HiGrad: Statistical Inference for Stochastic Approximation and Online Learning

Weijie Su

University of Pennsylvania

Collaborator

- Yuancheng Zhu (UPenn)

Learning by optimization

Sample $Z_1, \ldots, Z_N$, and $f(\theta, z)$ is cost function

Learning model by minimizing

$$\underset{\theta}{\operatorname{argmin}} \ \frac{1}{N} \sum_{n=1}^{N} f(\theta, Z_n)$$

Learning by optimization

Sample $Z_1, \ldots, Z_N$, and $f(\theta, z)$ is cost function

Learning model by minimizing

$$\underset{\theta}{\operatorname{argmin}} \ \frac{1}{N} \sum_{n=1}^{N} f(\theta, Z_n)$$

- Maximum likelihood estimation (MLE). More generally, M-estimation
- Often no closed-form solution
- Need optimization

Gradient descent

- Start at some θ_0
- Iterate
$$\theta_j = \theta_{j-1} - \gamma_j \frac{\sum_{n=1}^{N} \nabla f(\theta_{j-1}, Z_n)}{N},$$
where γ_j are step sizes

Dates back to Newton, Gauss, and Cauchy

Difficulty with gradient descent

Modern machine learning

Gradient descent often not feasible due to

Difficulty with gradient descent

Modern machine learning

- Data arrives in a stream

Gradient descent often not feasible due to

- Essentially an *offline* algorithm

Difficulty with gradient descent

Modern machine learning

- Data arrives in a stream
- Number of data points N is exceedingly large

Gradient descent often not feasible due to

- Essentially an *offline* algorithm
- Evaluating full gradient is *computationally* expensive

Stochastic gradient descent (SGD)

Aka incremental gradient descent

- Start at some θ_0
- Iterate

$$\theta_j = \theta_{j-1} - \gamma_j \nabla f(\theta_{j-1}, Z_j)$$

Stochastic gradient descent (SGD)

Aka incremental gradient descent

- Start at some θ_0
- Iterate

$$\theta_j = \theta_{j-1} - \gamma_j \nabla f(\theta_{j-1}, Z_j)$$

SGD resolved these challenges

- Online in nature

Stochastic gradient descent (SGD)

Aka incremental gradient descent

- Start at some θ_0
- Iterate

$$\theta_j = \theta_{j-1} - \gamma_j \nabla f(\theta_{j-1}, Z_j)$$

SGD resolved these challenges

- Online in nature
- One pass over data

Stochastic gradient descent (SGD)

Aka incremental gradient descent

- Start at some θ_0
- Iterate

$$\theta_j = \theta_{j-1} - \gamma_j \nabla f(\theta_{j-1}, Z_j)$$

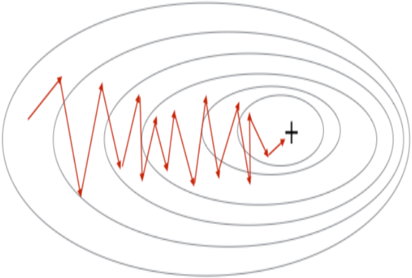
SGD resolved these challenges

- Online in nature
- One pass over data
- Optimal properties (Nemirovski & Yudin, 1983; Bertsekas, 1999; Agarwal et al, 2012; Rakhlin et al, 2012; Hardt et al, 2015)

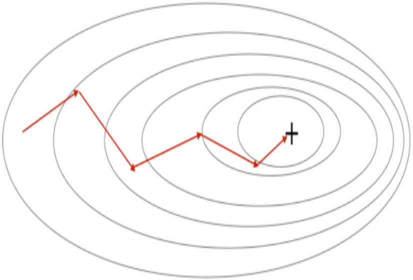
SGD in one line

SGD vs GD

SGD



GD



SGD: past and now

Statistics

- Robbins & Monro (1951); Kiefer & Wolfowitz (1952); Robbins & Siegmund (1971); Ruppert (1988); Polyak & Juditsky (1992)

Machine learning and optimization

- Nesterov & Vial (2008); Nemirovski et al (2009); Bottou (2010); Bach and Moulines (2011); Duchi et al (2011); Diederik & Ba (2014)

Applications

- Deep learning, recommender systems, MCMC, Kalman filter, phase retrieval, networks, and many

Using SGD for prediction

Averaged SGD

An estimator of $\theta^* := \operatorname{argmin} \mathbb{E} f(\theta, Z)$ is given by averaging

$$\overline{\theta} = \frac{1}{N} \sum_{j=1}^{N} \theta_j$$

Recall that $\theta_j = \theta_{j-1} - \gamma_j \nabla f(\theta_{j-1}, Z_j)$ for $j = 1, \ldots, N$.

Using SGD for prediction

Averaged SGD

An estimator of $\theta^* := \operatorname{argmin} \mathbb{E} f(\theta, Z)$ is given by averaging

$$\overline{\theta} = \frac{1}{N} \sum_{j=1}^{N} \theta_j$$

Recall that $\theta_j = \theta_{j-1} - \gamma_j \nabla f(\theta_{j-1}, Z_j)$ for $j = 1, \ldots, N$.

Given a new instance $z = (x, y)$ with y unknown

Interested in $\mu_x(\overline{\theta})$

- Linear regression: $\mu_x(\overline{\theta}) = x' \, \overline{\theta}$
- Logistic regression: $\mu_x(\overline{\theta}) = \frac{\mathrm{e}^{x' \, \overline{\theta}}}{1 + \mathrm{e}^{x' \, \overline{\theta}}}$
- Generalized linear models: $\mu_x(\overline{\theta}) = \mathbb{E}_{\overline{\theta}}(Y | X = x)$

How much can we trust SGD predictions?

We would observe a different $\mu_x(\overline{\theta})$ if

- Re-sample $Z'_1, \ldots, Z'_N$
- Sample with replacement N times from a finite population $z_1, \ldots, z_m$

How much can we trust SGD predictions?

We would observe a different $\mu_x(\overline{\theta})$ if

- Re-sample $Z'_1, \ldots, Z'_N$
- Sample with replacement N times from a finite population $z_1, \ldots, z_m$

Decision-making requires uncertainty quantification

- Should I invest in Bitcoin?
- How early to leave to catch a flight?

A real data example

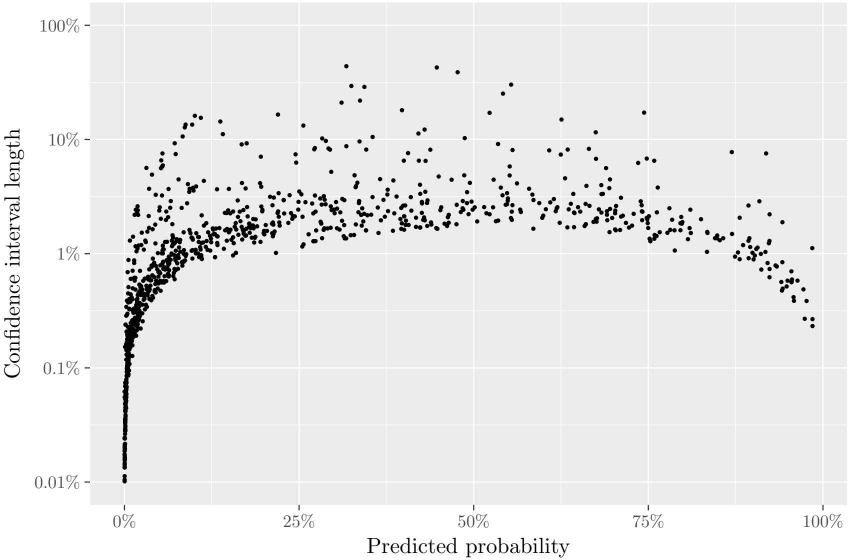
Adult dataset on UCI repository[1]

- 123 features
- $Y = 1$ if an individual's annual income exceeds $50,000
- 32,561 instances

Randomly pick 1,000 as a test set. Run SGD 500 times independently, each with 20 epochs and step sizes $\gamma_j = 0.5 j^{-0.55}$. Construct empirical confidence intervals with $\alpha = 10\%$

[1] `https://archive.ics.uci.edu/ml/datasets/Adult`

High variability of SGD predictions



What is desired

Can we construct a confidence interval for $\mu_x^* := \mu_x(\theta^*)$?

What is desired

Can we construct a confidence interval for $\mu_x^* := \mu_x(\theta^*)$?

