Adapting to Mixing Time in Stochastic Optimization with Markovian Data

Ron Dorfman Kfir Y. Levy

The Viterbi Faculty of Electrical and Computer Engineering

TECHNION
Israel Institute of Technology

ICML
International Conference On Machine Learning

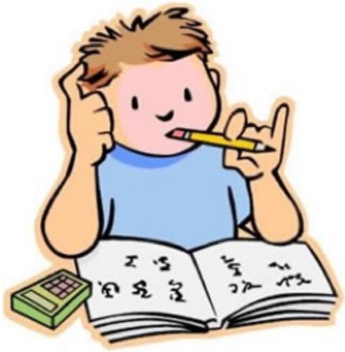
Motivation

- **Classical ML**: Learner uses data to optimize objective

(x_T, y_T) … (x_2, y_2) (x_1, y_1) Learner

…

→



Motivation

- **Classical ML**: Learner uses data to optimize objective

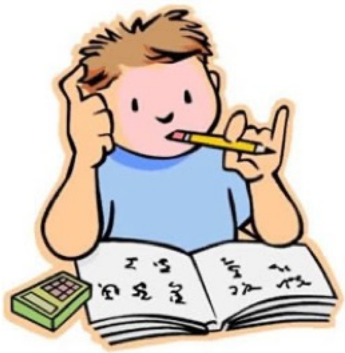
(x_T, y_T)

…

(x_2, y_2)

(x_1, y_1)

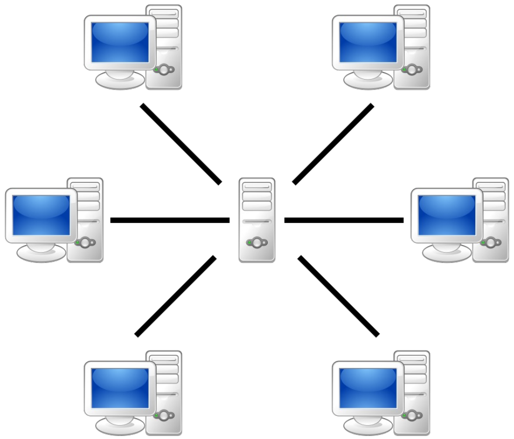
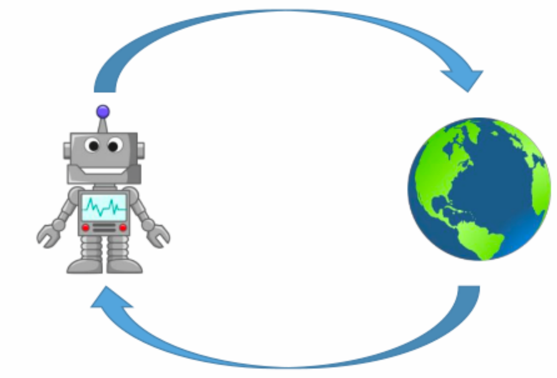
Learner



- **Standard assumption**: data points are <u>i.i.d.</u>

Motivation

- i.i.d. assumption is often violated, e.g., temporal data

Finance	Weather	Distributed opt.	RL
	 https://meteosim.com/en/will-artificial-intelligence-replace-the-conventional-weather-forecasting-system/		

Motivation

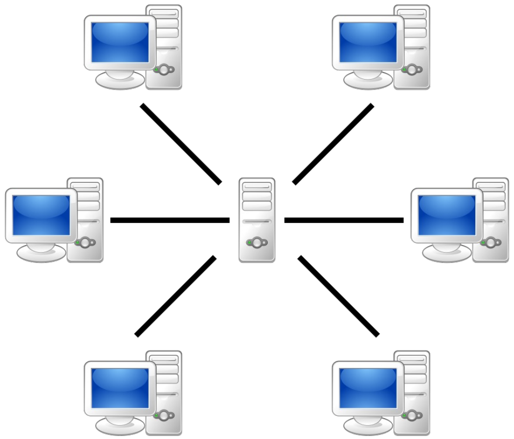
- i.i.d. assumption is often violated, e.g., temporal data

<u>Finance</u>

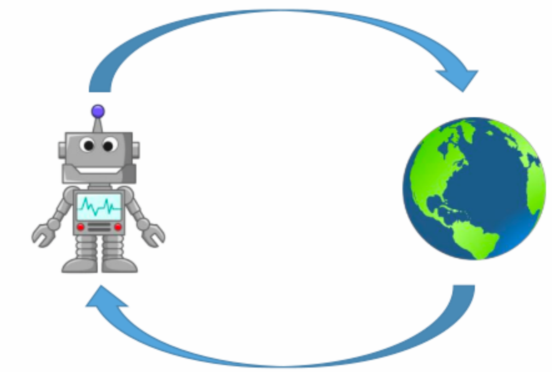
<u>Weather</u>

https://meteosim.com/en/will-artificial-intelligence-replace-the-conventional-weather-forecasting-system/

<u>Distributed opt.</u>



<u>RL</u>



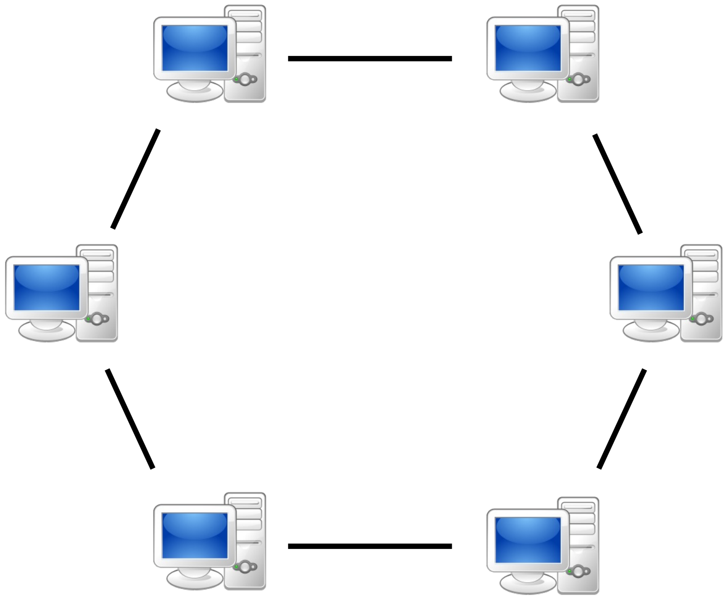
- Makrovian model: simple way to model temporal data

Example: Peer-to-Peer Distributed Optimization

- Each machine $i \in [n]$ holds data: $\{X^i, y^i\} \Longrightarrow$ empirical risk $f_i(w)$
- <u>Goal</u>: minimize risk over network

$$f(w) = \frac{1}{n}\sum_{i=1}^{n} f_i(w)$$

- At time t, machine $i(t)$ holds the parameters.
- $i(t)$ evolves according to a Markov chain.

