ETH zürich

Ellipsoidal Trust Region Methods for Neural Nets

L. Adolphs*, J. Kohler*, A. Lucchi

Institute for Machine Learning, ETH Zürich

Data Analytics Lab

Introduction

Introduction

Finite-sum optimization

$$\min_{\mathbf{w}\in\mathbb{R}^{\mathbf{d}}}\left[\mathcal{L}(\mathbf{w}) := \sum_{\mathbf{i}=\mathbf{1}}^{\mathbf{n}} \ell(\mathbf{f}(\mathbf{w},\mathbf{x_i},\mathbf{y_i}))\right]$$

Introduction

Finite-sum optimization

$$\min_{\mathbf{w}\in\mathbb{R}^{\mathbf{d}}} \left[\mathcal{L}(\mathbf{w}) := \sum_{\mathbf{i}=\mathbf{1}}^{\mathbf{n}} \ell(\mathbf{f}(\mathbf{w},\mathbf{x_i},\mathbf{y_i}))\right]$$

How to optimize?

Stochastic Gradient Descent

Introduction

Finite-sum optimization

$$\min_{\mathbf{w}\in\mathbb{R}^{\mathbf{d}}}\left[\mathcal{L}(\mathbf{w}) := \sum_{\mathbf{i}=\mathbf{1}}^{\mathbf{n}} \ell(\mathbf{f}(\mathbf{w},\mathbf{x}_{\mathbf{i}},\mathbf{y}_{\mathbf{i}}))\right]$$

How to optimize?

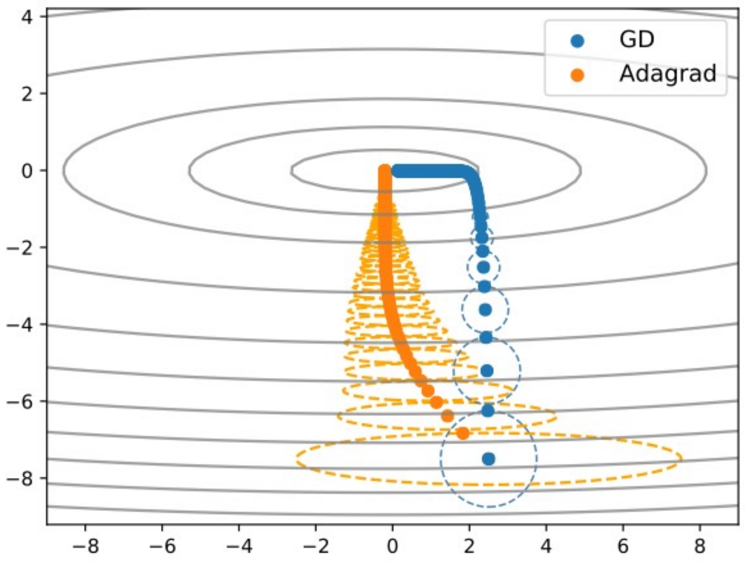
Stochastic Gradient Descent

⚠ inadequate for not well-conditioned functions

Introduction

Finite-sum optimization

$$\min_{\mathbf{w}\in\mathbb{R}^{\mathbf{d}}} \left[\mathcal{L}(\mathbf{w}) := \sum_{\mathbf{i}=\mathbf{1}}^{\mathbf{n}} \ell(\mathbf{f}(\mathbf{w}, \mathbf{x_i}, \mathbf{y_i})) \right]$$



Quadratic objective with condition number $\kappa = 2$

How to optimize?

Stochastic Gradient Descent

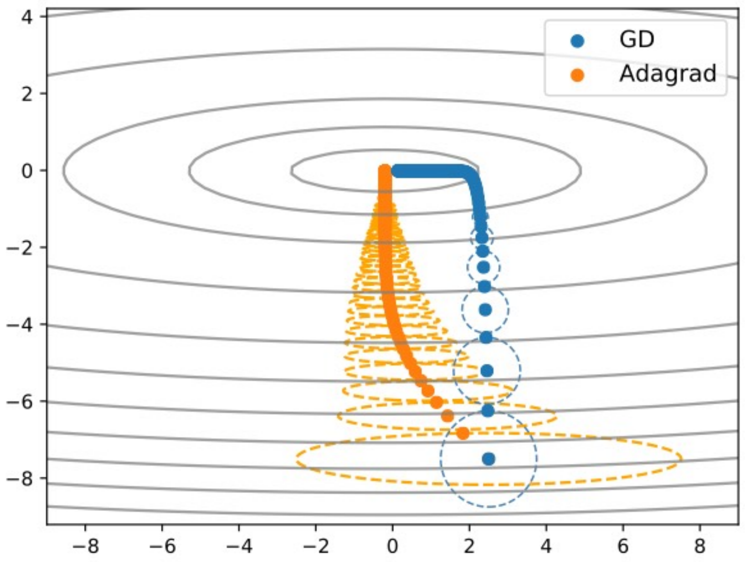
⚠ inadequate for not well-conditioned functions

⇨ adaptive 1st-order methods (e.g. RMSProp, Adagrad)

Introduction

Finite-sum optimization

$$\min_{\mathbf{w}\in\mathbb{R}^{\mathbf{d}}} \left[\mathcal{L}(\mathbf{w}) := \sum_{\mathbf{i}=\mathbf{1}}^{\mathbf{n}} \ell(\mathbf{f}(\mathbf{w}, \mathbf{x_i}, \mathbf{y_i}))\right]$$



Quadratic objective with condition number $\kappa = 2$

How to optimize?

Stochastic Gradient Descent

inadequate for not well-conditioned functions

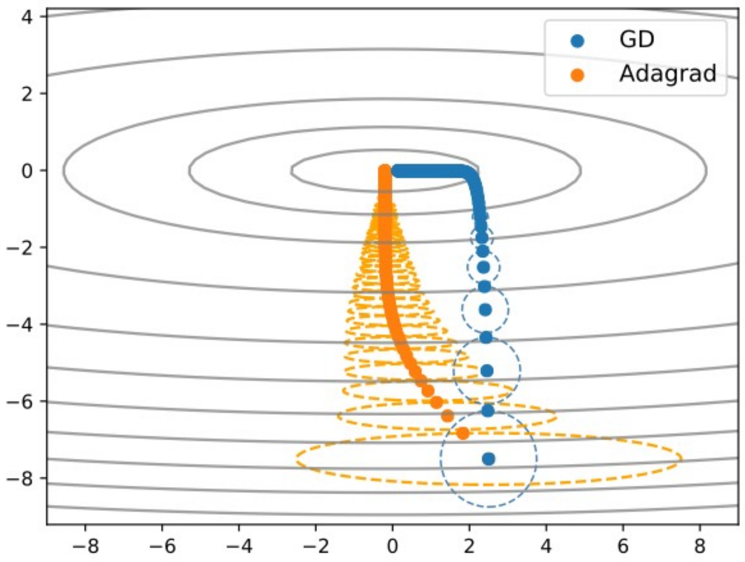
adaptive 1st-order methods (e.g. RMSProp, Adagrad)

Newton's Method

Introduction

Finite-sum optimization

$$\min_{\mathbf{w}\in\mathbb{R}^d}\left[\mathcal{L}(\mathbf{w}) := \sum_{\mathbf{i}=1}^{\mathbf{n}} \ell(\mathbf{f}(\mathbf{w},\mathbf{x_i},\mathbf{y_i}))\right]$$



Quadratic objective with condition number $\kappa = 2$

How to optimize?

