Reinforcement Learning Design with Optimal Learning (Convergence) Rate

PhD Thesis Defense: 10/29/2019

Adithya M. Devraj

Department of Electrical and Computer Engineering  University of Florida

Committee Chair: Sean Meyn

Committee Members: José Principe, Meera Sitharam, Murali Rao

Thanks to ARO and the National Science Foundation

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

Thanks to my advisor, commitee members, colloborators, teachers, mentors, lab-mates, friends, and family

Future Work (from PhD Proposal)

- Q-learning with function-approximation
  - *Obtain conditions for a stable algorithm in a general setting*
  - *Optimal stopping time problems*
- Computationally efficient Zap-Q

  Fast convergence with low computational complexity
- Finite-time analysis
  - *Large deviations principle*
  - *Law of iterated logarithm*
  - *?*

List of Publications

Zap Q-learning & Matrix Momentum Stochastic Approximation:

- A. M. Devraj and S. P. Meyn, **"Zap Q-learning."** *Advances in Neural Information Processing Systems* **(NeurIPS)**, *Dec 2017.*
- A. M. Devraj, A. Bŭsić, and S. P. Meyn, **"Optimal Matrix Momentum Stochastic Approximation and Applications to Q-learning."** *(Invited Paper) 57th Annual Allerton Conference. Sep. 2019.*
- A. M. Devraj, A. Bŭsić, and S. P. Meyn, **"Zap Q-Learning: A Users Guide."** *Indian Control Conference* **(ICC)**. *Jan. 2019 (Invited Paper – Best Student Paper Award)*

TD-learning in a Sobolev Space:

- A. M. Devraj and S. P. Meyn, **"Differential TD Learning for Value Function Approximation."** *IEEE Conference on Decision and Control* **(CDC)**, *Dec 2016.*
- A. M. Devraj, I. Kontoyiannis, and S. P. Meyn, **"Differential Temporal Difference Learning."** *Submitted to IEEE Transactions on Automatic Control* **(TAC)**, *Dec 2018.*
- A. M. Devraj, I. Kontoyiannis, and S. P. Meyn, **"Geometric Ergodicity in a Weighted Sobolev Space."** *To appear in the Annals of Probability* **(AoP)**, *Apr 2019.*

List of Publications

Zap Q-learning & Matrix Momentum Stochastic Approximation:

- A. M. Devraj and S. P. Meyn, **"Zap Q-learning."** *Advances in Neural Information Processing Systems* **(NeurIPS)**, *Dec 2017.*
- A. M. Devraj, A. Bŭsić, and S. P. Meyn, **"Optimal Matrix Momentum Stochastic Approximation and Applications to Q-learning."** *(Invited Paper) 57th Annual Allerton Conference. Sep. 2019.*
- A. M. Devraj, A. Bŭsić, and S. P. Meyn, **"Zap Q-Learning: A Users Guide."** *Indian Control Conference* **(ICC)**. *Jan. 2019 (Invited Paper – Best Student Paper Award)*

Works at Internships

- A. M. Devraj, and J. Chen, **"Stochastic Variance Reduced Primal Dual Algorithms for Empirical Composition Optimization."** *Advances in Neural Information Processing Systems* **(NeurIPS)**, *Dec 2019.*
- Y. Chen, A. Bernstein, A. M. Devraj, and S. P. Meyn, **"Model-Free Primal-Dual Methods for Network Optimization with Application to Real-Time Optimal Power Flow."** *submitted to IEEE American Control Conference* **(ACC)**, *2019*
- A. M. Devraj, M. Tomczak, S. V. Macua, E. M. de Cote, and S. P. Meyn, **"Zap Actor-Critic Algorithms."** *working paper, 2019*

List of Publications

Zap Q-learning & Matrix Momentum Stochastic Approximation:

- A. M. Devraj and S. P. Meyn, **"Zap Q-learning."** *Advances in Neural Information Processing Systems* **(NeurIPS)**, *Dec 2017.*
- A. M. Devraj, A. Bŭsić, and S. P. Meyn, **"Optimal Matrix Momentum Stochastic Approximation and Applications to Q-learning."** *(Invited Paper) 57th Annual Allerton Conference. Sep. 2019.*
- A. M. Devraj, A. Bŭsić, and S. P. Meyn, **"Zap Q-Learning: A Users Guide."** *Indian Control Conference* **(ICC)**. *Jan. 2019 (Invited Paper – Best Student Paper Award)*

Extensions and Applications of Zap

- S. Chen, A. M. Devraj, A. Bŭsić, and S. P. Meyn, **"Zap Q-Learning for Optimal Stopping."** *submitted to IEEE American Control Conference* **(ACC)**, *2019*
- S. Chen, A. M. Devraj, A. Bŭsić, and S. P. Meyn, **"Zap Q-Learning with Nonlinear Function Approximation."** *arXiv, 2019*
- N. S. Raman, A. M. Devraj, P. Barooah, and S. P. Meyn, **"Reinforcement Learning for Control of Building HVAC Systems."** *submitted to IEEE American Control Conference* **(ACC)**, *2019*

Newton, Polyak, Ruppert, and Nesterov

Outline

Detailed Comments:	1. In general, it is not clear how techniques from stochastic approximation can add to the literature of reinforcement learning. SA is concerned with stability and asymptotics, while RL is concerned with the efficiency of learning (sample complexity and regret). In general,I cannot convince myself why ppl care about the stability/asymptotic of SA in the context of RL. I think more justification is needed to bring together the theory of SA and RL.

Q-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 goal:

Find function Q^* that solves

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

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

Q-learning and Galerkin Relaxation

Dynamic programming goal:

Find function Q^* that solves

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

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

Q-learning goal:

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

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

Q-learning and Galerkin Relaxation

Dynamic programming goal:

Find function Q^* that solves

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

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

Q-learning goal:

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

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

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

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

Q-learning and Galerkin Relaxation

This is a Stochastic Approximation problem!

Q-learning goal:

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

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

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

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

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

Stochastic Approximation

What is Stochastic Approximation?

A simple goal: Find the solution θ^* to

$$\bar{f}(\theta^*) := \mathsf{E}[f(\theta, W_{n+1})]\Big|_{\theta=\theta^*} = 0\,, \qquad \theta \in \mathbb{R}^d\,, \bar{f} : \mathbb{R}^d \to \mathbb{R}^d$$

What is Stochastic Approximation?

A simple goal: Find the solution θ^* to

$$\bar{f}(\theta^*) := \mathsf{E}[f(\theta, W_{n+1})]\Big|_{\theta=\theta^*} = 0\,, \qquad \theta \in \mathbb{R}^d\,, \bar{f} : \mathbb{R}^d \to \mathbb{R}^d$$

In Q-learning,

$$\begin{aligned} W_{n+1} &\equiv \big(X_n, U_n, X_{n+1}, \zeta_n\big) \\ f(\theta, W_{n+1}) &\equiv \big[c(X_n, U_n) + \beta \underline{Q}^\theta(X_{n+1}) - Q^\theta(X_n, U_n)\big]\zeta_n \end{aligned}$$

What is Stochastic Approximation?

A simple goal: Find the solution θ^* to

$$\bar{f}(\theta^*) := \mathsf{E}[f(\theta, W_{n+1})]\Big|_{\theta=\theta^*} = 0\,, \qquad \theta \in \mathbb{R}^d\,, \bar{f} : \mathbb{R}^d \to \mathbb{R}^d$$

What makes this hard?

What is Stochastic Approximation?

A simple goal: Find the solution θ^* to

$$\bar{f}(\theta^*) := \mathsf{E}[f(\theta, W_{n+1})]\Big|_{\theta=\theta^*} = 0\,, \qquad \theta \in \mathbb{R}^d\,, \bar{f} : \mathbb{R}^d \to \mathbb{R}^d$$

What makes this hard?

Original Motivation:

1. The distribution of the random variable W_{n+1} 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

$$\bar{f}(\theta^*) := \mathsf{E}[f(\theta, W_{n+1})]\Big|_{\theta=\theta^*} = 0\,, \qquad \theta \in \mathbb{R}^d\,, \bar{f} : \mathbb{R}^d \to \mathbb{R}^d$$

What makes this hard?

