $\alpha_n(x,u)=\big[n(x,u)\big]^{-1}$

Linearization Matrix: $A=-[I-\gamma P_{\phi^*}]$

$\lambda_{\max}(A)=-(1-\gamma)\,,$ with right eigenvector $\mathbb{1}$

"Asymptotic" MSE convergence rate is $1/n^{2(1-\gamma)}$ if $\gamma>\frac{1}{2}$

Watkins' Q-learning

Q-learning is SA:

$$Q_{n+1}(X_n,U_n)=Q_n(X_n,U_n)+\alpha_{n+1}\big(c(X_n,U_n)+\gamma\underline{Q}_n(X_{n+1})-Q_n(X_n,U_n)\big)$$

Case 2: $\alpha_n(x,u)=\big[n(x,u)\big]^{-1}$

Linearization Matrix: $A=-[I-\gamma P_{\phi^*}]$

$\lambda_{\max}(A)=-(1-\gamma)\,,$ with right eigenvector $\mathbb{1}$

$\boxed{\text{"Asymptotic" MSE convergence rate is } 1/n^{2(1-\gamma)} \text{ if } \gamma>\tfrac{1}{2}}$

$\boxed{\text{Convergence rate is } 1/n, \text{ if } \alpha_n(x,u)=(1-\gamma)^{-1}\big[n(x,u)\big]^{-1}}$

Watkins' Q-learning

Q-learning is SA:

$$Q_{n+1}(X_n,U_n)=Q_n(X_n,U_n)+\alpha_{n+1}\big(c(X_n,U_n)+\gamma\underline{Q}_n(X_{n+1})-Q_n(X_n,U_n)\big)$$

Case 2: $\alpha_n(x,u)=\big[n(x,u)\big]^{-1}$

Linearization Matrix: $A=-[I-\gamma P_{\phi^*}]$

$$\lambda_{\max}(A)=-(1-\gamma)\,,\quad \text{with right eigenvector } \mathbb{1}$$

"Asymptotic" MSE convergence rate is $1/n^{2(1-\gamma)}$ if $\gamma>\frac{1}{2}$

Convergence rate is $1/n$, if $\alpha_n(x,u)=(1-\gamma)^{-1}\big[n(x,u)\big]^{-1}$

But... $\|\Sigma^\theta_\infty\|\propto(1-\gamma)^{-2}$

Relative Q-learning

Relative Q-learning Algorithm

$$H_{n+1}(X_n, U_n) = H_n(X_n, U_n) + \alpha_{n+1}\Big(c(X_n, U_n) + \gamma \underline{H}_n(X_{n+1}) - H_n(X_n, U_n) - \delta\langle\mu, H_n\rangle\Big)$$

Relative Q-learning

Relative Q-learning Algorithm

$$H_{n+1}(X_n,U_n)=H_n(X_n,U_n)+\alpha_{n+1}\big(c(X_n,U_n)+\gamma\underline{H}_n(X_{n+1})-H_n(X_n,U_n)-\delta\langle\mu,H_n\rangle\big)$$

Eigenvalue test [D., & Meyn, 2020]

$$\boxed{A=-[I-\gamma P_{\phi^*}+\delta\cdot\mathbb{1}\otimes\mu]}$$

- $\lambda_{\mathbb{1}}$ for eigenvector $\mathbb{1}$ is $-(1-\gamma+\delta)$

Relative Q-learning

Relative Q-learning Algorithm

$$H_{n+1}(X_n,U_n)=H_n(X_n,U_n)+\alpha_{n+1}\big(c(X_n,U_n)+\gamma\underline{H}_n(X_{n+1})-H_n(X_n,U_n)-\delta\langle\mu,H_n\rangle\big)$$

Eigenvalue test [D., & Meyn, 2020]

$$\boxed{A=-[I-\gamma P_{\phi^*}+\delta\cdot\mathbb{1}\otimes\mu]}$$

- $\lambda_{\mathbb{1}}$ for eigenvector $\mathbb{1}$ is $-(1-\gamma+\delta)$
- All other eigenvalues satisfy $\mathrm{Re}(\lambda(A))\le-(1-\gamma\rho^*)$,

$$\rho^*=\max\{\mathrm{Re}(\lambda(P_{\phi^*})):\lambda\neq\lambda_{\mathbb{1}}\},$$

