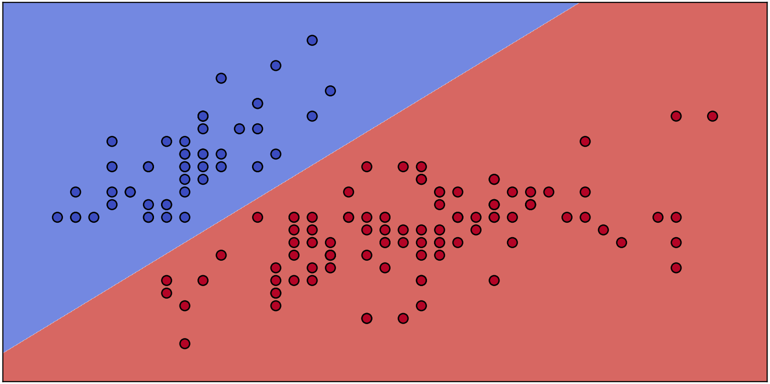


Replicable Learning of Large-Margin Halfspaces

A. Kalavasis, A. Karbasi, K.G. Larsen, G. Velegkas, F. Zhou

Yale University, Aarhus University

ICML 2024

Collaborators

Alkis Kalavasis
(Yale)

Amin Karbasi
(Yale, Google)

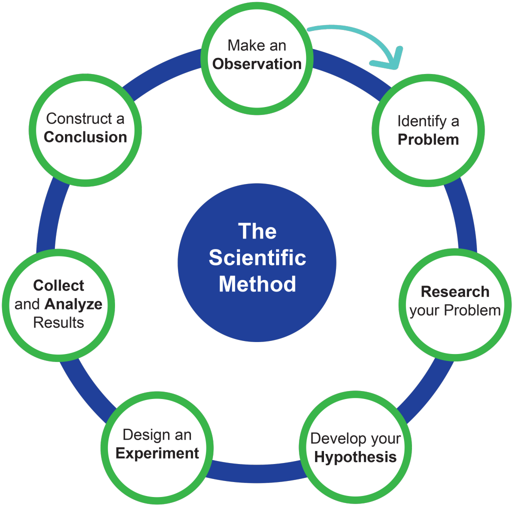
Kasper G. Larsen
(Aarhus)

Grigoris Velegkas
(Yale)

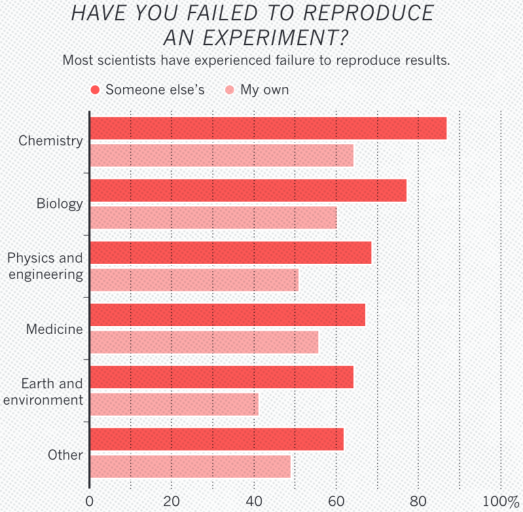
Table of Contents

Table of Contents

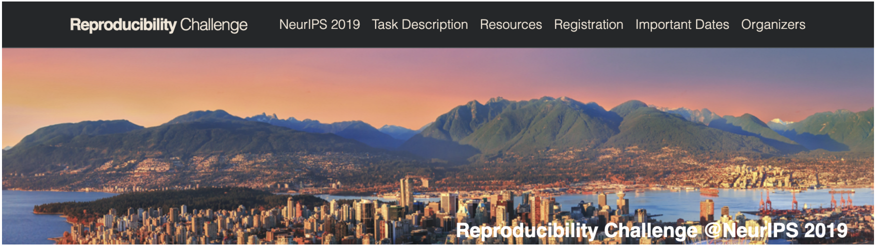
Replicability



"1500 Scientists Lift the Lid on Reproducibility."
Nature (2016)