Original Motivation:

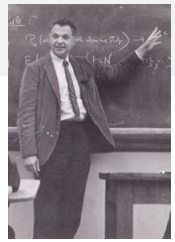
1. The distribution of the random variable W_{n+1} may not be known
2. Computation of the expectation may be expensive: root finding requires multiple evaluations of the expectation for different θ

New:

3. The recursive algorithms we come up with are often slow, and their variance may be **infinite**: typical in Q-learning [D & Meyn 2017]

Algorithm & Convergence

$\bar{f}(\theta^*) = \mathsf{E}[f(\theta^*, W_{n+1})] = 0$



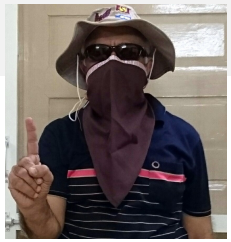
SA Algorithm:

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

We take $\alpha_n = 1/n$

Algorithm & Convergence

$\bar{f}(\theta^*) = \mathsf{E}[f(\theta^*, W_{n+1})] = 0$



SA Algorithm:

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

We take $\alpha_n = 1/n$

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

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

SA is a noisy Euler discretization:

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

Stability of the ODE $\oplus$ (Borkar'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 $\{\Sigma_n\}$:

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

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

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

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 $\{\Sigma_n\}$:

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

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

Asymptotic Variance Theory

1. If all $\operatorname{Re}\lambda(A) < -\frac{1}{2}$, $\Sigma = \lim_{n\to\infty} \Sigma_n$ solves the Lyapunov equation:

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

2. If $\operatorname{Re}\lambda(A) \geq -\frac{1}{2}$ for some eigenvalue then Σ is (typically) infinite

Optimal Asymptotic Covariance

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

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

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 all $\operatorname{Re}\lambda(GA) < -\frac{1}{2}$, Σ^G solves the Lyapunov equation:

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

- If $\operatorname{Re}\lambda(GA) \geq -\frac{1}{2}$ for some eigenvalue then Σ^G is (typically) infinite

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

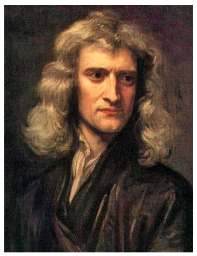
Case 1: $\bar{f}(\theta) = A\theta - b$ $\quad \frac{\partial}{\partial\theta}(\bar{f}(\theta)) = A$ (Example: TD-learning)

Optimal Variance and SNR

Case 1: $\bar{f}(\theta) = A\theta - b$ $\quad \frac{\partial}{\partial\theta}(\bar{f}(\theta)) = A$ (Example: TD-learning)

Stochastic Newton Raphson:

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



SNR Algorithm:

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

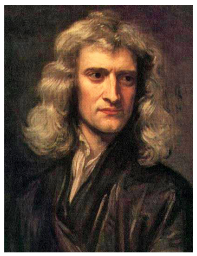
$$G_n^{-1} = -\frac{1}{n+1}\sum_{k=1}^{n+1} A_k \qquad A_{n+1} = \frac{d}{d\theta} f(\theta_n, W_{n+1})$$

Optimal Variance and SNR

Case 1: $\bar{f}(\theta) = A\theta - b$ $\quad \frac{\partial}{\partial\theta}(\bar{f}(\theta)) = A$ (Example: TD-learning)

Stochastic Newton Raphson:

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



SNR Algorithm:

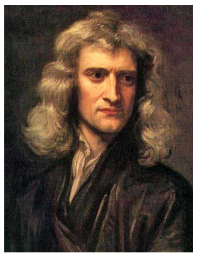
$$\theta_{n+1} = \theta_n + \alpha_{n+1}(-\widehat{A}_{n+1})^{-1} f(\theta_n, W_{n+1})$$
$$\widehat{A}_{n+1} = \widehat{A}_n + \alpha_{n+1}(A_{n+1} - \widehat{A}_n)$$

Optimal Variance and SNR

Case 1: $\bar{f}(\theta) = A\theta - b$ $\quad \frac{\partial}{\partial\theta}(\bar{f}(\theta)) = A$ (Example: TD-learning)

Stochastic Newton Raphson:

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



SNR Algorithm:

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

ODE for linear SA:

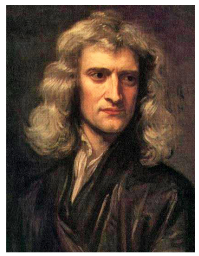
$$\frac{d}{dt}x_t = (-\mathcal{A}_t^{-1})\big[Ax_t - b\big] \qquad \frac{d}{dt}\mathcal{A}_t = -\mathcal{A}_t + A$$

Optimal Variance and SNR

Case 1: $\bar{f}(\theta) = A\theta - b$ $\quad \frac{\partial}{\partial\theta}(\bar{f}(\theta)) = A$ (Example: TD-learning)

Stochastic Newton Raphson:

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



SNR Algorithm:

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

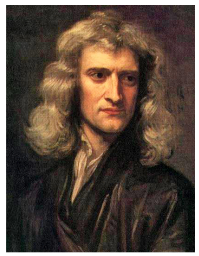
Example [D. & Meyn 2017]: LSTD(λ), *but this was* **not** *their motivation!*

Optimal Variance and SNR

Case 1: $\bar{f}(\theta) = A\theta - b \qquad \frac{\partial}{\partial\theta}(\bar{f}(\theta)) = A \qquad$ (Example: TD-learning)

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 [D. & Meyn 2017]: LSTD(λ), *but this was not their motivation!*

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

Optimal Variance and Zap-SNR

Case 2: $\bar{f}(\theta)$ nonlinear in θ $\quad A(\theta) = \frac{\partial}{\partial \theta}(\bar{f}(\theta))$ is a function of θ

(Example: Q-learning)

Optimal Variance and Zap-SNR

Case 2: $\bar{f}(\theta)$ nonlinear in θ $\quad A(\theta) = \frac{\partial}{\partial\theta}(\bar{f}(\theta))$ is a function of θ

(Example: Q-learning)

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

Case 2: $\bar{f}(\theta)$ nonlinear in θ $\quad A(\theta) = \frac{\partial}{\partial\theta}(\bar{f}(\theta))$ is a function of θ

(Example: Q-learning)

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

Case 2: $\bar{f}(\theta)$ nonlinear in θ $\quad A(\theta) = \frac{\partial}{\partial\theta}(\bar{f}(\theta))$ is a function of θ

(Example: Q-learning)

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, $\gamma_n/\alpha_n \to \infty, \qquad n \to \infty$

Optimal Variance and Zap-SNR

Case 2: $\bar{f}(\theta)$ nonlinear in θ $\quad A(\theta) = \frac{\partial}{\partial\theta}(\bar{f}(\theta))$ is a function of θ

(Example: Q-learning)

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, $\gamma_n/\alpha_n \to \infty, \qquad n \to \infty$

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

Optimal Variance and Zap-SNR

Case 2: $\bar{f}(\theta)$ nonlinear in θ $\quad A(\theta) = \frac{\partial}{\partial\theta}(\bar{f}(\theta))$ is a function of θ

(Example: Q-learning)

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, $\gamma_n/\alpha_n \to \infty, \qquad n \to \infty$

ODE for Zap-SNR

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

Optimal Variance and Zap-SNR

Case 2: $\bar{f}(\theta)$ nonlinear in θ $\quad A(\theta) = \frac{\partial}{\partial\theta}(\bar{f}(\theta))$ is a function of θ

(Example: Q-learning)

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, $\gamma_n/\alpha_n \to \infty, \qquad n \to \infty$

ODE for Zap-SNR

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

Stability? *Virtually universal*

Optimal Variance and Zap-SNR

Case 2: $\bar{f}(\theta)$ nonlinear in θ $\quad A(\theta) = \frac{\partial}{\partial\theta}(\bar{f}(\theta))$ is a function of θ

(Example: Q-learning)