Stochastic Gradient Descent

⚠ inadequate for not well-conditioned functions

➔ adaptive 1st-order methods (e.g. RMSProp, Adagrad)

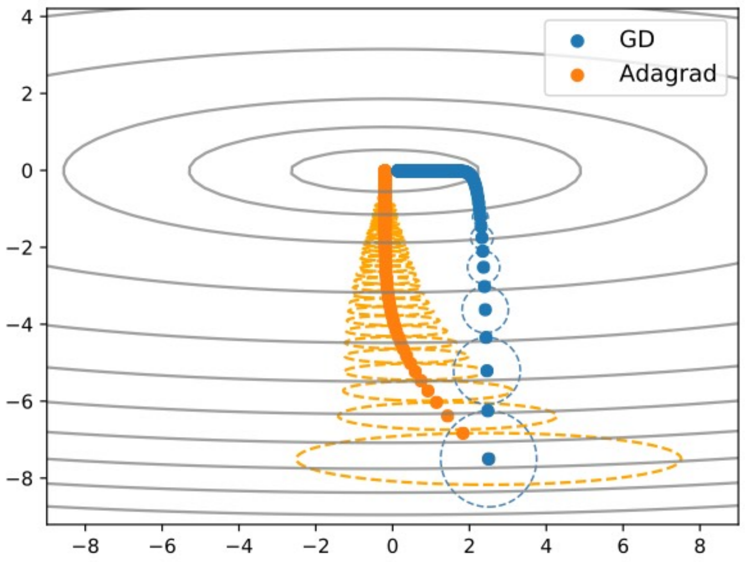
Newton's Method

✓ strong theoretical guarantees

Introduction

Finite-sum optimization

$$\min_{\mathbf{w}\in\mathbb{R}^{\mathbf{d}}} \left[\mathcal{L}(\mathbf{w}) := \sum_{\mathbf{i}=1}^{\mathbf{n}} \ell(\mathbf{f}(\mathbf{w}, \mathbf{x_i}, \mathbf{y_i}))\right]$$



Quadratic objective with condition number $\kappa = 2$

How to optimize?

Stochastic Gradient Descent

inadequate for not well-conditioned functions

adaptive 1st-order methods (e.g. RMSProp, Adagrad)

Newton's Method

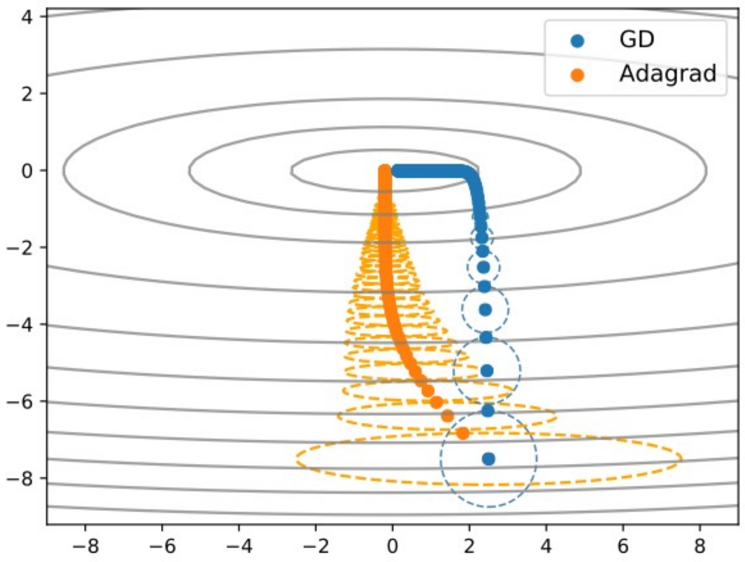
strong theoretical guarantees

expensive

Introduction

Finite-sum optimization

$$\min_{\mathbf{w}\in\mathbb{R}^{\mathbf{d}}} \left[\mathcal{L}(\mathbf{w}) := \sum_{\mathbf{i=1}}^{\mathbf{n}} \ell(\mathbf{f}(\mathbf{w}, \mathbf{x_i}, \mathbf{y_i})) \right]$$



Quadratic objective with condition number $\kappa = 2$

How to optimize?

Stochastic Gradient Descent

⚠ inadequate for not well-conditioned functions

⇨ adaptive 1st-order methods (e.g. RMSProp, Adagrad)

Newton's Method

✓ strong theoretical guarantees

⚠ expensive

⇨ *Stochastic Hessian-Free* Trust Region methods

Alternative View on Adaptive Gradient Descent

Alternative View on Adaptive Gradient Descent

*Gradient Descent methods can be interpreted as **first-order** TR methods…*

$$\mathbf{w_{t+1}} - \mathbf{w_t} = \mathbf{s_t} := -\eta_\mathbf{t} \mathbf{A_t^{-1}} \mathbf{g_t} \qquad \Big| \; \mathbf{g_t} := \nabla \mathcal{L}(\mathbf{w_t})$$

Alternative View on Adaptive Gradient Descent

*Gradient Descent methods can be interpreted as **first-order** TR methods…*

$$\mathbf{w_{t+1}} - \mathbf{w_t} = \mathbf{s_t} := -\eta_\mathbf{t} \mathbf{A_t^{-1}} \mathbf{g_t} \qquad \mid \mathbf{g_t} := \nabla \mathcal{L}(\mathbf{w_t})$$

$$\mathbf{s_t} := \arg\min_{\mathbf{s} \in \mathbb{R}^\mathbf{d}} \left[m_t(\mathbf{s}) = \mathcal{L}(\mathbf{w_t}) + \mathbf{s}^\intercal \mathbf{g_t} \right]$$

$$\text{s.t.} \quad \|\mathbf{s}\|_{\color{red}{\mathbf{A_t}}} \leq \eta_\mathbf{t} \|\mathbf{g_t}\|_{\mathbf{A_t^{-1}}}$$

