Newton, Polyak and Nesterov

Quickest Stochastic Approximation and Reinforcement Learning

Reinforcement Learning and Data-Driven Control
INFORMS 2019

Adithya M. Devraj

Department of Electrical and Computer Engineering  University of Florida

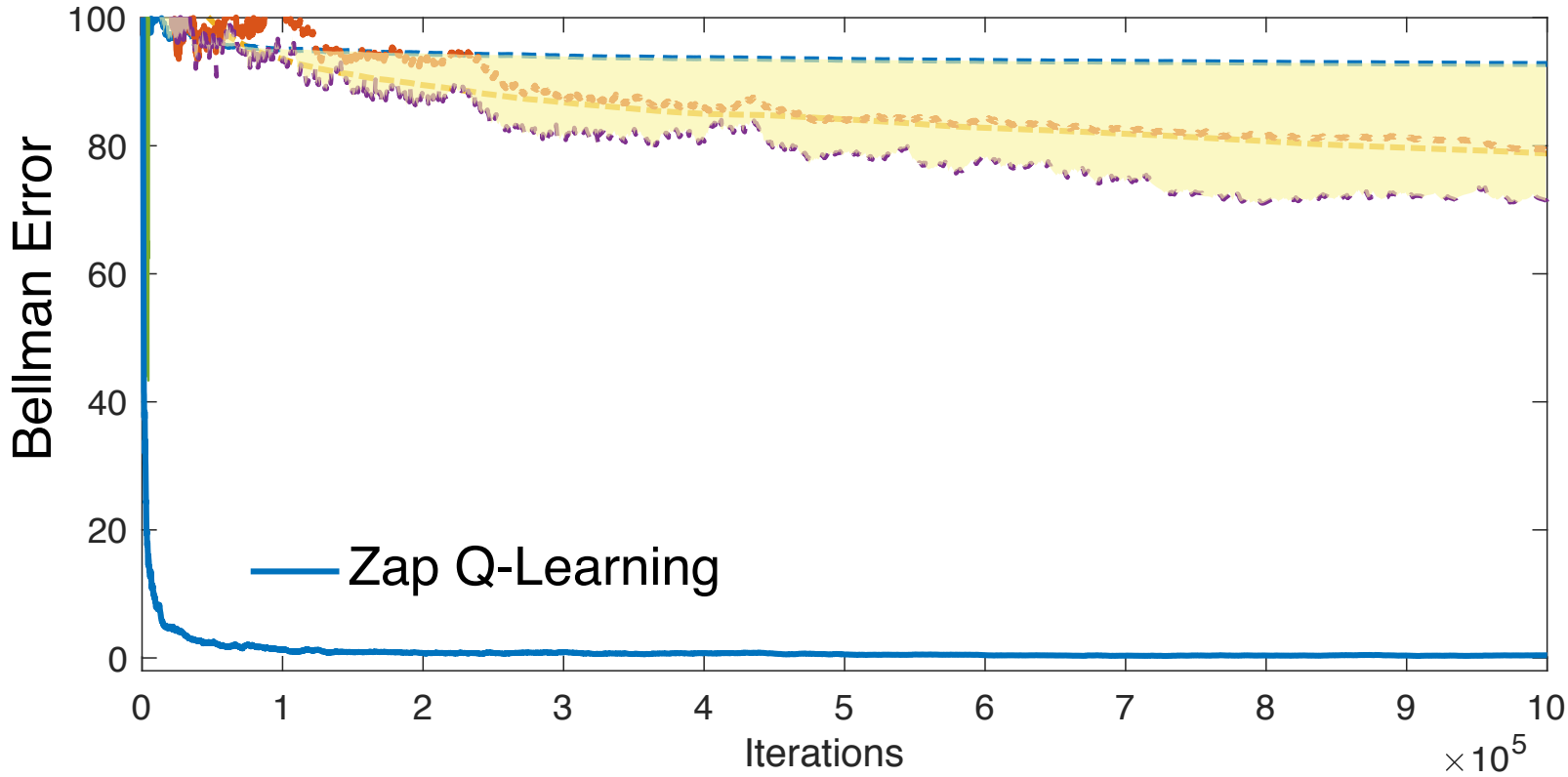
Based on joint research with Ana Bušić and Sean Meyn

Thanks to ARO and the National Science Foundation

Thanks to the Simons Institute RTDM program, Berkeley, Spring 2018

Newton, Polyak, and Nesterov

Outline



Reinforcement Learning and Stochastic Approximation

Stochastic Optimal Control

MDP Model

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

- For all states x and sets A,

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

- $c\colon \mathsf{X} \times \mathsf{U} \to \mathbb{R}$ is a cost function
- $\beta < 1$ a discount factor

Stochastic Optimal Control

MDP Model

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

- For all states x and sets A,

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

- $c\colon \mathsf{X} \times \mathsf{U} \to \mathbb{R}$ is a cost function
- $\beta < 1$ a discount factor

Q-function:

$$Q^*(x,u) = \min_{\boldsymbol{U}} \sum_{n=0}^{\infty} \beta^n \mathsf{E}[c(X_n, U_n) \mid X(0) = x, U(0) = u]$$

Stochastic Optimal Control

MDP Model

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

- For all states x and sets A,

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

- $c\colon \mathsf{X} \times \mathsf{U} \to \mathbb{R}$ is a cost function
- $\beta < 1$ a discount factor

Q-function:

$$Q^*(x,u) = \min_{\boldsymbol{U}} \sum_{n=0}^{\infty} \beta^n \mathsf{E}[c(X_n, U_n) \mid X(0) = x, U(0) = u]$$

Bellman equation:

$$Q^*(x,u) = c(x,u) + \beta \mathsf{E}\big[\min_{u'} Q^*(X_{n+1}, u') \mid X_n = x,\ U_n = u\big]$$

Q-learning and Galerkin Relaxation

Dynamic programming

Find function Q^* that solves

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

Q-learning and Galerkin Relaxation

Dynamic programming

Find function Q^* that solves

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

Q-learning

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

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

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

Q-learning and Stochastic Approximation

Q-learning with linear parameterization:

$Q^\theta(x,u) = \theta^\intercal \psi(x,u) \qquad \theta \in \mathbb{R}^d, \quad \psi : \mathsf{X} \times \mathsf{U} \to \mathbb{R}^d$

Find θ^* such that:

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

Example: $\zeta_n = \nabla_\theta Q^\theta(X_n,U_n) = \psi(X_n,U_n)$

Q-learning and Stochastic Approximation

Q-learning with linear parameterization:

$Q^\theta(x,u) = \theta^\intercal \psi(x,u) \qquad \theta \in \mathbb{R}^d, \quad \psi : \mathsf{X} \times \mathsf{U} \to \mathbb{R}^d$

Find θ^* such that:

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

Example: $\zeta_n = \nabla_\theta Q^\theta(X_n,U_n) = \psi(X_n,U_n)$

Q-learning and SA

Root finding problem: $\bar{f}(\theta^*) = \boldsymbol{A}(\theta^*)\theta^* - \boldsymbol{b} = 0$

Q-learning and Stochastic Approximation

Q-learning with linear parameterization:

$Q^\theta(x,u) = \theta^\intercal \psi(x,u) \qquad \theta \in \mathbb{R}^d, \quad \psi : \mathsf{X} \times \mathsf{U} \to \mathbb{R}^d$

Find θ^* such that:

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

Example: $\zeta_n = \nabla_\theta Q^\theta(X_n,U_n) = \psi(X_n,U_n)$

Q-learning and SA

Root finding problem: $\bar{f}(\theta^*) = \boldsymbol{A}(\theta^*)\theta^* - \boldsymbol{b} = 0$

$$\boldsymbol{A}(\theta) = \mathsf{E}\big[\zeta_n\big[\beta\psi(X_{n+1},\phi^\theta(X_{n+1})) - \psi(X_n,U_n)\big]^\intercal\big]$$
$$\boldsymbol{b} := \mathsf{E}\big[\zeta_n c(X_n,U_n)\big]$$
$$\phi^\theta(x) := \arg\min_u Q^\theta(x,u)$$

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

Stochastic Approximation

What is Stochastic Approximation?

A simple goal: Find the solution θ^* to

$$\bar{f}(\theta^*) := \mathsf{E}[f(\theta, W)]\Big|_{\theta=\theta^*} = 0\,, \qquad \theta \in \mathbb{R}^d\,, \bar{f} : \mathbb{R}^d \to \mathbb{R}^d$$

What is Stochastic Approximation?

A simple goal: Find the solution θ^* to

$$\bar{f}(\theta^*) := \mathsf{E}[f(\theta, W)]\Big|_{\theta=\theta^*} = 0\,, \qquad \theta \in \mathbb{R}^d\,, \bar{f} : \mathbb{R}^d \to \mathbb{R}^d$$

What makes this hard?

What is Stochastic Approximation?

A simple goal: Find the solution θ^* to

$$\bar{f}(\theta^*) := \mathsf{E}[f(\theta, W)]\Big|_{\theta=\theta^*} = 0\,, \qquad \theta \in \mathbb{R}^d\,, \bar{f} : \mathbb{R}^d \to \mathbb{R}^d$$

What makes this hard?

1. The distribution of the random variable W may not be known
2. Computation of the expectation may be expensive: root finding requires multiple evaluations of the expectation for different θ

What is Stochastic Approximation?

A simple goal: Find the solution θ^* to

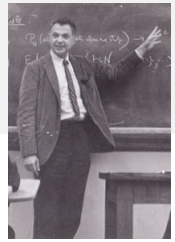
$$\bar{f}(\theta^*) := \mathsf{E}[f(\theta, W)]\Big|_{\theta=\theta^*} = 0\,, \qquad \theta \in \mathbb{R}^d\,, \bar{f} : \mathbb{R}^d \to \mathbb{R}^d$$

What makes this hard?

1. The distribution of the random variable W may not be known
2. Computation of the expectation may be expensive: root finding requires multiple evaluations of the expectation for different θ
3. The recursive algorithms we come up with are often slow, and their variance may be **infinite**: typical in Q-learning [Devraj & M 2017]

Algorithm & Convergence

$\bar{f}(\theta^*) = \mathsf{E}[f(\theta^*, W)] = 0$



SA Algorithm:

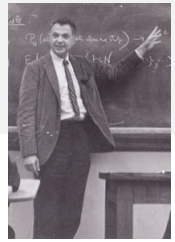
$$\theta_{n+1} = \theta_n + \alpha_{n+1} f(\theta_n, W_{n+1}) \qquad \text{[Robbins \& Monro 1951]}$$

The step-size satisfies:

- $\sum \alpha_n = \infty \qquad \sum \alpha_n^2 < \infty$
- Usually we take $\alpha_n = 1/n$

Algorithm & Convergence

$\bar{f}(\theta^*) = \mathsf{E}[f(\theta^*, W)] = 0$



SA Algorithm:

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

The step-size satisfies:

- $\sum \alpha_n = \infty \qquad \sum \alpha_n^2 < \infty$
- Usually we take $\alpha_n = 1/n$

Analysis: θ^* : *stationary point* of the ODE $\dfrac{d}{dt} x(t) = \bar{f}(x(t))$

SA is a noisy Euler approximation:

$$\theta_{n+1} = \theta_n + \alpha_{n+1}[\bar{f}(\theta_n) + \Delta_{n+1}]$$

Algorithm & Convergence

$\bar{f}(\theta^*) = \mathsf{E}[f(\theta^*, W)] = 0$

SA Algorithm:

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

The step-size satisfies:

- $\sum \alpha_n = \infty \qquad \sum \alpha_n^2 < \infty$
- Usually we take $\alpha_n = 1/n$

Analysis: θ^* : *stationary point* of the ODE $\dfrac{d}{dt} x(t) = \bar{f}(x(t))$

SA is a noisy Euler approximation:

$$\theta_{n+1} = \theta_n + \alpha_{n+1}[\bar{f}(\theta_n) + \Delta_{n+1}]$$

Stability of the ODE $\oplus$ (See Vivek's monograph) $\Longrightarrow$

$$\lim_{n\to\infty} \theta_n = \theta^*$$

Performance Criteria

Error sequence:

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

Performance Criteria

Error sequence:

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

Two standard approaches to evaluate performance,

1. Finite-n bound:

$$\mathsf{P}\{\|\tilde{\theta}_n\| \geq \varepsilon\} \leq ?$$

2. Asymptotic covariance (CLT):

$$\Sigma = \lim_{n\to\infty} n\mathsf{E}\left[\tilde{\theta}_n \tilde{\theta}_n^{\mathsf{T}}\right], \qquad \sqrt{n}\tilde{\theta}_n \approx N(0, \Sigma)$$