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

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

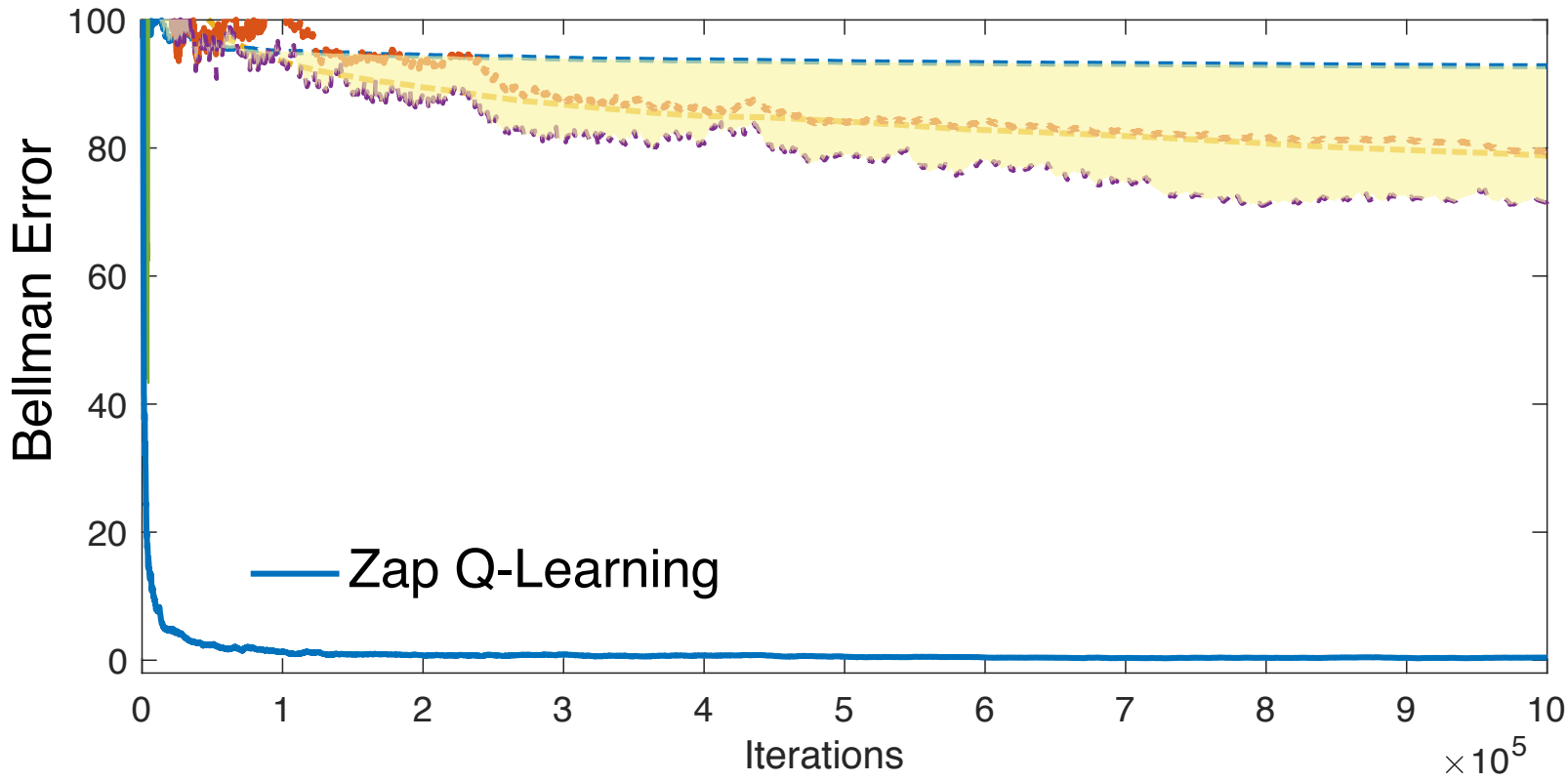
$$\widehat{A}_{n+1} = \widehat{A}_n + \gamma_{n+1}(A_{n+1} - \widehat{A}_n), \qquad A_{n+1} = \frac{d}{d\theta} f(\theta_n, W_{n+1})$$

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

ODE for Zap-SNR

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

Ruppert's "SqNR" technique also tries to mimic this ODE



Zap Q-learning

Watkins' Q-learning

Goal: Find θ^* that solves the *Projected Q-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: Find θ^* that solves the *Projected Q-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:

- Linear parameterization: $Q^\theta(x,u) = \theta^{\intercal}\psi(x,u)$
- $\zeta_n \equiv \nabla_\theta Q^\theta(X_n, U_n) = \psi(X_n, U_n)$
- $\psi_i(x,u) = \mathbb{I}\{x = x^i, u = u^i\}$ (complete basis)

Watkins' Q-learning

Goal: Find θ^* that solves the *Projected Q-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:

- Linear parameterization: $Q^\theta(x,u) = \theta^{\intercal}\psi(x,u)$
- $\zeta_n \equiv \nabla_\theta Q^\theta(X_n, U_n) = \psi(X_n, U_n)$
- $\psi_i(x,u) = \mathbb{I}\{x = x^i, u = u^i\}$ (complete basis)

Q-learning 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: Find θ^* that solves the *Projected Q-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:

- Linear parameterization: $Q^\theta(x,u) = \theta^{\intercal}\psi(x,u)$
- $\zeta_n \equiv \nabla_\theta Q^\theta(X_n, U_n) = \psi(X_n, U_n)$
- $\psi_i(x,u) = \mathbb{I}\{x = x^i, u = u^i\}$ (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: Find θ^* that solves the *Projected Q-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:

- Linear parameterization: $Q^\theta(x,u) = \theta^{\intercal}\psi(x,u)$
- $\zeta_n \equiv \nabla_\theta Q^\theta(X_n, U_n) = \psi(X_n, U_n)$
- $\psi_i(x,u) = \mathbb{I}\{x = x^i, u = u^i\}$ (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:

- Linear parameterization: $Q^\theta(x,u) = \theta^\intercal \psi(x,u)$
- $\zeta_n \equiv \nabla_\theta Q^\theta(X_n, U_n) = \psi(X_n, U_n)$
- $\psi_i(x,u) = \mathbb{I}\{x = x^i, u = u^i\}$ (complete basis)

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

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

[D & Meyn, 2017]

Zap Q-learning ($\equiv$ Zap-SNR for Q-Learning)

Algorithm:

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

Zap Q-learning ($\equiv$ Zap-SNR for Q-Learning)

Algorithm:

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

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

Zap Q-learning ($\equiv$ Zap-SNR for Q-Learning)

Algorithm:

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

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 ($\equiv$ Zap-SNR for Q-Learning)

Algorithm:

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

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

Zap Q-learning ($\equiv$ Zap-SNR for Q-Learning)

Algorithm:

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

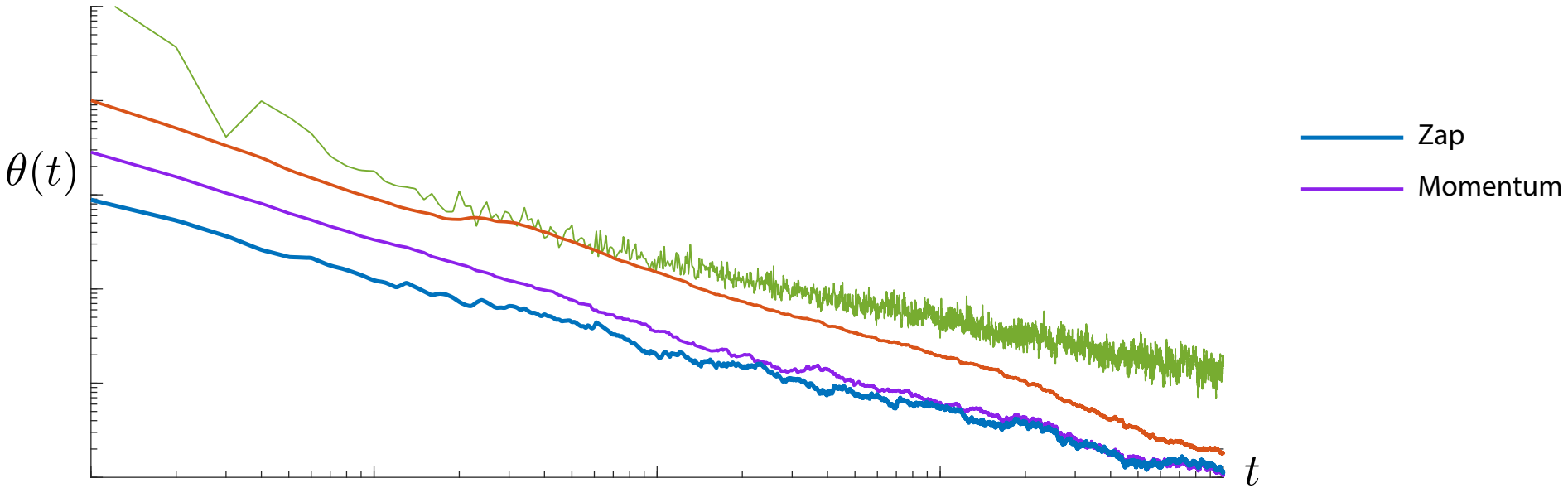
ODE Analysis: change of variables $q = \mathcal{Q}^*(\varsigma)$

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

- Convergence $\oplus$ optimal covariance $\oplus$ great transients
- Similar results for linear *and non-linear* function approximation Zap Q



Optimal Momentum Stochastic Approximation

Momentum based Stochastic Approximation

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

Momentum based Stochastic Approximation

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

Matrix Gain Stochastic Approximation:

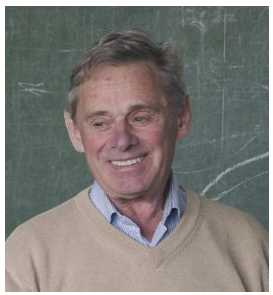
$$\Delta\theta_{n+1} = \alpha_n G_{n+1} f_{n+1}(\theta_n)$$

Momentum based Stochastic Approximation

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

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

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

Momentum based Stochastic Approximation

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

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

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

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 $G_n = \zeta I$

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

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

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

following Nesterov, 1983

Momentum based Stochastic Approximation

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

Special case $G_n = \zeta I$

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

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

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

Coupling of SNR and PoISA: $\|\theta_n^{\text{SNR}} - \theta_n^{\text{PoISA}}\|^2 = O(n^{-2})$

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: $\|\theta_n^{\text{SNR}} - \theta_n^{\text{PolSA}}\|^2 = O(n^{-2})$