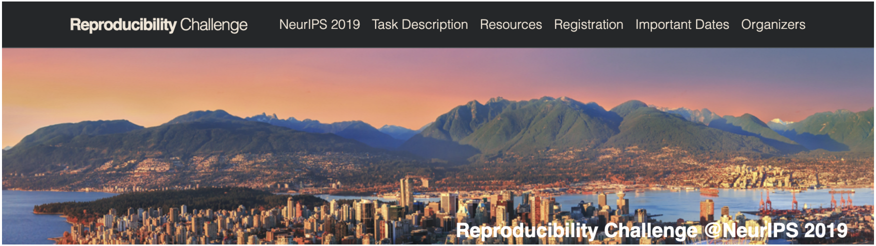


Reproducibility Crisis



- 2019 NeurIPS/ICLR Reproducibility Challenge (github.com/reproducibility-challenge)

Reproducibility Crisis



- 2019 NeurIPS/ICLR Reproducibility Challenge (github.com/reproducibility-challenge)
- Ongoing ML Reproducibility Challenge (paperswithcode.com/rc2022)

Reproducibility Crisis

Trying to develop agreed-upon set of replicable practices is difficult.

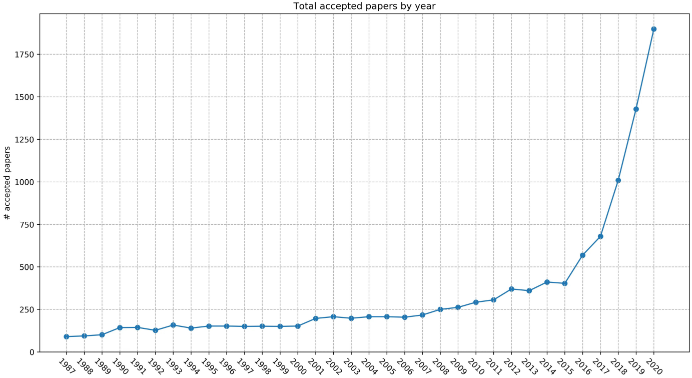


Figure: Number of accepted NeurIPS papers by year.

Algorithmic Replicability

Goal: Design ML algorithms with replicability as theoretical guarantee.

Initiated by [Impagliazzo, Lei, Pitassi, and Sorell '22] (STOC'22).

Algorithmic Replicability

- X data domain

Algorithmic Replicability

- X data domain
- $\mathcal{D}$ distribution over X

Algorithmic Replicability

- X data domain
- $\mathcal{D}$ distribution over X
- $S_1, S_2 \sim_{i.i.d.} \mathcal{D}^n$ datasets of size n

Algorithmic Replicability

- X data domain
- $\mathcal{D}$ distribution over X
- $S_1, S_2 \sim_{i.i.d.} \mathcal{D}^n$ datasets of size n
- ξ uniformly random binary string

Algorithmic Replicability

- X data domain
- $\mathcal{D}$ distribution over X
- $S_1, S_2 \sim_{i.i.d.} \mathcal{D}^n$ datasets of size n
- ξ uniformly random binary string

DEFINITION (REPLICABLE ALGORITHM; [ILPS '22])

A randomized algorithm $\mathcal{A} : X^n \to Y$ is *replicable* if $\mathcal{A}$ produces the same output on two independently drawn datasets from the same distribution.