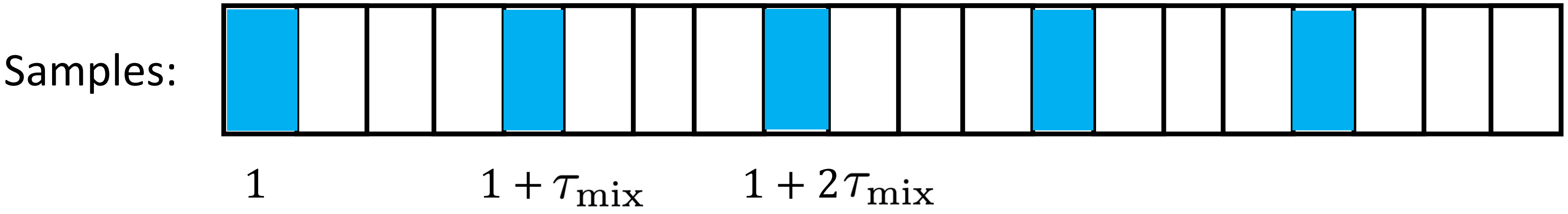


Markov Chains and Mixing Time

- **Assumption**: Markov Chain has <u>finite</u> state space and is <u>ergodic</u>

  $\Longrightarrow$ There exists a unique stationary distribution μ.

- **Mixing time** $\tau_{\text{mix}}(\epsilon)$:
  - Samples separated by $\tau_{\text{mix}}(\epsilon)$ are "ϵ-independent".



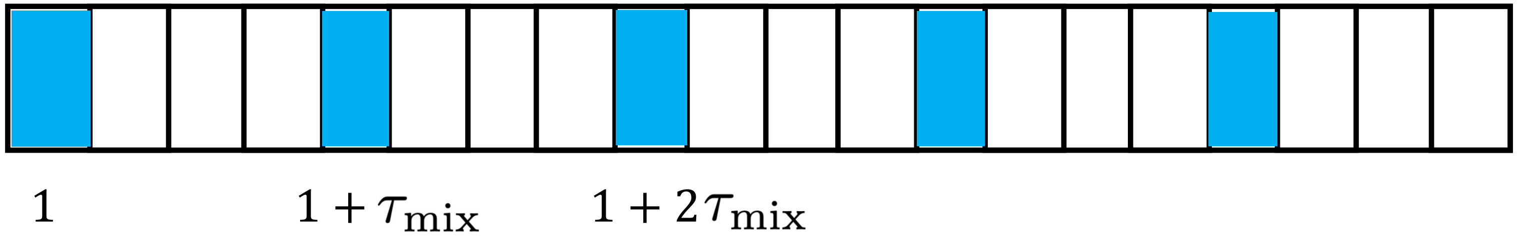
Markov Chains and Mixing Time

- **Assumption**: Markov Chain has <u>finite</u> state space and is <u>ergodic</u>

  $\Longrightarrow$ There exists a unique stationary distribution μ.

- **Mixing time** $\tau_{\text{mix}}(\epsilon)$:
  - Samples separated by $\tau_{\text{mix}}(\epsilon)$ are "ϵ-independent".

Samples:



  - Usually: $\tau_{\text{mix}} \coloneqq \tau_{\text{mix}}(1/4)\,, \quad \tau_{\text{mix}}(\epsilon) \propto \tau_{\text{mix}} \cdot \log(1/\epsilon)$

Stochastic Optimization with Markovian Data

- Access to $z_1, z_2, \dots, z_T$ from a Markov chain
- Minimize expected loss w.r.t. **stationary distribution**

$$\min_{w \in \mathcal{K}} F(w) \coloneqq \mathbb{E}_{\boldsymbol{z \sim \mu}}[f(w; z)]$$

- **Performance measure**: $\mathrm{err}(\bar{w}_T) = F(\bar{w}_T) - \min_{w \in \mathcal{K}} F(w)$

Stochastic Optimization with Markovian Data

- Access to $z_1, z_2, \dots, z_T$ from a Markov chain
- Minimize expected loss w.r.t. **stationary distribution**

$$\min_{w \in \mathcal{K}} F(w) \coloneqq \mathbb{E}_{\boldsymbol{z \sim \mu}}[f(w; z)]$$

- **Performance measure**: $\mathrm{err}(\overline{w}_T) = F(\overline{w}_T) - \min_{w \in \mathcal{K}} F(w)$
- Mainly focus on the convex case: $\{w \mapsto f(\cdot; z)\}$ and $\mathcal{K}$ are convex

Simply use SGD?

SGD:

$$\left.\begin{aligned} g_t &= \nabla f(w_t; z_t) \\ w_{t+1} &= \Pi_{\mathcal{K}}(w_t - \eta_t g_t) \end{aligned}\right., \quad t = 1, \ldots, T.$$

Simply use SGD?

SGD:

$$\begin{aligned} g_t &= \nabla f(w_t; z_t) \\ w_{t+1} &= \Pi_{\mathcal{K}}(w_t - \eta_t g_t) \end{aligned}, \quad t = 1, \dots, T.$$

The i.i.d. case

$$z_t \sim \mu \implies \mathbb{E}[g_t | w_t] = \nabla F(w_t)$$

Convergence:

$$\eta_t \propto 1/\sqrt{T} \implies \text{err} \le \mathcal{O}\left(1/\sqrt{T}\right)$$ [1]

Optimal!

[1] Agarwal et al., "Information-theoretic lower bounds on the oracle complexity of stochastic convex optimization", 2012.

Simply use SGD?

SGD:

$$\begin{aligned} g_t &= \nabla f(w_t; z_t) \\ w_{t+1} &= \Pi_{\mathcal{K}}(w_t - \eta_t g_t) \end{aligned}, \quad t = 1, \dots, T.$$

The i.i.d. case

$$z_t \sim \mu \implies \mathbb{E}[g_t | w_t] = \nabla F(w_t)$$

Convergence:

$$\eta_t \propto 1/\sqrt{T} \implies \text{err} \leq \mathcal{O}\left(1/\sqrt{T}\right)$$ [1]

Optimal!

The non-i.i.d. case

$$z_t \sim P(\cdot \,| z_1, \dots, z_{t-1}) \implies \mathbb{E}[g_t | w_t] \neq \nabla F(w_t)$$

Bias!

[1] Agarwal et al., "Information-theoretic lower bounds on the oracle complexity of stochastic convex optimization", 2012.

Simply use SGD?

SGD:
$$\begin{aligned} g_t &= \nabla f(w_t; z_t) \\ w_{t+1} &= \Pi_{\mathcal{K}}(w_t - \eta_t g_t) \end{aligned}, \quad t = 1, \dots, T.$$

The i.i.d. case

$$z_t \sim \mu \implies \mathbb{E}[g_t | w_t] = \nabla F(w_t)$$

Convergence:

$$\eta_t \propto 1/\sqrt{T} \implies \text{err} \leq \mathcal{O}\left(1/\sqrt{T}\right)$$ [1]

Optimal!