Alternative View on Adaptive Gradient Descent

Gradient Descent methods can be interpreted as ***first-order*** *TR methods…*

$$\mathbf{w_{t+1}} - \mathbf{w_t} = \mathbf{s_t} := -\eta_\mathbf{t} \mathbf{A_t^{-1}} \mathbf{g_t} \qquad \mid \mathbf{g_t} := \nabla \mathcal{L}(\mathbf{w_t})$$

$$\mathbf{s_t} := \arg\min_{\mathbf{s} \in \mathbb{R}^\mathbf{d}} \left[m_t(\mathbf{s}) = \mathcal{L}(\mathbf{w_t}) + \mathbf{s}^\intercal \mathbf{g_t} \right]$$

$$\text{s.t.} \quad \|\mathbf{s}\|_{\mathbf{A_t}} \leq \eta_\mathbf{t} \|\mathbf{g_t}\|_{\mathbf{A_t^{-1}}}$$

…with the ‘preconditioning’ matrix determining the shape of the constraint.

Alternative View on Adaptive Gradient Descent

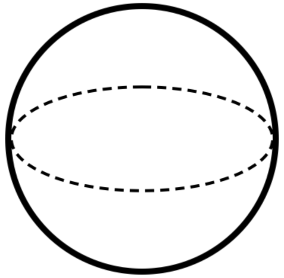
Gradient Descent methods can be interpreted as ***first-order*** *TR methods…*

$$\mathbf{w_{t+1}} - \mathbf{w_t} = \mathbf{s_t} := -\eta_\mathbf{t} \mathbf{A_t^{-1}} \mathbf{g_t} \qquad \mid \mathbf{g_t} := \nabla \mathcal{L}(\mathbf{w_t})$$

$$\mathbf{s_t} := \arg\min_{\mathbf{s} \in \mathbb{R}^\mathbf{d}} \left[m_t(\mathbf{s}) = \mathcal{L}(\mathbf{w_t}) + \mathbf{s}^\intercal \mathbf{g_t} \right]$$

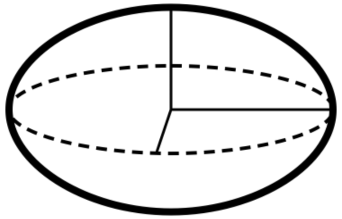
$$\text{s.t.} \quad \|\mathbf{s}\|_{\mathbf{A_t}} \leq \eta_\mathbf{t} \|\mathbf{g_t}\|_{\mathbf{A_t^{-1}}}$$

…with the ‘preconditioning’ matrix determining the shape of the constraint.



Gradient Descent

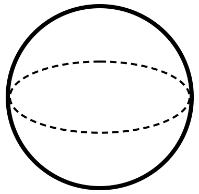
$$\mathbf{A_t} = \mathbb{I}$$



RMSProp
Adagrad
Adam

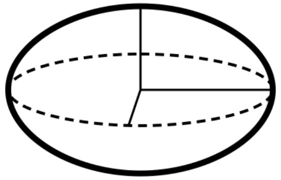
Second-order Trust Region Methods

$$\min_{\mathbf{s}\in\mathbb{R}^{\mathbf{d}}} \left[m_t(\mathbf{s}) := \mathcal{L}(\mathbf{w_t}) + \mathbf{g_t}^{\intercal}\mathbf{s} + \frac{\mathbf{1}}{\mathbf{2}}\mathbf{s}^{\intercal}\underbrace{\mathbf{B_t}}_{\nabla^2\mathcal{L}(\mathbf{w_t})}\mathbf{s} \right] \quad \text{s.t. } \|\mathbf{s}\| \leq \mathbf{\Delta_t}$$



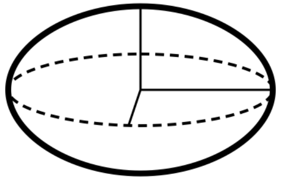
Second-order Trust Region Methods

$$\min_{\mathbf{s}\in\mathbb{R}^{\mathbf{d}}} \left[m_t(\mathbf{s}) := \mathcal{L}(\mathbf{w_t}) + \mathbf{g_t^\intercal s} + \frac{\mathbf{1}}{\mathbf{2}}\mathbf{s^\intercal B_t s} \right] \quad \text{s.t. } \|\mathbf{s}\|_{\mathbf{A_t}} \leq \mathbf{\Delta_t}$$



Second-order Trust Region Methods

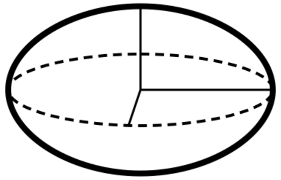
$$\min_{\mathbf{s}\in\mathbb{R}^{\mathbf{d}}} \left[m_t(\mathbf{s}) := \mathcal{L}(\mathbf{w_t}) + \mathbf{g_t}^\intercal \mathbf{s} + \frac{\mathbf{1}}{\mathbf{2}} \mathbf{s}^\intercal \mathbf{B_t} \mathbf{s} \right] \quad \text{s.t. } \|\mathbf{s}\|_{\mathbf{A_t}} \leq \mathbf{\Delta_t}$$



Why Ellipsoids?

Second-order Trust Region Methods

$$\min_{\mathbf{s}\in\mathbb{R}^{\mathbf{d}}} \left[m_t(\mathbf{s}) := \mathcal{L}(\mathbf{w_t}) + \mathbf{g_t^\intercal s} + \frac{\mathbf{1}}{\mathbf{2}}\mathbf{s^\intercal B_t s} \right] \quad \text{s.t. } \|\mathbf{s}\|_{\color{red}{\mathbf{A_t}}} \leq \mathbf{\Delta_t}$$

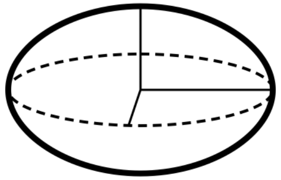


Why Ellipsoids?

Many sources for ill-conditioning in Neural Nets

Second-order Trust Region Methods

$$\min_{\mathbf{s}\in\mathbb{R}^{\mathbf{d}}} \left[m_t(\mathbf{s}) := \mathcal{L}(\mathbf{w_t}) + \mathbf{g_t^\intercal s} + \frac{\mathbf{1}}{\mathbf{2}}\mathbf{s^\intercal B_t s} \right] \quad \text{s.t. } \|\mathbf{s}\|_{\color{red}{\mathbf{A_t}}} \leq \mathbf{\Delta_t}$$



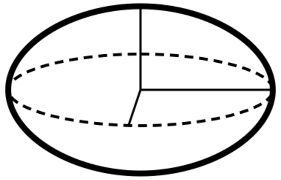
Why Ellipsoids?