Remarks

- Bootstrap is computationally infeasible
- Most existing works concern bounding generalization errors or minimizing regrets (Shalev-Shwartz et al, 2011; Rakhlin et al, 2012)
- Chen et al (2016) proposed a batch-mean estimator of SGD covariance, and Fang et al (2017) proposed a perturbation-based resampling procedure

This talk: HiGrad

A new method: Hierarchical Incremental GRAdient Descent

This talk: HiGrad

A new method: Hierarchical Incremental GRAdient Descent

Properties of HiGrad

- Online in nature with same computational cost as vanilla SGD

This talk: HiGrad

A new method: Hierarchical Incremental GRAdient Descent

Properties of HiGrad

- Online in nature with same computational cost as vanilla SGD
- A confidence interval for μ_x^* in addition to an estimator

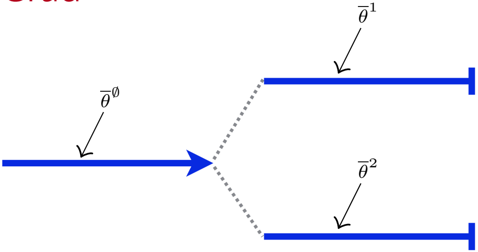
This talk: HiGrad

A new method: **H**ierarchical **I**ncremental **GRA**dient **D**escent

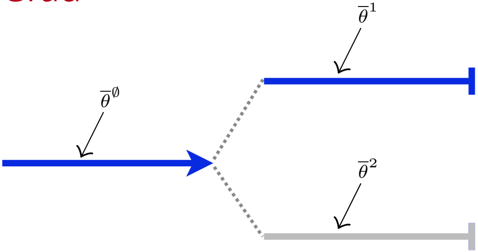
Properties of HiGrad

- Online in nature with same computational cost as vanilla SGD
- A confidence interval for μ_x^* in addition to an estimator
- Estimator (almost) as accurate as vanilla SGD

Preview of HiGrad

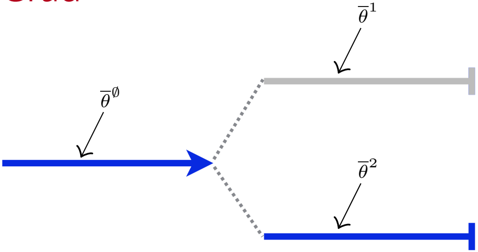


Preview of HiGrad



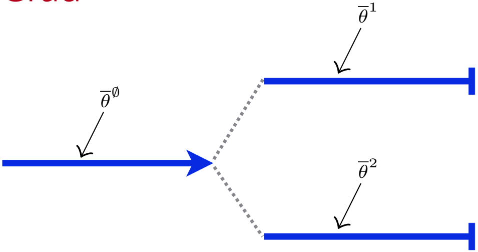
- $\overline{\theta}_1 = \frac{1}{3}\overline{\theta}^{\emptyset} + \frac{2}{3}\overline{\theta}^{1}, \quad \overline{\theta}_2 = \frac{1}{3}\overline{\theta}^{\emptyset} + \frac{2}{3}\overline{\theta}^{2}$

Preview of HiGrad



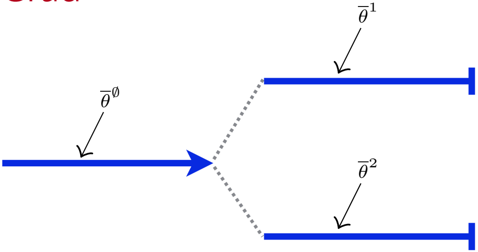
- $\overline{\theta}_1 = \frac{1}{3}\overline{\theta}^{\emptyset} + \frac{2}{3}\overline{\theta}^{1}, \quad \overline{\theta}_2 = \frac{1}{3}\overline{\theta}^{\emptyset} + \frac{2}{3}\overline{\theta}^{2}$

Preview of HiGrad



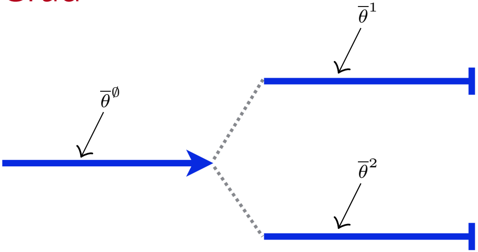
- $\overline{\theta}_1 = \frac{1}{3}\overline{\theta}^{\emptyset} + \frac{2}{3}\overline{\theta}^{1}, \quad \overline{\theta}_2 = \frac{1}{3}\overline{\theta}^{\emptyset} + \frac{2}{3}\overline{\theta}^{2}$
- $\mu_x^1 := \mu_x(\overline{\theta}_1) = 0.15, \quad \mu_x^2 := \mu_x(\overline{\theta}_2) = 0.11$

Preview of HiGrad



- $\overline{\theta}_1 = \frac{1}{3}\overline{\theta}^{\emptyset} + \frac{2}{3}\overline{\theta}^{1}, \quad \overline{\theta}_2 = \frac{1}{3}\overline{\theta}^{\emptyset} + \frac{2}{3}\overline{\theta}^{2}$
- $\mu_x^1 := \mu_x(\overline{\theta}_1) = 0.15, \quad \mu_x^2 := \mu_x(\overline{\theta}_2) = 0.11$
- HiGrad estimator is $\overline{\mu}_x = \frac{\mu_x^1 + \mu_x^2}{2} = 0.13$

Preview of HiGrad



- $\overline{\theta}_1 = \frac{1}{3}\overline{\theta}^{\emptyset} + \frac{2}{3}\overline{\theta}^{1}, \quad \overline{\theta}_2 = \frac{1}{3}\overline{\theta}^{\emptyset} + \frac{2}{3}\overline{\theta}^{2}$
- $\mu_x^1 := \mu_x(\overline{\theta}_1) = 0.15, \quad \mu_x^2 := \mu_x(\overline{\theta}_2) = 0.11$
- HiGrad estimator is $\overline{\mu}_x = \frac{\mu_x^1 + \mu_x^2}{2} = 0.13$
- The 90% HiGrad confidence interval for μ_x^* is

$$\left[\overline{\mu}_x - t_{1,0.95}\sqrt{0.375}|\mu_x^1 - \mu_x^2|,\ \overline{\mu}_x + t_{1,0.95}\sqrt{0.375}|\mu_x^1 - \mu_x^2|\right]$$
$$= [-0.025, 0.285]$$

Outline

1. Deriving HiGrad

2. Constructing Confidence Intervals

3. Configuring HiGrad

4. Empirical Performance

Problem statement

Minimizing convex f

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \; f(\theta) \equiv \mathbb{E} f(\theta, Z)$$

Observe i.i.d. $Z_1, \ldots, Z_N$ and can evaluate unbiased noisy gradient $g(\theta; Z)$

$$\mathbb{E}\, g(\theta, Z) = \nabla f(\theta) \text{ for all } \theta$$

To be fulfilled

- Online in nature with same computational cost as vanilla SGD
- A confidence interval for μ_x^* in addition to an estimator
- Estimator (almost) as accurate as vanilla SGD

The idea of contrasting and sharing

- Need more than one value μ_x to quantify variability: **contrasting**

The idea of contrasting and sharing

- Need more than one value μ_x to quantify variability: **contrasting**
- Need to share gradient information to elongate threads: **sharing**

The HiGrad tree

- $K + 1$ levels
- each k-level segment is of length n_k and is split into B_{k+1} segments

$$n_0 + B_1 n_1 + B_1 B_2 n_2 + B_1 B_2 B_3 n_3 + \cdots + B_1 B_2 \cdots B_K n_K = N$$

The HiGrad tree

- $K + 1$ levels
- each k-level segment is of length n_k and is split into B_{k+1} segments

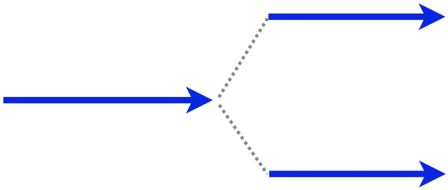
$$n_0 + B_1 n_1 + B_1 B_2 n_2 + B_1 B_2 B_3 n_3 + \cdots + B_1 B_2 \cdots B_K n_K = N$$

An example of HiGrad tree: $B_1 = 2, B_2 = 3, K = 2$

The HiGrad tree

- $K+1$ levels
- each k-level segment is of length n_k and is split into B_{k+1} segments

$$n_0 + B_1 n_1 + B_1 B_2 n_2 + B_1 B_2 B_3 n_3 + \cdots + B_1 B_2 \cdots B_K n_K = N$$