Relative Q-learning

Relative Q-learning Algorithm

$$H_{n+1}(X_n, U_n) = H_n(X_n, U_n) + \alpha_{n+1}\big(c(X_n, U_n) + \gamma \underline{H}_n(X_{n+1}) - H_n(X_n, U_n) - \delta\langle \mu, H_n\rangle\big)$$

Eigenvalue test [D., & Meyn, 2020]

$$\boxed{A = -[I - \gamma P_{\phi^*} + \delta \cdot \mathbb{1} \otimes \mu]}$$

- $\lambda_{\mathbb{1}}$ for eigenvector $\mathbb{1}$ is $-(1 - \gamma + \delta)$
- All other eigenvalues satisfy $\mathrm{Re}(\lambda(A)) \leq -(1 - \gamma\rho^*)$,

$$\rho^* = \max\{\mathrm{Re}(\lambda(P_{\phi^*})) : \lambda \neq \lambda_{\mathbb{1}}\},$$

- Finite asymptotic variance with

$$\alpha_n(x, u) = \big[n(x, u)\big]^{-1} \cdot (1 - \rho^*\gamma)^{-1}$$

Relative Q-learning

Relative Q-learning Algorithm

$$H_{n+1}(X_n,U_n)=H_n(X_n,U_n)+\alpha_{n+1}\big(c(X_n,U_n)+\gamma\underline{H}_n(X_{n+1})-H_n(X_n,U_n)-\delta\langle\mu,H_n\rangle\big)$$

Eigenvalue test [D., & Meyn, 2020]

$$\boxed{A=-[I-\gamma P_{\phi^*}+\delta\cdot\mathbb{1}\otimes\mu]}$$

- $\lambda_{\mathbb{1}}$ for eigenvector $\mathbb{1}$ is $-(1-\gamma+\delta)$
- All other eigenvalues satisfy $\mathrm{Re}(\lambda(A))\le-(1-\gamma\rho^*)$,

$$\rho^*=\max\{\mathrm{Re}(\lambda(P_{\phi^*})):\lambda\neq\lambda_{\mathbb{1}}\},$$

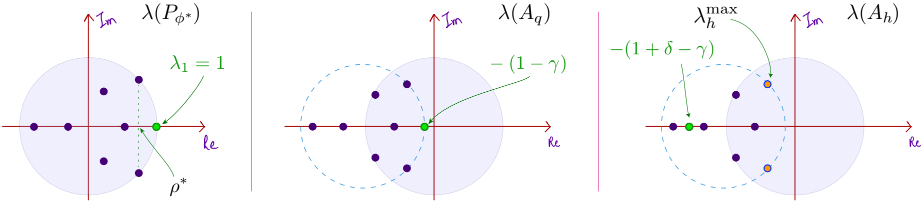
- Finite asymptotic variance with

$$\alpha_n(x,u)=\big[n(x,u)\big]^{-1}\cdot(1-\rho^*\gamma)^{-1}$$

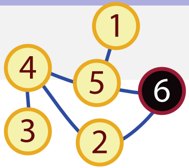
$$\boxed{\|\Sigma_\infty^\theta\|\text{ is proportional to }(1-\rho^*\gamma)^{-2}\text{ !!}}$$