Performance Criteria

Error sequence:

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

Two standard approaches to evaluate performance,

1. Finite-n bound:

$$\mathsf{P}\{\|\tilde{\theta}_n\| \geq \varepsilon\} \leq ?$$

2. Asymptotic covariance (CLT):

$$\Sigma = \lim_{n\to\infty} n\mathsf{E}\left[\tilde{\theta}_n \tilde{\theta}_n^{\mathsf{T}}\right], \qquad \sqrt{n}\tilde{\theta}_n \approx N(0, \Sigma)$$

Asymptotic Covariance

$$\Sigma = \lim_{n\to\infty} \Sigma_n = \lim_{n\to\infty} n\mathsf{E}\left[\tilde{\theta}_n \tilde{\theta}_n^T\right]$$

Recall the SA recursion:

$$\theta_{n+1} = \theta_n + \alpha_{n+1}[\bar{f}(\theta_n) + \Delta_{n+1}]$$

Asymptotic Covariance

$$\Sigma = \lim_{n\to\infty} \Sigma_n = \lim_{n\to\infty} n\mathsf{E}\big[\tilde{\theta}_n \tilde{\theta}_n^{\intercal}\big]$$

SA recursion for $\{\Sigma_n\}$:

$$\Sigma_{n+1} \approx \Sigma_n + \tfrac{1}{n}\Big\{(A + \tfrac{1}{2}I)\Sigma_n + \Sigma_n(A + \tfrac{1}{2}I)^{\intercal} + \Sigma_\Delta\Big\}$$

$$A = \frac{d}{d\theta}\bar{f}\,(\theta^*)$$

$$\Sigma_\Delta = \mathsf{E}[\Delta_{n+1}\Delta_{n+1}^{\intercal}]$$

Asymptotic Covariance

$$\Sigma = \lim_{n\to\infty} \Sigma_n = \lim_{n\to\infty} n\mathsf{E}\big[\tilde{\theta}_n \tilde{\theta}_n^{\intercal}\big]$$

SA recursion for $\{\Sigma_n\}$:

$$\Sigma_{n+1} \approx \Sigma_n + \tfrac{1}{n}\Big\{(A+\tfrac{1}{2}I)\Sigma_n + \Sigma_n(A+\tfrac{1}{2}I)^{\intercal} + \Sigma_\Delta\Big\}$$

$$A = \tfrac{d}{d\theta}\bar{f}\,(\theta^*)$$
$$\Sigma_\Delta = \mathsf{E}[\Delta_{n+1}\Delta_{n+1}^{\intercal}]$$

Asymptotic Variance Theory

1. If $\operatorname{Re}\lambda(A) \geq -\frac{1}{2}$ for some eigenvalue then Σ is (typically) infinite
2. If all $\operatorname{Re}\lambda(A) < -\frac{1}{2}$, $\Sigma = \lim_{n\to\infty} \Sigma_n$ solves the Lyapunov equation:

$$\boxed{0 = (A+\tfrac{1}{2}I)\Sigma + \Sigma(A+\tfrac{1}{2}I)^{\intercal} + \Sigma_\Delta}$$

Optimal Asymptotic Covariance

Basis of Ruppert's Stochastic Newton Raphson, and Polyak-Ruppert Averaging

Introduce a $d \times d$ matrix gain sequence $\{G_n\}$:

$$\theta_{n+1} = \theta_n + \alpha_{n+1} G_{n+1} f(\theta_n, W_{n+1})$$

Optimal Asymptotic Covariance

Basis of Ruppert's Stochastic Newton Raphson, and Polyak-Ruppert Averaging

Introduce a $d \times d$ matrix gain sequence $\{G_n\}$:

$$\theta_{n+1} = \theta_n + \alpha_{n+1} G_{n+1} f(\theta_n, W_{n+1})$$

Assume it converges:

$$\lim_{n \to \infty} G_n = G$$

Optimal Asymptotic Covariance

Basis of Ruppert's Stochastic Newton Raphson, and Polyak-Ruppert Averaging

Introduce a $d \times d$ matrix gain sequence $\{G_n\}$:

$$\theta_{n+1} = \theta_n + \alpha_{n+1} G_{n+1} f(\theta_n, W_{n+1})$$

Assume it converges:

$$\lim_{n\to\infty} G_n = G$$

Asymptotic Variance Theory

- If $\operatorname{Re}\lambda(GA) \geq -\frac{1}{2}$ for some eigenvalue then Σ^G is (typically) infinite
- If $\operatorname{Re}\lambda(GA) < -\frac{1}{2}$ for all, Σ^G solves the Lyapunov equation:

$$\boxed{0 = (GA + \tfrac{1}{2}I)\Sigma^G + \Sigma^G(GA + \tfrac{1}{2}I)^\intercal + G\Sigma_\Delta G^\intercal}$$

Optimal Asymptotic Covariance

Basis of Ruppert's Stochastic Newton Raphson, and Polyak-Ruppert Averaging

Introduce a $d \times d$ matrix gain sequence $\{G_n\}$:

$$\theta_{n+1} = \theta_n + \alpha_{n+1} G_{n+1} f(\theta_n, W_{n+1})$$

Assume it converges:

$$\lim_{n\to\infty} G_n = G$$

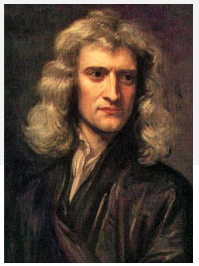
Optimal Matrix Gain: $G^* := -A^{-1}$

- Resembles Newton-Rapshon
- It is optimal: $\Sigma^* = G^* \Sigma_\Delta G^{*\intercal} \le \Sigma^G$ *any other* G

$$0 = (GA + \tfrac{1}{2} I)\Sigma^G + \Sigma^G (GA + \tfrac{1}{2} I)^\intercal + G\Sigma_\Delta G^\intercal$$

Optimal Variance and SNR

$\bar{f}(\theta) = A\theta - b \qquad \frac{\partial}{\partial\theta}(\bar{f}(\theta)) = A$



Stochastic Newton Raphson: Matrix gain algorithm with $G_n \approx G^* = -A^{-1}$:

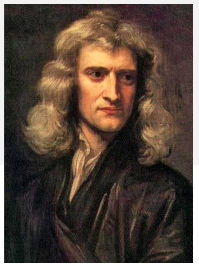
SNR Algorithm:

$$\theta_{n+1} = \theta_n + \alpha_{n+1} G_n f(\theta_n, W_{n+1})$$

$$G_n^{-1} = -\frac{1}{n+1}\sum_{k=1}^{n+1} A_k \qquad A_{n+1} = \frac{d}{d\theta} f(\theta_n, W_{n+1})$$

Optimal Variance and SNR

$\bar{f}(\theta) = A\theta - b \qquad \frac{\partial}{\partial\theta}(\bar{f}(\theta)) = A$



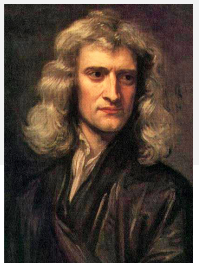
Stochastic Newton Raphson: Matrix gain algorithm with $G_n \approx G^* = -A^{-1}$:

SNR Algorithm:

$$\theta_{n+1} = \theta_n + \alpha_{n+1}(-\widehat{A}_{n+1})^{-1} f(\theta_n, W_{n+1})$$
$$\widehat{A}_{n+1} = \widehat{A}_n + \alpha_{n+1}(A_{n+1} - \widehat{A}_n)$$

Optimal Variance and SNR

$\bar{f}(\theta) = A\theta - b \qquad \frac{\partial}{\partial\theta}(\bar{f}(\theta)) = A$



Stochastic Newton Raphson: Matrix gain algorithm with $G_n \approx G^* = -A^{-1}$:

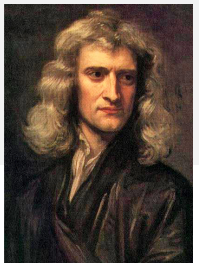
SNR Algorithm:

$$\theta_{n+1} = \theta_n + \alpha_{n+1}(-\widehat{A}_{n+1})^{-1} f(\theta_n, W_{n+1})$$
$$\widehat{A}_{n+1} = \widehat{A}_n + \alpha_{n+1}(A_{n+1} - \widehat{A}_n)$$

Example: LSTD(λ), *but this was* **not** *their motivation!*

Optimal Variance and SNR

$\bar{f}(\theta) = A\theta - b \qquad \frac{\partial}{\partial\theta}(\bar{f}(\theta)) = A$



Stochastic Newton Raphson: Matrix gain algorithm with $G_n \approx G^* = -A^{-1}$:

SNR Algorithm:

$$\theta_{n+1} = \theta_n + \alpha_{n+1}(-\widehat{A}_{n+1})^{-1} f(\theta_n, W_{n+1})$$
$$\widehat{A}_{n+1} = \widehat{A}_n + \alpha_{n+1}(A_{n+1} - \widehat{A}_n)$$

Example: LSTD(λ), *but this was* **not** *their motivation!*

Yes, TD(λ) can have ***infinite*** *asymptotic variance!*

Optimal Variance and Zap-SNR

$A(\theta) = \frac{\partial}{\partial\theta}\bar{f}(\theta)$ is a function of θ

Zap-SNR (designed to emulate deterministic Newton-Raphson)

$$\text{Requires} \quad \widehat{A}_{n+1} \approx A(\theta_n) := \frac{d}{d\theta}\bar{f}\,(\theta_n)$$

Optimal Variance and Zap-SNR

$A(\theta) = \frac{\partial}{\partial\theta}\bar{f}(\theta)$ is a function of θ

Zap-SNR (designed to emulate deterministic Newton-Raphson)

$$\theta_{n+1} = \theta_n + \alpha_{n+1}(-\widehat{A}_{n+1})^{-1} f(\theta_n, W_{n+1})$$

$$\widehat{A}_{n+1} = \widehat{A}_n + \gamma_{n+1}(A_{n+1} - \widehat{A}_n), \qquad A_{n+1} = \frac{d}{d\theta} f(\theta_n, W_{n+1})$$

Optimal Variance and Zap-SNR

$A(\theta) = \frac{\partial}{\partial\theta}\bar{f}(\theta)$ is a function of θ

Zap-SNR (designed to emulate deterministic Newton-Raphson)

$$\theta_{n+1} = \theta_n + \alpha_{n+1}(-\widehat{A}_{n+1})^{-1} f(\theta_n, W_{n+1})$$

$$\widehat{A}_{n+1} = \widehat{A}_n + \gamma_{n+1}(A_{n+1} - \widehat{A}_n), \qquad A_{n+1} = \frac{d}{d\theta} f(\theta_n, W_{n+1})$$

$\widehat{A}_{n+1} \approx A(\theta_n)$ requires high-gain, $\dfrac{\gamma_n}{\alpha_n} \to \infty, \qquad n \to \infty$

Optimal Variance and Zap-SNR

$A(\theta) = \frac{\partial}{\partial\theta}\bar{f}(\theta)$ is a function of θ

Zap-SNR (designed to emulate deterministic Newton-Raphson)

$$\theta_{n+1} = \theta_n + \alpha_{n+1}(-\widehat{A}_{n+1})^{-1} f(\theta_n, W_{n+1})$$

$$\widehat{A}_{n+1} = \widehat{A}_n + \gamma_{n+1}(A_{n+1} - \widehat{A}_n), \qquad A_{n+1} = \frac{d}{d\theta} f(\theta_n, W_{n+1})$$

$\widehat{A}_{n+1} \approx A(\theta_n)$ requires high-gain, $\dfrac{\gamma_n}{\alpha_n} \to \infty, \qquad n \to \infty$

Always: $\alpha_n = 1/n$. Numerics that follow: $\gamma_n = (1/n)^\rho$, $\rho \in (0.5, 1)$

Optimal Variance and Zap-SNR

$A(\theta) = \frac{\partial}{\partial\theta}\bar{f}(\theta)$ is a function of θ

Zap-SNR (designed to emulate deterministic Newton-Raphson)

$$\theta_{n+1} = \theta_n + \alpha_{n+1}(-\widehat{A}_{n+1})^{-1} f(\theta_n, W_{n+1})$$

$$\widehat{A}_{n+1} = \widehat{A}_n + \gamma_{n+1}(A_{n+1} - \widehat{A}_n), \qquad A_{n+1} = \frac{d}{d\theta} f(\theta_n, W_{n+1})$$