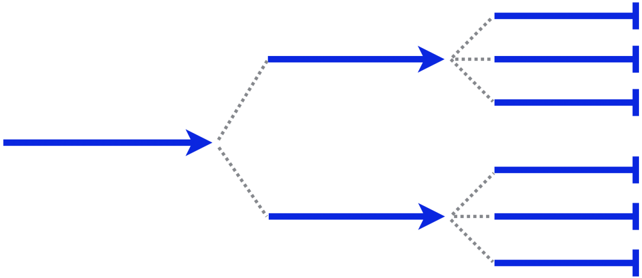


An example of HiGrad tree: $B_1 = 2, B_2 = 3, K = 2$

The HiGrad tree

- $K + 1$ levels
- each k-level segment is of length n_k and is split into B_{k+1} segments

$$n_0 + B_1 n_1 + B_1 B_2 n_2 + B_1 B_2 B_3 n_3 + \cdots + B_1 B_2 \cdots B_K n_K = N$$



An example of HiGrad tree: $B_1 = 2, B_2 = 3, K = 2$

Iterate along HiGrad tree

Recall: noisy gradient $g(\theta, Z)$ unbiased for $\nabla f(\theta)$; partition $\{Z^{\boldsymbol{s}}\}$ of $\{Z_1, \ldots, Z_N\}$; and $L_k := n_0 + \cdots + n_k$

Iterate along HiGrad tree

Recall: noisy gradient $g(\theta, Z)$ unbiased for $\nabla f(\theta)$; partition $\{Z^{\boldsymbol{s}}\}$ of $\{Z_1, \ldots, Z_N\}$; and $L_k := n_0 + \cdots + n_k$

- Iterate along level 0 segment: $\theta_j = \theta_{j-1} - \gamma_j \nabla f(\theta_{j-1}, Z_j)$ for $j = 1, \ldots, n_0$, starting from some θ_0

Iterate along HiGrad tree

Recall: noisy gradient $g(\theta, Z)$ unbiased for $\nabla f(\theta)$; partition $\{Z^{\boldsymbol{s}}\}$ of $\{Z_1, \ldots, Z_N\}$; and $L_k := n_0 + \cdots + n_k$

- Iterate along level 0 segment: $\theta_j = \theta_{j-1} - \gamma_j \nabla f(\theta_{j-1}, Z_j)$ for $j = 1, \ldots, n_0$, starting from some θ_0
- Iterate along each level 1 segment $\boldsymbol{s} = (b_1)$ for $1 \le b_1 \le B_1$

$$\theta_j^{\boldsymbol{s}} = \theta_{j-1}^{\boldsymbol{s}} - \gamma_{j+L_0} g(\theta_{j-1}^{\boldsymbol{s}}, Z_j^{\boldsymbol{s}})$$

  for $j = 1, \ldots, n_1$, starting from θ_{n_0}

Iterate along HiGrad tree

Recall: noisy gradient $g(\theta, Z)$ unbiased for $\nabla f(\theta)$; partition $\{Z^{\boldsymbol{s}}\}$ of $\{Z_1, \ldots, Z_N\}$; and $L_k := n_0 + \cdots + n_k$

- Iterate along level 0 segment: $\theta_j = \theta_{j-1} - \gamma_j \nabla f(\theta_{j-1}, Z_j)$ for $j = 1, \ldots, n_0$, starting from some θ_0
- Iterate along each level 1 segment $\boldsymbol{s} = (b_1)$ for $1 \le b_1 \le B_1$

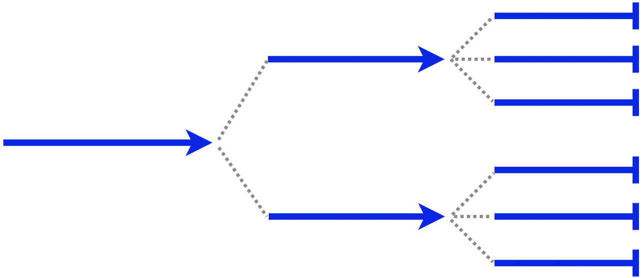
  $$\theta_j^{\boldsymbol{s}} = \theta_{j-1}^{\boldsymbol{s}} - \gamma_{j+L_0} g(\theta_{j-1}^{\boldsymbol{s}}, Z_j^{\boldsymbol{s}})$$

  for $j = 1, \ldots, n_1$, starting from θ_{n_0}
- Generally, for the segment $\boldsymbol{s} = (b_1 \cdots b_k)$, iterate

  $$\theta_j^{\boldsymbol{s}} = \theta_{j-1}^{\boldsymbol{s}} - \gamma_{j+L_{k-1}}\, g(\theta_{j-1}^{\boldsymbol{s}}, Z_j^{\boldsymbol{s}})$$

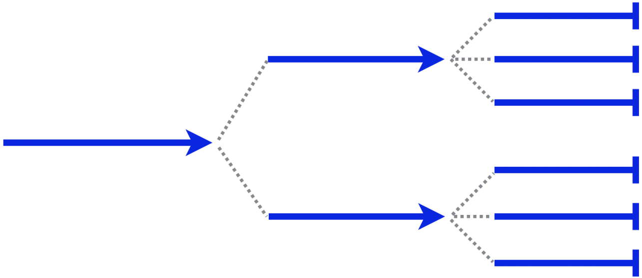
  for $j = 1, \ldots, n_k$, starting from $\theta_{n_{k-1}}^{(b_1 \cdots b_{k-1})}$

A second look at the HiGrad tree



An example of HiGrad tree: $B_1 = 2, B_2 = 3, K = 2$

A second look at the HiGrad tree

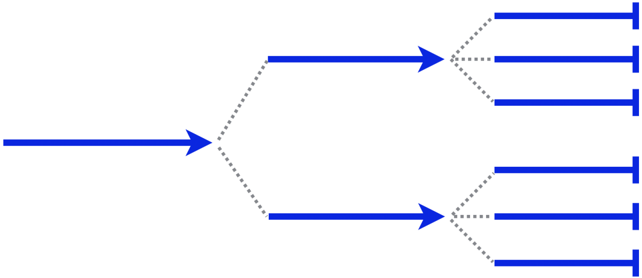


An example of HiGrad tree: $B_1 = 2, B_2 = 3, K = 2$

Fulfilled

- Online in nature with same computational cost as vanilla SGD ✓

A second look at the HiGrad tree



An example of HiGrad tree: $B_1 = 2, B_2 = 3, K = 2$

Fulfilled

- Online in nature with same computational cost as vanilla SGD ✓

Bonus

Easier to parallelize than vanilla SGD!

The HiGrad algorithm in action

Require: $g(\cdot,\cdot), Z_1, \ldots, Z_N, (n_0, n_1, \ldots, n_K), (B_1, \ldots, B_K), (\gamma_1, \ldots, \gamma_{N_K}), \theta_0$

$\overline{\theta}^{\boldsymbol{s}} = 0$ for all segments $\boldsymbol{s}$
function NodeTreeSGD($\theta, \boldsymbol{s}$)
$\theta_0^{\boldsymbol{s}} = \theta$
$k = \#\boldsymbol{s}$
for $j = 1$ **to** n_k **do**
    $\theta_j^{\boldsymbol{s}} \leftarrow \theta_{j-1}^{\boldsymbol{s}} - \gamma_{j+L_{k-1}}\, g(\theta_{j-1}^{\boldsymbol{s}}, Z_j^{\boldsymbol{s}})$
    $\overline{\theta}^{\boldsymbol{s}} \leftarrow \overline{\theta}^{\boldsymbol{s}} + \theta_j^{\boldsymbol{s}}/n_k$
end for
if $k < K$ **then**
    for $b_{k+1} = 1$ **to** B_{k+1} **do**
        $\boldsymbol{s}^+ \leftarrow (\boldsymbol{s}, b_{k+1})$
        execute NodeTreeSGD$\left(\theta_{n_k}^{\boldsymbol{s}}, \boldsymbol{s}^+\right)$
    end for
end if
end function
execute NodeTreeSGD($\theta_0, \emptyset$)
output: $\overline{\theta}^{\boldsymbol{s}}$ for all segments $\boldsymbol{s}$

Outline

1. Deriving HiGrad

2. Constructing Confidence Intervals

3. Configuring HiGrad

4. Empirical Performance

Estimate μ_x^* through each thread

Average over each segment $\boldsymbol{s} = (b_1, \ldots, b_k)$

$$\overline{\theta}^{\boldsymbol{s}} = \frac{1}{n_k} \sum_{j=1}^{n_k} \theta_j^{\boldsymbol{s}}$$

Given weights $w_0, w_1, \ldots, w_K$ that sum up to 1, weighted average along thread $\boldsymbol{t} = (b_1, \ldots, b_K)$ is

$$\overline{\theta}_{\boldsymbol{t}} = \sum_{k=0}^{K} w_k \overline{\theta}^{(b_1, \ldots, b_k)}$$

Estimate μ_x^* through each thread

Average over each segment $\boldsymbol{s} = (b_1, \ldots, b_k)$

$$\overline{\theta}^{\boldsymbol{s}} = \frac{1}{n_k} \sum_{j=1}^{n_k} \theta_j^{\boldsymbol{s}}$$

Given weights $w_0, w_1, \ldots, w_K$ that sum up to 1, weighted average along thread $\boldsymbol{t} = (b_1, \ldots, b_K)$ is

$$\overline{\theta}_{\boldsymbol{t}} = \sum_{k=0}^{K} w_k \overline{\theta}^{(b_1, \ldots, b_k)}$$

Estimator yielded by thread $\boldsymbol{t}$

$$\mu_x^{\boldsymbol{t}} := \mu_x(\overline{\theta}_{\boldsymbol{t}})$$

How to construct a confidence interval based on $T := B_1 B_2 \cdots B_K$ many such $\mu_x^{\boldsymbol{t}}$ estimates?