The non-i.i.d. case

$$z_t \sim P(\cdot\, | z_1, \dots, z_{t-1}) \implies \mathbb{E}[g_t | w_t] \neq \nabla F(w_t)$$

Bias!

Convergence:

$$\eta_t \propto 1/\sqrt{T} \implies \text{err} \leq \mathcal{O}\left(\tau_{\text{mix}}/\sqrt{T}\right)$$

[1] Agarwal et al., "Information-theoretic lower bounds on the oracle complexity of stochastic convex optimization", 2012.

Simply use SGD?

SGD:

$$\begin{aligned} g_t &= \nabla f(w_t; z_t) \\ w_{t+1} &= \Pi_{\mathcal{K}}(w_t - \eta_t g_t) \end{aligned}, \quad t = 1, \dots, T.$$

The i.i.d. case

$$z_t \sim \mu \implies \mathbb{E}[g_t | w_t] = \nabla F(w_t)$$

Convergence:

$$\eta_t \propto 1/\sqrt{T} \implies \mathrm{err} \leq \mathcal{O}(1/\sqrt{T})$$ [1]

Optimal!

The non-i.i.d. case

$$z_t \sim P(\cdot | z_1, \dots, z_{t-1}) \implies \mathbb{E}[g_t | w_t] \neq \nabla F(w_t)$$

Bias!

Convergence:

$$\eta_t \propto 1/\sqrt{T} \implies \mathrm{err} \leq \mathcal{O}(\tau_{\mathrm{mix}}/\sqrt{T})$$

Suboptimal! Lower bound: $\Omega(\sqrt{\tau_{\mathrm{mix}}}/\sqrt{T})$ [2]

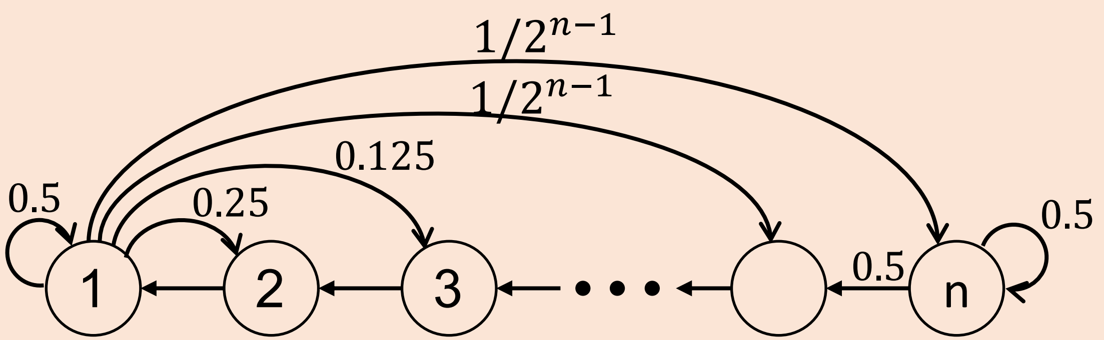
[1] Agarwal et al., "Information-theoretic lower bounds on the oracle complexity of stochastic convex optimization", 2012.
[2] Duchi et al., "Ergodic mirror descent", 2012.

Simply use SGD?

SGD:

$$\begin{aligned} g_t &= \nabla f(w_t; z_t) \\ w_{t+1} &= \Pi_{\mathcal{K}}(w_t - \eta_t g_t) \end{aligned}, \quad t = 1, \dots, T.$$

Time reversal of the winning streak



$$\tau_{\text{mix}} = \Theta(n)$$

The non-i.i.d. case

$$z_t \sim P(\cdot \mid z_1, \dots, z_{t-1}) \implies \mathbb{E}[g_t \mid w_t] \neq \nabla F(w_t)$$

Bias!

Convergence:

$$\eta_t \propto 1/\sqrt{T} \implies \text{err} \leq \mathcal{O}\left(\tau_{\text{mix}}/\sqrt{T}\right)$$

Suboptimal! Lower bound: $\Omega\left(\sqrt{\tau_{\text{mix}}}/\sqrt{T}\right)$[2]

[1] Agarwal et al., “Information-theoretic lower bounds on the oracle complexity of stochastic convex optimization”, 2012.
[2] Duchi et al., “Ergodic mirror descent”, 2012.

How to Obtain Optimal Convergence?

- **Smaller learning rate[1]:**
  - $\eta_t \propto 1/\sqrt{\tau_{\text{mix}} T} \implies \text{err} \leq \mathcal{O}\left(\sqrt{\tau_{\text{mix}}}/\sqrt{T}\right)$

[1] Duchi et al., "Ergodic mirror descent", 2012.

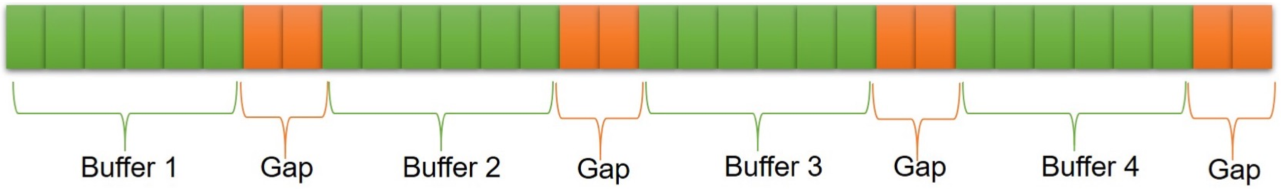
How to Obtain Optimal Convergence?

- **Smaller learning rate[1]**:
  - $\eta_t \propto 1/\sqrt{\tau_{\text{mix}} T} \implies \text{err} \leq \mathcal{O}\left(\sqrt{\tau_{\text{mix}}}/\sqrt{T}\right)$
- **SGD-Data Drop (SGD-DD)**:
  - SGD step once in every τ_{mix} samples ($\eta_t \propto 1/\sqrt{T}$)
  - Effectively T/τ_{mix} SGD updates: $\text{err} \leq \mathcal{O}\left(1/\sqrt{T_{\text{eff}}}\right) = \mathcal{O}\left(\sqrt{\tau_{\text{mix}}}/\sqrt{T}\right)$

[1] Duchi et al., “Ergodic mirror descent”, 2012.

How to Obtain Optimal Convergence?