$\widehat{A}_{n+1} \approx A(\theta_n)$ requires high-gain, $\dfrac{\gamma_n}{\alpha_n} \to \infty, \qquad n \to \infty$

ODE for Zap-SNR

$$\frac{d}{dt}x_t = -\big[A(x_t)\big]^{-1}\bar{f}\,(x_t), \qquad A(x) = \frac{d}{dx}\bar{f}\,(x)$$

Optimal Variance and Zap-SNR

$A(\theta) = \frac{\partial}{\partial\theta}\bar{f}(\theta)$ is a function of θ

Zap-SNR (designed to emulate deterministic Newton-Raphson)

$$\theta_{n+1} = \theta_n + \alpha_{n+1}(-\widehat{A}_{n+1})^{-1} f(\theta_n, W_{n+1})$$

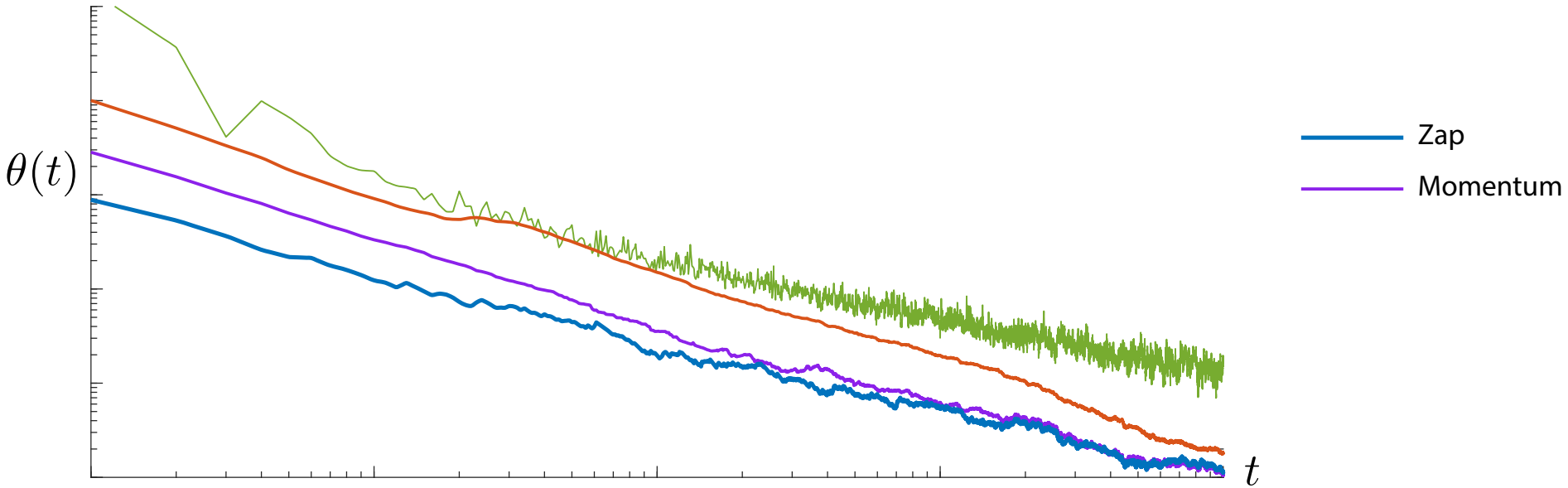
$$\widehat{A}_{n+1} = \widehat{A}_n + \gamma_{n+1}(A_{n+1} - \widehat{A}_n), \qquad A_{n+1} = \frac{d}{d\theta} f(\theta_n, W_{n+1})$$

$\widehat{A}_{n+1} \approx A(\theta_n)$ requires high-gain, $\dfrac{\gamma_n}{\alpha_n} \to \infty, \qquad n \to \infty$

ODE for Zap-SNR

$$\frac{d}{dt}x_t = -\big[A(x_t)\big]^{-1}\bar{f}\,(x_t), \qquad A(x) = \frac{d}{dx}\bar{f}\,(x)$$

Stability? *Virtually universal*



Optimal Momentum Stochastic Approximation

Momentum based Stochastic Approximation

$\Delta\theta_n := \theta_n - \theta_{n-1}$ $\quad$ $f_{n+1}(\theta) := f(\theta, W_{n+1})$

Matrix Gain Stochastic Approximation:

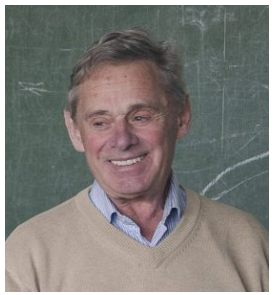
$$\Delta\theta_{n+1} = \alpha_n G_{n+1} f_{n+1}(\theta_n)$$

Momentum based Stochastic Approximation

$\Delta\theta_n := \theta_n - \theta_{n-1}$ $\quad$ $f_{n+1}(\theta) := f(\theta, W_{n+1})$

Matrix Gain Stochastic Approximation:

$$\Delta\theta_{n+1} = \alpha_n G_{n+1} f_{n+1}(\theta_n)$$



Heavy Ball Stochastic Approximation:

following Polyak, 1964 [12]

$$\Delta\theta_{n+1} = m\Delta\theta_n + \alpha_n f_{n+1}(\theta_n)$$

Momentum based Stochastic Approximation

$\Delta\theta_n := \theta_n - \theta_{n-1}$ $\quad$ $f_{n+1}(\theta) := f(\theta, W_{n+1})$

Matrix Gain Stochastic Approximation:

$$\Delta\theta_{n+1} = \alpha_n G_{n+1} f_{n+1}(\theta_n)$$

Heavy Ball Stochastic Approximation:

following Polyak, 1964 [12]

$$\Delta\theta_{n+1} = m\Delta\theta_n + \alpha_n f_{n+1}(\theta_n)$$

Matrix Heavy Ball Stochastic Approximation:

$$\Delta\theta_{n+1} = M_{n+1}\Delta\theta_n + \alpha_n G_{n+1} f_{n+1}(\theta_n)$$

Matrix Heavy Ball Stochastic Approximation

Optimizing $\{M_{n+1}\}$ and $\{G_{n+1}\}$

Matrix Heavy Ball Stochastic Approximation:

$$\Delta\theta_{n+1} = M_{n+1}\Delta\theta_n + \alpha_{n+1}G_{n+1}f_{n+1}(\theta_n)$$

Heuristic: Assume $\Delta\theta_n \to 0$ much faster than $\theta_n \to \theta^*$

Matrix Heavy Ball Stochastic Approximation

Optimizing $\{M_{n+1}\}$ and $\{G_{n+1}\}$

Matrix Heavy Ball Stochastic Approximation:

$$\Delta\theta_{n+1} = M_{n+1}\Delta\theta_n + \alpha_{n+1}G_{n+1}f_{n+1}(\theta_n)$$

Heuristic: Assume $\Delta\theta_n \to 0$ much faster than $\theta_n \to \theta^*$

$$\Delta\theta_{n+1} \approx M_{n+1}\Delta\theta_{n+1} + \alpha_{n+1}G_{n+1}f_{n+1}(\theta_n)$$

Matrix Heavy Ball Stochastic Approximation

Optimizing $\{M_{n+1}\}$ and $\{G_{n+1}\}$

Matrix Heavy Ball Stochastic Approximation:

$$\Delta\theta_{n+1} = M_{n+1}\Delta\theta_n + \alpha_{n+1}G_{n+1}f_{n+1}(\theta_n)$$

Heuristic: Assume $\Delta\theta_n \to 0$ much faster than $\theta_n \to \theta^*$

$$\Delta\theta_{n+1} \approx M_{n+1}\Delta\theta_{n+1} + \alpha_{n+1}G_{n+1}f_{n+1}(\theta_n)$$

$$\Delta\theta_{n+1} \approx [I - M_{n+1}]^{-1}\alpha_{n+1}G_{n+1}f_{n+1}(\theta_n)$$

Matrix Heavy Ball Stochastic Approximation

Optimizing $\{M_{n+1}\}$ and $\{G_{n+1}\}$

Matrix Heavy Ball Stochastic Approximation:

$$\Delta\theta_{n+1} = M_{n+1}\Delta\theta_n + \alpha_{n+1}G_{n+1}f_{n+1}(\theta_n)$$

Heuristic: Assume $\Delta\theta_n \to 0$ much faster than $\theta_n \to \theta^*$

$$\Delta\theta_{n+1} \approx M_{n+1}\Delta\theta_{n+1} + \alpha_{n+1}G_{n+1}f_{n+1}(\theta_n)$$

$$\Delta\theta_{n+1} \approx [I - M_{n+1}]^{-1}\alpha_{n+1}G_{n+1}f_{n+1}(\theta_n)$$

Try this: $M_{n+1} = I + G_{n+1}\widehat{A}_{n+1}$

Matrix Heavy Ball Stochastic Approximation

Optimizing $\{M_{n+1}\}$ and $\{G_{n+1}\}$

Matrix Heavy Ball Stochastic Approximation:

$$\Delta\theta_{n+1} = M_{n+1}\Delta\theta_n + \alpha_{n+1}G_{n+1}f_{n+1}(\theta_n)$$

Heuristic: Assume $\Delta\theta_n \to 0$ much faster than $\theta_n \to \theta^*$

$$\Delta\theta_{n+1} \approx M_{n+1}\Delta\theta_{n+1} + \alpha_{n+1}G_{n+1}f_{n+1}(\theta_n)$$

$$\Delta\theta_{n+1} \approx [I - M_{n+1}]^{-1}\alpha_{n+1}G_{n+1}f_{n+1}(\theta_n)$$

Try this: $M_{n+1} = I + G_{n+1}\widehat{A}_{n+1}$

$$= -\alpha_{n+1}\widehat{A}_{n+1}^{-1}f_{n+1}(\theta_n) \qquad \text{SNR!}$$

Momentum based Stochastic Approximation

$\widehat{A}_{n+1} \approx \frac{d}{d\theta} \bar{f}(\theta_n)$

SNR: $\Delta\theta_{n+1} = -\alpha_{n+1} \widehat{A}_{n+1}^{-1} f_{n+1}(\theta_n)$

Momentum based Stochastic Approximation

$\widehat{A}_{n+1} \approx \frac{d}{d\theta} \bar{f}\,(\theta_n)$

Special case $G_n = \zeta I$

$$\text{SNR:} \quad \Delta\theta_{n+1} = -\alpha_{n+1} \widehat{A}_{n+1}^{-1} f_{n+1}(\theta_n)$$

$$\text{PolSA:} \quad \Delta\theta_{n+1} = [I + \zeta \widehat{A}_{n+1}]\Delta\theta_n + \alpha_{n+1}\zeta f_{n+1}(\theta_n)$$

Momentum based Stochastic Approximation

$\widehat{A}_{n+1} \approx \frac{d}{d\theta} \bar{f}\,(\theta_n)$

Special case $G_n = \zeta I$

SNR: $\Delta\theta_{n+1} = -\alpha_{n+1} \widehat{A}_{n+1}^{-1} f_{n+1}(\theta_n)$

PolSA: $\Delta\theta_{n+1} = [I + \zeta \widehat{A}_{n+1}]\Delta\theta_n + \alpha_{n+1}\zeta f_{n+1}(\theta_n)$

NeSA: $\Delta\theta_{n+1} = \Delta\theta_n + \zeta[f_{n+1}(\theta_n) - f_{n+1}(\theta_{n-1})]$
$\qquad + \alpha_{n+1}\zeta f_{n+1}(\theta_n)$

following Nesterov, 1983 [13]

Momentum based Stochastic Approximation

$\widehat{A}_{n+1} \approx \frac{d}{d\theta} \bar{f}\,(\theta_n)$

Special case $G_n = \zeta I$

$$\text{SNR:} \quad \Delta\theta_{n+1} = -\alpha_{n+1} \widehat{A}_{n+1}^{-1} f_{n+1}(\theta_n)$$

$$\text{PolSA:} \quad \Delta\theta_{n+1} = [I + \zeta \widehat{A}_{n+1}]\Delta\theta_n + \alpha_{n+1}\zeta f_{n+1}(\theta_n)$$

$$\text{(linearized) NeSA:} \quad \Delta\theta_{n+1} = [I + \zeta A_{n+1}]\Delta\theta_n + \alpha_{n+1}\zeta f_{n+1}(\theta_n)$$