Algorithmic Replicability

- X data domain
- $\mathcal{D}$ distribution over X
- $S_1, S_2 \sim_{i.i.d.} \mathcal{D}^n$ datasets of size n
- ξ uniformly random binary string

DEFINITION (REPLICABLE ALGORITHM; [ILPS '22])

A randomized algorithm $\mathcal{A} : X^n \to Y$ is *ρ-replicable* if $\mathcal{A}$ produces the same output on two independently drawn datasets from the same distribution.

$$\Pr_{S_1, S_2, \xi}[\mathcal{A}(S_1; \xi) \neq \mathcal{A}(S_2; \xi)] \leq \rho.$$

Yale

Algorithmic Replicability

- X data domain
- $\mathcal{D}$ distribution over X
- $S_1, S_2 \sim_{i.i.d.} \mathcal{D}^n$ datasets of size n
- ξ uniformly random binary string

DEFINITION (REPLICABLE ALGORITHM; [ILPS '22])

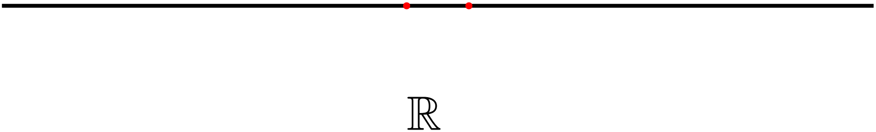
A randomized algorithm $\mathcal{A} : X^n \to Y$ is *ρ-replicable* if $\mathcal{A}$ produces the same output on two independently drawn datasets from the same distribution.

$$\Pr_{S_1, S_2, \xi} [\mathcal{A}(S_1; \xi) \neq \mathcal{A}(S_2; \xi)] \leq \rho.$$

Remark: Replicability is trivial to obtain by itself!

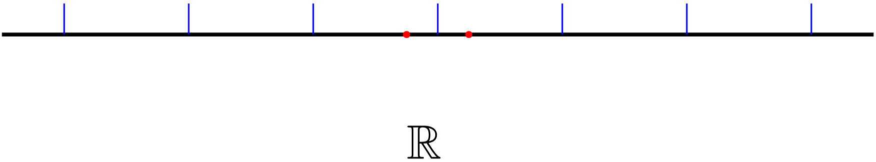
Example: Replicable Mean Estimation

1. Concentration of Measure



Example: Replicable Mean Estimation

1. Concentration of Measure
2. Discretization



Example: Replicable Mean Estimation

1. Concentration of Measure
2. Discretization
3. Shared Random Offset

$\mathbb{R}$

Table of Contents

Large-Margin Halfspaces

- Distribution $\mathcal{D}$ over $\mathbb{R}^d$

Large-Margin Halfspaces

- Distribution $\mathcal{D}$ over $\mathbb{R}^d$
  - features e.g. unlabeled images

Large-Margin Halfspaces

- Distribution $\mathcal{D}$ over $\mathbb{R}^d$
  - features e.g. unlabeled images
- Binary classifiers: halfspaces $\mathcal{H} = \{w \in \mathbb{R}^d\}$

Large-Margin Halfspaces

- Distribution $\mathcal{D}$ over $\mathbb{R}^d$
  - features e.g. unlabeled images
- Binary classifiers: halfspaces $\mathcal{H} = \{w \in \mathbb{R}^d\}$
- **Given:** Samples $(x, y^\star)$ where $x \sim \mathcal{D}$, $y^\star(x) = \mathrm{sgn}(\langle w^\star, x\rangle)$

Large-Margin Halfspaces

- Distribution $\mathcal{D}$ over $\mathbb{R}^d$
  - features e.g. unlabeled images
- Binary classifiers: halfspaces $\mathcal{H} = \{w \in \mathbb{R}^d\}$
- **Given:** Samples $(x, y^\star)$ where $x \sim \mathcal{D}$, $y^\star(x) = \operatorname{sgn}(\langle w^\star, x\rangle)$
  - label determined by "side" of the halfspace

Large-Margin Halfspaces

- Distribution $\mathcal{D}$ over $\mathbb{R}^d$
  - features e.g. unlabeled images
- Binary classifiers: halfspaces $\mathcal{H} = \{w \in \mathbb{R}^d\}$
- **Given:** Samples $(x, y^\star)$ where $x \sim \mathcal{D}$, $y^\star(x) = \mathrm{sgn}(\langle w^\star, x \rangle)$
  - label determined by "side" of the halfspace
- **Want:** $\hat{w} = \mathcal{A}(S_1)$ such that $\mathbb{E}_x[\hat{y}(x) \neq y^\star(x)] \leq \varepsilon$