Many sources for ill-conditioning in Neural Nets

More flexibility to reshape constraint set to loss landscape

Second-order Trust Region Methods

$$\min_{\mathbf{s}\in\mathbb{R}^{\mathbf{d}}} \left[m_t(\mathbf{s}) := \mathcal{L}(\mathbf{w_t}) + \mathbf{g_t}^{\intercal}\mathbf{s} + \frac{\mathbf{1}}{\mathbf{2}}\mathbf{s}^{\intercal}\mathbf{B_t}\mathbf{s} \right] \quad \text{s.t. } \|\mathbf{s}\|_{\mathbf{A_t}} \leq \mathbf{\Delta_t}$$



Why Ellipsoids?

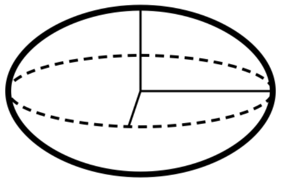
Many sources for ill-conditioning in Neural Nets

More flexibility to reshape constraint set to loss landscape

What Ellipsoid?

Second-order Trust Region Methods

$$\min_{\mathbf{s}\in\mathbb{R}^{\mathbf{d}}} \left[m_t(\mathbf{s}) := \mathcal{L}(\mathbf{w_t}) + \mathbf{g_t}^{\intercal}\mathbf{s} + \frac{\mathbf{1}}{\mathbf{2}}\mathbf{s}^{\intercal}\mathbf{B_t}\mathbf{s} \right] \quad \text{s.t. } \|\mathbf{s}\|_{\mathbf{A_t}} \leq \mathbf{\Delta_t}$$



Why Ellipsoids?

Many sources for ill-conditioning in Neural Nets

More flexibility to reshape constraint set to loss landscape

What Ellipsoid?

Preconditioners from most-common first-order methods, that fulfill *Uniform Equivalence*

RMSProp, Adam, …

Algorithm

Algorithm 1 Stochastic Ellipsoidal Trust Region Method

1: **Input:** $\mathbf{w}_0 \in \mathbb{R}^d, \gamma_1, \gamma_2 > 1, 1 > \eta_2 > \eta_1 > 0, \Delta_0 > 0, T \geq 1, |\mathcal{S}_0|, \mu \geq 1, \epsilon > 0$

2: **for** $t = 0, 1, \ldots,$ until convergence **do**

3: Sample $\mathcal{L}_t$, $\mathbf{g}_t$ and $\mathbf{B}_t$ with batch sizes $|\mathcal{S}_{\mathcal{L},t}|, |\mathcal{S}_{\mathbf{g},t}|, |\mathcal{S}_{\mathbf{B},t}|$

4: Compute preconditioner $\mathbf{A}_t$

5: Obtain $\mathbf{s}_t$ by solving $m_t(\mathbf{s}_t)$

6: Compute actual over predicted decrease on batch

$$\rho_{\mathcal{S},t} = \frac{\mathcal{L}_{\mathcal{S}}(\mathbf{w}_t) - \mathcal{L}_{\mathcal{S}}(\mathbf{w}_t + \mathbf{s}_t)}{m_t(\mathbf{0}) - m_t(\mathbf{s}_t)}$$

7: Set

$$\Delta_{t+1} = \begin{cases} \gamma_1 \Delta_t & \text{if } \rho_{\mathcal{S},t} > \eta_2 \text{ (very successful)} \\ \Delta_t & \text{if } \eta_2 \geq \rho_{\mathcal{S},t} \geq \eta_1 \text{ (successful)} \\ \Delta_t / \gamma_2 & \text{if } \rho_{\mathcal{S},t} < \eta_1 \text{ (unsuccessful)} \end{cases}, \quad \mathbf{w}_{t+1} = \begin{cases} \mathbf{w}_t + \mathbf{s}_t & \text{if } \rho_{\mathcal{S},t} \geq \eta_1 \\ \mathbf{w}_t & \text{otherwise} \end{cases}$$

8: **end for**

Algorithm

Algorithm 1 Stochastic Ellipsoidal Trust Region Method

Input: $\mathbf{w}_0 \in \mathbb{R}^d, \gamma_1, \gamma_2 > 1, 1 > \eta_2 > \eta_1 > 0, \Delta_0 > 0, T \geq 1, |\mathcal{S}_0|, \mu \geq 1, \epsilon > 0$

for $t = 0, 1, \dots,$ until convergence **do**

  Sample $\mathcal{L}_t$, $\mathbf{g}_t$ and $\mathbf{B}_t$ with batch sizes $|\mathcal{S}_{\mathcal{L},t}|, |\mathcal{S}_{\mathbf{g},t}|, |\mathcal{S}_{\mathbf{B},t}|$

  Compute preconditioner $\mathbf{A}_t$

  Obtain $\mathbf{s}_t$ by solving $m_t(\mathbf{s}_t)$

  Compute actual over predicted decrease on batch

$$\rho_{\mathcal{S},t} = \frac{\mathcal{L}_{\mathcal{S}}(\mathbf{w}_t) - \mathcal{L}_{\mathcal{S}}(\mathbf{w}_t + \mathbf{s}_t)}{m_t(\mathbf{0}) - m_t(\mathbf{s}_t)}$$

  Set

$$\Delta_{t+1} = \begin{cases} \gamma_1 \Delta_t & \text{if } \rho_{\mathcal{S},t} > \eta_2 \text{ (very successful)} \\ \Delta_t & \text{if } \eta_2 \geq \rho_{\mathcal{S},t} \geq \eta_1 \text{ (successful)} \\ \Delta_t / \gamma_2 & \text{if } \rho_{\mathcal{S},t} < \eta_1 \text{ (unsuccessful)} \end{cases}, \quad \mathbf{w}_{t+1} = \begin{cases} \mathbf{w}_t + \mathbf{s}_t & \text{if } \rho_{\mathcal{S},t} \geq \eta_1 \\ \mathbf{w}_t & \text{otherwise} \end{cases}$$

end for

Loss, Gradient, approx. Hessian for batch

Algorithm

Algorithm 1 Stochastic Ellipsoidal Trust Region Method

Input: $\mathbf{w}_0 \in \mathbb{R}^d, \gamma_1, \gamma_2 > 1, 1 > \eta_2 > \eta_1 > 0, \Delta_0 > 0, T \geq 1, |\mathcal{S}_0|, \mu \geq 1, \epsilon > 0$

for $t = 0, 1, \ldots,$ until convergence **do**

  Sample $\mathcal{L}_t$, $\mathbf{g}_t$ and $\mathbf{B}_t$ with batch sizes $|\mathcal{S}_{\mathcal{L},t}|, |\mathcal{S}_{\mathbf{g},t}|, |\mathcal{S}_{\mathbf{B},t}|$