Momentum based Stochastic Approximation

$\widehat{A}_{n+1} \approx \frac{d}{d\theta} \bar{f}\,(\theta_n)$

Special case $G_n = \zeta I$

$$\text{SNR:} \quad \Delta\theta_{n+1} = -\alpha_{n+1} \widehat{A}_{n+1}^{-1} f_{n+1}(\theta_n)$$

$$\text{PolSA:} \quad \Delta\theta_{n+1} = [I + \zeta \widehat{A}_{n+1}]\Delta\theta_n + \alpha_{n+1}\zeta f_{n+1}(\theta_n)$$

$$\text{(linearized) NeSA:} \quad \Delta\theta_{n+1} = [I + \zeta A_{n+1}]\Delta\theta_n + \alpha_{n+1}\zeta f_{n+1}(\theta_n)$$

Coupling of SNR and PolSA: $\|\theta_n^{\text{SNR}} - \theta_n^{\text{PolSA}}\| = O(n^{-1})$

Momentum based Stochastic Approximation

$\widehat{A}_{n+1} \approx \frac{d}{d\theta} \bar{f}\,(\theta_n)$

Special case $G_n = \zeta I$

$$\text{SNR:} \quad \Delta\theta_{n+1} = -\alpha_{n+1} \widehat{A}_{n+1}^{-1} f_{n+1}(\theta_n)$$

$$\text{PolSA:} \quad \Delta\theta_{n+1} = [I + \zeta \widehat{A}_{n+1}]\Delta\theta_n + \alpha_{n+1}\zeta f_{n+1}(\theta_n)$$

$$\text{(linearized) NeSA:} \quad \Delta\theta_{n+1} = [I + \zeta A_{n+1}]\Delta\theta_n + \alpha_{n+1}\zeta f_{n+1}(\theta_n)$$

Coupling of SNR and PolSA: $\|\theta_n^{\text{SNR}} - \theta_n^{\text{PolSA}}\| = O(n^{-1})$

- Linear model: $f_{n+1}(\theta) = A_{n+1}\theta - b_{n+1}$ $\qquad \big(\bar{f}(\theta) = A\theta - b\big)$
- $\{\Delta_{n+1}\}$ square-integrable martingale difference sequence
- A Hurwitz *and* eig$(I + \zeta A) \in$ open unit disk

Momentum based Stochastic Approximation

$\widehat{A}_{n+1} \approx \frac{d}{d\theta} \bar{f}\,(\theta_n)$

Special case $G_n = \zeta I$

$$\text{SNR:} \quad \Delta\theta_{n+1} = -\alpha_{n+1} \widehat{A}_{n+1}^{-1} f_{n+1}(\theta_n)$$

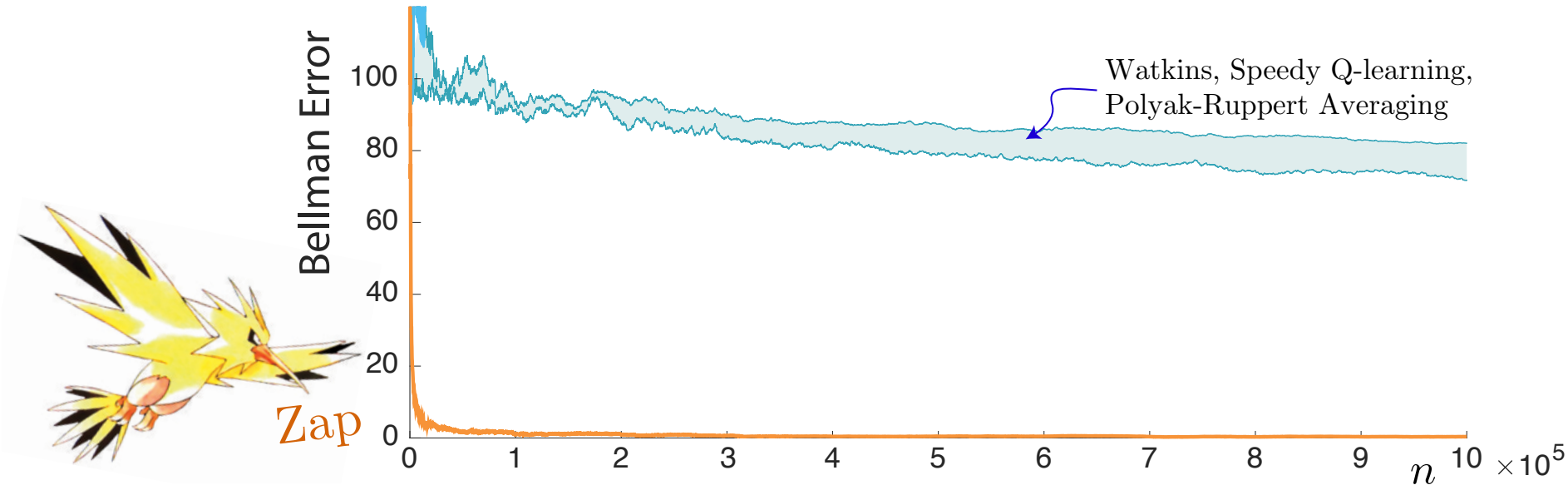
$$\text{PolSA:} \quad \Delta\theta_{n+1} = [I + \zeta \widehat{A}_{n+1}]\Delta\theta_n + \alpha_{n+1}\zeta f_{n+1}(\theta_n)$$

$$\text{(linearized) NeSA:} \quad \Delta\theta_{n+1} = [I + \zeta A_{n+1}]\Delta\theta_n + \alpha_{n+1}\zeta f_{n+1}(\theta_n)$$

Coupling of SNR and PolSA: $\|\theta_n^{\text{SNR}} - \theta_n^{\text{PolSA}}\| = O(n^{-1})$

- Linear model: $f_{n+1}(\theta) = A_{n+1}\theta - b_{n+1}$ $\qquad \big(\bar{f}(\theta) = A\theta - b\big)$
- $\{\Delta_{n+1}\}$ square-integrable martingale difference sequence
- A Hurwitz *and* eig$(I + \zeta A) \in$ open unit disk

PolSA has optimal asymptotic variance



Zap & Momentum in Reinforcement Learning

Watkins' Q-learning

Goal: *Galerkin relaxation of Bellman equation*

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

Watkins' Q-learning

Goal: *Galerkin relaxation of Bellman equation*

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

Watkin's algorithm is Stochastic Approximation

The family $\{Q^\theta\}$ and *eligibility vectors* $\{\zeta_n\}$ in this design:

- Linearly parameterized family of functions: $Q^\theta(x,u) = \theta^\intercal \psi(x,u)$
- $\zeta_n \equiv \psi(X_n, U_n)$
- $\psi_i(x,u) = \mathbb{I}\{x = x^i, u = u^i\}$ (complete basis)

Watkins' Q-learning

Goal: *Galerkin relaxation of Bellman equation*

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

Watkin's algorithm is Stochastic Approximation

The family $\{Q^\theta\}$ and *eligibility vectors* $\{\zeta_n\}$ in this design:

- Linearly parameterized family of functions: $Q^\theta(x,u) = \theta^{\intercal}\psi(x,u)$
- $\zeta_n \equiv \psi(X_n, U_n)$
- $\psi_i(x,u) = \mathbb{I}\{x = x^i, u = u^i\}$ (complete basis)

Algorithm:

$$\theta_{n+1} = \theta_n + \alpha_{n+1}\big(c(X_n, U_n) + \beta \underline{Q}^{\theta^*}(X_{n+1}) - Q^{\theta^*}(X_n, U_n)\big)\zeta_n$$

Watkins' Q-learning

Goal: *Galerkin relaxation of Bellman equation*

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

Watkin's algorithm is Stochastic Approximation

The family $\{Q^\theta\}$ and *eligibility vectors* $\{\zeta_n\}$ in this design:

- Linearly parameterized family of functions: $Q^\theta(x,u) = \theta^\intercal \psi(x,u)$
- $\zeta_n \equiv \psi(X_n, U_n)$
- $\psi_i(x,u) = \mathbb{I}\{x = x^i, u = u^i\}$ (complete basis)

Converges, but has infinite asymptotic variance if $\beta > \frac{1}{2}$:

$$\lambda_{\max}\big(\boldsymbol{A}(\theta^*)\big) > -\tfrac{1}{2}$$

[D & Meyn, 2017]

Watkins' Q-learning

Goal: *Galerkin relaxation of Bellman equation*

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

Watkin's algorithm is Stochastic Approximation

The family $\{Q^\theta\}$ and *eligibility vectors* $\{\zeta_n\}$ in this design:

- Linearly parameterized family of functions: $Q^\theta(x,u) = \theta^\intercal \psi(x,u)$
- $\zeta_n \equiv \psi(X_n, U_n)$
- $\psi_i(x,u) = \mathbb{I}\{x = x^i, u = u^i\}$ (complete basis)

Convergence rate for $\beta > \frac{1}{2}$:

$$\mathcal{O}(1/n^{1-\beta})$$

[D & Meyn, 2017]

Watkins' Q-learning

Can we *Zap* Q-Learning?

Watkin's algorithm is Stochastic Approximation

The family $\{Q^\theta\}$ and *eligibility vectors* $\{\zeta_n\}$ in this design:

- Linearly parameterized family of functions: $Q^\theta(x,u) = \theta^\intercal \psi(x,u)$
- $\zeta_n \equiv \psi(X_n, U_n)$
- $\psi_i(x,u) = \mathbb{I}\{x = x^i, u = u^i\}$ (complete basis)

Convergence rate for $\beta > \frac{1}{2}$:

$$\mathcal{O}(1/n^{1-\beta})$$

[D & Meyn, 2017]

Zap Q-learning

Zap Q-Learning ≡ Zap-SNR for Q-Learning

Algorithm:

$$\theta_{n+1} = \theta_n + \alpha_n \widehat{A}_{n+1}^{-1} \big(c(X_n, U_n) + \beta \underline{Q}^{\theta_n}(X_{n+1}) - Q^{\theta_n}(X_n, U_n) \big) \zeta_n$$

Zap Q-learning

Zap Q-Learning ≡ Zap-SNR for Q-Learning

Algorithm:

$$\theta_{n+1} = \theta_n + \alpha_n \widehat{A}_{n+1}^{-1} \big(c(X_n, U_n) + \beta \underline{Q}^{\theta_n}(X_{n+1}) - Q^{\theta_n}(X_n, U_n) \big) \zeta_n$$

$$\widehat{A}_{n+1} \approx A(\theta_n)$$

Zap Q-learning

Zap Q-Learning ≡ Zap-SNR for Q-Learning

Algorithm:

$$\theta_{n+1} = \theta_n + \alpha_n \widehat{A}_{n+1}^{-1} \big(c(X_n, U_n) + \beta \underline{Q}^{\theta_n}(X_{n+1}) - Q^{\theta_n}(X_n, U_n) \big) \zeta_n$$

ODE Analysis: change of variables $q = \mathcal{Q}^*(\varsigma)$

Functional $\mathcal{Q}^*$ maps cost functions to Q-functions:

$$q(x,u) = \varsigma(x,u) + \beta \sum_{x'} P_u(x,x') \min_{u'} q(x',u')$$

Zap Q-learning

Zap Q-Learning ≡ Zap-SNR for Q-Learning

Algorithm:

$$\theta_{n+1} = \theta_n + \alpha_n \widehat{A}_{n+1}^{-1} \big(c(X_n, U_n) + \beta \underline{Q}^{\theta_n}(X_{n+1}) - Q^{\theta_n}(X_n, U_n) \big) \zeta_n$$

ODE Analysis: change of variables $q = \mathcal{Q}^*(\varsigma)$

Functional $\mathcal{Q}^*$ maps cost functions to Q-functions:

$$q(x,u) = \varsigma(x,u) + \beta \sum_{x'} P_u(x,x') \min_{u'} q(x',u')$$

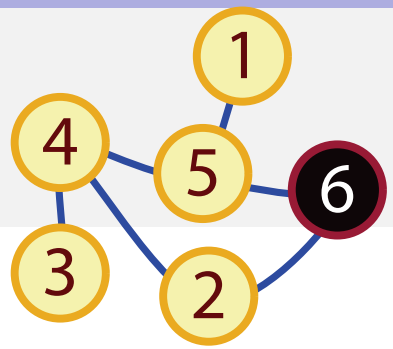
ODE for Zap-Q:

$$q_t = \mathcal{Q}^*(\varsigma_t), \qquad \frac{d}{dt}\varsigma_t = -\varsigma_t + c$$