- **Smaller learning rate[1]**:
  - $\eta_t \propto 1/\sqrt{\tau_{\text{mix}} T} \implies \text{err} \le \mathcal{O}\left(\sqrt{\tau_{\text{mix}}}/\sqrt{T}\right)$
- **SGD-Data Drop (SGD-DD)**:
  - SGD step once in every τ_{mix} samples ($\eta_t \propto 1/\sqrt{T}$)
  - Effectively T/τ_{mix} SGD updates: $\text{err} \le \mathcal{O}\left(1/\sqrt{T_{\text{eff}}}\right) = \mathcal{O}\left(\sqrt{\tau_{\text{mix}}}/\sqrt{T}\right)$
- **Replay buffer[2,3]**:
  - Store past data to decorrelate samples



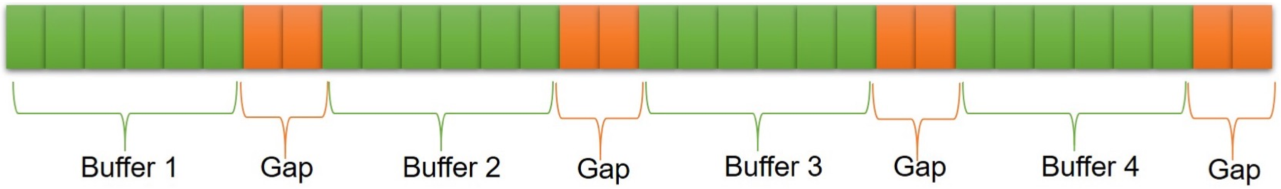
[1] Duchi et al., “Ergodic mirror descent”, 2012.
[2] Bresler et al., “Least squares regression with markovian data: fundamental limits and algorithms”, 2020.
[3] Jain et al., “Streaming linear system identification with reverse experience replay ”, 2021.

How to Obtain Optimal Convergence?

- **Smaller learning rate**[1]:
  - $\eta_t \propto 1$
- **SGD-Dat**
  - SGD st
  - Effectively T/τ_{mix} SGD updates: $\text{err} \leq \mathcal{O}(1/\sqrt{T_{\text{eff}}}) = \mathcal{O}(\sqrt{\tau_{\text{mix}}}/\sqrt{T})$

Problem: The prior knowledge of τ_{mix} is required

- **Replay buffer**[2,3]:
  - Store past data to decorrelate samples



[1] Duchi et al., “Ergodic mirror descent”, 2012.
[2] Bresler et al., “Least squares regression with markovian data: fundamental limits and algorithms”, 2020.
[3] Jain et al., “Streaming linear system identification with reverse experience replay ”, 2021.

How to Obtain Optimal Convergence?

- **Smaller learning rate[1]:**
  - $\eta_t \propto 1$
- **SGD-Dat**
  - SGD st
  - Effectively T/τ_{mix} SGD updates: $\text{err} \leq \mathcal{O}(1/\sqrt{T_{\text{eff}}}) = \mathcal{O}(\sqrt{\tau_{\text{mix}}}/\sqrt{T})$
- **Repla**
  - Sto

Problem: The prior knowledge of τ_{mix} is required

This Work: Optimal performance without knowing τ_{mix}

[1] Duchi et al., “Ergodic mirror descent”, 2012.
[2] Bresler et al., “Least squares regression with markovian data: fundamental limits and algorithms”, 2020.
[3] Jain et al., “Streaming linear system identification with reverse experience replay ”, 2021.

Our Technique

- **Challenge**: Gradient estimates are biased

$$\mathbb{E}[g_t|w_t] = \nabla F(w_t) \pm \mathcal{O}(1)\,, \quad \& \quad \mathbf{Var}(g_t) = \mathcal{O}(1)$$

Our Technique

- **Challenge**: Gradient estimates are biased

$$\mathbb{E}[g_t|w_t] = \nabla F(w_t) \pm \mathcal{O}(1)\,, \quad \& \quad \mathbf{Var}(g_t) = \mathcal{O}(1)$$

- **Straightforward approach**: Reduce bias by averaging

Using a mini-batch of size T ensures (via concentration for MCs):

$$\mathbb{E}[\tilde{g}_t|w_t] = \nabla F(w_t) \pm \mathcal{O}(\sqrt{\tau_{\text{mix}}/T})\,, \ \& \ \mathbf{Var}(\tilde{g}_t) = \mathcal{O}(\tau_{\text{mix}}/T)$$

Our Technique

- **Challenge**: Gradient estimates are biased

$$\mathbb{E}[g_t | w_t] = \nabla F(w_t) \pm \mathcal{O}(1) \,, \quad \& \quad \mathbf{Var}(g_t) = \mathcal{O}(1)$$

- **Straightforward approach**: Reduce bias by averaging

Using a mini-batch of size T ensures (via concentration for MCs):

$$\mathbb{E}[\tilde{g}_t | w_t] = \nabla F(w_t) \pm \mathcal{O}(\sqrt{\tau_{\text{mix}}/T}) \,, \ \& \ \mathbf{Var}(\tilde{g}_t) = \mathcal{O}(\tau_{\text{mix}}/T)$$

Too Costly!

Our Technique

- **Challenge**: Gradient estimates are biased

$$\mathbb{E}[g_t|w_t] = \nabla F(w_t) \pm \mathcal{O}(1)\,, \quad \& \quad \mathbf{Var}(g_t) = \mathcal{O}(1)$$

- **Straightforward approach**: Reduce bias by averaging

  Using a mini-batch of size T ensures (via concentration for MCs):

$$\mathbb{E}[\tilde{g}_t|w_t] = \nabla F(w_t) \pm \mathcal{O}(\sqrt{\tau_{\mathrm{mix}}/T})\,, \;\; \& \;\; \mathbf{Var}(\tilde{g}_t) = \mathcal{O}(\tau_{\mathrm{mix}}/T)$$

- **Wishful thinking**: cheap way to <u>trade bias for variance</u>

$$\mathbb{E}[\tilde{g}_t|w_t] = \nabla F(w_t) \pm \mathcal{O}(\sqrt{\tau_{\mathrm{mix}}/T})\,, \;\; \& \;\; \mathbf{Var}(\tilde{g}_t) = \mathcal{O}(\tau_{\mathrm{mix}})$$

Our Technique

- **Wishful thinking**: **cheap** way to trade bias for variance

$$\mathbb{E}[\tilde{g}_t | w_t] = \nabla F(w_t) \pm \mathcal{O}(\sqrt{\tau_{\mathrm{mix}}/T}), \quad \& \quad \mathbf{Var}(\tilde{g}_t) = \mathcal{O}(\tau_{\mathrm{mix}})$$

Our Technique

- **Wishful thinking**: **cheap** way to trade bias for variance

$$\mathbb{E}[\tilde{g}_t|w_t] = \nabla F(w_t) \pm \mathcal{O}(\sqrt{\tau_{\text{mix}}/T}), \quad \& \quad \mathbf{Var}(\tilde{g}_t) = \mathcal{O}(\tau_{\text{mix}})$$