Relative Q-learning

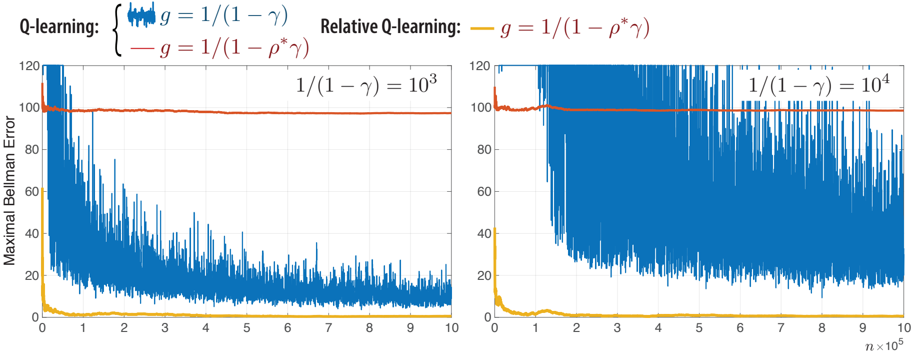
Eigenvalue Analysis



Application to Stochastic Shortest Path



Maximal Bellman error for $\gamma = 0.999$ and $\gamma = 0.9999$



A Twist in the Tail

A Twist in the Tail

Q-learning vs. Relative Q-learning

More Eigenvalue analysis [D., & Meyn, 2020]

$$\boxed{A_h = -[I - \gamma P_{\phi^*} + \delta \cdot \mathbb{1} \otimes \mu]} \qquad \lambda_{\mathbb{1}} = -(1 - \gamma + \delta)$$

$$\boxed{A_q = -[I - \gamma P_{\phi^*}]} \qquad \lambda_{\mathbb{1}} = -(1 - \gamma)$$

A Twist in the Tail

Q-learning vs. Relative Q-learning

More Eigenvalue analysis [D., & Meyn, 2020]

$$\boxed{A_h = -[I - \gamma P_{\phi^*} + \delta \cdot \mathbb{1} \otimes \mu]} \qquad \lambda_{\mathbb{1}} = -(1 - \gamma + \delta)$$

$$\boxed{A_q = -[I - \gamma P_{\phi^*}]} \qquad \lambda_{\mathbb{1}} = -(1 - \gamma)$$

- All other eigenvalues coincide: $(\lambda(A_h)) = (\lambda(A_q))$, $\lambda \neq \lambda_{\mathbb{1}}$

A Twist in the Tail

Q-learning vs. Relative Q-learning

More Eigenvalue analysis [D., & Meyn, 2020]

$$\boxed{A_h = -[I - \gamma P_{\phi^*} + \delta \cdot \mathbb{1} \otimes \mu]} \qquad \lambda_{\mathbb{1}} = -(1 - \gamma + \delta)$$

$$\boxed{A_q = -[I - \gamma P_{\phi^*}]} \qquad \lambda_{\mathbb{1}} = -(1 - \gamma)$$

- All other eigenvalues coincide: $(\lambda(A_h)) = (\lambda(A_q))$, $\lambda \neq \lambda_{\mathbb{1}}$
- For all $\nu, w \in \{v : v^\dagger \mathbb{1} = 0\}$, $\nu^\dagger \Sigma_\infty^\theta w$ is the same for both algorithms, *provided, same g is used!*

A Twist in the Tail

Q-learning vs. Relative Q-learning

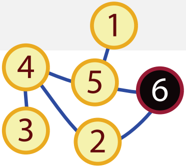
More Eigenvalue analysis [D., & Meyn, 2020]

$$\boxed{A_h = -[I - \gamma P_{\phi^*} + \delta \cdot \mathbb{1} \otimes \mu]} \qquad \lambda_{\mathbb{1}} = -(1 - \gamma + \delta)$$

$$\boxed{A_q = -[I - \gamma P_{\phi^*}]} \qquad \lambda_{\mathbb{1}} = -(1 - \gamma)$$

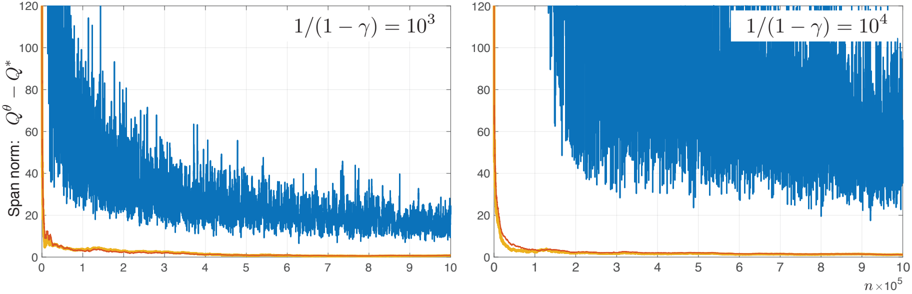
- All other eigenvalues coincide: $(\lambda(A_h)) = (\lambda(A_q))$, $\lambda \neq \lambda_{\mathbb{1}}$
- For all $\nu, w \in \{v : v^\dagger \mathbb{1} = 0\}$, $\nu^\dagger \Sigma_\infty^\theta w$ is the same for both algorithms, *provided, same g is used!*
- Convergence rate of the two algorithms is same, *except in the subspace corresponding to the constant basis function*