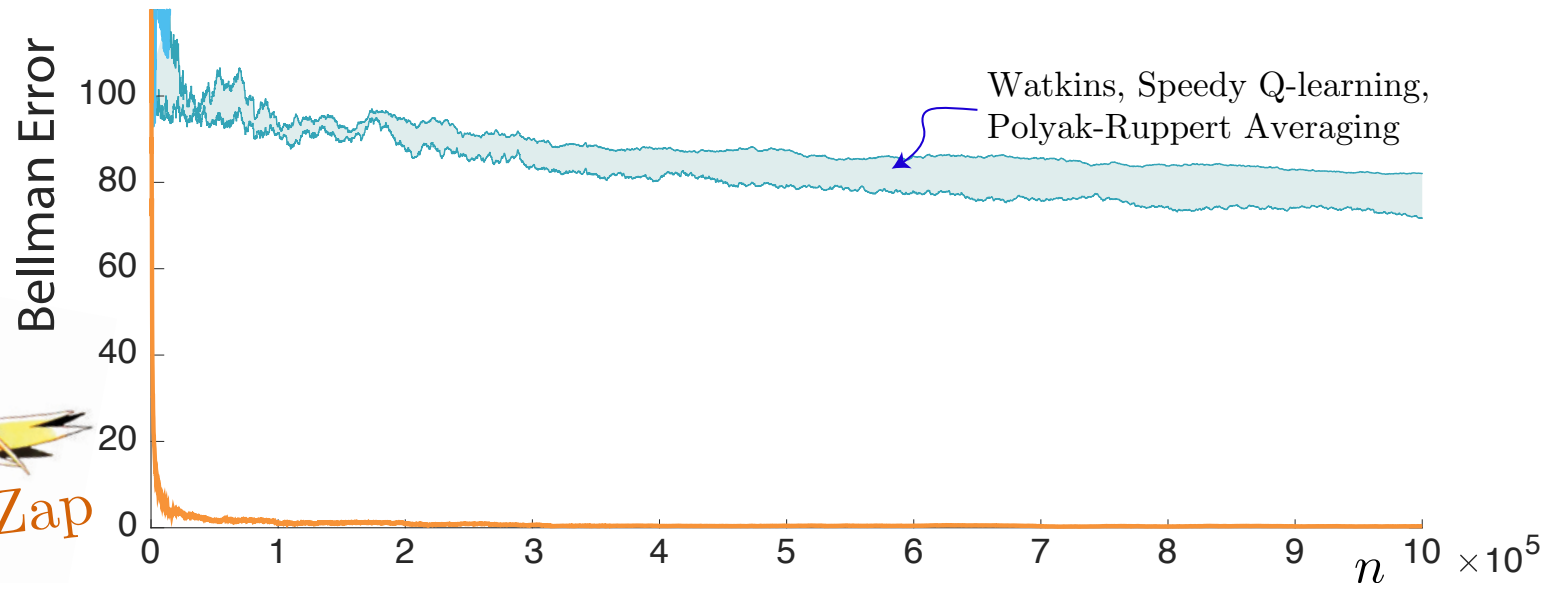
$\Rightarrow$ convergence, optimal covariance, ...