Assume normality

Denote by $\boldsymbol{\mu}_x$ the T-dimensional vector consisting of all $\mu_x^{\boldsymbol{t}}$

Normality of $\boldsymbol{\mu}_x$ (to be proved soon)

$\sqrt{N}(\boldsymbol{\mu}_x - \mu_x^* \mathbf{1})$ converges weakly to normal distribution $\mathcal{N}(\mathbf{0}, \Sigma)$ as $N \to \infty$

Convert to simple linear regression

From $\boldsymbol{\mu}_x \overset{a}{\sim} \mathcal{N}(\mu_x^* \mathbf{1}, \Sigma/N)$ we get

$$\Sigma^{-\frac{1}{2}} \boldsymbol{\mu}_x \approx (\Sigma^{-\frac{1}{2}} \mathbf{1}) \mu_x^* + \tilde{\boldsymbol{z}}, \quad \tilde{\boldsymbol{z}} \sim \mathcal{N}(0, \boldsymbol{I}/N)$$

Convert to simple linear regression

From $\boldsymbol{\mu}_x \overset{a}{\sim} \mathcal{N}(\mu_x^* \mathbf{1}, \Sigma/N)$ we get

$$\Sigma^{-\frac{1}{2}} \boldsymbol{\mu}_x \approx (\Sigma^{-\frac{1}{2}} \mathbf{1}) \mu_x^* + \tilde{\boldsymbol{z}}, \quad \tilde{\boldsymbol{z}} \sim \mathcal{N}(0, \boldsymbol{I}/N)$$

Simple linear regression! Least-squares estimator of μ_x^* given as

$$\begin{aligned}
&(\mathbf{1}' \Sigma^{-\frac{1}{2}} \Sigma^{-\frac{1}{2}} \mathbf{1})^{-1} \mathbf{1}' \Sigma^{-\frac{1}{2}} \Sigma^{-\frac{1}{2}} \boldsymbol{\mu}_x \\
&= (\mathbf{1}' \Sigma^{-1} \mathbf{1})^{-1} \mathbf{1}' \Sigma^{-1} \boldsymbol{\mu}_x \\
&= \frac{1}{T} \sum_{\boldsymbol{t} \in \mathcal{T}} \mu_x^{\boldsymbol{t}} \equiv \overline{\mu}_x
\end{aligned}$$

HiGrad estimator

Just the sample mean $\overline{\mu}_x$

A t-based confidence interval

A *pivot* for μ_x^*

$$\frac{\overline{\mu}_x - \mu_x^*}{\mathrm{SE}_x} \overset{a}{\sim} t_{T-1},$$

where the standard error is given as

$$\mathrm{SE}_x = \sqrt{\frac{(\boldsymbol{\mu}_x' - \overline{\mu}_x \mathbf{1}')\Sigma^{-1}(\boldsymbol{\mu}_x - \overline{\mu}_x \mathbf{1})}{T-1}} \cdot \frac{\sqrt{\mathbf{1}'\Sigma\mathbf{1}}}{T}$$

A t-based confidence interval

A *pivot* for μ_x^*

$$\frac{\overline{\mu}_x - \mu_x^*}{\mathrm{SE}_x} \overset{a}{\sim} t_{T-1},$$

where the standard error is given as

$$\mathrm{SE}_x = \sqrt{\frac{(\boldsymbol{\mu}_x' - \overline{\mu}_x \mathbf{1}')\Sigma^{-1}(\boldsymbol{\mu}_x - \overline{\mu}_x \mathbf{1})}{T-1}} \cdot \frac{\sqrt{\mathbf{1}'\Sigma\mathbf{1}}}{T}$$

HiGrad confidence interval of coverage $1-\alpha$

$$\left[\overline{\mu}_x - t_{T-1,1-\frac{\alpha}{2}}\,\mathrm{SE}_x, \quad \overline{\mu}_x + t_{T-1,1-\frac{\alpha}{2}}\,\mathrm{SE}_x\right]$$

Do we know the covariance Σ?

An extension of Ruppert–Polyak normality

Given a thread $\boldsymbol{t} = (b_1, \ldots, b_K)$, denote by segments $\boldsymbol{s}_k = (b_1, b_2, \ldots, b_k)$

Fact (informal)

$\sqrt{n_0}(\overline{\theta}^{\boldsymbol{s}_0} - \theta^*), \sqrt{n_1}(\overline{\theta}^{\boldsymbol{s}_1} - \theta^*), \ldots, \sqrt{n_K}(\overline{\theta}^{\boldsymbol{s}_K} - \theta^*)$ converge to i.i.d. centered normal distributions

An extension of Ruppert–Polyak normality

Given a thread $\boldsymbol{t} = (b_1, \ldots, b_K)$, denote by segments $\boldsymbol{s}_k = (b_1, b_2, \ldots, b_k)$

Fact (informal)

$\sqrt{n_0}(\overline{\theta}^{\boldsymbol{s}_0} - \theta^*), \sqrt{n_1}(\overline{\theta}^{\boldsymbol{s}_1} - \theta^*), \ldots, \sqrt{n_K}(\overline{\theta}^{\boldsymbol{s}_K} - \theta^*)$ converge to i.i.d. centered normal distributions

- Hessian $H = \nabla^2 f(\theta^*)$ and $V = \mathbb{E}\left[g(\theta^*, Z)g(\theta^*, Z)'\right]$. Ruppert (1988), Polyak (1990), and Polyak and Juditsky (1992) prove

$$\sqrt{N}(\overline{\theta}_N - \theta^*) \Rightarrow \mathcal{N}(0, H^{-1}VH^{-1})$$

An extension of Ruppert–Polyak normality

Given a thread $\boldsymbol{t} = (b_1, \ldots, b_K)$, denote by segments $\boldsymbol{s}_k = (b_1, b_2, \ldots, b_k)$

Fact (informal)

$\sqrt{n_0}(\overline{\theta}^{\boldsymbol{s}_0} - \theta^*), \sqrt{n_1}(\overline{\theta}^{\boldsymbol{s}_1} - \theta^*), \ldots, \sqrt{n_K}(\overline{\theta}^{\boldsymbol{s}_K} - \theta^*)$ converge to i.i.d. centered normal distributions

- Hessian $H = \nabla^2 f(\theta^*)$ and $V = \mathbb{E}\left[g(\theta^*, Z)g(\theta^*, Z)'\right]$. Ruppert (1988), Polyak (1990), and Polyak and Juditsky (1992) prove

$$\sqrt{N}(\overline{\theta}_N - \theta^*) \Rightarrow \mathcal{N}(0, H^{-1}VH^{-1})$$

- Difficult to estimate sandwich covariance $H^{-1}VH^{-1}$ (Chen et al, 2016)

An extension of Ruppert–Polyak normality

Given a thread $\boldsymbol{t} = (b_1, \dots, b_K)$, denote by segments $\boldsymbol{s}_k = (b_1, b_2, \dots, b_k)$

Fact (informal)

$\sqrt{n_0}(\overline{\theta}^{\boldsymbol{s}_0} - \theta^*), \sqrt{n_1}(\overline{\theta}^{\boldsymbol{s}_1} - \theta^*), \dots, \sqrt{n_K}(\overline{\theta}^{\boldsymbol{s}_K} - \theta^*)$ converge to i.i.d. centered normal distributions

- Hessian $H = \nabla^2 f(\theta^*)$ and $V = \mathbb{E}\left[g(\theta^*, Z)g(\theta^*, Z)'\right]$. Ruppert (1988), Polyak (1990), and Polyak and Juditsky (1992) prove

$$\sqrt{N}(\overline{\theta}_N - \theta^*) \Rightarrow \mathcal{N}(0, H^{-1}VH^{-1})$$

- Difficult to estimate sandwich covariance $H^{-1}VH^{-1}$ (Chen et al, 2016)
- *To know covariance of $\{\mu_x(\overline{\theta}_{\boldsymbol{t}})\}$, really need to know $H^{-1}VH^{-1}$?*