- Use method that implicitly adapt to variance:

AdaGrad:

$$w_{t+1} = \Pi_{\mathcal{K}}(w_t - \eta_t \tilde{g}_t), \qquad \eta_t \propto 1/\sqrt{\sum_{k=1}^{t} \|\tilde{g}_k\|^2}\,.$$

Our Technique

- **Wishful thinking**: **cheap** way to trade bias for variance

$$\mathbb{E}[\tilde{g}_t|w_t] = \nabla F(w_t) \pm \mathcal{O}(\sqrt{\tau_{\mathrm{mix}}/T}), \quad \& \quad \mathbf{Var}(\tilde{g}_t) = \mathcal{O}(\tau_{\mathrm{mix}})$$

- Use method that implicitly adapt to variance:

AdaGrad:

$$w_{t+1} = \Pi_{\mathcal{K}}(w_t - \eta_t \tilde{g}_t), \qquad \eta_t \propto 1/\sqrt{\sum_{k=1}^{t} \|\tilde{g}_k\|^2}\,.$$

AdaGrad Guarantees:

$$\mathrm{err} \propto \tfrac{1}{T}\sum_{t=1}^{T} \mathbf{Bias}(\tilde{g}_t) + \tfrac{1}{T}\sqrt{\sum_{t=1}^{T} \mathbf{Var}(\tilde{g}_t)}$$

Our Technique

- **Wishful thinking**: **cheap** way to trade bias for variance

$$\mathbb{E}[\tilde{g}_t|w_t] = \nabla F(w_t) \pm \mathcal{O}(\sqrt{\tau_{\text{mix}}/T})\,, \quad \& \quad \mathbf{Var}(\tilde{g}_t) = \mathcal{O}(\tau_{\text{mix}})$$

- Use method that implicitly adapt to variance:

AdaGrad:

$$w_{t+1} = \Pi_{\mathcal{K}}(w_t - \eta_t \tilde{g}_t), \qquad \eta_t \propto 1/\sqrt{\sum_{k=1}^{t} \|\tilde{g}_k\|^2}\,.$$

AdaGrad Guarantees: **Optimal!**

$$\text{err} \propto \frac{1}{T}\sum_{t=1}^{T} \mathbf{Bias}(\tilde{g}_t) + \frac{1}{T}\sqrt{\sum_{t=1}^{T} \mathbf{Var}(\tilde{g}_t)} = \mathcal{O}(\sqrt{\tau_{\text{mix}}/T})$$

Trading Bias for Variance

g_t

$$\mathbf{Bias} = \mathcal{O}(1),$$
$$\mathbf{Var} = \mathcal{O}(1)$$

→ **Cheap Bias-Variance Transducer** →

$\tilde{g}_t$

$$\mathbf{Bias} = \mathcal{O}(\sqrt{\tau_{\text{mix}}/T}),$$
$$\mathbf{Var} = \mathcal{O}(\tau_{\text{mix}})$$

Trading Bias for Variance

g_t

$\mathbf{Bias} = \mathcal{O}(1),$
$\mathbf{Var} = \mathcal{O}(1)$

→ **Multi-Level Monte Carlo (MLMC)** →

$\tilde{g}_t$

$\mathbf{Bias} = \mathcal{O}(\sqrt{\tau_{\text{mix}}/T}),$
$\mathbf{Var} = \mathcal{O}(\tau_{\text{mix}})$

Only $\mathcal{O}(\log T)$ **Samples!**

Trading Bias for Variance

g_t

$\mathbf{Bias} = \mathcal{O}(1),$
$\mathbf{Var} = \mathcal{O}(1)$

→ **Multi-Level Monte Carlo (MLMC)** →

$\tilde{g}_t$

$\mathbf{Bias} = \mathcal{O}(\sqrt{\tau_{\text{mix}}/T}),$
$\mathbf{Var} = \mathcal{O}(\tau_{\text{mix}})$

Only $\mathcal{O}(\log T)$ **Samples!**

- **MLMC Applications[1,2,3]:** Distributionally robust optimization, fast projections, conditional stochastic optimization, …

[1] Levy et al., “Large-scale methods for distributionally robust optimization ”, 2020.
[2] Asi et al., “Stochastic bias-reduced gradient methods”, 2021.
[3] Hu et al., “On the bias-variance-cost tradeoff of stochastic optimization”, 2021.

MLMC Gradient Estimator

- Denote: g_t^j =*average gradient over batch of size* 2^j *at* w_t

MLMC Gradient Estimator

- Denote: g_t^j =*average gradient over batch of size* 2^j *at* w_t
- **MLMC Estimator:** Draw $J_t \sim \mathrm{Geom}(1/2)$ and set:

$$\tilde{g}_t^{\mathrm{MLMC}} = g_t^0 + \begin{cases} 2^{J_t}(g_t^{J_t} - g_t^{J_t-1}), & 2^{J_t} \leq T \\ 0, & \text{otherwise} \end{cases}.$$

MLMC Gradient Estimator

- Denote: g_t^j =*average gradient over batch of size* 2^j *at* w_t
- **MLMC Estimator:** Draw $J_t \sim \mathrm{Geom}(1/2)$ and set:

$$\tilde{g}_t^{\mathrm{MLMC}} = g_t^0 + \begin{cases} 2^{J_t}\left(g_t^{J_t} - g_t^{J_t-1}\right), & 2^{J_t} \leq T \\ 0, & \text{otherwise} \end{cases}.$$

<u>For example</u>: $J_t = 3 \rightarrow$ draw $2^{J_t} = 8$ samples $\sim$ MC

$\tilde{\nabla}_1$	$\tilde{\nabla}_2$	$\tilde{\nabla}_3$	$\tilde{\nabla}_4$	$\tilde{\nabla}_5$	$\tilde{\nabla}_6$	$\tilde{\nabla}_7$	$\tilde{\nabla}_8$

MLMC Gradient Estimator

- Denote: g_t^j =*average gradient over batch of size* 2^j *at* w_t
- **MLMC Estimator:** Draw $J_t \sim \mathrm{Geom}(1/2)$ and set:

$$\tilde{g}_t^{\mathrm{MLMC}} = g_t^0 + \begin{cases} 2^{J_t}\left(g_t^{J_t} - g_t^{J_t-1}\right), & 2^{J_t} \le T \\ 0, & \text{otherwise} \end{cases}.$$

<u>For example</u>: $J_t = 3 \rightarrow$ draw $2^{J_t} = 8$ samples $\sim$ MC

$\tilde{\nabla}_1$	$\tilde{\nabla}_2$	$\tilde{\nabla}_3$	$\tilde{\nabla}_4$	$\tilde{\nabla}_5$	$\tilde{\nabla}_6$	$\tilde{\nabla}_7$	$\tilde{\nabla}_8$