  Compute preconditioner $\mathbf{A}_t$

  Obtain $\mathbf{s}_t$ by solving $m_t(\mathbf{s}_t)$

  Compute actual over predicted decrease on batch

$$\rho_{\mathcal{S},t} = \frac{\mathcal{L}_{\mathcal{S}}(\mathbf{w}_t) - \mathcal{L}_{\mathcal{S}}(\mathbf{w}_t + \mathbf{s}_t)}{m_t(\mathbf{0}) - m_t(\mathbf{s}_t)}$$

  Set

$$\Delta_{t+1} = \begin{cases} \gamma_1 \Delta_t & \text{if } \rho_{\mathcal{S},t} > \eta_2 \text{ (very successful)} \\ \Delta_t & \text{if } \eta_2 \geq \rho_{\mathcal{S},t} \geq \eta_1 \text{ (successful)} \\ \Delta_t/\gamma_2 & \text{if } \rho_{\mathcal{S},t} < \eta_1 \text{ (unsuccessful)} \end{cases}, \quad \mathbf{w}_{t+1} = \begin{cases} \mathbf{w}_t + \mathbf{s}_t & \text{if } \rho_{\mathcal{S},t} \geq \eta_1 \\ \mathbf{w}_t & \text{otherwise} \end{cases}$$

end for

Algorithm

Algorithm 1 Stochastic Ellipsoidal Trust Region Method

1: **Input:** $\mathbf{w}_0 \in \mathbb{R}^d$, $\gamma_1, \gamma_2 > 1, 1 > \eta_2 > \eta_1 > 0$, $\Delta_0 > 0$, $T \geq 1$, $|\mathcal{S}_0|, \mu \geq 1, \epsilon > 0$

2: **for** $t = 0, 1, \ldots,$ until convergence **do**

3: Sample $\mathcal{L}_t$, $\mathbf{g}_t$ and $\mathbf{B}_t$ with batch sizes $|\mathcal{S}_{\mathcal{L},t}|, |\mathcal{S}_{\mathbf{g},t}|, |\mathcal{S}_{\mathbf{B},t}|$

4: Compute preconditioner $\mathbf{A}_t$

5: Obtain $\mathbf{s}_t$ by solving $m_t(\mathbf{s}_t)$

$$\min_{\mathbf{s}\in\mathbb{R}^d} \mathcal{L}(\mathbf{w}_t) + \mathbf{g}_t^\intercal \mathbf{s} + \frac{1}{2}\mathbf{s}^\intercal \mathbf{B}_t \mathbf{s} \quad \text{s.t. } \|\mathbf{s}\|_{\mathbf{A}_t} \leq \Delta_t$$

6: Compute actual over predicted decrease on batch

$$\rho_{\mathcal{S},t} = \frac{\mathcal{L}_{\mathcal{S}}(\mathbf{w}_t) - \mathcal{L}_{\mathcal{S}}(\mathbf{w}_t + \mathbf{s}_t)}{m_t(\mathbf{0}) - m_t(\mathbf{s}_t)}$$

7: Set

$$\Delta_{t+1} = \begin{cases} \gamma_1 \Delta_t & \text{if } \rho_{\mathcal{S},t} > \eta_2 \text{ (very successful)} \\ \Delta_t & \text{if } \eta_2 \geq \rho_{\mathcal{S},t} \geq \eta_1 \text{ (successful)} \\ \Delta_t/\gamma_2 & \text{if } \rho_{\mathcal{S},t} < \eta_1 \text{ (unsuccessful)} \end{cases}, \quad \mathbf{w}_{t+1} = \begin{cases} \mathbf{w}_t + \mathbf{s}_t & \text{if } \rho_{\mathcal{S},t} \geq \eta_1 \\ \mathbf{w}_t & \text{otherwise} \end{cases}$$

8: **end for**

Algorithm

Algorithm 1 Stochastic Ellipsoidal Trust Region Method

Input: $\mathbf{w}_0 \in \mathbb{R}^d, \gamma_1, \gamma_2 > 1, 1 > \eta_2 > \eta_1 > 0, \Delta_0 > 0, T \geq 1, |\mathcal{S}_0|, \mu \geq 1, \epsilon > 0$

for $t = 0, 1, \ldots,$ until convergence **do**

  Sample $\mathcal{L}_t$, $\mathbf{g}_t$ and $\mathbf{B}_t$ with batch sizes $|\mathcal{S}_{\mathcal{L},t}|, |\mathcal{S}_{\mathbf{g},t}|, |\mathcal{S}_{\mathbf{B},t}|$

  Compute preconditioner $\mathbf{A}_t$

  Obtain $\mathbf{s}_t$ by solving $m_t(\mathbf{s}_t)$

  Compute actual over predicted decrease on batch

$$\rho_{\mathcal{S},t} = \frac{\mathcal{L}_{\mathcal{S}}(\mathbf{w}_t) - \mathcal{L}_{\mathcal{S}}(\mathbf{w}_t + \mathbf{s}_t)}{m_t(\mathbf{0}) - m_t(\mathbf{s}_t)}$$

Real loss

Model loss

  Set

$$\Delta_{t+1} = \begin{cases} \gamma_1 \Delta_t & \text{if } \rho_{\mathcal{S},t} > \eta_2 \text{ (very successful)} \\ \Delta_t & \text{if } \eta_2 \geq \rho_{\mathcal{S},t} \geq \eta_1 \text{ (successful)} \\ \Delta_t / \gamma_2 & \text{if } \rho_{\mathcal{S},t} < \eta_1 \text{ (unsuccessful)} \end{cases}, \quad \mathbf{w}_{t+1} = \begin{cases} \mathbf{w}_t + \mathbf{s}_t & \text{if } \rho_{\mathcal{S},t} \geq \eta_1 \\ \mathbf{w}_t & \text{otherwise} \end{cases}$$