Covariance determined by number of shared segments

Consider $\mu_x(\theta) = T(x)'\,\theta$ and observe

- $\sqrt{n_0}(\mu_x(\overline{\theta}^{\boldsymbol{s}_0}) - \mu_x^*), \sqrt{n_1}(\mu_x(\overline{\theta}^{\boldsymbol{s}_1}) - \mu_x^*), \dots, \sqrt{n_K}(\mu_x(\overline{\theta}^{\boldsymbol{s}_K}) - \mu_x^*)$ converge to i.i.d. centered univariate normal distributions
- $\mu_x^{\boldsymbol{t}} - \mu_x^* = \mu_x(\overline{\theta}_{\boldsymbol{t}}) - \mu_x^* = \sum_{k=0}^{K} w_k \left(\mu_x(\overline{\theta}^{\boldsymbol{s}_k}) - \mu_x^*\right)$

Covariance determined by number of shared segments

Consider $\mu_x(\theta) = T(x)'\,\theta$ and observe

- $\sqrt{n_0}(\mu_x(\overline{\theta}^{\boldsymbol{s}_0}) - \mu_x^*), \sqrt{n_1}(\mu_x(\overline{\theta}^{\boldsymbol{s}_1}) - \mu_x^*), \ldots, \sqrt{n_K}(\mu_x(\overline{\theta}^{\boldsymbol{s}_K}) - \mu_x^*)$ converge to i.i.d. centered univariate normal distributions
- $\mu_x^{\boldsymbol{t}} - \mu_x^* = \mu_x(\overline{\theta}_{\boldsymbol{t}}) - \mu_x^* = \sum_{k=0}^{K} w_k \left(\mu_x(\overline{\theta}^{\boldsymbol{s}_k}) - \mu_x^*\right)$

Fact (informal)

For any two threads $\boldsymbol{t}$ and $\boldsymbol{t}'$ that agree at the first k segments and differ henceforth, we have

$$\mathrm{Cov}\left(\mu_x^{\boldsymbol{t}}, \mu_x^{\boldsymbol{t}'}\right) = (1 + o(1))\sigma^2 \sum_{i=0}^{k} \frac{w_i^2}{n_i}$$

Specify Σ up to a multiplicative factor

If $\mu_x(\theta) = T(x)'\,\theta$, then for any two threads $\boldsymbol{t}$ and $\boldsymbol{t}'$ that agree only at the first k segments,

$$\Sigma_{\boldsymbol{t},\boldsymbol{t}'} = (1 + o(1))C \sum_{i=0}^{k} \frac{\omega_i^2 N}{n_i}$$

Specify Σ up to a multiplicative factor

If $\mu_x(\theta) = T(x)'\,\theta$, then for any two threads $\boldsymbol{t}$ and $\boldsymbol{t}'$ that agree only at the first k segments,

$$\Sigma_{\boldsymbol{t},\boldsymbol{t}'} = (1 + o(1))C \sum_{i=0}^{k} \frac{\omega_i^2 N}{n_i}$$

- Do we need to know C as well?

Specify Σ up to a multiplicative factor

If $\mu_x(\theta) = T(x)'\,\theta$, then for any two threads $\boldsymbol{t}$ and $\boldsymbol{t}'$ that agree only at the first k segments,

$$\Sigma_{\boldsymbol{t},\boldsymbol{t}'} = (1 + o(1))C \sum_{i=0}^{k} \frac{\omega_i^2 N}{n_i}$$

- Do we need to know C as well?
- No! Standard error of $\overline{\mu}_x$ invariant under multiplying Σ by a scalar

$$\mathrm{SE}_x = \sqrt{\frac{(\boldsymbol{\mu}_x' - \overline{\mu}_x \mathbf{1}')\Sigma^{-1}(\boldsymbol{\mu}_x - \overline{\mu}_x \mathbf{1})}{T-1}} \cdot \frac{\sqrt{\mathbf{1}'\Sigma\mathbf{1}}}{T}$$

Some remarks

- In generalized linear models, μ_x often takes the form $\mu_x(\theta) = \eta^{-1}(T(x)'\theta)$ for an increasing η. Construct confidence interval for $\eta(\mu_x)$ and then invert
- For general nonlinear but smooth $\mu_x(\theta)$, use delta method
- Need less than Ruppert–Polyak: remains to hold if $\sqrt{N}(\bar{\theta}_N - \theta^*)$ converges to some centered normal distribution

Formal statement of theoretical results

Assumptions

1. **Local strong convexity.** $f(\theta) \equiv \mathbb{E}f(\theta, Z)$ *convex*, differentiable, with Lipschitz gradients. Hessian $\nabla^2 f(\theta)$ locally Lipschitz and *positive-definite* at θ^*

2. **Noise regularity.** $V(\theta) = \mathbb{E}\left[g(\theta, Z)g(\theta, Z)'\right]$ Lipschitz and does not grow too fast. Noisy gradient $g(\theta, Z)$ has $2 + o(1)$ moment locally at θ^*

Examples satisfying assumptions

- **Linear regression**: $f(\theta, z) = \frac{1}{2}(y - x^\top\theta)^2$.
- **Logistic regression**: $f(\theta, z) = -yx^\top\theta + \log\left(1 + \mathrm{e}^{x^\top\theta}\right)$.
- **Penalized regression**: Add a ridge penalty $\lambda\|\theta\|^2$.
- **Huber regression**: $f(\theta, z) = \rho_\lambda(y - x^\top\theta)$, where $\rho_\lambda(a) = a^2/2$ for $|a| \le \lambda$ and $\rho_\lambda(a) = \lambda|a| - \lambda^2/2$ otherwise.

Sufficient conditions

X in *generic* position, and $\mathbb{E}\|X\|^{4+o(1)} < \infty$ and $\mathbb{E}|Y|^{2+o(1)}\|X\|^{2+o(1)} < \infty$

Main theoretical results

Theorem (S. and Zhu)

Assume K and $B_1, \ldots, B_K$ are fixed, $n_k \propto N$ as $N \to \infty$, and μ_x has a nonzero derivative at θ^*. Taking $\gamma_j \asymp j^{-\alpha}$ for $\alpha \in (0.5, 1)$ gives

$$\frac{\overline{\mu}_x - \mu_x^*}{\mathrm{SE}_x} \Longrightarrow t_{T-1}$$

Main theoretical results

Theorem (S. and Zhu)

Assume K and $B_1, \ldots, B_K$ are fixed, $n_k \propto N$ as $N \to \infty$, and μ_x has a nonzero derivative at θ^*. Taking $\gamma_j \asymp j^{-\alpha}$ for $\alpha \in (0.5, 1)$ gives

$$\frac{\overline{\mu}_x - \mu_x^*}{\mathrm{SE}_x} \Longrightarrow t_{T-1}$$

Confidence intervals

$$\lim_{N\to\infty} \mathbb{P}\left(\mu_x^* \in \left[\overline{\mu}_x - t_{T-1,1-\frac{\alpha}{2}}\,\mathrm{SE}_x, \quad \overline{\mu}_x + t_{T-1,1-\frac{\alpha}{2}}\,\mathrm{SE}_x\right]\right) = 1 - \alpha$$

Main theoretical results

Theorem (S. and Zhu)

Assume K and $B_1, \ldots, B_K$ are fixed, $n_k \propto N$ as $N \to \infty$, and μ_x has a nonzero derivative at θ^*. Taking $\gamma_j \asymp j^{-\alpha}$ for $\alpha \in (0.5, 1)$ gives

$$\frac{\overline{\mu}_x - \mu_x^*}{\mathrm{SE}_x} \Longrightarrow t_{T-1}$$

Confidence intervals

$$\lim_{N \to \infty} \mathbb{P}\left(\mu_x^* \in \left[\overline{\mu}_x - t_{T-1,1-\frac{\alpha}{2}}\, \mathrm{SE}_x, \quad \overline{\mu}_x + t_{T-1,1-\frac{\alpha}{2}}\, \mathrm{SE}_x\right]\right) = 1 - \alpha$$

Fulfilled

- Online in nature with same computational cost as vanilla SGD ✓
- A confidence interval for μ_x^* in addition to an estimator ✓

How accurate is the HiGrad estimator?