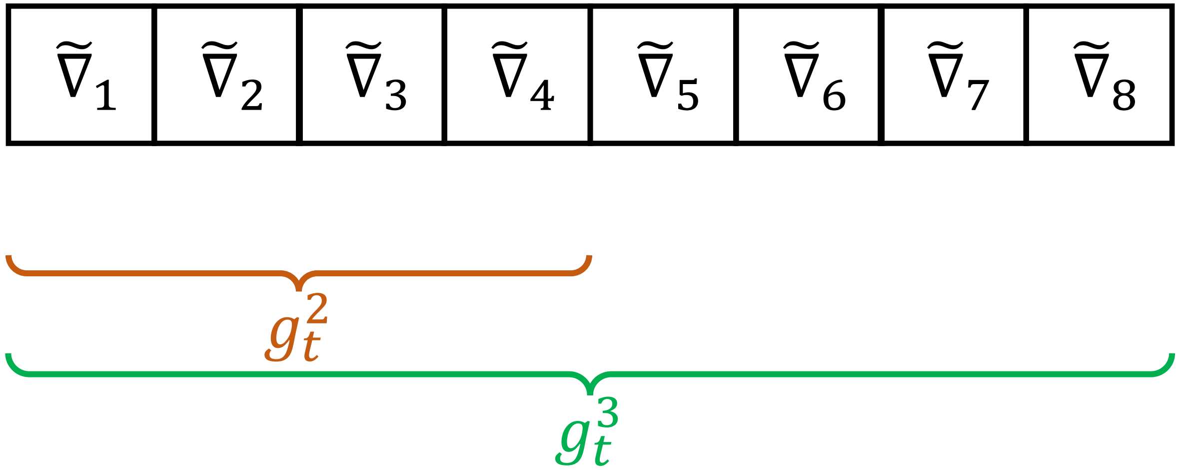
$$g_t^3$$

MLMC Gradient Estimator

- Denote: g_t^j =*average gradient over batch of size* 2^j *at* w_t
- **MLMC Estimator:** Draw $J_t \sim \mathrm{Geom}(1/2)$ and set:

$$\tilde{g}_t^{\mathrm{MLMC}} = g_t^0 + \begin{cases} 2^{J_t}\left(g_t^{J_t} - g_t^{J_t-1}\right), & 2^{J_t} \le T \\ 0, & \text{otherwise} \end{cases}.$$

<u>For example</u>: $J_t = 3 \rightarrow$ draw $2^{J_t} = 8$ samples $\sim$ MC

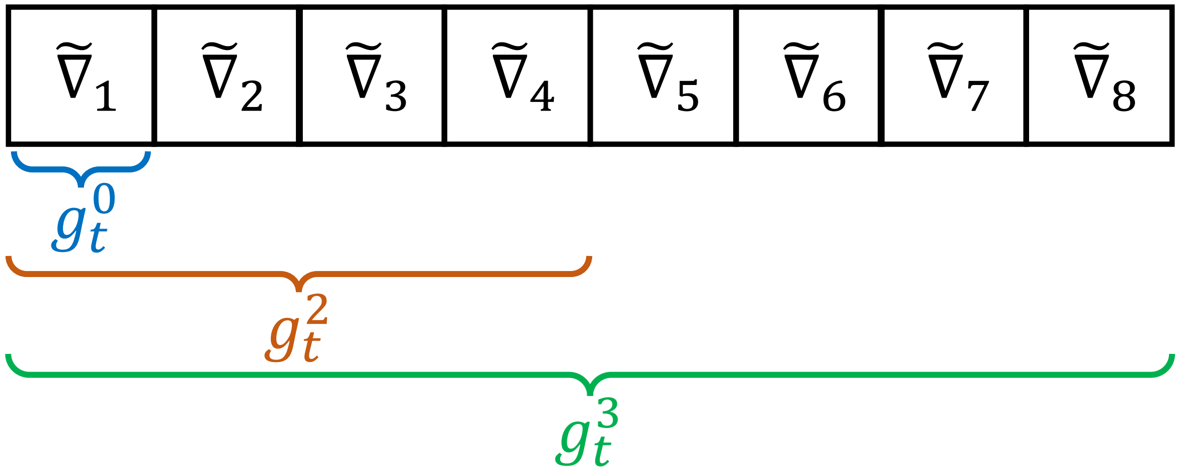


MLMC Gradient Estimator

- Denote: g_t^j =*average gradient over batch of size* 2^j *at* w_t
- **MLMC Estimator:** Draw $J_t \sim \mathrm{Geom}(1/2)$ and set:

$$\tilde{g}_t^{\mathrm{MLMC}} = g_t^0 + \begin{cases} 2^{J_t}\left(g_t^{J_t} - g_t^{J_t-1}\right), & 2^{J_t} \leq T \\ 0, & \text{otherwise} \end{cases}.$$

<u>For example</u>: $J_t = 3 \rightarrow$ draw $2^{J_t} = 8$ samples $\sim$ MC



MLMC Gradient Estimator

$$J_t \sim \mathrm{Geom}(1/2), \quad \tilde{g}_t^{\mathrm{MLMC}} = g_t^0 + \begin{cases} 2^{J_t}(g_t^{J_t} - g_t^{J_t-1}), & 2^{J_t} \le T \\ 0, & \text{otherwise} \end{cases}.$$

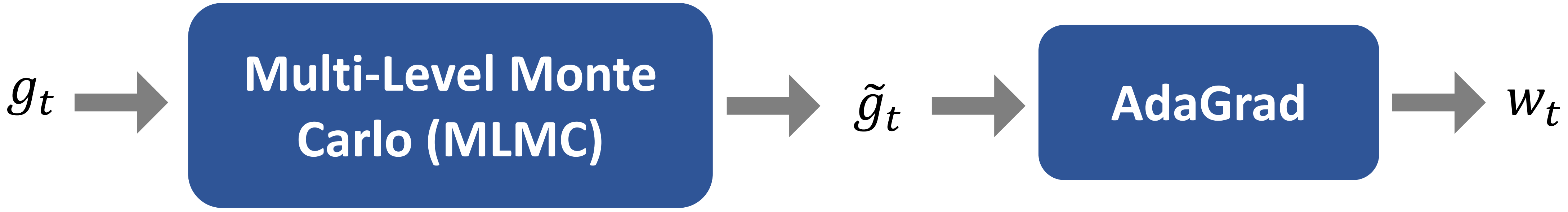
Lemma (informal):

$$\mathbf{Bias}\left(\tilde{g}_t^{\mathrm{MLMC}}\right) = \mathcal{O}\left(\sqrt{\tau_{\mathrm{mix}}/T}\right), \quad \mathbf{Var}\left(\tilde{g}_t^{\mathrm{MLMC}}\right) = \mathcal{O}(\tau_{\mathrm{mix}})$$