end for

Algorithm

Algorithm 1 Stochastic Ellipsoidal Trust Region Method

1: **Input:** $\mathbf{w}_0 \in \mathbb{R}^d$, $\gamma_1, \gamma_2 > 1, 1 > \eta_2 > \eta_1 > 0$, $\Delta_0 > 0$, $T \geq 1$, $|\mathcal{S}_0|, \mu \geq 1, \epsilon > 0$
2: **for** $t = 0, 1, \ldots,$ until convergence **do**
3: Sample $\mathcal{L}_t$, $\mathbf{g}_t$ and $\mathbf{B}_t$ with batch sizes $|\mathcal{S}_{\mathcal{L},t}|, |\mathcal{S}_{\mathbf{g},t}|, |\mathcal{S}_{\mathbf{B},t}|$
4: Compute preconditioner $\mathbf{A}_t$
5: Obtain $\mathbf{s}_t$ by solving $m_t(\mathbf{s}_t)$
6: Compute actual over predicted decrease on batch

$$\rho_{\mathcal{S},t} = \frac{\mathcal{L}_{\mathcal{S}}(\mathbf{w}_t) - \mathcal{L}_{\mathcal{S}}(\mathbf{w}_t + \mathbf{s}_t)}{m_t(\mathbf{0}) - m_t(\mathbf{s}_t)}$$

7: Set

Adjust TR radius

$$\Delta_{t+1} = \begin{cases} \gamma_1 \Delta_t & \text{if } \rho_{\mathcal{S},t} > \eta_2 \text{ (very successful)} \\ \Delta_t & \text{if } \eta_2 \geq \rho_{\mathcal{S},t} \geq \eta_1 \text{ (successful)} \\ \Delta_t/\gamma_2 & \text{if } \rho_{\mathcal{S},t} < \eta_1 \text{ (unsuccessful)} \end{cases}, \quad \mathbf{w}_{t+1} = \begin{cases} \mathbf{w}_t + \mathbf{s}_t & \text{if } \rho_{\mathcal{S},t} \geq \eta_1 \\ \mathbf{w}_t & \text{otherwise} \end{cases}$$

8: **end for**

Algorithm

Algorithm 1 Stochastic Ellipsoidal Trust Region Method

1: **Input:** $\mathbf{w}_0 \in \mathbb{R}^d, \gamma_1, \gamma_2 > 1, 1 > \eta_2 > \eta_1 > 0, \Delta_0 > 0, T \geq 1, |\mathcal{S}_0|, \mu \geq 1, \epsilon > 0$

2: **for** $t = 0, 1, \ldots,$ until convergence **do**

3: Sample $\mathcal{L}_t$, $\mathbf{g}_t$ and $\mathbf{B}_t$ with batch sizes $|\mathcal{S}_{\mathcal{L},t}|, |\mathcal{S}_{\mathbf{g},t}|, |\mathcal{S}_{\mathbf{B},t}|$

4: Compute preconditioner $\mathbf{A}_t$

5: Obtain $\mathbf{s}_t$ by solving $m_t(\mathbf{s}_t)$

6: Compute actual over predicted decrease on batch

$$\rho_{\mathcal{S},t} = \frac{\mathcal{L}_{\mathcal{S}}(\mathbf{w}_t) - \mathcal{L}_{\mathcal{S}}(\mathbf{w}_t + \mathbf{s}_t)}{m_t(\mathbf{0}) - m_t(\mathbf{s}_t)}$$

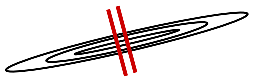
7: Set

Accept only successful updates

$$\Delta_{t+1} = \begin{cases} \gamma_1 \Delta_t & \text{if } \rho_{\mathcal{S},t} > \eta_2 \text{ (very successful)} \\ \Delta_t & \text{if } \eta_2 \geq \rho_{\mathcal{S},t} \geq \eta_1 \text{ (successful)} \\ \Delta_t/\gamma_2 & \text{if } \rho_{\mathcal{S},t} < \eta_1 \text{ (unsuccessful)} \end{cases}, \quad \mathbf{w}_{t+1} = \begin{cases} \mathbf{w}_t + \mathbf{s}_t & \text{if } \rho_{\mathcal{S},t} \geq \eta_1 \\ \mathbf{w}_t & \text{otherwise} \end{cases}$$

8: **end for**

Convergence guarantees



$$\exists \mu \geq 1, \quad \frac{1}{\mu}\|\mathbf{w}\|_{\mathbf{A}_t} \leq \|\mathbf{w}\|_2 \leq \mu\|\mathbf{w}\|_{\mathbf{A}_t}, \ \forall t = 1, 2, \dots$$

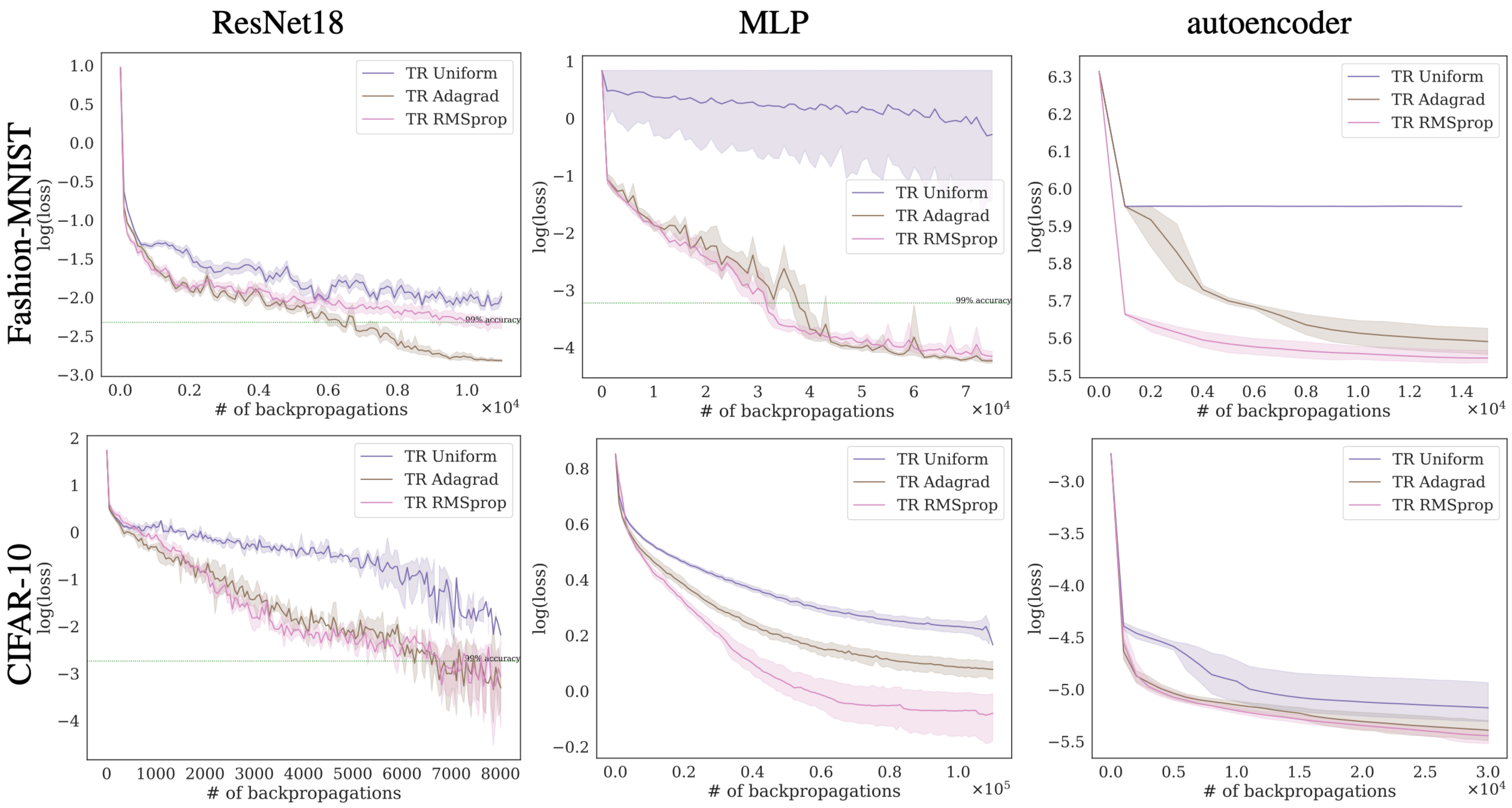
(Conn et al. 2000) prove convergence assuming uniform equivalent norms