Optimal variance with optimal weights

By Cauchy–Schwarz

$$
\begin{aligned}
N\,\mathbb{V}\mathsf{ar}(\overline{\mu}_x) &= (1+o(1))\sigma^2 \left[\sum_{k=0}^{K} n_k \prod_{i=1}^{k} B_i\right] \left[\sum_{k=0}^{K} \frac{w_k^2}{n_k \prod_{i=1}^{k} B_i}\right] \\
&\geq (1+o(1))\sigma^2 \left[\sum_{k=0}^{K} \sqrt{w_k^2}\right]^2 = (1+o(1))\sigma^2,
\end{aligned}
$$

with equality if

$$
w_k^* = \frac{n_k \prod_{i=1}^{k} B_i}{N}
$$

Optimal variance with optimal weights

By Cauchy–Schwarz

$$
\begin{aligned}
N\,\mathbb{V}\mathrm{ar}(\overline{\mu}_x) &= (1+o(1))\sigma^2 \left[\sum_{k=0}^{K} n_k \prod_{i=1}^{k} B_i\right] \left[\sum_{k=0}^{K} \frac{w_k^2}{n_k \prod_{i=1}^{k} B_i}\right] \\
&\geq (1+o(1))\sigma^2 \left[\sum_{k=0}^{K} \sqrt{w_k^2}\right]^2 = (1+o(1))\sigma^2,
\end{aligned}
$$

with equality if

$$
w_k^* = \frac{n_k \prod_{i=1}^{k} B_i}{N}
$$

- Segments at an early level weighted less

Optimal variance with optimal weights

By Cauchy–Schwarz

$$
\begin{aligned}
N\,\mathbb{V}\mathrm{ar}(\overline{\mu}_x) &= (1+o(1))\sigma^2 \left[\sum_{k=0}^{K} n_k \prod_{i=1}^{k} B_i\right] \left[\sum_{k=0}^{K} \frac{w_k^2}{n_k \prod_{i=1}^{k} B_i}\right] \\
&\geq (1+o(1))\sigma^2 \left[\sum_{k=0}^{K} \sqrt{w_k^2}\right]^2 = (1+o(1))\sigma^2,
\end{aligned}
$$

with equality if

$$
w_k^* = \frac{n_k \prod_{i=1}^{k} B_i}{N}
$$

- Segments at an early level weighted less
- HiGrad estimator has the *same* asymptotic variance as vanilla SGD

Optimal variance with optimal weights

By Cauchy–Schwarz

$$
\begin{aligned}
N\,\mathbb{V}\mathrm{ar}(\overline{\mu}_x) &= (1+o(1))\sigma^2 \left[\sum_{k=0}^{K} n_k \prod_{i=1}^{k} B_i\right] \left[\sum_{k=0}^{K} \frac{w_k^2}{n_k \prod_{i=1}^{k} B_i}\right] \\
&\geq (1+o(1))\sigma^2 \left[\sum_{k=0}^{K} \sqrt{w_k^2}\right]^2 = (1+o(1))\sigma^2,
\end{aligned}
$$

with equality if

$$
w_k^* = \frac{n_k \prod_{i=1}^{k} B_i}{N}
$$

- Segments at an early level weighted less
- HiGrad estimator has the *same* asymptotic variance as vanilla SGD
- Achieves Cramér–Rao lower bound when model specified

Prediction intervals for vanilla SGD

Theorem (S. and Zhu)

Run vanilla SGD on a fresh dataset of the same size, producing μ_x^{SGD}. Then, with optimal weights,

$$\lim_{N\to\infty} \mathbb{P}\left(\mu_x^{\mathrm{SGD}} \in \left[\overline{\mu}_x - \sqrt{2}t_{T-1,1-\frac{\alpha}{2}}\,\mathrm{SE}_x, \quad \overline{\mu}_x + \sqrt{2}t_{T-1,1-\frac{\alpha}{2}}\,\mathrm{SE}_x\right]\right) = 1 - \alpha.$$

- μ_x^{SGD} can be replaced by the HiGrad estimator with the same structure
- Interpretable even under model misspecification

HiGrad enjoys three appreciable properties

Under certain assumptions, for example, f being locally strongly convex

Fulfilled

- Online in nature with same computational cost as vanilla SGD ✓
- A confidence interval for μ_x^* in addition to an estimator ✓
- Estimator (almost) as accurate as vanilla SGD ✓

Outline

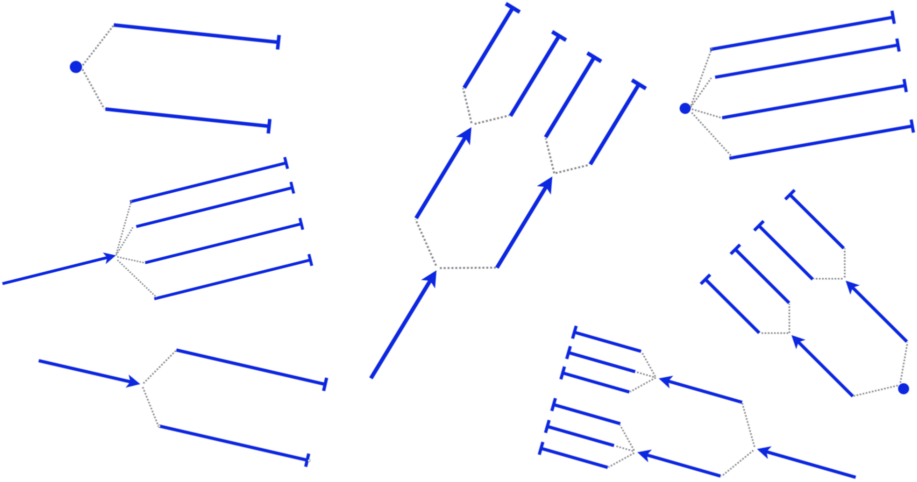
1. Deriving HiGrad

2. Constructing Confidence Intervals

3. Configuring HiGrad

4. Empirical Performance

Which one?



Length of confidence intervals

Denote by $L_{\mathrm{CI}} = 2t_{T-1,1-\frac{\alpha}{2}}\,\mathrm{SE}_x$ the length of HiGrad confidence interval

Proposition (S. and Zhu)

$$\sqrt{N}\mathbb{E}L_{\mathrm{CI}} \to \frac{2\sigma\sqrt{2}t_{T-1,1-\frac{\alpha}{2}}\Gamma\left(\frac{T}{2}\right)}{\sqrt{T-1}\,\Gamma\left(\frac{T-1}{2}\right)}$$

Length of confidence intervals

Denote by $L_{\mathrm{CI}} = 2t_{T-1,1-\frac{\alpha}{2}}\,\mathrm{SE}_x$ the length of HiGrad confidence interval

Proposition (S. and Zhu)

$$\sqrt{N}\mathbb{E}L_{\mathrm{CI}} \to \frac{2\sigma\sqrt{2}t_{T-1,1-\frac{\alpha}{2}}\Gamma\left(\frac{T}{2}\right)}{\sqrt{T-1}\,\Gamma\left(\frac{T-1}{2}\right)}$$

- The function $\dfrac{t_{T-1,1-\frac{\alpha}{2}}\Gamma\left(\frac{T}{2}\right)}{\sqrt{T-1}\,\Gamma\left(\frac{T-1}{2}\right)}$ is decreasing in $T \geq 2$

Length of confidence intervals

Denote by $L_{\mathrm{CI}} = 2t_{T-1,1-\frac{\alpha}{2}}\,\mathrm{SE}_x$ the length of HiGrad confidence interval

Proposition (S. and Zhu)

$$\sqrt{N}\mathbb{E}L_{\mathrm{CI}} \to \frac{2\sigma\sqrt{2}t_{T-1,1-\frac{\alpha}{2}}\Gamma\left(\frac{T}{2}\right)}{\sqrt{T-1}\,\Gamma\left(\frac{T-1}{2}\right)}$$

- The function $\dfrac{t_{T-1,1-\frac{\alpha}{2}}\Gamma\left(\frac{T}{2}\right)}{\sqrt{T-1}\,\Gamma\left(\frac{T-1}{2}\right)}$ is decreasing in $T \geq 2$
- The more threads, the shorter the HiGrad confidence interval on average

Length of confidence intervals

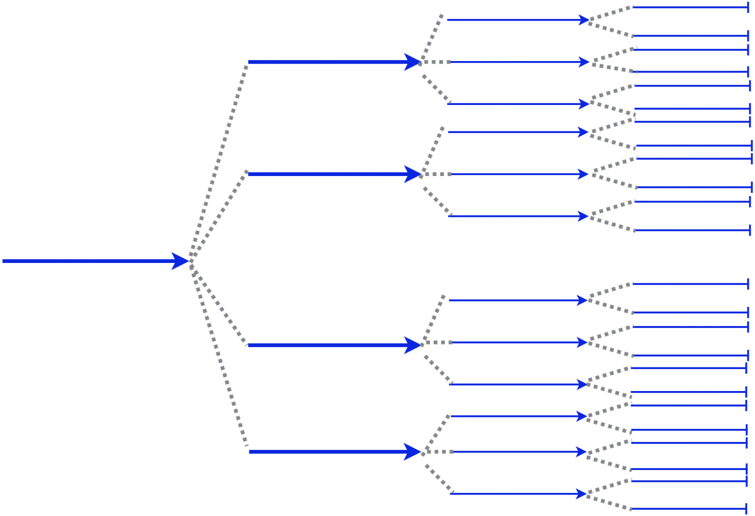
Denote by $L_{\mathrm{CI}} = 2t_{T-1,1-\frac{\alpha}{2}}\,\mathrm{SE}_x$ the length of HiGrad confidence interval