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

Momentum based Stochastic Approximation

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

Special case $G_n = \zeta I$

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

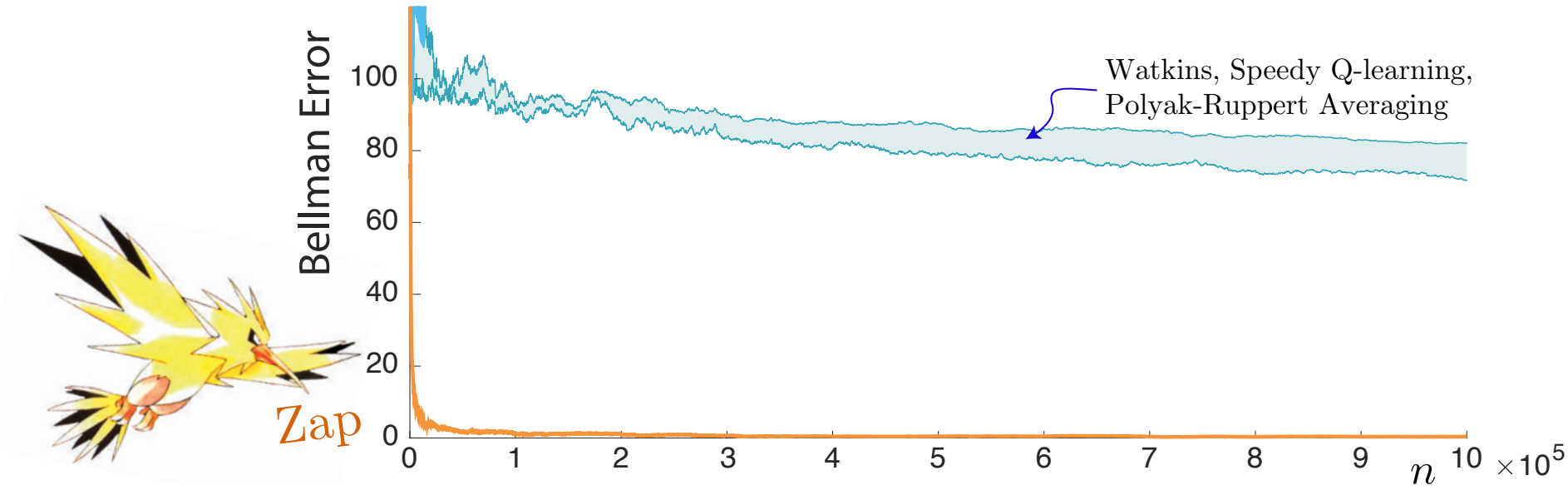
$$\text{PolSA:} \quad \Delta\theta_{n+1} = [I + \zeta \widehat{A}_{n+1}]\Delta\theta_n + \alpha_{n+1}\zeta f_{n+1}(\theta_n)$$

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

Coupling of SNR and PolSA: $\|\theta_n^{\text{SNR}} - \theta_n^{\text{PolSA}}\|^2 = O(n^{-2})$

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

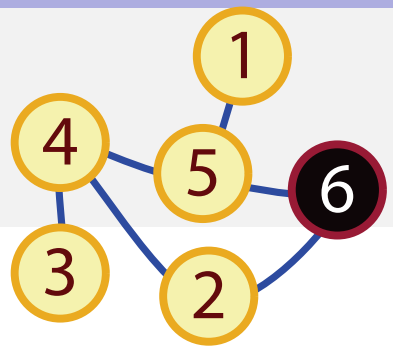
PolSA has optimal asymptotic variance



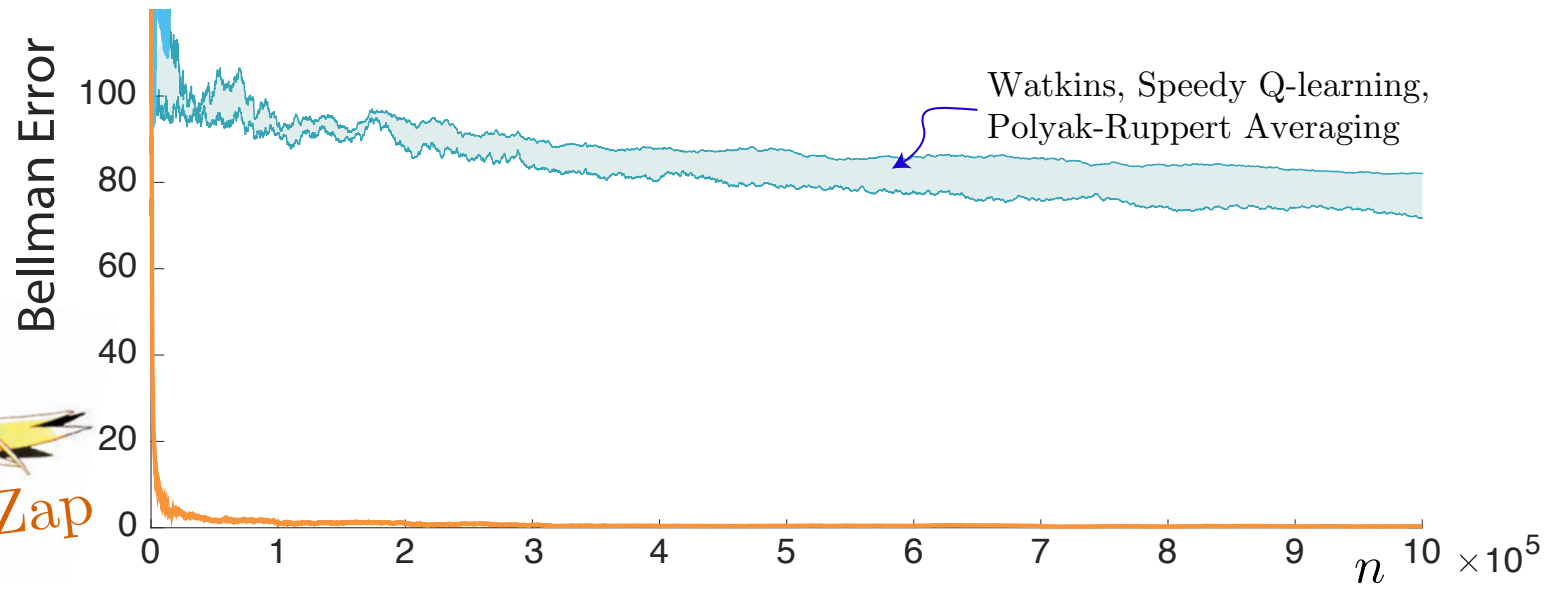
Zap & Momentum in Reinforcement Learning