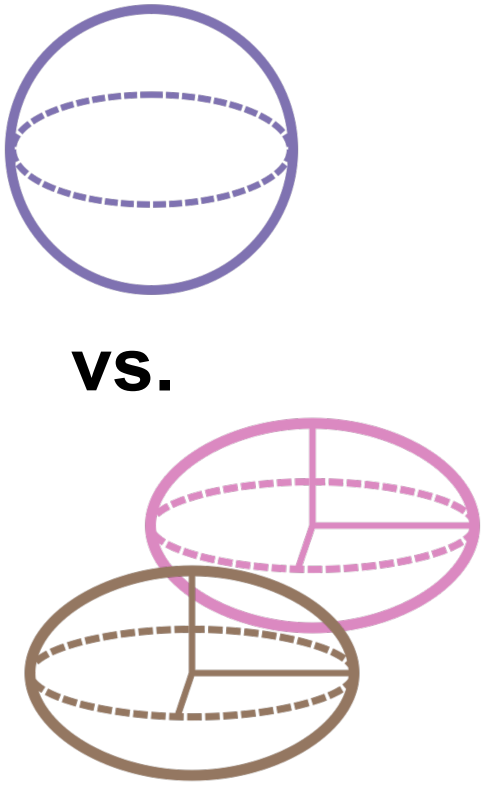
We extend this result to prove a convergence **rate** for the semi-stochastic TR framework of (Yao et al. 2018) with ellipsoidal constraints.

- full function - but inexact derivative information

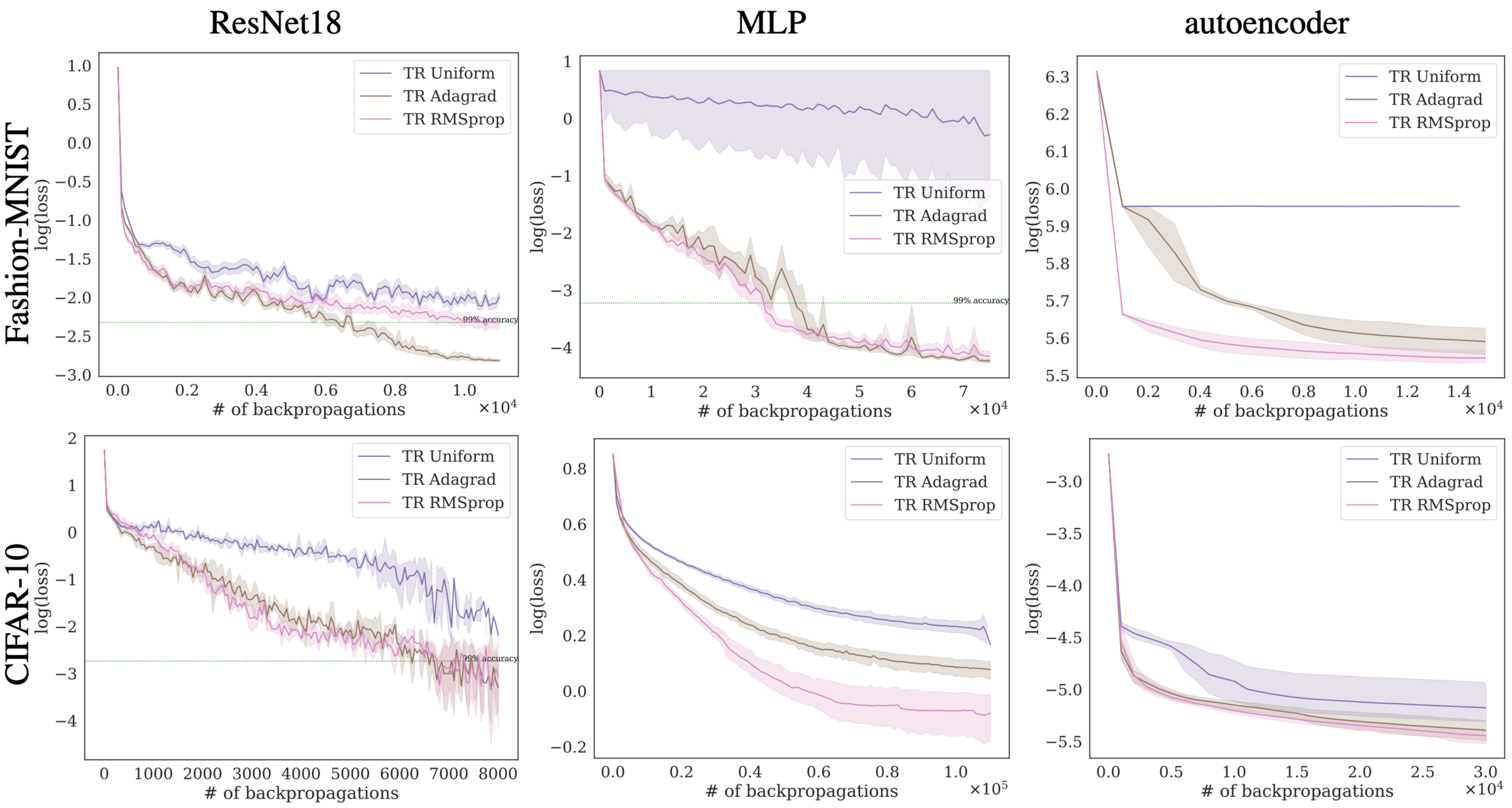
Theorem 2 (Convergence rate of Algorithm 1)**.** *Assume that $\mathcal{L}(\mathbf{w})$ is second-order smooth with Lipschitz constants L_g and L_H. Furthermore, let Assumption 1 and 2 hold. Then Algorithm 1 finds an $\mathcal{O}(\epsilon_g, \epsilon_H)$ first- and second-order stationary point in at most $\mathcal{O}\left(\max\left\{\epsilon_g^{-2}\epsilon_H^{-1}, \epsilon_H^{-3}\right\}\right)$ iterations.*