Proposition (S. and Zhu)

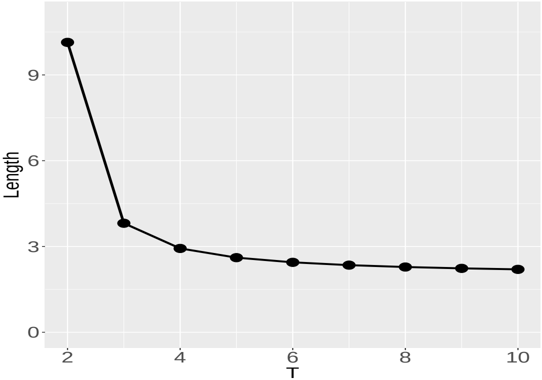
$$\sqrt{N}\mathbb{E}L_{\mathrm{CI}} \to \frac{2\sigma\sqrt{2}t_{T-1,1-\frac{\alpha}{2}}\Gamma\left(\frac{T}{2}\right)}{\sqrt{T-1}\,\Gamma\left(\frac{T-1}{2}\right)}$$

- The function $\dfrac{t_{T-1,1-\frac{\alpha}{2}}\Gamma\left(\frac{T}{2}\right)}{\sqrt{T-1}\,\Gamma\left(\frac{T-1}{2}\right)}$ is decreasing in $T \geq 2$
- The more threads, the shorter the HiGrad confidence interval on average
- More contrasting leads to shorter confidence interval

Really want to set $T = 1000$?



$T = 4$ is sufficient



Plot of $\dfrac{t_{T-1,0.975}\Gamma\left(T/2\right)}{\sqrt{T-1}\,\Gamma\left(T/2-0.5\right)}$

- Too many threads result in inaccurate normality (unless N is huge)
- Large T leads to much *contrasting* and little *sharing*

How to choose $(n_0, \ldots, n_K)$?

$$n_0 + B_1 n_1 + B_1 B_2 n_2 + B_1 B_2 B_3 n_3 + \cdots + B_1 B_2 \cdots B_K n_K = N$$

Length of each thread

$$L_K := n_0 + n_1 + \cdots + n_K$$

How to choose $(n_0, \ldots, n_K)$?

$$n_0 + B_1 n_1 + B_1 B_2 n_2 + B_1 B_2 B_3 n_3 + \cdots + B_1 B_2 \cdots B_K n_K = N$$

Length of each thread

$$L_K := n_0 + n_1 + \cdots + n_K$$

- Sharing: want a larger L_K by setting $n_0 > n_1 > \cdots > n_K$

How to choose $(n_0, \ldots, n_K)$?

$$n_0 + B_1 n_1 + B_1 B_2 n_2 + B_1 B_2 B_3 n_3 + \cdots + B_1 B_2 \cdots B_K n_K = N$$

Length of each thread

$$L_K := n_0 + n_1 + \cdots + n_K$$

- Sharing: want a larger L_K by setting $n_0 > n_1 > \cdots > n_K$
- Contrasting: want $n_0 < n_1 < \cdots < n_K$

Outline

1. Deriving HiGrad

2. Constructing Confidence Intervals

3. Configuring HiGrad

4. Empirical Performance

General simulation setup

X generated as i.i.d. $\mathcal{N}(0,1)$ and $Z = (X, Y) \in \mathbb{R}^d \times \mathbb{R}$. Set $N = 10^6$ and use $\gamma_j = 0.5 j^{-0.55}$

- Linear regression $Y \sim \mathcal{N}(\mu_X(\theta^*), 1)$, where $\mu_x(\theta) = x'\theta$
- Logistic regression $Y \sim \mathrm{Bernoulli}(\mu_X(\theta^*))$, where

$$\mu_x(\theta) = \frac{\mathrm{e}^{x'\theta}}{1 + \mathrm{e}^{x'\theta}}$$

Criteria

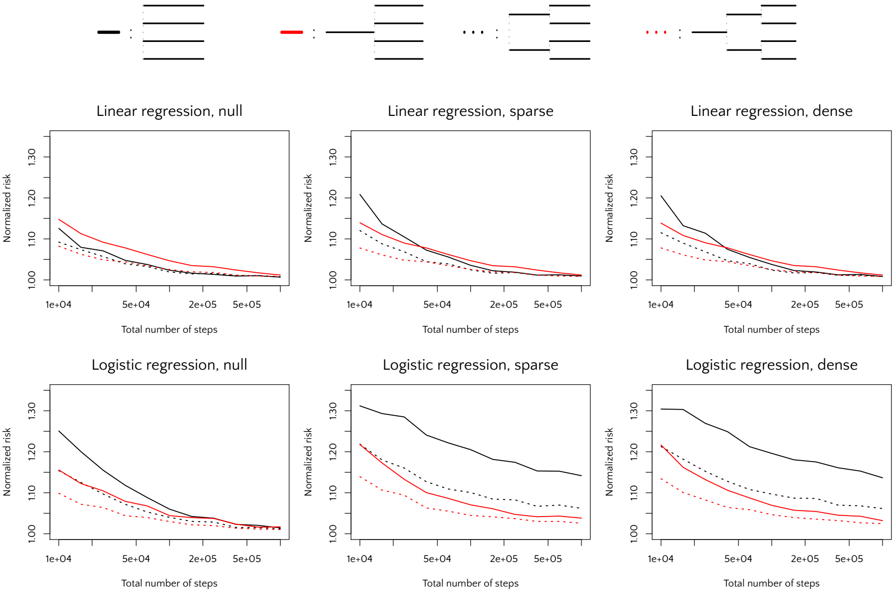
- Accuracy: $\|\overline{\theta} - \theta^*\|^2$, where $\overline{\theta}$ averaged over T threads
- Coverage probability and length of confidence interval

Accuracy

Dimension $d = 50$. MSE $\|\overline{\theta} - \theta^*\|^2$ normalized by that of vanilla SGD

- *null* case where $\theta_1 = \cdots = \theta_{50} = 0$
- *dense* case where $\theta_1 = \cdots = \theta_{50} = \frac{1}{\sqrt{50}}$
- *sparse* case where $\theta_1 = \cdots = \theta_5 = \frac{1}{\sqrt{5}}$, $\theta_6 = \cdots = \theta_{50} = 0$

Accuracy



Linear regression, null
Linear regression, sparse
Linear regression, dense
Logistic regression, null
Logistic regression, sparse
Logistic regression, dense
Normalized risk
1.00
1.10
1.20
1.30
1e+04
5e+04
2e+05
5e+05
Total number of steps

Coverage and CI length

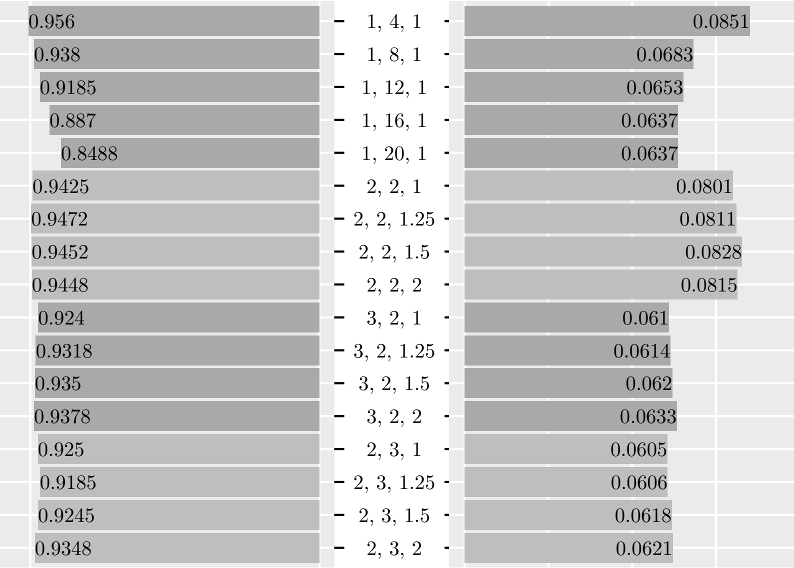
HiGrad configurations

- $K = 1$, then $n_1 = n_0 = r = 1$;
- $K = 2$, then $n_1/n_0 = n_2/n_1 = r \in \{0.75, 1, 1.25, 1.5\}$