Zap Q-learning

Example: Stochastic Shortest Path



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

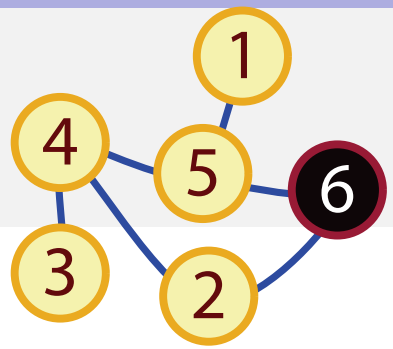


Convergence of Zap-Q Learning

Discount factor: $\beta = 0.99$

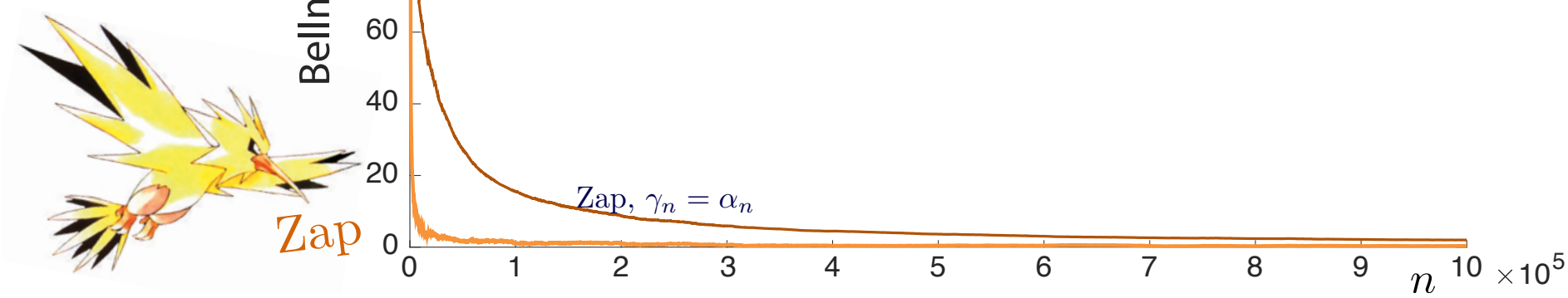
Zap Q-learning

Example: Stochastic Shortest Path



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

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

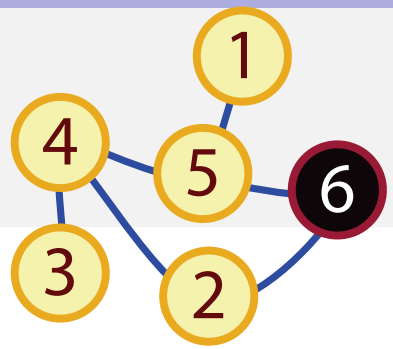


Convergence of Zap-Q Learning

Discount factor: $\beta = 0.99$

Zap Q-learning

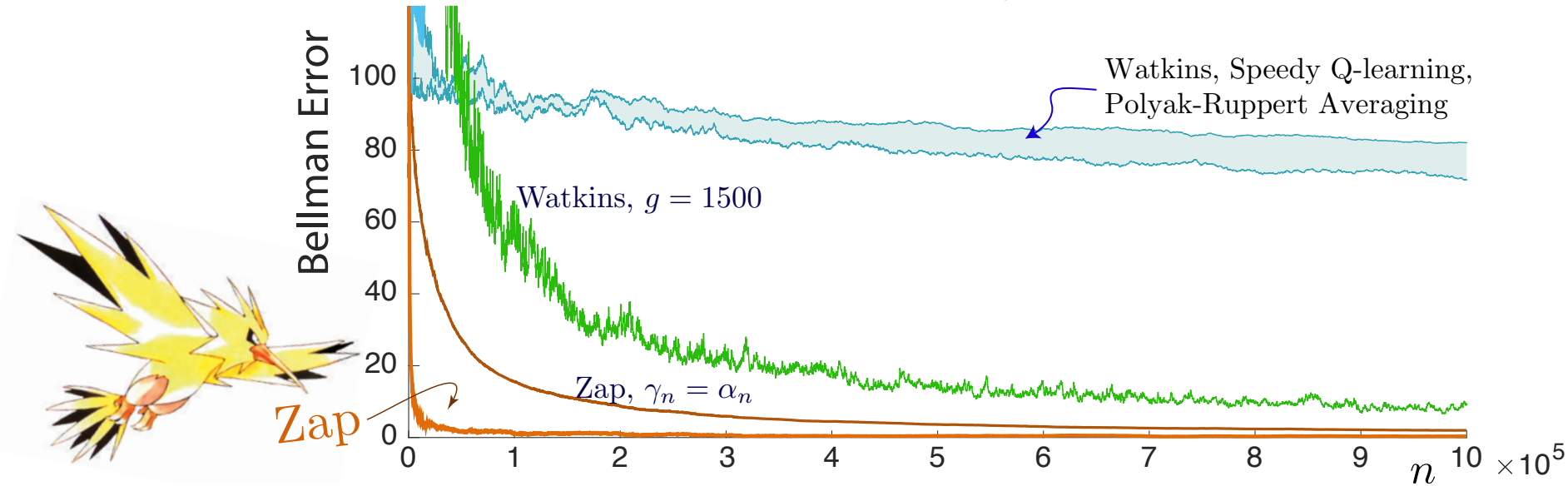
Example: Stochastic Shortest Path



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 $\alpha_n = 1500/n$

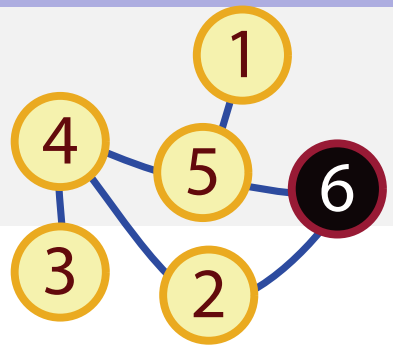


Convergence of Zap-Q Learning

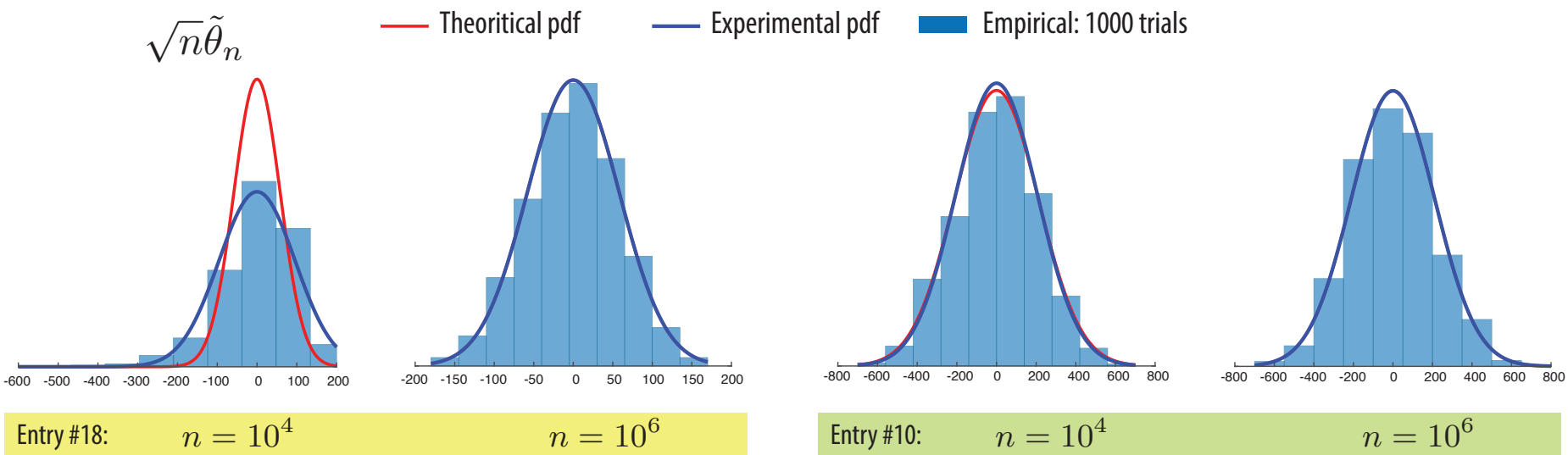
Discount factor: $\beta = 0.99$

Zap Q-learning

Optimize Walk to Cafe



Convergence with Zap gain $\gamma_n = n^{-0.85}$:
Histogram of $\sqrt{n}\tilde{\theta}_n$



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

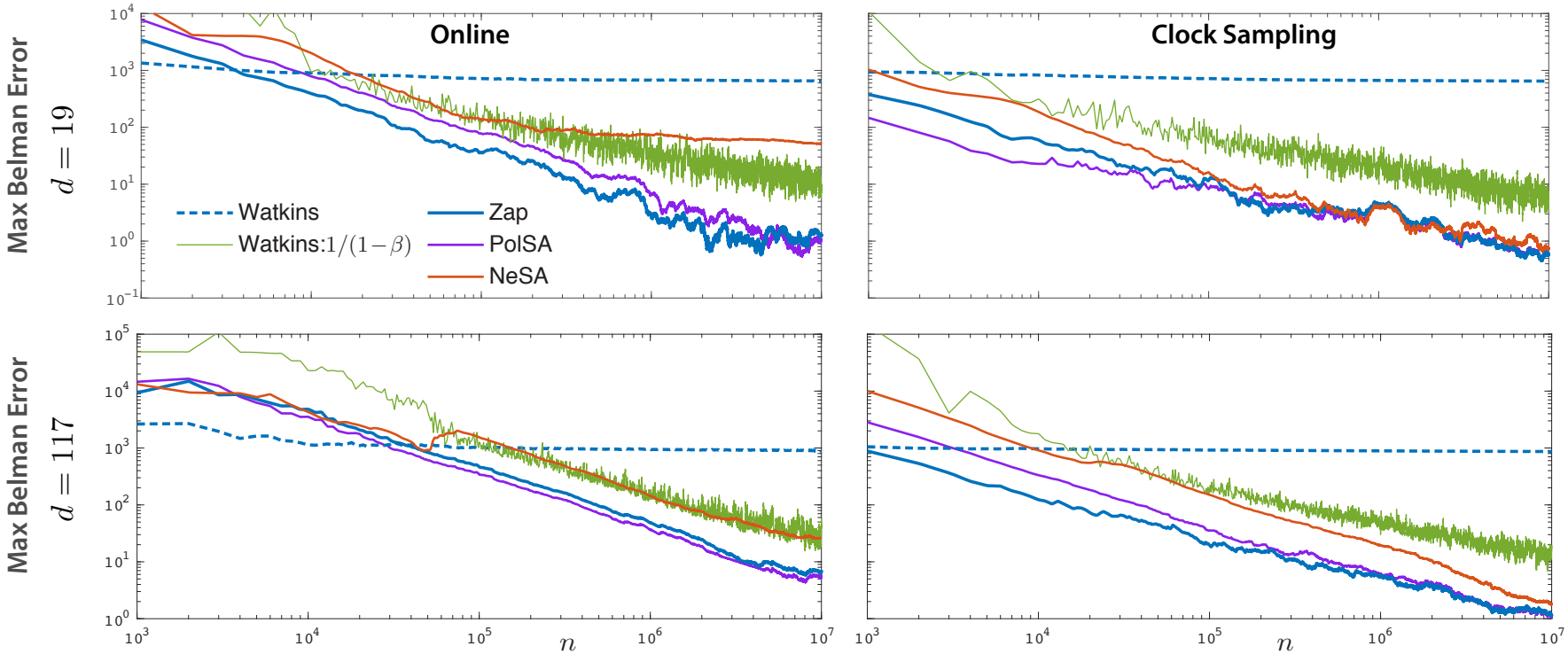
Entry #10: $n = 10^4$ $n = 10^6$

CLT gives good prediction of finite-n performance

Discount factor: $\beta = 0.99$

Zap Q-learning and Momentum

Coupling and convergence for larger models:



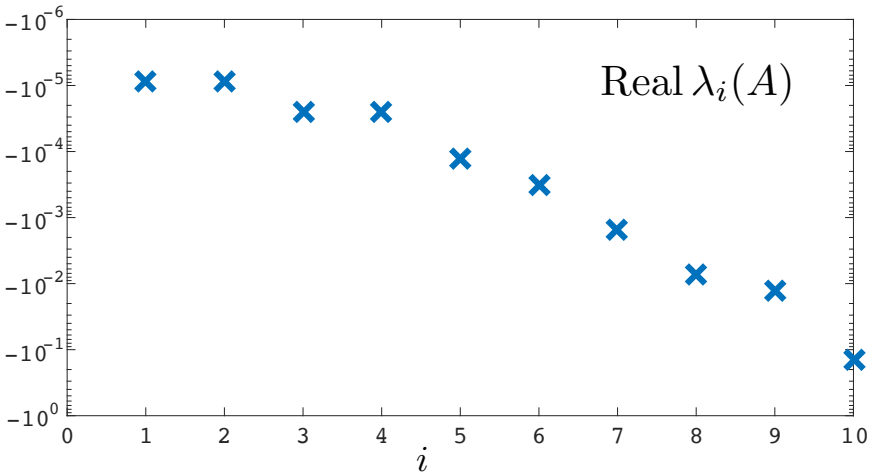
Coupling is amazing

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

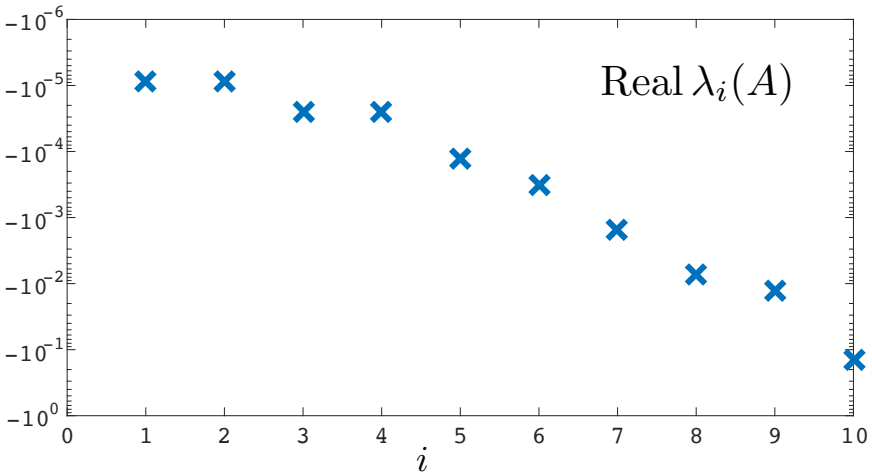
Asymptotic covariance is infinite

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

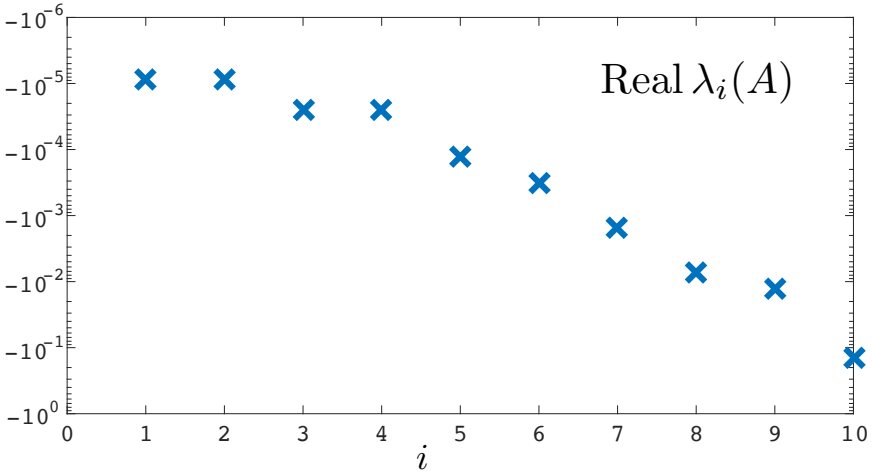
$\{G_n\}$ (see refs for details)

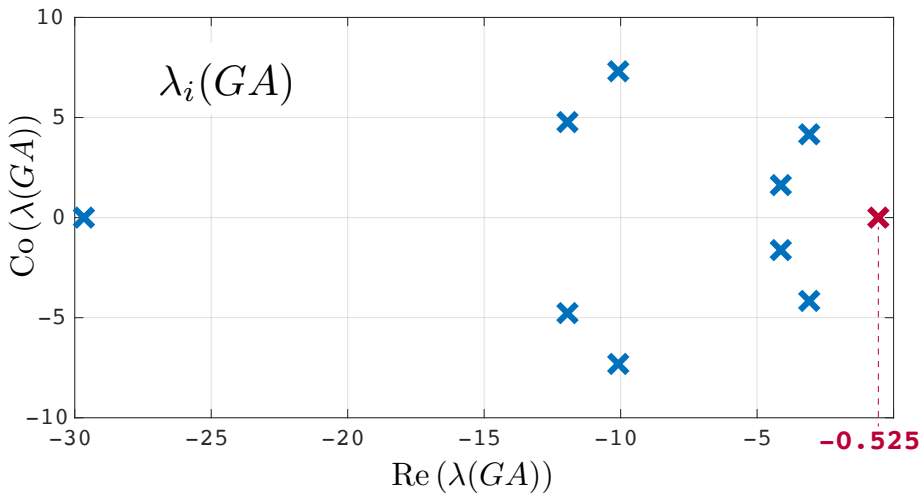
Zap Q-learning

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 of the "KF" algorithm for the finance example

Favorite choice of gain in [Choi & Van Roy 2006] "just" meets the criterion $\text{Re}(\lambda(GA)) < -\frac{1}{2}$

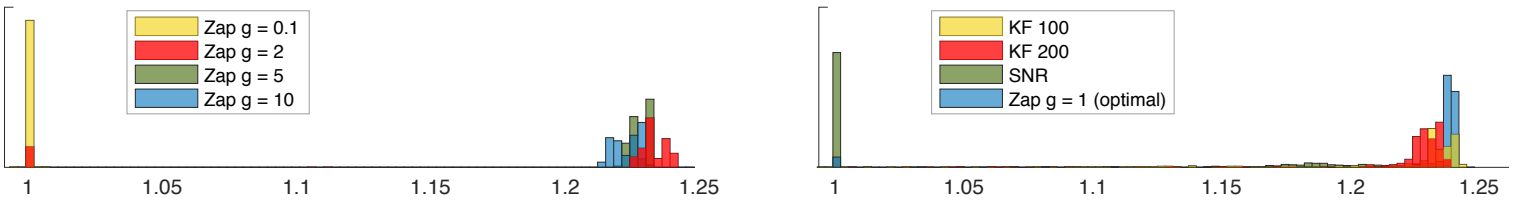
Zap Q-learning

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

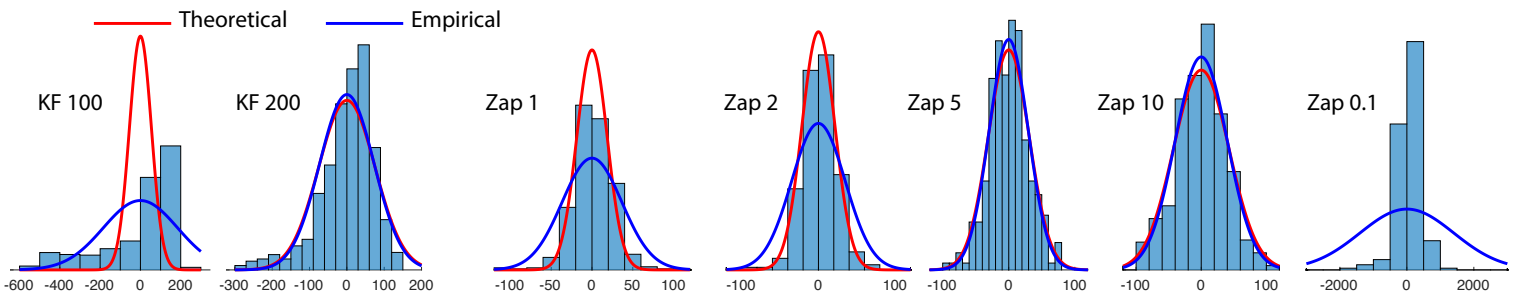
State space: $\mathbb{R}^{100}$

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:



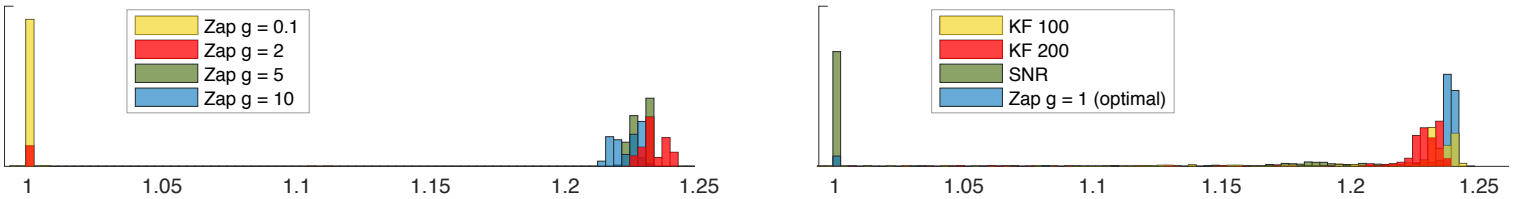
Zap Q-learning

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

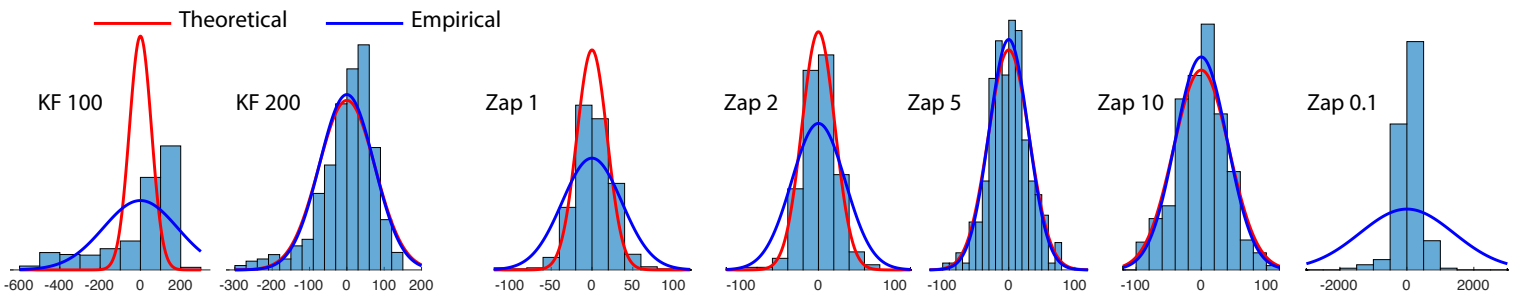
State space: $\mathbb{R}^{100}$

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

Histograms of the average reward obtained using the different algorithms:



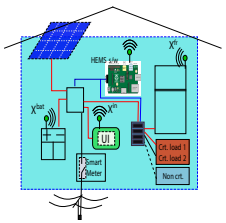
Asymptotic variance for of the algorithms:

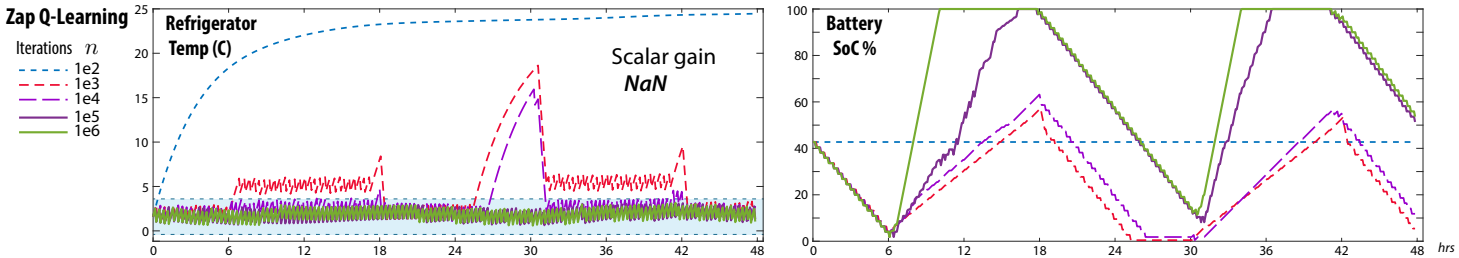


Zap-Q $\gg$ KF

Zap Q-learning for Energy Control in Buildings

Joint work with N. S. Raman and P. Barooah @ UF MAE

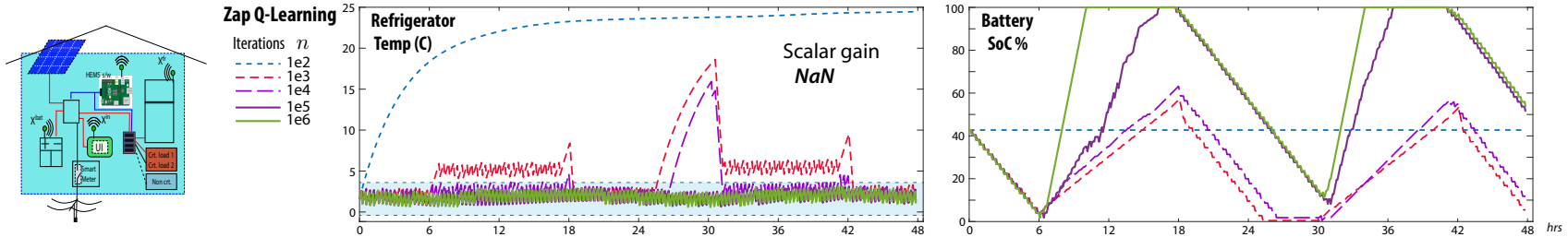




"Hurricane Resiliency": Stay alive as long as possible

Zap Q-learning for Energy Control in Buildings

Joint work with N. S. Raman and P. Barooah @ UF MAE

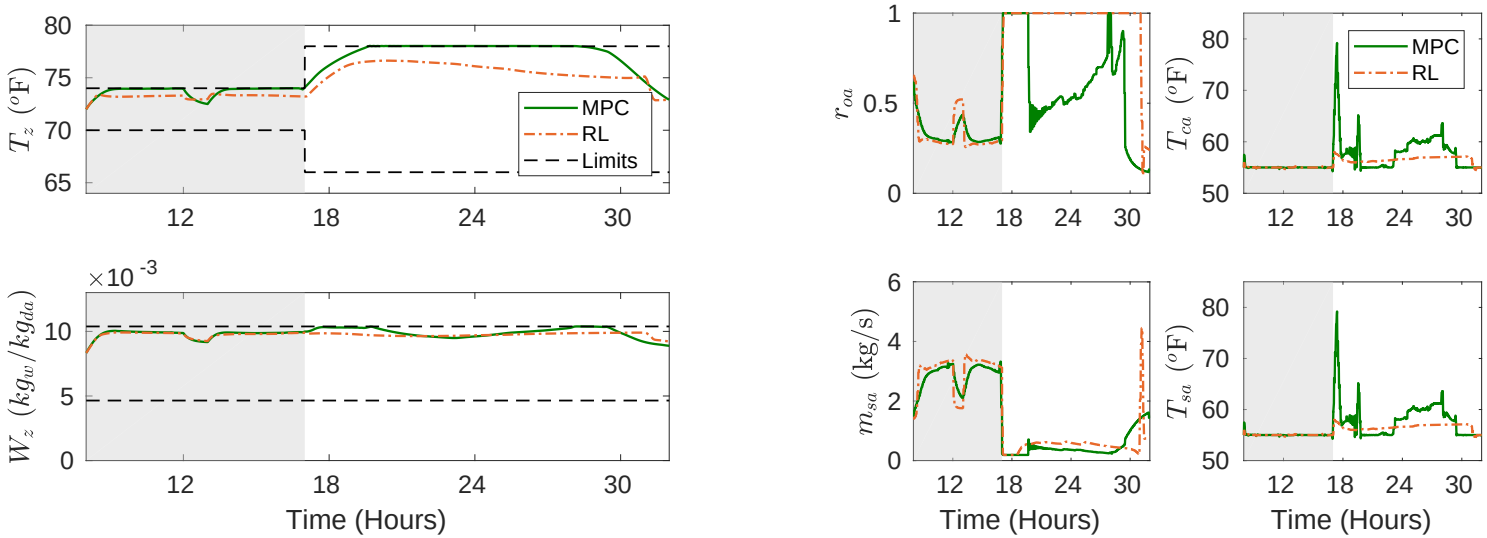


"Hurricane Resiliency": Stay alive as long as possible

- Goal: Maintain refrigerator temperature within a band, relying only on renewable energy
- Control: Switching the refrigerator on or off
- *State feedback controller* was learnt using Zap-Q on Gainesville weather data
- MPC based approaches is complicated $\oplus$ needs model information