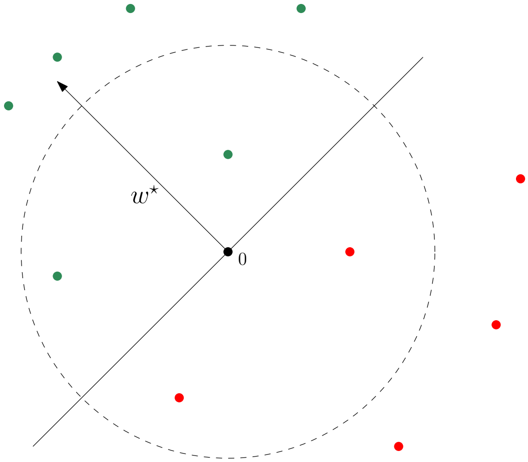
Large-Margin Halfspaces

- Distribution $\mathcal{D}$ over $\mathbb{R}^d$
  - features e.g. unlabeled images
- Binary classifiers: halfspaces $\mathcal{H} = \{w \in \mathbb{R}^d\}$
- **Given:** Samples $(x, y^\star)$ where $x \sim \mathcal{D}$, $y^\star(x) = \mathrm{sgn}(\langle w^\star, x \rangle)$
  - label determined by "side" of the halfspace
- **Want:** $\hat{w} = \mathcal{A}(S_1)$ such that $\mathbb{E}_x[\hat{y}(x) \neq y^\star(x)] \leq \varepsilon$
- And $\Pr[\mathcal{A}(S_1) \neq \mathcal{A}(S_2)] \leq \rho$ when $S_1, S_2 \sim_{i.i.d.} \mathcal{D}^n$ (ρ-Replicability)

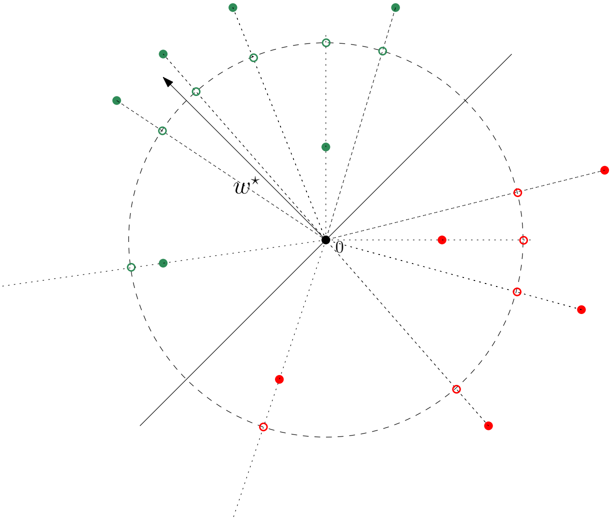
Large-Margin Halfspaces

- Distribution $\mathcal{D}$ over $\mathbb{R}^d$
  - features e.g. unlabeled images
- Binary classifiers: halfspaces $\mathcal{H} = \{w \in \mathbb{R}^d\}$
- **Given:** Samples $(x, y^\star)$ where $x \sim \mathcal{D}$, $y^\star(x) = \mathrm{sgn}(\langle w^\star, x\rangle)$
  - label determined by "side" of the halfspace
- **Want:** $\hat{w} = \mathcal{A}(S_1)$ such that $\mathbb{E}_x[\hat{y}(x) \neq y^\star(x)] \leq \varepsilon$
- And $\Pr[\mathcal{A}(S_1) \neq \mathcal{A}(S_2)] \leq \rho$ when $S_1, S_2 \sim_{i.i.d.} \mathcal{D}^n$ (ρ-Replicability)
- τ-**Margin Assumption:** "every x is τ-far from decision boundary"