Application to Stochastic Shortest Path

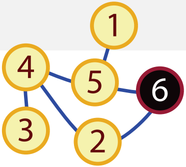


Span semi-norm of error for $\gamma = 0.999$ and $\gamma = 0.9999$

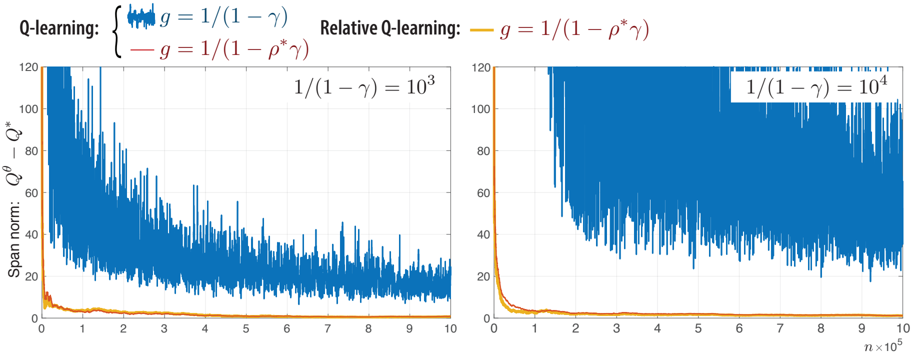
Q-learning: $\begin{cases} g = 1/(1-\gamma) \\ g = 1/(1-\rho^*\gamma) \end{cases}$ **Relative Q-learning:** $g = 1/(1-\rho^*\gamma)$



Application to Stochastic Shortest Path



Span semi-norm of error for $\gamma = 0.999$ and $\gamma = 0.9999$



- Does this property extend beyond tabular setting?

Conclusions & Future Work

- We used *asymptotic theory of SA* to design & analyze relative Q-learning

Conclusions & Future Work

- We used *asymptotic theory of SA* to design & analyze relative Q-learning
- Most "complexity" in classical Q-learning seems to be spent on estimating the "constant"

Conclusions & Future Work

- We used *asymptotic theory of SA* to design & analyze relative Q-learning
- Most "complexity" in classical Q-learning seems to be spent on estimating the "constant" *Policy is all that we care about..*

Conclusions & Future Work

- We used *asymptotic theory of SA* to design & analyze relative Q-learning
- Most "complexity" in classical Q-learning seems to be spent on estimating the "constant" *Policy is all that we care about..*
  Keeping everything relative doesn't hurt

Conclusions & Future Work

- We used *asymptotic theory of SA* to design & analyze relative Q-learning
- Most "complexity" in classical Q-learning seems to be spent on estimating the "constant" *Policy is all that we care about..*
  Keeping everything relative doesn't hurt
- The *relative* Q-learning algorithm results in asymptotic variance that is *uniformly bounded for all* $\gamma < 1$

Conclusions & Future Work

- We used *asymptotic theory of SA* to design & analyze relative Q-learning
- Most "complexity" in classical Q-learning seems to be spent on estimating the "constant" *Policy is all that we care about..*

  Keeping everything relative doesn't hurt
- The *relative* Q-learning algorithm results in asymptotic variance that is *uniformly bounded for all* $\gamma < 1$

  It also directly gives us the Q-function..