Zap Q-learning for Energy Control in Buildings

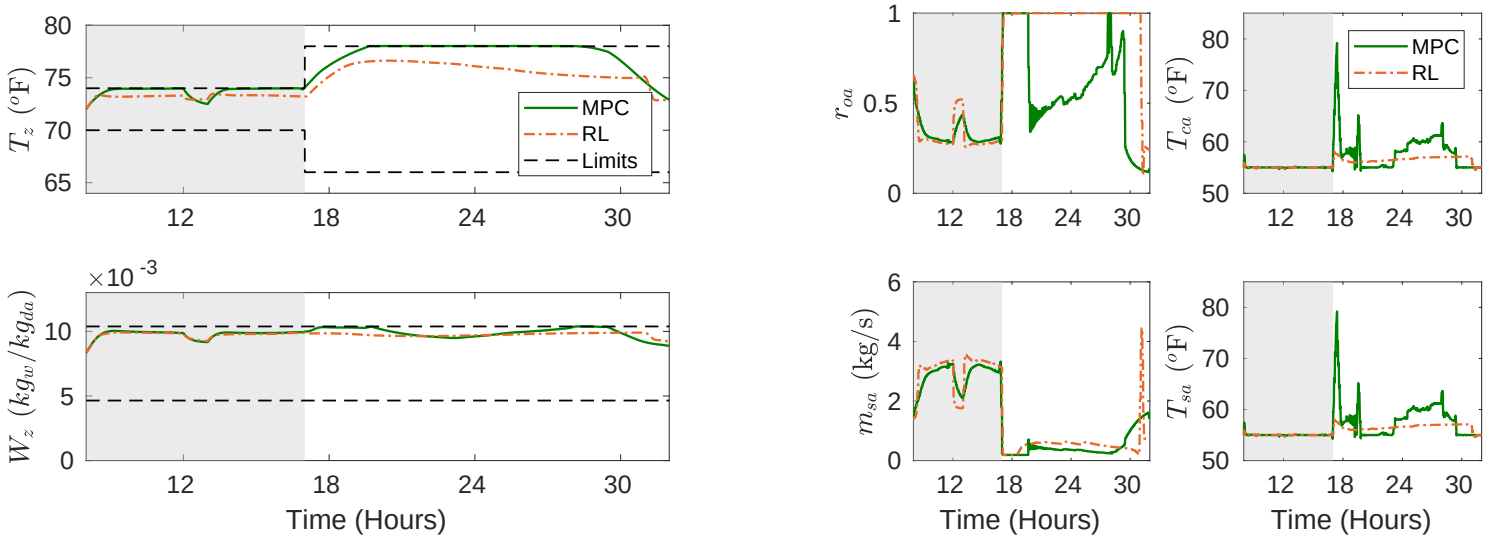
Joint work with N. S. Raman and P. Barooah @ UF MAE



Energy Control in Pugh Hall: Comparison of RL and MPC

Zap Q-learning for Energy Control in Buildings

Joint work with N. S. Raman and P. Barooah @ UF MAE

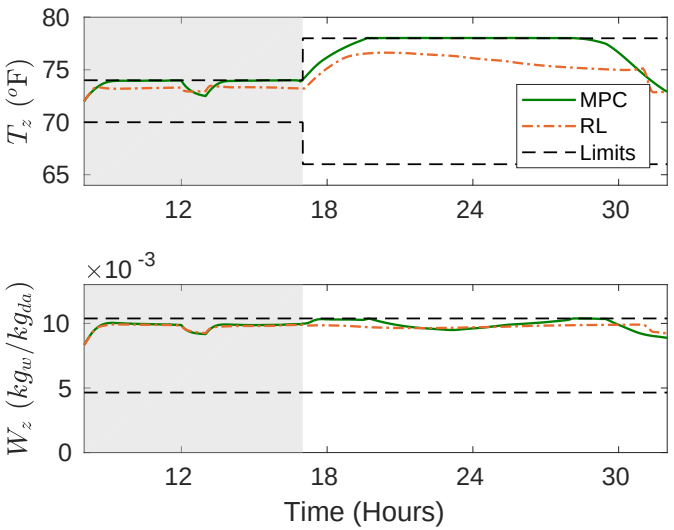


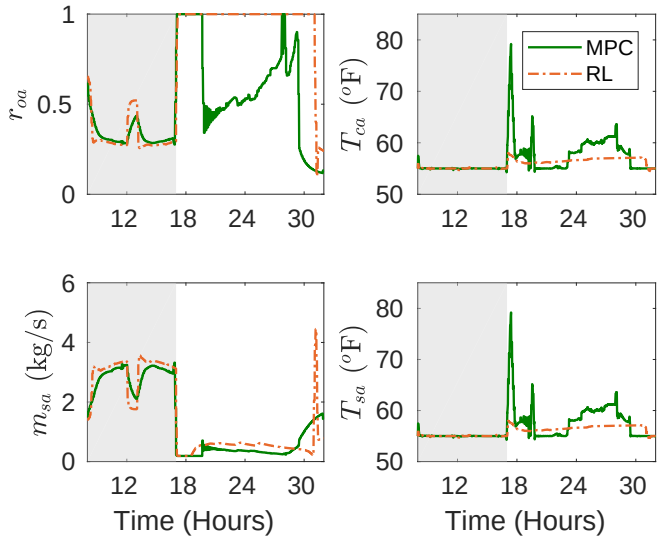
Energy Control in Pugh Hall: Comparison of RL and MPC

- Goal: Maintain zone temperature and humidity, *and* minimize the energy consumption
- Control: Air-flow rate, out-door air ratio, conditioned air temperature, supply air temperature
- *Zap-Q is the only Q-learning algorithm that worked!*

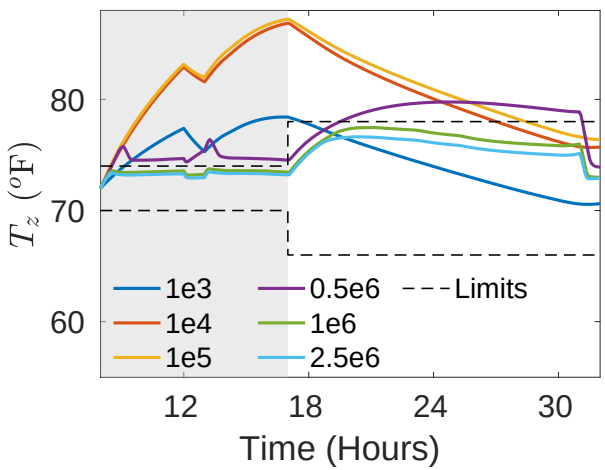
Zap Q-learning for Energy Control in Buildings

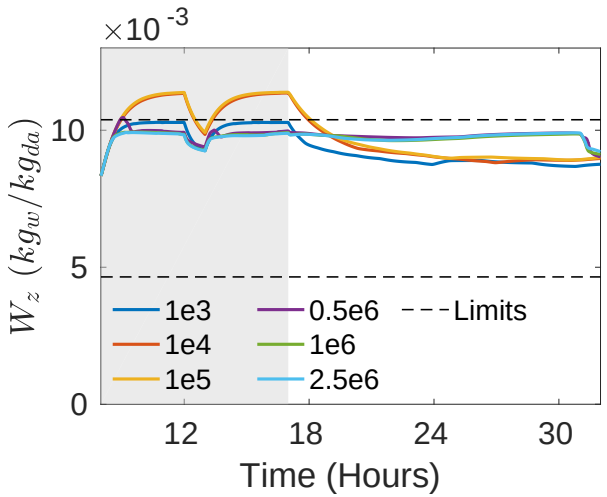
Joint work with N. S. Raman and P. Barooah @ UF MAE





Energy Control in Pugh Hall: The Learning Process





Conclusions & Future Work

Conclusions:

- The Zap Q-learning algorithm achieves optimal rate of convergence,

Conclusions & Future Work

Conclusions:

- The Zap Q-learning algorithm achieves optimal rate of convergence, *and also leads to stable algorithms in a function approximation setting!*

Conclusions & Future Work

Conclusions:

- The Zap Q-learning algorithm achieves optimal rate of convergence, *and also leads to stable algorithms in a function approximation setting!*
- We proposed "computationally efficient" algorithms in the form of *matrix momentum techniques*

Conclusions & Future Work

Conclusions:

- The Zap Q-learning algorithm achieves optimal rate of convergence, *and also leads to stable algorithms in a function approximation setting!*
- We proposed "computationally efficient" algorithms in the form of *matrix momentum techniques*
- Applied the algorithms to *solve real problems*

Conclusions & Future Work

Conclusions:

- The Zap Q-learning algorithm achieves optimal rate of convergence, *and also leads to stable algorithms in a function approximation setting!*
- We proposed "computationally efficient" algorithms in the form of *matrix momentum techniques*
- Applied the algorithms to *solve real problems*

Future work:

- Gradient free Zap Q (Zap-SqNR)
  - More suitable for non-linear Zap Q-learning
- Acceleration techniques (momentum and matrix momentum) for Q-learning
- Further variance reduction using control variates


Thank you!