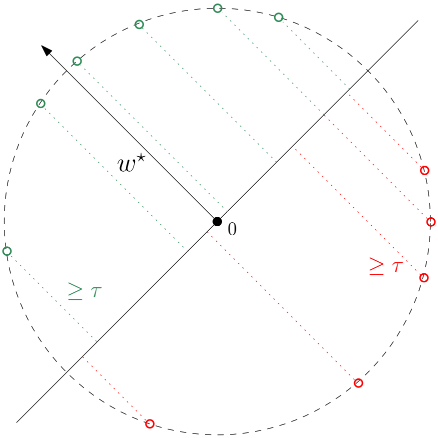
Large-Margin Halfspaces



Large-Margin Halfspaces



Large-Margin Halfspaces



Remarks

- Learning halfspaces is fundamental ML task, inspired perceptron, SVMs, AdaBoost

Remarks

- Learning halfspaces is fundamental ML task,
  inspired perceptron, SVMs, AdaBoost
- τ-margin assumption is necessary for replicably learning halfspaces

Remarks

- Learning halfspaces is fundamental ML task, inspired perceptron, SVMs, AdaBoost
- τ-margin assumption is necessary for replicably learning halfspaces
- We design first replicable algorithms for learning large-margin halfspaces with dimension-independent sample complexity.

Table of Contents

THEOREM (KKLVZ '23)

- ρ-replicable learner for τ-margin halfspaces with ε-error

Our Results

THEOREM (KKLVZ '23)

- ρ-replicable learner for τ-margin halfspaces with ε-error
- $m = \tilde{O}\left(\frac{1}{\varepsilon^2 \tau^6 \rho^2}\right)$ sample complexity

Our Results

THEOREM (KKLVZ '23)

- ρ-replicable learner for τ-margin halfspaces with ε-error
- $m = \tilde{O}\left(\frac{1}{\varepsilon^2\tau^6\rho^2}\right)$ sample complexity
- poly(m, d) running time

High-Level Idea

1. Divide data into B disjoint equisized batches

High-Level Idea

1. Divide data into B disjoint equisized batches
2. Run a (non-replicable) algorithm on each batch independently $\hat{w}_1, \ldots, \hat{w}_B$

High-Level Idea

1. Divide data into B disjoint equisized batches

2. Run a (non-replicable) algorithm on each batch independently $\hat{w}_1, \ldots, \hat{w}_B$

3. Take the average of output vectors $\bar{w} = \frac{1}{B} \sum_i \hat{w}_i$

High-Level Idea

1. Divide data into B disjoint equisized batches
2. Run a (non-replicable) algorithm on each batch independently $\hat{w}_1, \ldots, \hat{w}_B$
3. Take the average of output vectors $\bar{w} = \frac{1}{B} \sum_i \hat{w}_i$
4. Apply random projection $\bar{w} \mapsto A\bar{w}$ to reduce dimension

High-Level Idea

1. Divide data into B disjoint equisized batches
2. Run a (non-replicable) algorithm on each batch independently $\hat{w}_1, \ldots, \hat{w}_B$
3. Take the average of output vectors $\bar{w} = \frac{1}{B} \sum_i \hat{w}_i$
4. Apply random projection $\bar{w} \mapsto A\bar{w}$ to reduce dimension
5. Replicably round $A\bar{w}$ using *Alon-Klartag* [AK'17] rounding scheme

Table of Contents

Replicability

1. Initiated by Impagliazzo, Lei, Pitassi, Sorell (STOC'22), studied statistical queries and large-margin halfspaces

Replicability

1. Initiated by Impagliazzo, Lei, Pitassi, Sorell (STOC'22), studied statistical queries and large-margin halfspaces
2. Bandits [EKKKMV'23], Clustering [EKMVZ'23], Reinforcement Learning [KVYZ'23, EHKS'23]

Replicability

1. Initiated by Impagliazzo, Lei, Pitassi, Sorell (STOC'22), studied statistical queries and large-margin halfspaces
2. Bandits [EKKKMV'23], Clustering [EKMVZ'23], Reinforcement Learning [KVYZ'23, EHKS'23]
3. Equivalences to Differential Privacy (DP) etc. [KKSV'23, MSS'23, KKVZ'24]