$$\mathbb{E}[\mathrm{cost}(\tilde{g}_t^{\mathrm{MLMC}})] = \mathcal{O}(\log T)$$

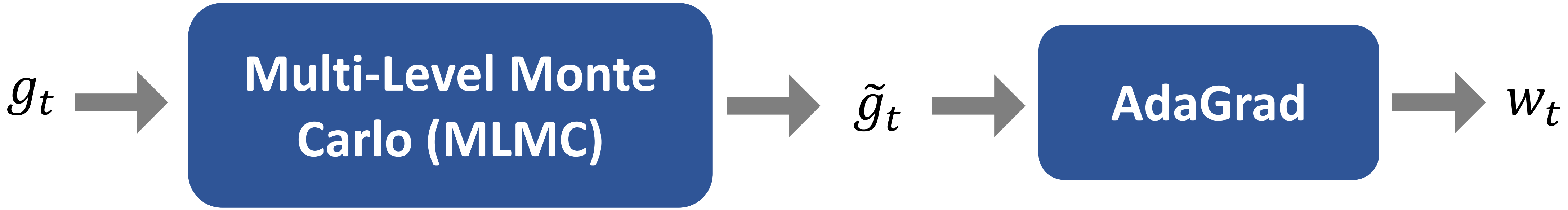
- MLMC shape ensures **telescoping cancellation** → small bias

Overall Scheme – MAG (MLMC-AdaGrad)



Cost: $\mathcal{O}(\log T)$ **Samples**

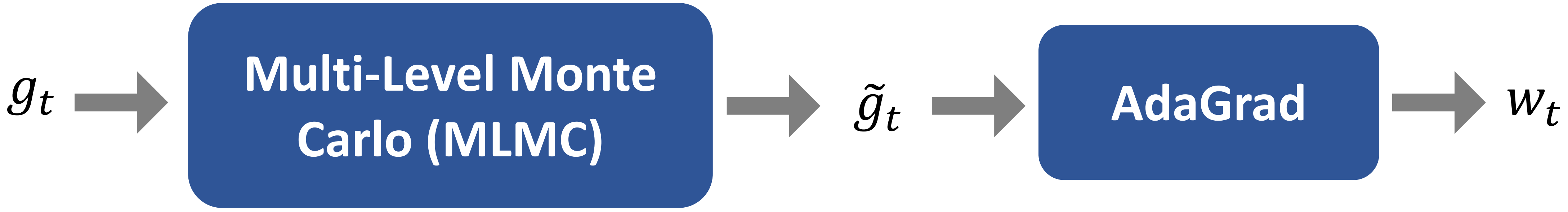
Overall Scheme – MAG (MLMC-AdaGrad)



Cost: $\mathcal{O}(\log T)$ **Samples**

- Main Result: **optimal** $\mathrm{err} = \mathcal{O}(\sqrt{\tau_{\mathrm{mix}}/T})$ in the **convex** setting

Overall Scheme – MAG (MLMC-AdaGrad)



Cost: $\mathcal{O}(\log T)$ **Samples**

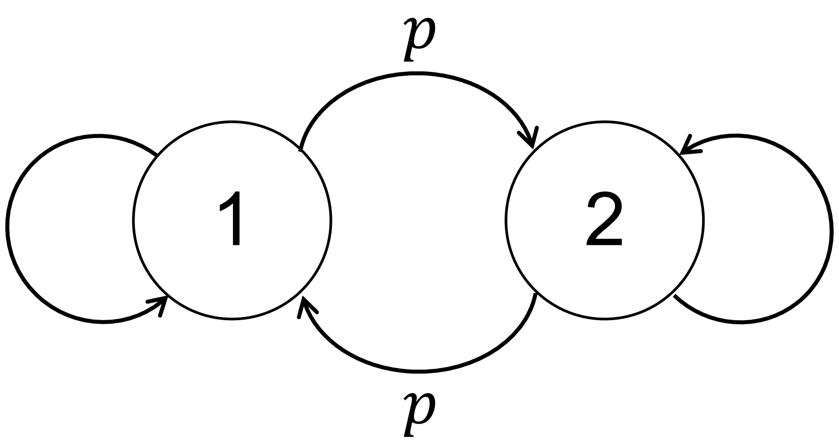
- Main Result: **optimal** $\text{err} = \mathcal{O}(\sqrt{\tau_{\text{mix}}/T})$ in the **convex** setting
- Extensions:
  - Smooth **non-convex** optimization: $\mathbb{E}[\|\nabla F(\tilde{w}_t)\|] = \mathcal{O}((\tau_{\text{mix}}/T)^{1/4})$
  - **TD Learning**: $\text{WMSE} = \mathcal{O}(\sqrt{\tau_{\text{mix}}/T})$ compared to $\mathcal{O}(\tau_{\text{mix}}/\sqrt{T})$[1]

[1] Bhandari et al., "A Finite Time Analysis of Temporal Difference Learning With Linear Function Approximation", 2018.

Experiment – Linear Regression

- For each machine $i \in \{1,2\}$ draw $w_i^* \in \mathbb{R}^d$ and $X_i \in \mathbb{R}^{n \times d}$ from $\mathcal{N}(0, I)$
- Set $y_i = X_i w_i^* + \epsilon$ $(\epsilon \sim \mathcal{N}(0,10^{-3}))$