Conclusions & Future Work

- We used *asymptotic theory of SA* to design & analyze relative Q-learning
- Most "complexity" in classical Q-learning seems to be spent on estimating the "constant" *Policy is all that we care about..*

  Keeping everything relative doesn't hurt
- The *relative* Q-learning algorithm results in asymptotic variance that is *uniformly bounded for all* $\gamma < 1$

  It also directly gives us the Q-function.. ***but do we need it?***

Conclusions & Future Work

- We used *asymptotic theory of SA* to design & analyze relative Q-learning
- Most "complexity" in classical Q-learning seems to be spent on estimating the "constant" *Policy is all that we care about..*
  Keeping everything relative doesn't hurt
- The *relative* Q-learning algorithm results in asymptotic variance that is *uniformly bounded for all* $\gamma < 1$
  It also directly gives us the Q-function.. ***but do we need it?***
- Same for TD-learning algorithms when used as a part of actor-critic or LSPI

Conclusions & Future Work

- We used *asymptotic theory of SA* to design & analyze relative Q-learning
- Most "complexity" in classical Q-learning seems to be spent on estimating the "constant" *Policy is all that we care about..*

  Keeping everything relative doesn't hurt
- The *relative* Q-learning algorithm results in asymptotic variance that is *uniformly bounded for all* $\gamma < 1$

  It also directly gives us the Q-function.. ***but do we need it?***
- Same for TD-learning algorithms when used as a part of actor-critic or LSPI
- The algorithm can be Zapped: $A_h^{-1}\Sigma_\Delta A_h^{-1} \ll A_q^{-1}\Sigma_\Delta A_q^{-1}$

  Globally stable even with non-linear function approximation

Conclusions & Future Work

- We used *asymptotic theory of SA* to design & analyze relative Q-learning
- Most "complexity" in classical Q-learning seems to be spent on estimating the "constant" *Policy is all that we care about..*
  Keeping everything relative doesn't hurt
- The *relative* Q-learning algorithm results in asymptotic variance that is *uniformly bounded for all* $\gamma < 1$
  It also directly gives us the Q-function.. ***but do we need it?***
- Same for TD-learning algorithms when used as a part of actor-critic or LSPI
- The algorithm can be Zapped: $A_h^{-1}\Sigma_\Delta A_h^{-1} \ll A_q^{-1}\Sigma_\Delta A_q^{-1}$
  Globally stable even with non-linear function approximation
- Can apply averaging, acceleration, variance reduction, etc.

Conclusions & Future Work

- We used *asymptotic theory of SA* to design & analyze relative Q-learning
- Most "complexity" in classical Q-learning seems to be spent on estimating the "constant" *Policy is all that we care about..*

  Keeping everything relative doesn't hurt
- The *relative* Q-learning algorithm results in asymptotic variance that is *uniformly bounded for all* $\gamma < 1$

  It also directly gives us the Q-function.. ***but do we need it?***
- Same for TD-learning algorithms when used as a part of actor-critic or LSPI
- The algorithm can be Zapped: $A_h^{-1}\Sigma_\Delta A_h^{-1} \ll A_q^{-1}\Sigma_\Delta A_q^{-1}$

  Globally stable even with non-linear function approximation
- Can apply averaging, acceleration, variance reduction, etc.
- Open problem: Finite-n analysis, and extension of theory to episodic RL

$$\mathsf{E}\|\theta_n - \theta^*\|^2 \leq (1-\rho^*\gamma)^{-p} \cdot B/n \ ?$$

References

- A. M. Devraj, and S. P. Meyn, *Q-learning with Uniformly Bounded Variance: Large Discounting is Not a Barrier to Fast Learning. Submitted to IEEE Transactions on Automatic Control. Available on arXiv.* 2020.
- J. Abounadi, D. Bertsekas, and V. S. Borkar, *Learning algorithms for Markov decision processes with average cost. SIAM Journal on Control and Optimization.* 2001.
- S. C. Chen, A. M. Devraj, A. Bušić, and S. P. Meyn, *Explicit MSE Bounds for Monte-Carlo and SA. AISTATS,* 2020.
- A. M. Devraj, A. Bušić, and S. P. Meyn, *Fundamental design principles for reinforcement learning algorithms. Handbook on Reinforcement Learning and Control.* Springer, 2020.
- A. M. Devraj, *Reinforcement Learning Design with Optimal Learning Rate.* PhD Thesis. Dec. 2019.

Thank you!