Replicability

1. Initiated by Impagliazzo, Lei, Pitassi, Sorell (STOC'22), studied statistical queries and large-margin halfspaces
2. Bandits [EKKKMV'23], Clustering [EKMVZ'23], Reinforcement Learning [KVYZ'23, EHKS'23]
3. Equivalences to Differential Privacy (DP) etc. [KKSV'23, MSS'23, KKVZ'24]
4. List-Replicability [CMY'23, DPWV'23, CCMY'24]

(Non-Replicable) Large-Margin Halfspaces

1. $\Theta\left(\frac{1}{\varepsilon\tau^2}\right)$ sample complexity

(Non-Replicable) Large-Margin Halfspaces

1. $\Theta\left(\frac{1}{\varepsilon\tau^2}\right)$ sample complexity
2. DP algorithms [BDMN'05, NUZ'20, BCS'20, BMS'22]

(Non-Replicable) Large-Margin Halfspaces

1. $\Theta\left(\frac{1}{\varepsilon\tau^2}\right)$ sample complexity
2. DP algorithms [BDMN'05, NUZ'20, BCS'20, BMS'22]
3. Margin-independent algorithms under finite domain [BMNS'19, KMST'20]

(Non-Replicable) Large-Margin Halfspaces

1. $\Theta\left(\frac{1}{\varepsilon\tau^2}\right)$ sample complexity
2. DP algorithms [BDMN'05, NUZ'20, BCS'20, BMS'22]
3. Margin-independent algorithms under finite domain [BMNS'19, KMST'20]
4. Robust algorithms [DKM'19, DDKW'23]

Comparison with Prior Work

Replicable Algorithms for Large-Margin Halfspaces			
Algorithm	Sample Complexity	Running Time	Proper
Prior Work			
[ILPS'22] with foams rounding	$(d\varepsilon^{-3}\tau^{-8}\rho^{-2})^{1.01}$	$2^d \cdot \mathrm{poly}(1/\varepsilon, 1/\rho, 1/\tau)$	No
[ILPS'22] with box rounding	$(d^3\varepsilon^{-4}\tau^{-10}\rho^{-2})^{1.01}$	$\mathrm{poly}(d, 1/\varepsilon, 1/\rho, 1/\tau)$	No
Our Work			
SGD + Alon-Klartag rounding	$\varepsilon^{-2}\tau^{-6}\rho^{-2}$	$\mathrm{poly}(d, 1/\varepsilon, 1/\rho, 1/\tau)$	Yes
SVM + Alon-Klartag rounding	$\varepsilon^{-1}\tau^{-7}\rho^{-2}$	$\mathrm{poly}(d, 1/\varepsilon, 1/\rho, 1/\tau)$	Yes

Table of Contents

Table of Contents

Step I: Solving the Non-Replicable Problem

- Minimize population loss $\min_w \mathbb{E}_{x \sim \mathcal{D}}[\mathbb{1}\{y^\star \cdot (w^\top x) < \tau/2\}]$

Step I: Solving the Non-Replicable Problem

- Minimize population loss $\min_w \mathbb{E}_{x \sim \mathcal{D}}[\mathbb{1}\{y^\star \cdot (w^\top x) < \tau/2\}]$
  - "threshold" function of the inner product

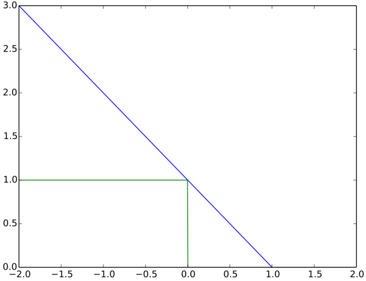
Step I: Solving the Non-Replicable Problem

- Minimize population loss $\min_w \mathbb{E}_{x \sim \mathcal{D}}[\mathbb{1}\{y^\star \cdot (w^\top x) < \tau/2\}]$
  - "threshold" function of the inner product
  - discrete, non-convex, difficult to efficiently minimize!