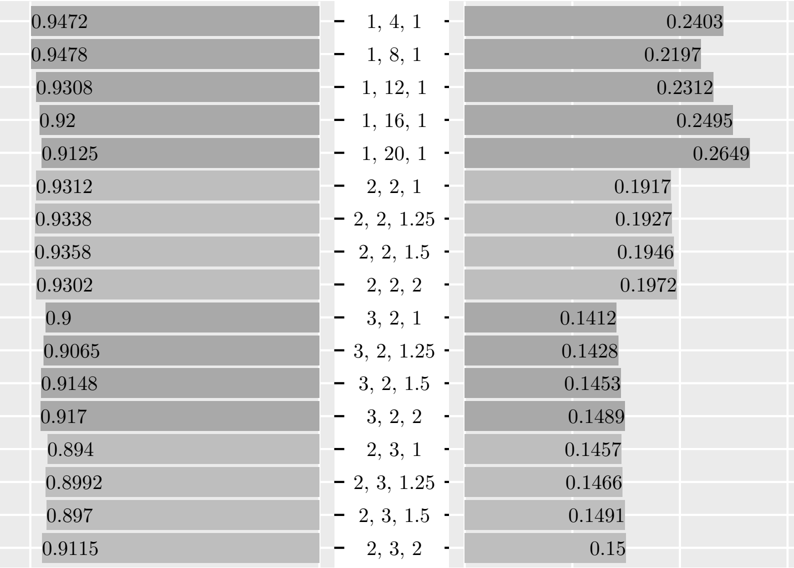
Set $\theta_i^* = (i-1)/d$ for $i = 1, \dots, d$ and $\alpha = 5\%$. Use measure

$$\frac{1}{20} \sum_{i=1}^{20} \mathbf{1}(\mu_{x_i}(\theta^*) \in \mathsf{CI}_{x_i})$$

Linear regression: $d = 20$



Linear regression: $d = 100$



A real data example: setup

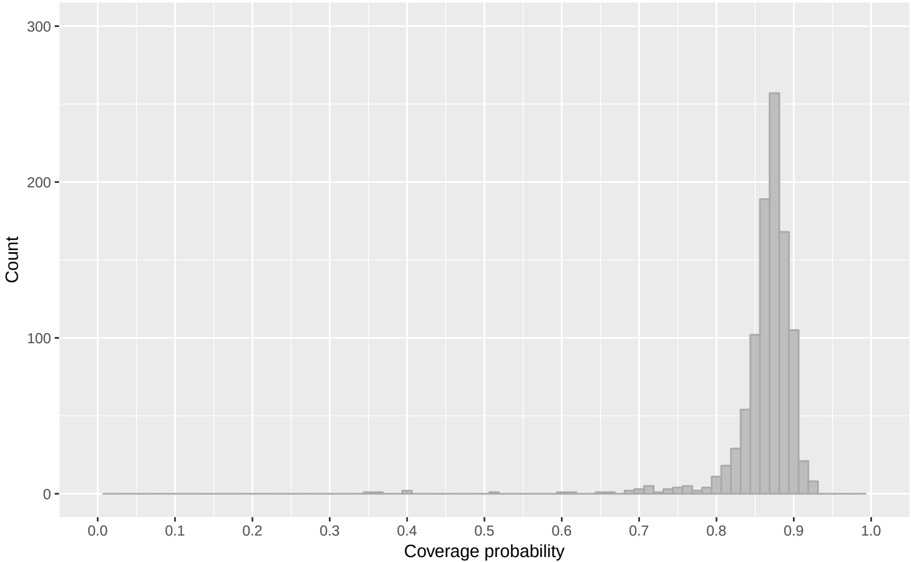
From the 1994 census data based on UCI repository. Y indicates if an individual's annual income exceeds $50,000

- 123 features
- 32,561 instances
- Randomly pick 1,000 as a test set

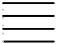
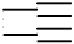
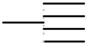
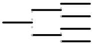
Use $N = 10^6, \alpha = 10\%$, and $\gamma_j = 0.5 j^{-0.55}$. Run HiGrad for $L = 500$ times. Use measure

$$\text{coverage}_i = \frac{1}{L(L-1)} \sum_{\ell_1}^{L} \sum_{\ell_2 \neq \ell_1} \mathbf{1}\left(\hat{p}_{i\ell_1} \in \mathrm{PI}_{i\ell_2}\right)$$

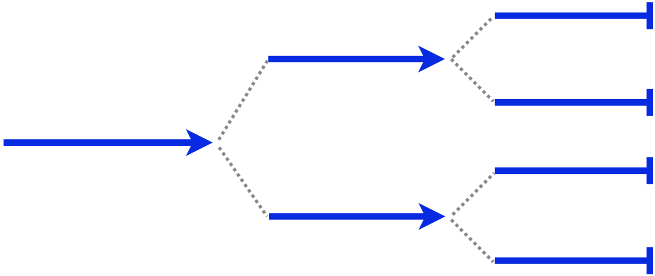
A real data example: histogram



Comparisons of HiGrad configurations

Configurations	Accuracy	Coverage	CI length
	★★★★★	☆☆☆☆☆	☆☆☆☆☆
	★★★★☆	★★★★★	★★☆☆☆
	★★★☆☆	★★★★½	★★★★☆
	★★★½☆	★★★★☆	★★★★☆
	★★★½☆	★★★★☆	★★★★☆
	★★★★☆	★★★★☆	★★★★☆

Default HiGrad parameters



HiGrad R package default values

$$K = 2, B_1 = 2, B_2 = 2, n_0 = n_1 = n_2 = \frac{N}{7}$$

Concluding Remarks

Straightforward extensions

- **Flexible tree structures**
  HiGrad tree can be asymmetric
- N **unknown**
  Grow the tree assuming a lower bound on N
- **Burn-in**
  Get a better initial point
- **A criterion for stopping**
  Need to incorporate selective inference
- **Mini-batch sizes**
  Evaluate (less) noisy gradient

$$\bar{g}(\theta, Z_{1:m}) = \frac{1}{m} \sum_{i=1}^{m} g(\theta, Z_i)$$

Future extensions

Improving statistical properties

- Finite-sample guarantee
  - Better coverage probability

Future extensions

Improving statistical properties

- Finite-sample guarantee
  - Better coverage probability
- Extend Ruppert-Polyak to high dimensions
  - Number of unknown variables growing

Future extensions

Improving statistical properties

- Finite-sample guarantee
  - Better coverage probability
- Extend Ruppert-Polyak to high dimensions
  - Number of unknown variables growing

A new template for online learning

- Adaptive step sizes and pre-conditioned SGD
  - AdaGrad (Duchi et al, 2011) and Adam (Diederik & Ba, 2014)

Future extensions

Improving statistical properties

- Finite-sample guarantee
  - Better coverage probability
- Extend Ruppert-Polyak to high dimensions
  - Number of unknown variables growing

A new template for online learning

- Adaptive step sizes and pre-conditioned SGD
  - AdaGrad (Duchi et al, 2011) and Adam (Diederik & Ba, 2014)
- General convex optimization and non-convex problems
  - SVM, regularized GLM, and deep learning

Take-home messages

Idea

Contrasting and sharing through hierarchical splitting

Take-home messages

Idea

Contrasting and sharing through hierarchical splitting

Properties (local strong convexity)

- Online in nature with same computational cost as vanilla SGD
- A confidence interval for μ_x^* in addition to an estimator
- Estimator (almost) as accurate as vanilla SGD

Take-home messages

Idea

Contrasting and sharing through hierarchical splitting

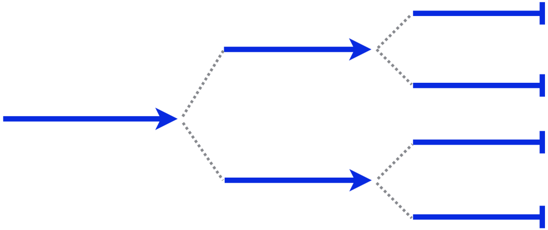
Properties (local strong convexity)

- Online in nature with same computational cost as vanilla SGD
- A confidence interval for μ_x^* in addition to an estimator
- Estimator (almost) as accurate as vanilla SGD

Bonus

Easier to parallelize than vanilla SGD!

Thanks!



- **Reference.** *Statistical Inference for Stochastic Approximation and Online Learning via Hierarchical Incremental Gradient Descent*, Weijie Su and Yuancheng Zhu, coming soon

- **Software.** R package `HiGrad`, coming soon