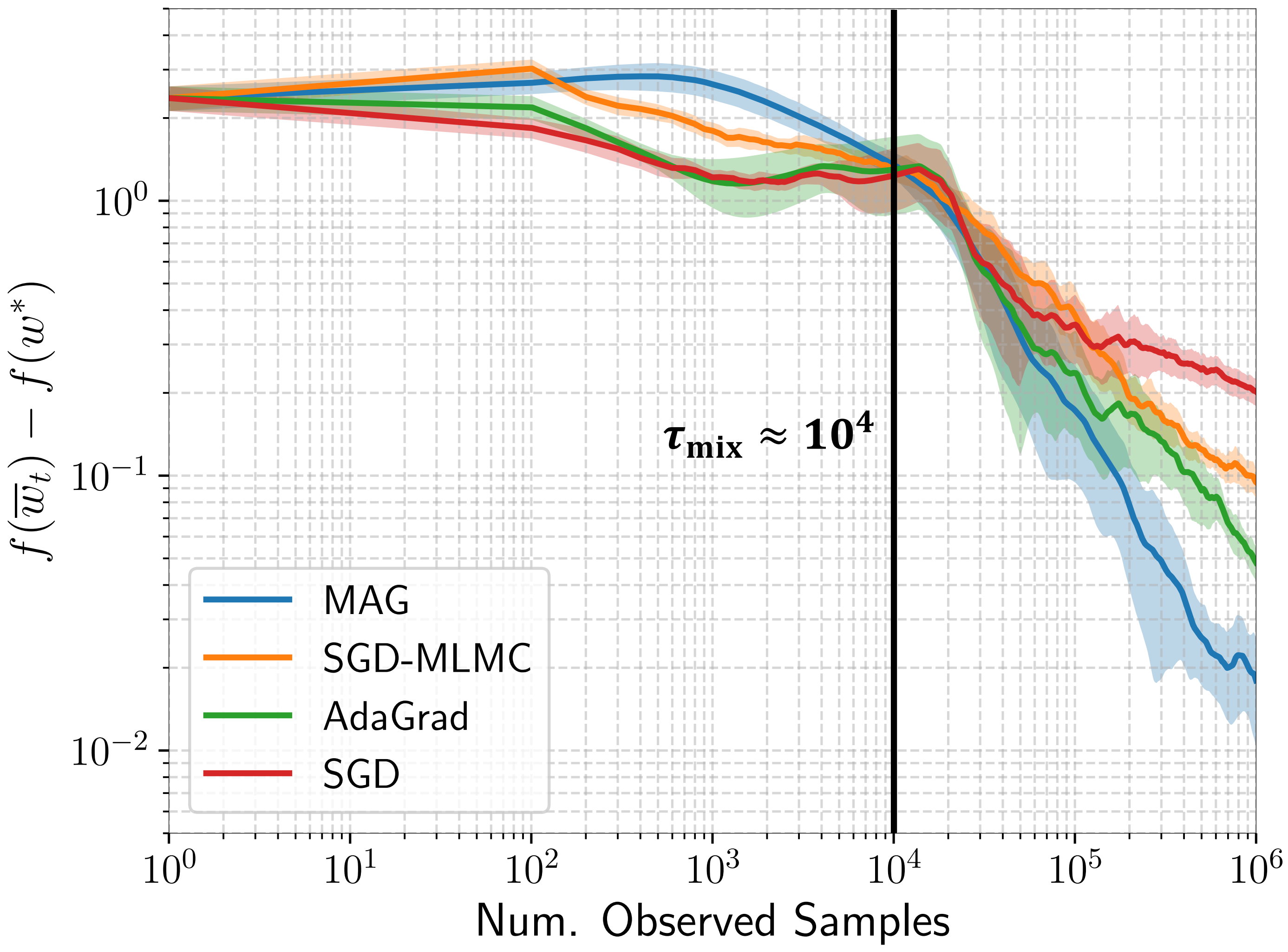
$$\min_{\|w\| \leq r} \frac{1}{4n} \left\| \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} w - \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \right\|^2$$



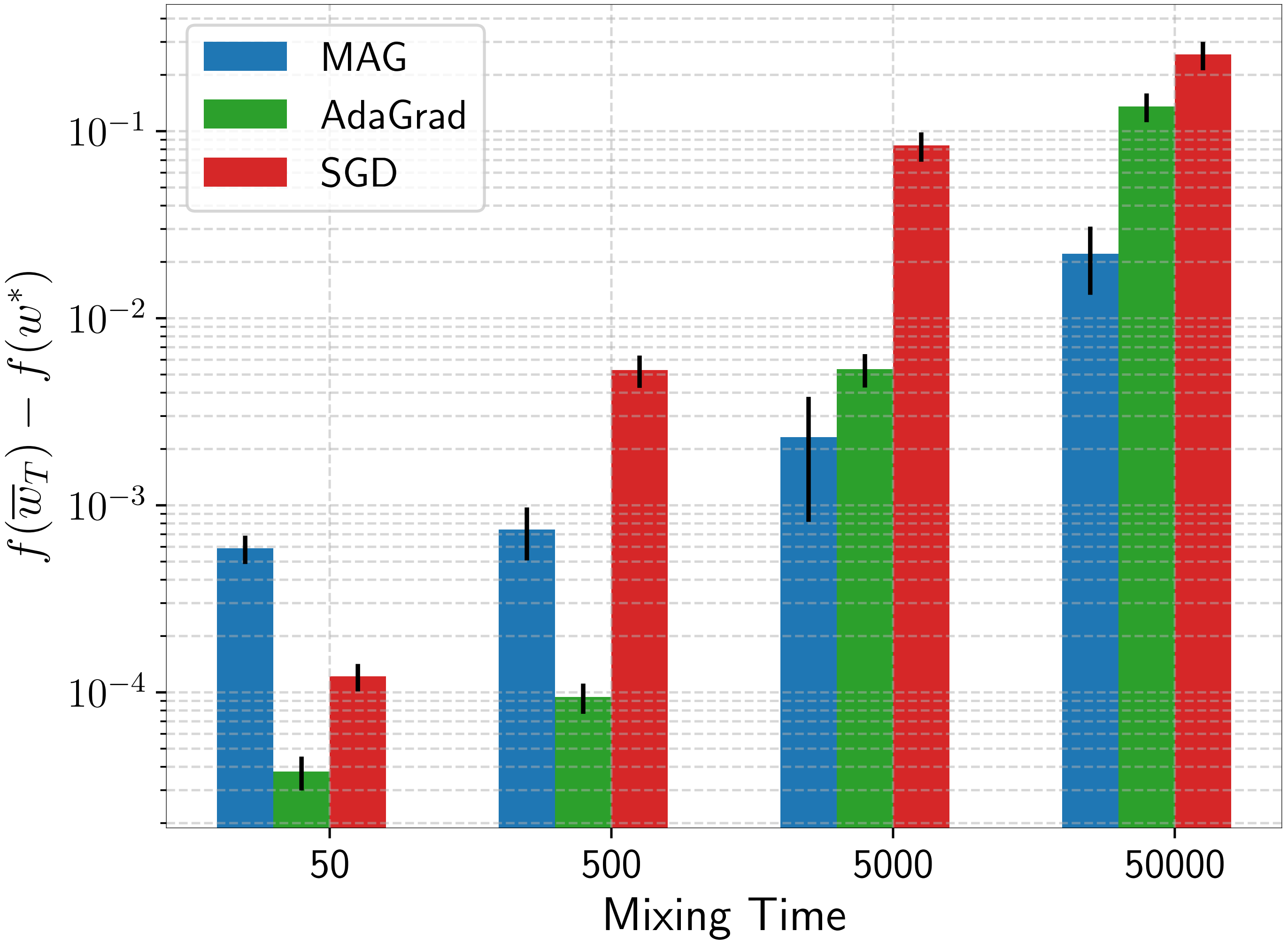
$\tau_{\text{mix}} = \Theta(1/p)$

- $n = 250,\ d = 100,\ p = 10^{-4}$

Experiment – Linear Regression



Experiment – Linear Regression



Take-Home Message

- **MAG**: Optimal SO with Markovian data, adaptive to mixing time
- Combines 2 components:
  - **MLMC** – Cheap bias-variance transducer
  - **AdaGrad** – Adaptive to variance