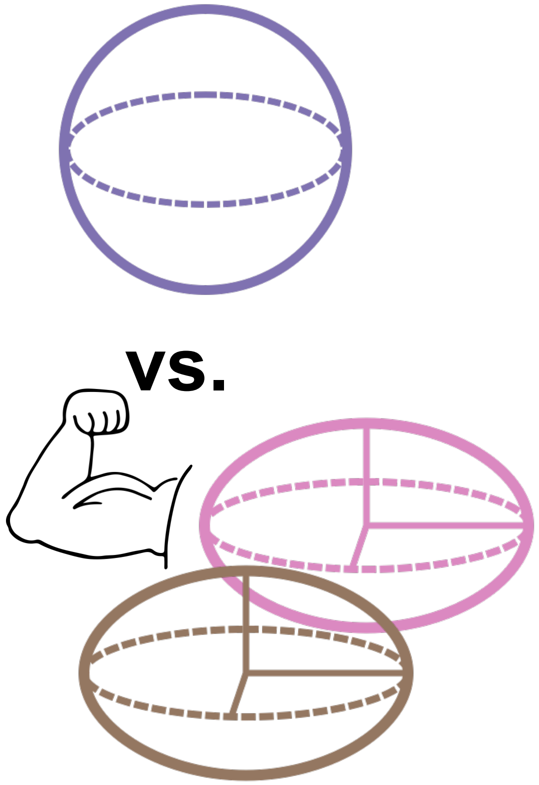
Experiments



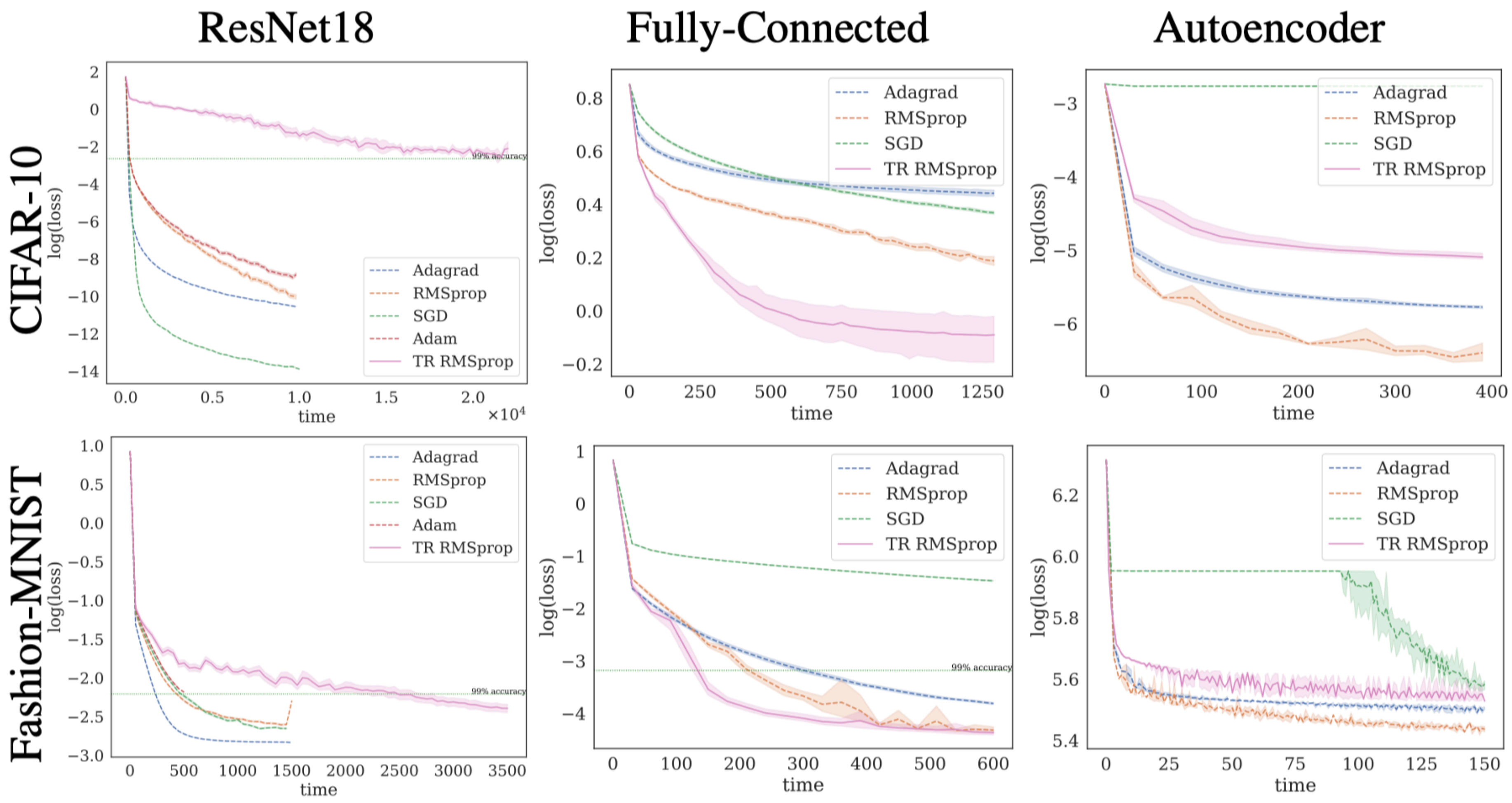


Experiments





Experiments



Comparison against 1st-order methods in wall-clock-time

Thank you!
Any Questions?