Zap Q-learning

Example: Stochastic Shortest Path



Convergence with Zap gain $\gamma_n = n^{-0.85}$

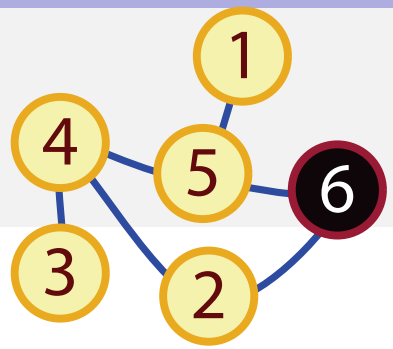


Convergence of Zap-Q Learning

Discount factor: $\beta = 0.99$

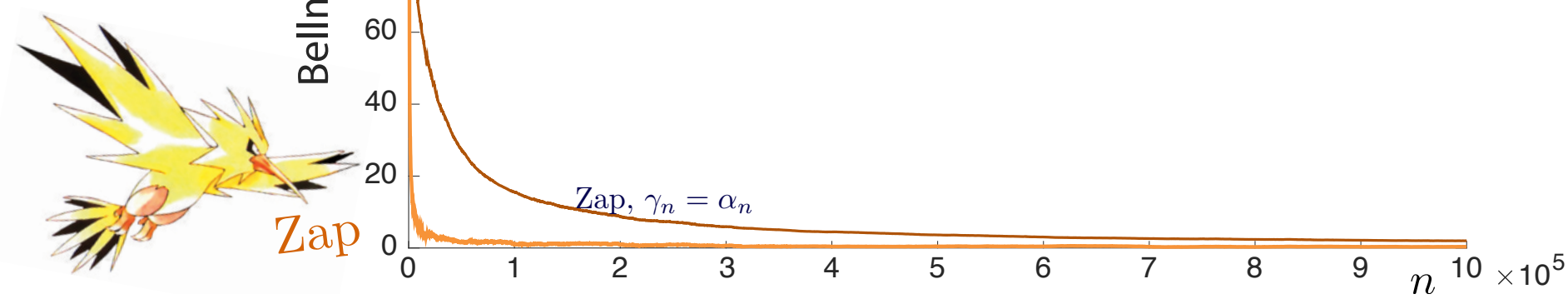
Zap Q-learning

Example: Stochastic Shortest Path



Convergence with Zap gain $\gamma_n = n^{-0.85}$

Watkins' algorithm has infinite asymptotic covariance with $\alpha_n = 1/n$

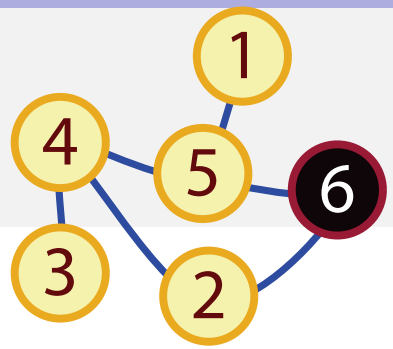


Convergence of Zap-Q Learning

Discount factor: $\beta = 0.99$

Zap Q-learning

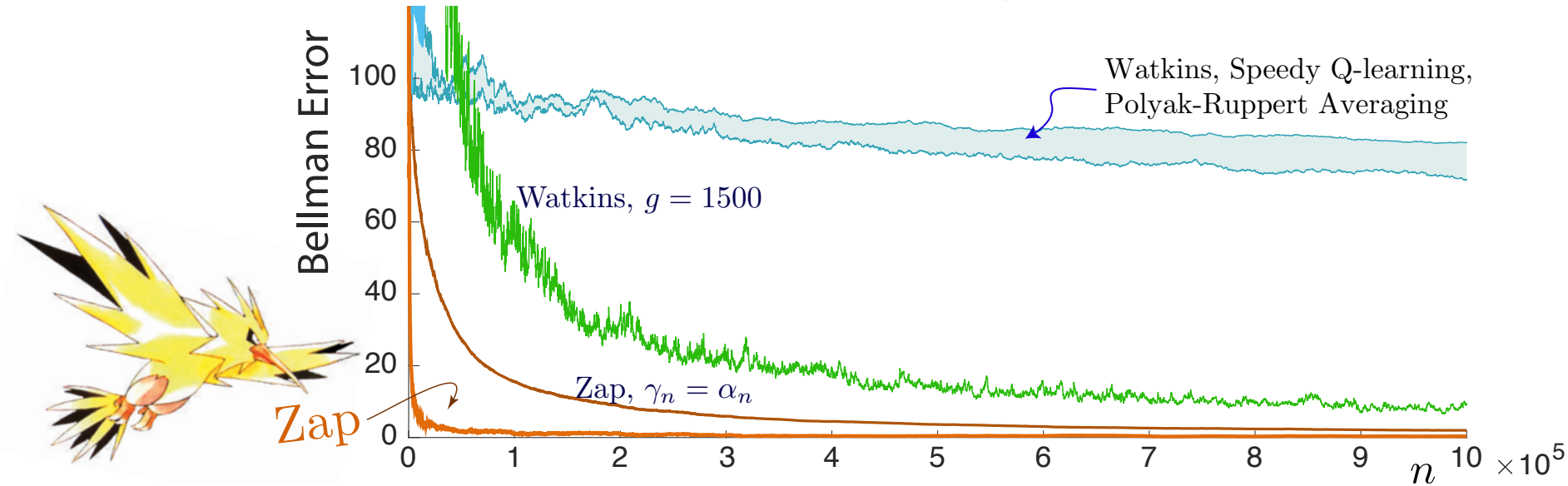
Example: Stochastic Shortest Path



Convergence with Zap gain $\gamma_n = n^{-0.85}$

Watkins' algorithm has infinite asymptotic covariance with $\alpha_n = 1/n$

Optimal scalar gain is approximately $\alpha_n = 1500/n$

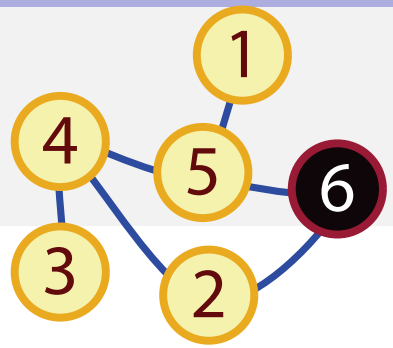


Convergence of Zap-Q Learning

Discount factor: $\beta = 0.99$

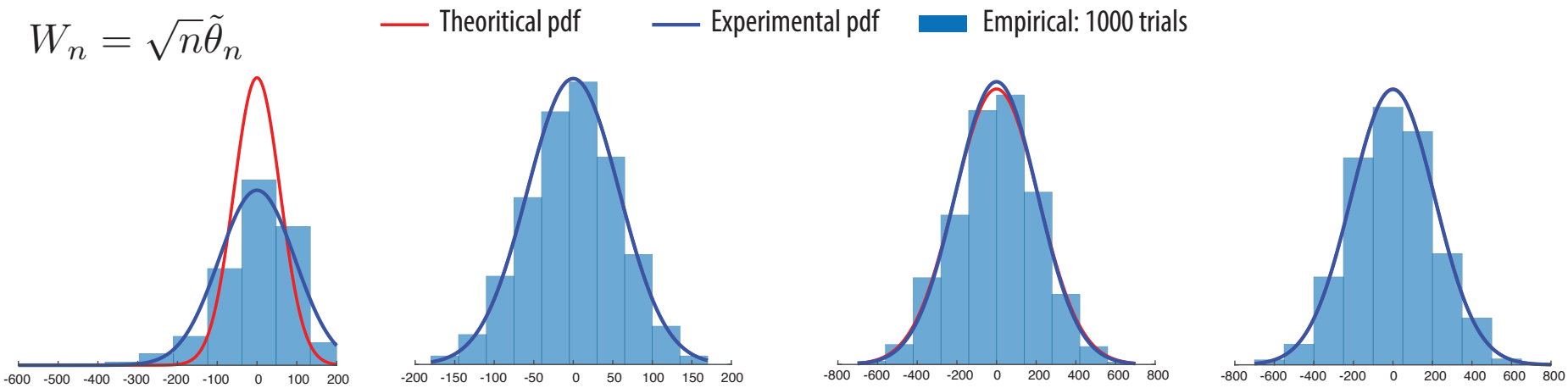
Zap Q-Learning

Optimize Walk to Cafe



Convergence with Zap gain $\gamma_n = n^{-0.85}$

$W_n = \sqrt{n}\tilde{\theta}_n$



Entry #18: $n = 10^4$ $n = 10^6$

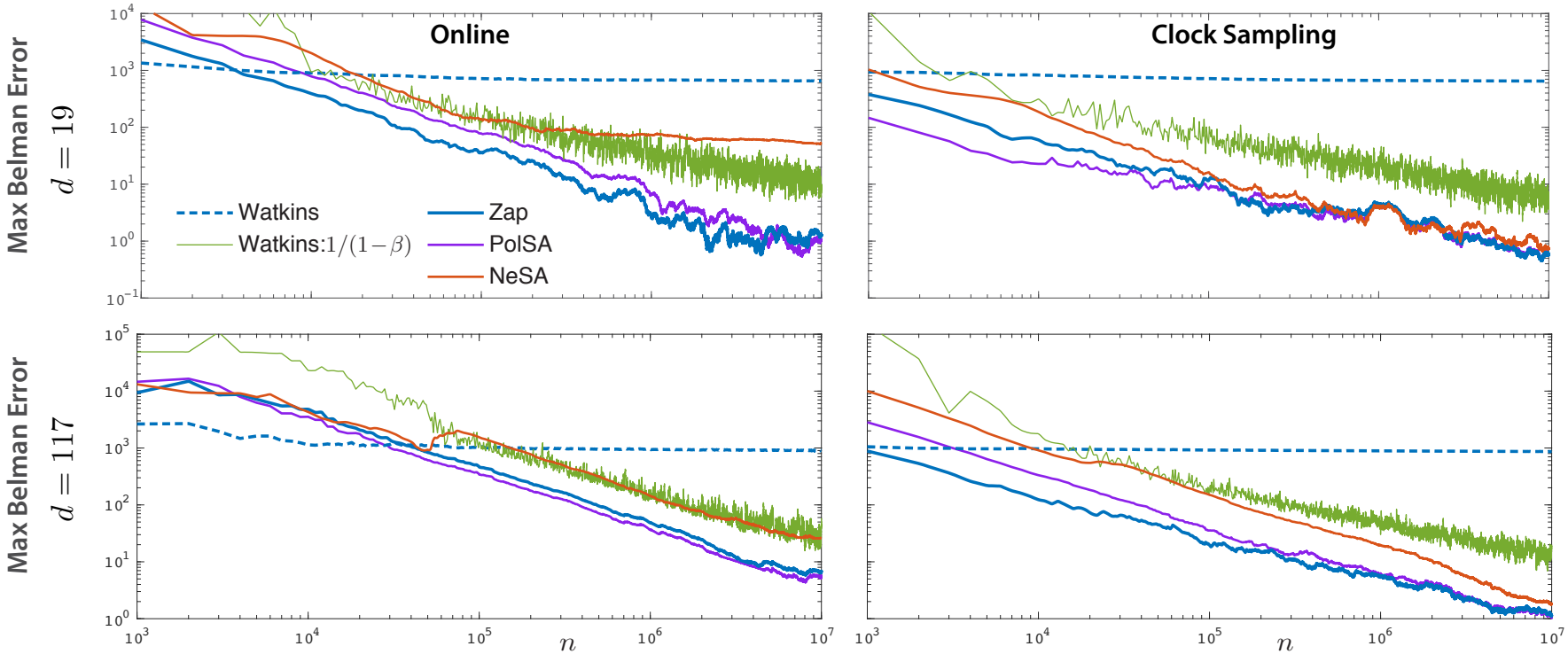
Entry #10: $n = 10^4$ $n = 10^6$

CLT gives good prediction of finite-n performance

Discount factor: $\beta = 0.99$

Zap Q-learning and Momentum

Coupling and convergence for larger models:



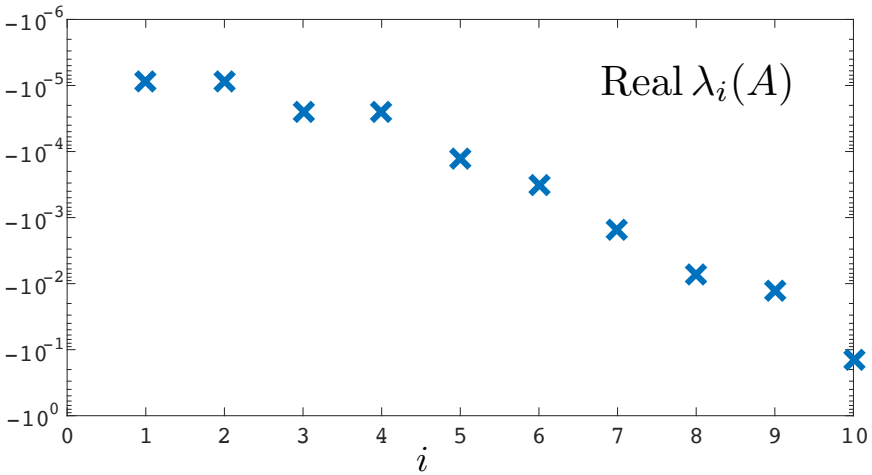
Coupling is amazing

Zap Q-Learning

Model of Tsitsiklis and Van Roy: **Optimal Stopping Time in Finance**

State space: $\mathbb{R}^{100}$

Parameterized Q-function: Q^θ with $\theta \in \mathbb{R}^{10}$



$\mathrm{Real}\,\lambda > -\frac{1}{2}$ for every eigenvalue λ

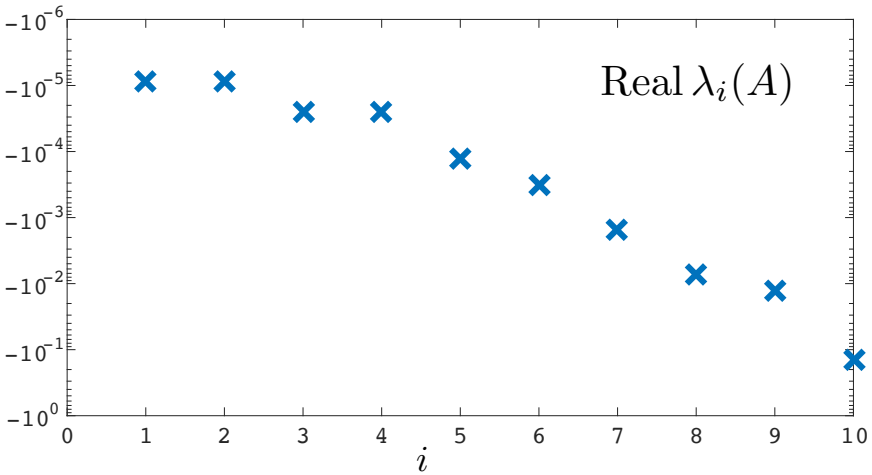
Asymptotic covariance is infinite

Zap Q-Learning

Model of Tsitsiklis and Van Roy: **Optimal Stopping Time in Finance**

State space: $\mathbb{R}^{100}$

Parameterized Q-function: Q^θ with $\theta \in \mathbb{R}^{10}$



$\mathrm{Real}\,\lambda > -\frac{1}{2}$ for every eigenvalue λ

Asymptotic covariance is infinite

Authors observed slow convergence
Proposed a matrix gain sequence

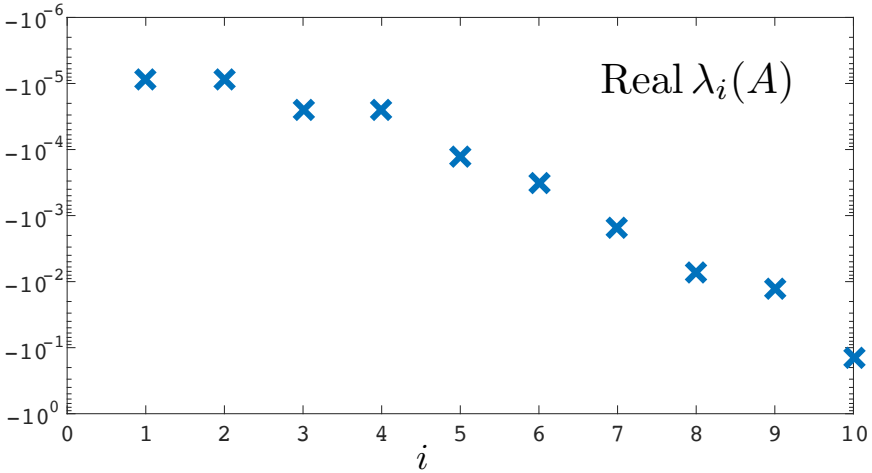
$\{G_n\}$ (see refs for details)

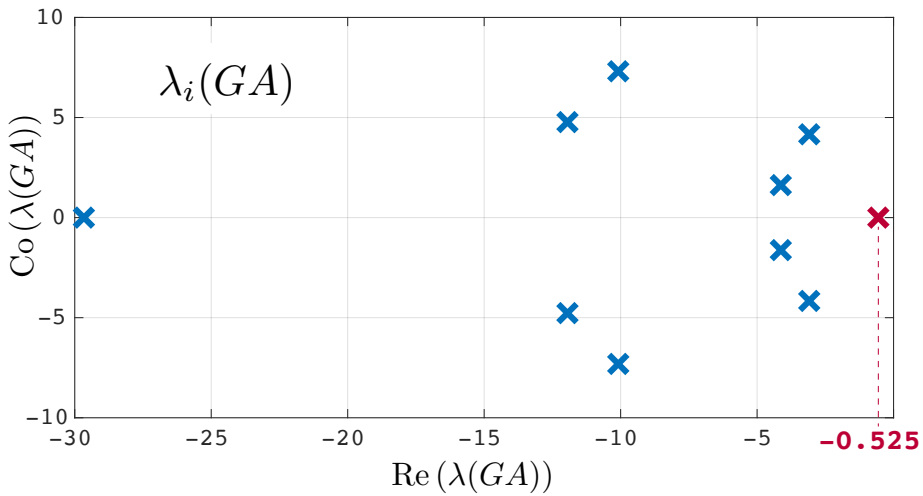
Zap Q-Learning

Model of Tsitsiklis and Van Roy: **Optimal Stopping Time in Finance**

State space: $\mathbb{R}^{100}$

Parameterized Q-function: Q^θ with $\theta \in \mathbb{R}^{10}$





Eigenvalues of A and GA for the finance example

Favorite choice of gain in [25] barely meets the criterion $\text{Re}(\lambda(GA)) < -\frac{1}{2}$

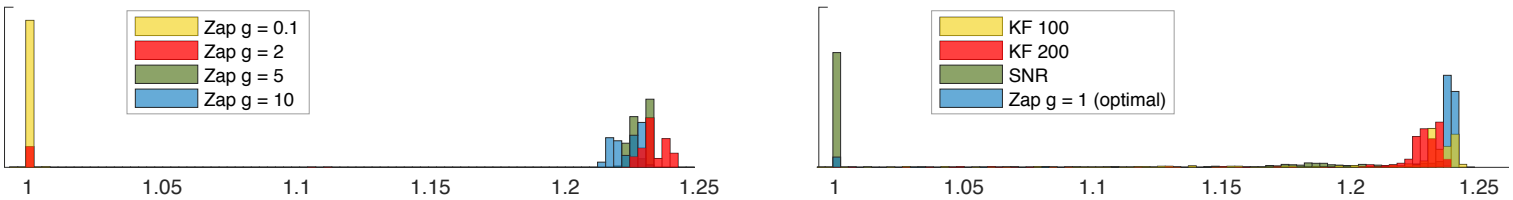
Zap Q-learning

Model of Tsitsiklis and Van Roy: **Optimal Stopping Time in Finance**

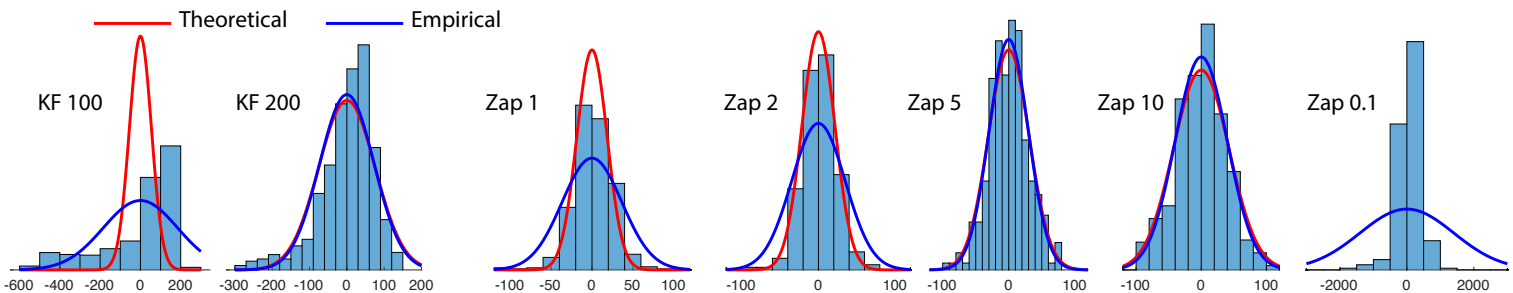
State space: $\mathbb{R}^{100}$

Parameterized Q-function: Q^θ with $\theta \in \mathbb{R}^{10}$

Histograms of the average reward obtained using the different algorithms:



Asymptotic variance for of the algorithms:



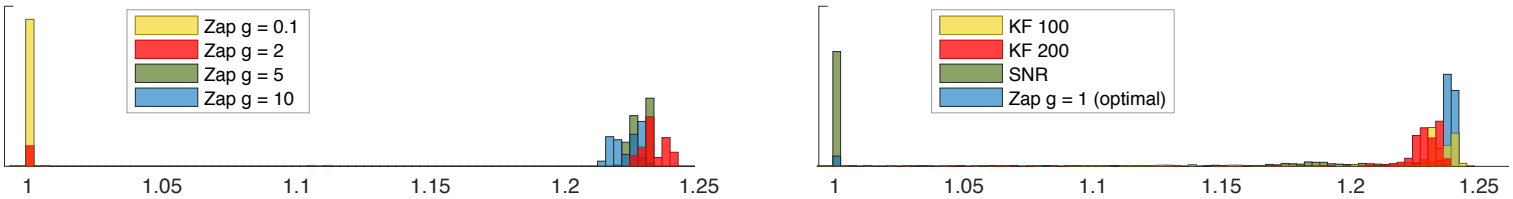
Zap Q-learning

Model of Tsitsiklis and Van Roy: **Optimal Stopping Time in Finance**

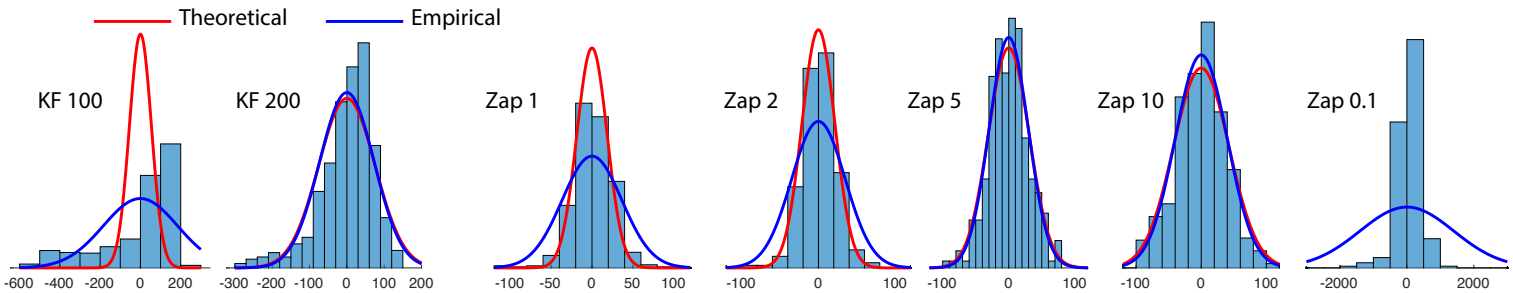
State space: $\mathbb{R}^{100}$

Parameterized Q-function: Q^θ with $\theta \in \mathbb{R}^{10}$

Histograms of the average reward obtained using the different algorithms:



Asymptotic variance for of the algorithms:



Zap-Q $\gg$ KF

Conclusions & Future Work

Conclusions:

- *Reinforcement Learning is not just cursed by dimension, but also by variance!*
- RL algorithms in their raw form are NO GOOD without careful gain selection
- *We need a richer set of tools for algorithm design*

Conclusions & Future Work

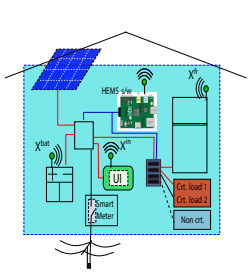
Future work:

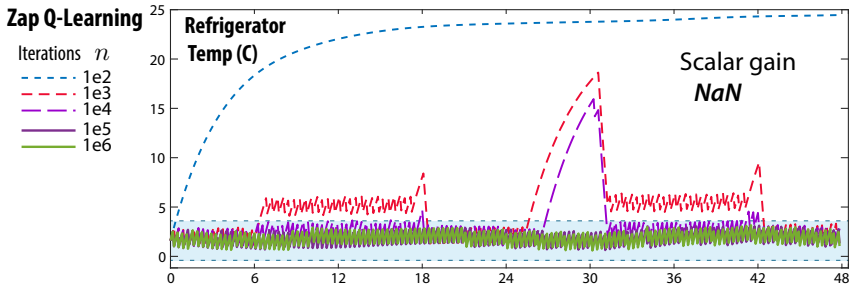
Conclusions & Future Work

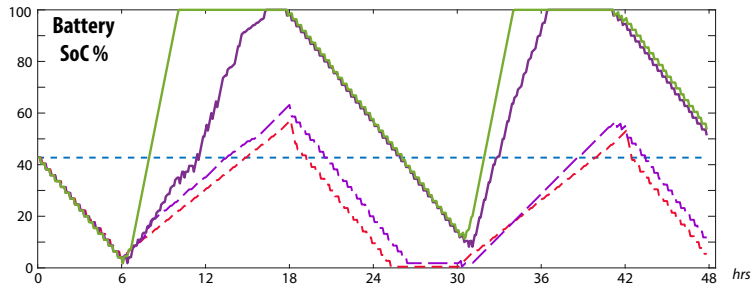
Future work:

- Q-learning with function-approximation
  - *Obtain conditions for a stable algorithm in a general setting*

Conclusions & Future Work





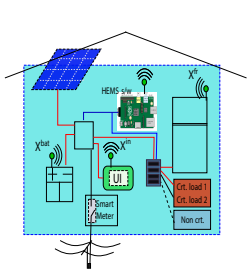


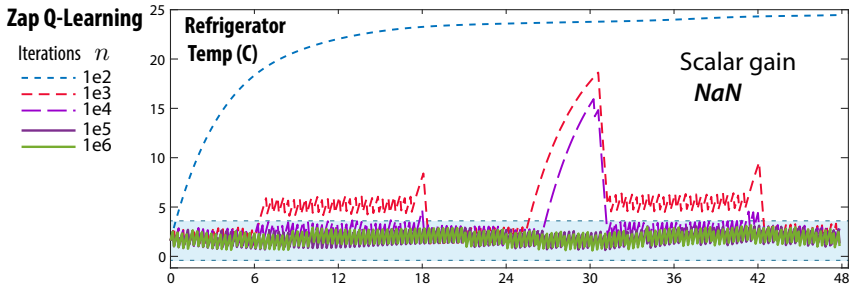
Update in 2019: *stability resolved using second order methods!*

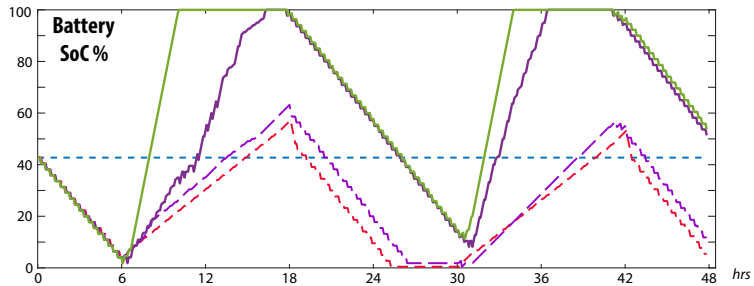
Future work:

- Q-learning with function-approximation
  - *Obtain conditions for a stable algorithm in a general setting*

Conclusions & Future Work





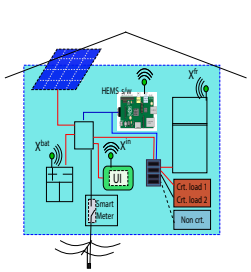


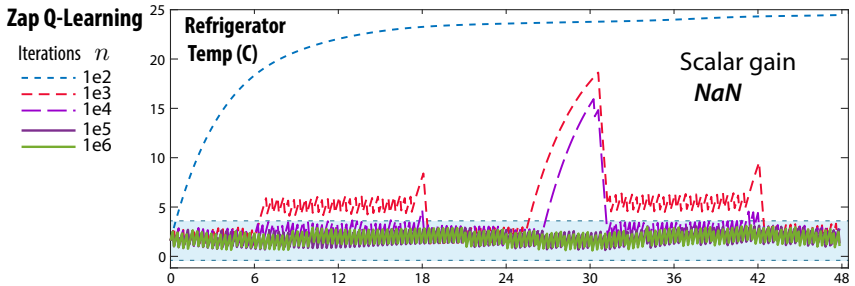
Update in 2019: *stability resolved using second order methods!*

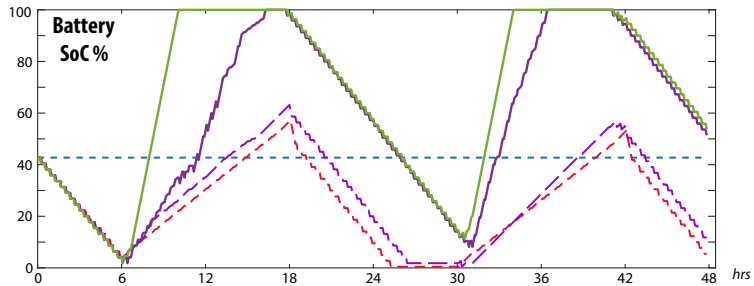
Future work:

- Q-learning with function-approximation
  - *Obtain conditions for a stable algorithm in a general setting*
- Acceleration techniques (momentum and matrix momentum)

Conclusions & Future Work





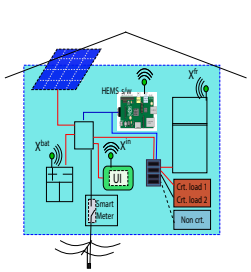


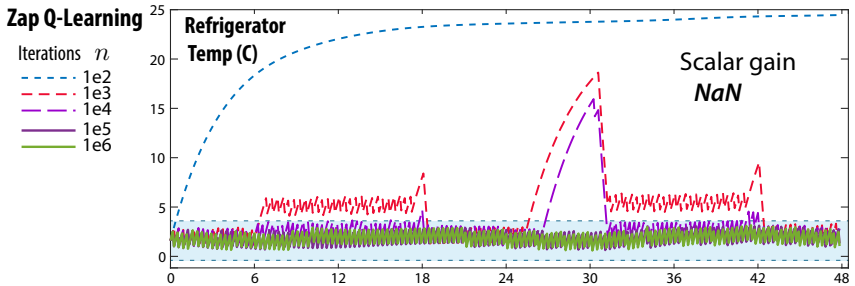
Update in 2019: *stability resolved using second order methods!*

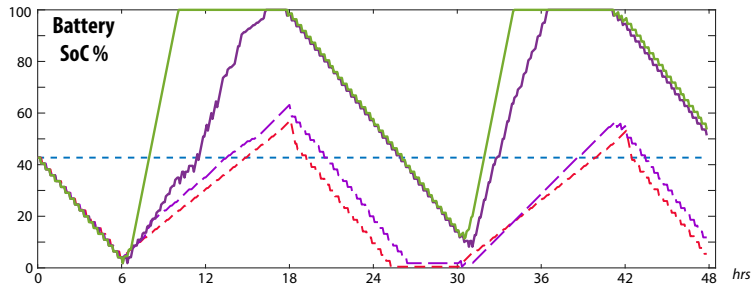
Future work:

- Q-learning with function-approximation
  - *Obtain conditions for a stable algorithm in a general setting*
- Acceleration techniques (momentum and matrix momentum)
- Efficient performance evaluation of a policy – *would revolutionize RL*

Conclusions & Future Work





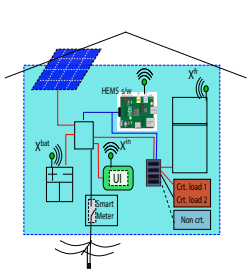


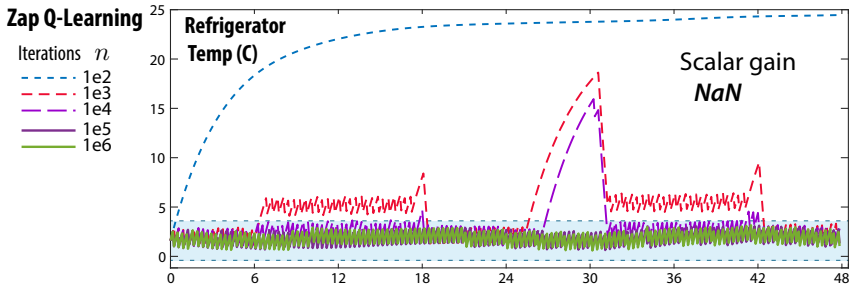
Update in 2019: *stability resolved using second order methods!*

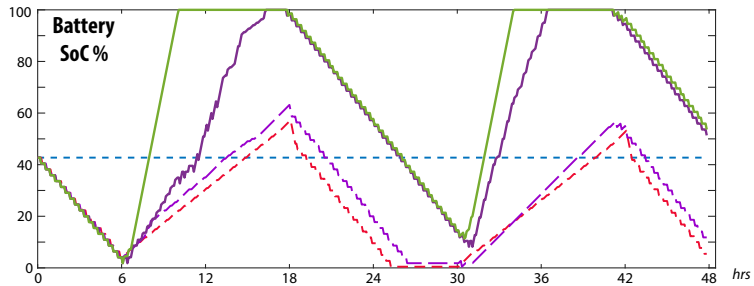
Future work:

- Q-learning with function-approximation
  - *Obtain conditions for a stable algorithm in a general setting*
- Acceleration techniques (momentum and matrix momentum)
- Efficient performance evaluation of a policy – *would revolutionize RL*
- Further variance reduction using control variates

Conclusions & Future Work







Update in 2019: *stability resolved using second order methods!*

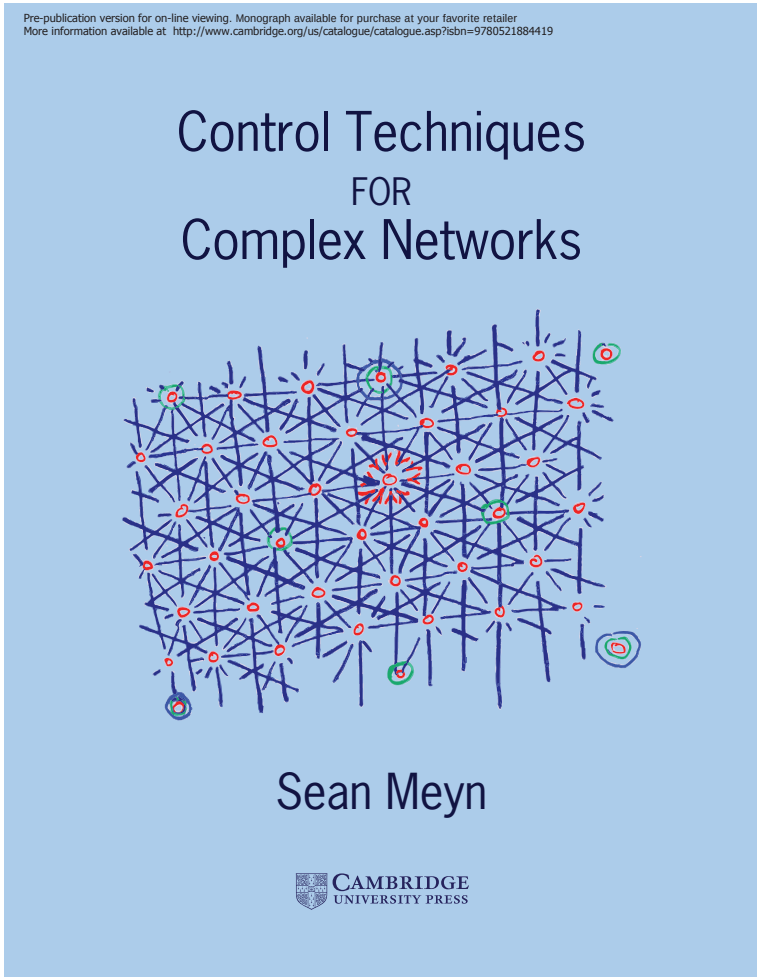
Future work:

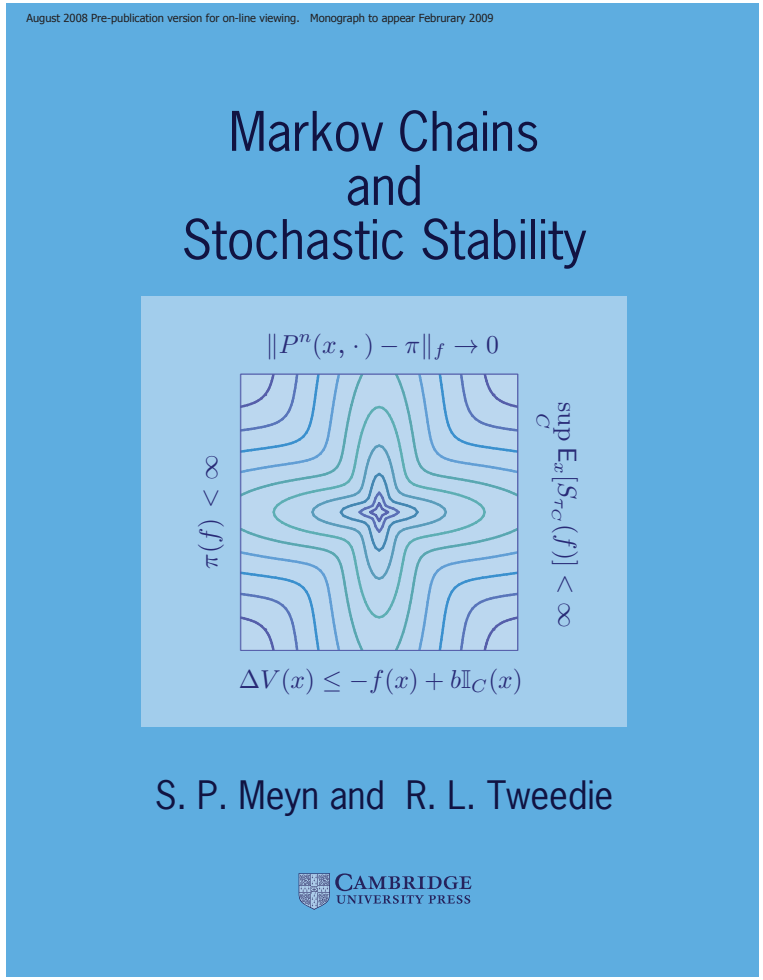
- Q-learning with function-approximation
  - *Obtain conditions for a stable algorithm in a general setting*
- Acceleration techniques (momentum and matrix momentum)
- Efficient performance evaluation of a policy – *would revolutionize RL*
- Further variance reduction using control variates
- Applications in Stochastic Optimization

This Presentation

- A. M. Devraj and S. P. Meyn, *Zap Q-learning. Advances in Neural Information Processing Systems (NIPS).* Dec. 2017.
- A. M. Devraj and S. P. Meyn, *Fastest convergence for Q-learning.* Available on *ArXiv*. Jul. 2017.
- A. M. Devraj, A. Bušić, and S. Meyn. *Optimal Matrix Momentum Stochastic Approximation and Applications to Q-learning*. *ArXiv e-prints*, Feb. 2019.
- S. Chen, A. M. Devraj, A. Bušić, and S. Meyn. *Zap Q-learning for Optimal Stopping Time Problems*. *ArXiv e-prints*, Apr. 2019.


Thank you!


Pre-publication version for on-line viewing. Monograph available for purchase at your favorite retailer
More information available at http://www.cambridge.org/us/catalogue/catalogue.asp?isbn=9780521884419
Control Techniques
FOR
Complex Networks
Sean Meyn
CAMBRIDGE
UNIVERSITY PRESS


August 2008 Pre-publication version for on-line viewing. Monograph to appear Februrary 2009
Markov Chains
and
Stochastic Stability
$\|P^n(x,\cdot) - \pi\|_f \to 0$
$\pi(f) < \infty$
$\sup_C \mathsf{E}_x[S_{\tau_C}(f)] < \infty$
$\Delta V(x) \le -f(x) + b\mathbb{I}_C(x)$
S. P. Meyn and R. L. Tweedie
CAMBRIDGE
UNIVERSITY PRESS

References

Selected References I

[1] A. M. Devraj and S. P. Meyn. *Fastest convergence for Q-learning. ArXiv* , July 2017 (extended version of NIPS 2017).

[2] A. M. Devraj, A. Bušić and S. P. Meyn. *Zap Meets Momentum: Stochastic Approximation Algorithms with Optimal Convergence Rate. ArXiv* , September 2018.

[3] A. Benveniste, M. Métivier, and P. Priouret. *Adaptive algorithms and stochastic approximations*, volume 22 of *Applications of Mathematics (New York)*. Springer-Verlag, Berlin, 1990. Translated from the French by Stephen S. Wilson.

[4] V. S. Borkar. *Stochastic Approximation: A Dynamical Systems Viewpoint*. Hindustan Book Agency and Cambridge University Press (jointly), Delhi, India and Cambridge, UK, 2008.

[5] V. S. Borkar and S. P. Meyn. *The ODE method for convergence of stochastic approximation and reinforcement learning. SIAM J. Control Optim.*, 38(2):447–469, 2000.

[6] S. P. Meyn and R. L. Tweedie. *Markov chains and stochastic stability*. Cambridge University Press, Cambridge, second edition, 2009. Published in the Cambridge Mathematical Library.

[7] S. P. Meyn. *Control Techniques for Complex Networks*. Cambridge University Press, 2007. *See last chapter on simulation and average-cost TD learning*

Selected References II

[8] D. Ruppert. *A Newton-Raphson version of the multivariate Robbins-Monro procedure. The Annals of Statistics*, 13(1):236–245, 1985.

[9] D. Ruppert. *Efficient estimators from a slowly convergent Robbins-Monro processes.* Technical Report Tech. Rept. No. 781, Cornell University, School of Operations Research and Industrial Engineering, Ithaca, NY, 1988.

[10] B. T. Polyak. *A new method of stochastic approximation type. Avtomatika i telemekhanika (in Russian). translated in Automat. Remote Control, 51 (1991)*, pages 98–107, 1990.

[11] B. T. Polyak and A. B. Juditsky. *Acceleration of stochastic approximation by averaging. SIAM J. Control Optim.*, 30(4):838–855, 1992.

[12] B. T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

[13] Y. Nesterov. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. In *Soviet Mathematics Doklady*, 1983.

[14] V. R. Konda and J. N. Tsitsiklis. *Convergence rate of linear two-time-scale stochastic approximation. Ann. Appl. Probab.*, 14(2):796–819, 2004.

Selected References III

[15] E. Moulines and F. R. Bach. *Non-asymptotic analysis of stochastic approximation algorithms for machine learning.* In *Advances in Neural Information Processing Systems 24*, pages 451–459. Curran Associates, Inc., 2011.

[16] C. Szepesvári. *Algorithms for Reinforcement Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2010.

[17] C. J. C. H. Watkins and P. Dayan. *Q-learning. Machine Learning*, 8(3-4):279–292, 1992.

[18] R. S. Sutton.*Learning to predict by the methods of temporal differences. Mach. Learn.*, 3(1):9–44, 1988.

[19] J. N. Tsitsiklis and B. Van Roy. *An analysis of temporal-difference learning with function approximation. IEEE Trans. Automat. Control*, 42(5):674–690, 1997.

[20] C. Szepesvári. *The asymptotic convergence-rate of Q-learning.* In *Proceedings of the 10th Internat. Conf. on Neural Info. Proc. Systems*, pages 1064–1070. MIT Press, 1997.

[21] M. G. Azar, R. Munos, M. Ghavamzadeh, and H. Kappen. *Speedy Q-learning.* In *Advances in Neural Information Processing Systems*, 2011.

[22] E. Even-Dar and Y. Mansour. *Learning rates for Q-learning. Journal of Machine Learning Research*, 5(Dec):1–25, 2003.

Selected References IV

[23] D. Huang, W. Chen, P. Mehta, S. Meyn, and A. Surana. *Feature selection for neuro-dynamic programming.* In F. Lewis, editor, *Reinforcement Learning and Approximate Dynamic Programming for Feedback Control*. Wiley, 2011.

[24] J. N. Tsitsiklis and B. Van Roy. *Optimal stopping of Markov processes: Hilbert space theory, approximation algorithms, and an application to pricing high-dimensional financial derivatives. IEEE Trans. Automat. Control*, 44(10):1840–1851, 1999.

[25] D. Choi and B. Van Roy. *A generalized Kalman filter for fixed point approximation and efficient temporal-difference learning. Discrete Event Dynamic Systems: Theory and Applications*, 16(2):207–239, 2006.

[26] S. J. Bradtke and A. G. Barto. *Linear least-squares algorithms for temporal difference learning. Mach. Learn.*, 22(1-3):33–57, 1996.

[27] J. A. Boyan. *Technical update: Least-squares temporal difference learning. Mach. Learn.*, 49(2-3):233–246, 2002.

[28] A. Nedic and D. Bertsekas. *Least squares policy evaluation algorithms with linear function approximation. Discrete Event Dyn. Systems: Theory and Appl.*, 13(1-2):79–110, 2003.

[29] P. G. Mehta and S. P. Meyn. *Q-learning and Pontryagin's minimum principle.* In *IEEE Conference on Decision and Control*, pages 3598–3605, Dec. 2009.