Step I: Solving the Non-Replicable Problem

- Minimize population loss $\min_w \mathbb{E}_{x\sim\mathcal{D}}[\mathbb{1}\{y^\star \cdot (w^\top x) < \tau/2\}]$
  - "threshold" function of the inner product
  - discrete, non-convex, difficult to efficiently minimize!

- Design a convex *surrogate* loss $\min_w \mathbb{E}_{x\sim\mathcal{D}}[\ell(w; x)]$



  -

Step I: Solving the Non-Replicable Problem

- Minimize population loss $\min_w \mathbb{E}_{x\sim\mathcal{D}}[\mathbb{1}\{y^\star \cdot (w^\top x) < \tau/2\}]$
  - "threshold" function of the inner product
  - discrete, non-convex, difficult to efficiently minimize!
- Design a convex *surrogate* loss $\min_w \mathbb{E}_{x\sim\mathcal{D}}[\ell(w; x)]$
  - cannot evaluate exactly due to expectation

Step I: Solving the Non-Replicable Problem

- Minimize population loss $\min_w \mathbb{E}_{x\sim\mathcal{D}}[\mathbb{1}\{y^\star \cdot (w^\top x) < \tau/2\}]$
  - "threshold" function of the inner product
  - discrete, non-convex, difficult to efficiently minimize!
- Design a convex *surrogate* loss $\min_w \mathbb{E}_{x\sim\mathcal{D}}[\ell(w; x)]$
  - cannot evaluate exactly due to expectation
  - cannot compute gradient exactly

Step I: Solving the Non-Replicable Problem

- Minimize population loss $\min_w \mathbb{E}_{x\sim\mathcal{D}}[\mathbb{1}\{y^\star \cdot (w^\top x) < \tau/2\}]$
  - "threshold" function of the inner product
  - discrete, non-convex, difficult to efficiently minimize!
- Design a convex *surrogate* loss $\min_w \mathbb{E}_{x\sim\mathcal{D}}[\ell(w;x)]$
  - cannot evaluate exactly due to expectation
  - cannot compute gradient exactly
- We have $\nabla \mathbb{E}_{x\sim\mathcal{D}}[\ell(w;x)] = \mathbb{E}_{x\sim\mathcal{D}}[\nabla \ell(w;x)]$

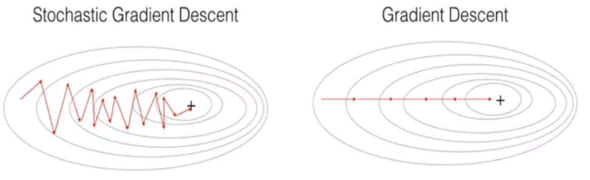
Step I: Solving the Non-Replicable Problem

- Minimize population loss $\min_w \mathbb{E}_{x\sim\mathcal{D}}[\mathbb{1}\{y^\star \cdot (w^\top x) < \tau/2\}]$
  - "threshold" function of the inner product
  - discrete, non-convex, difficult to efficiently minimize!
- Design a convex *surrogate* loss $\min_w \mathbb{E}_{x\sim\mathcal{D}}[\ell(w;x)]$
  - cannot evaluate exactly due to expectation
  - cannot compute gradient exactly
- We have $\nabla \mathbb{E}_{x\sim\mathcal{D}}[\ell(w;x)] = \mathbb{E}_{x\sim\mathcal{D}}[\nabla \ell(w;x)]$
  - $\nabla \ell(w;x)$ is *unbiased* estimate of gradient for $x \sim \mathcal{D}$

Stochastic Gradient Descent

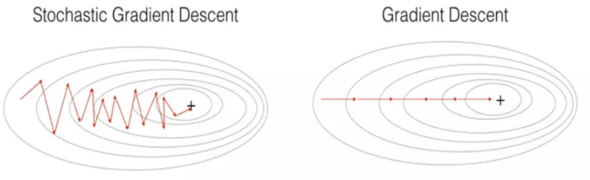
THEOREM

Assume an unbiased stochastic gradient oracle $\hat{g}(w)$ for "nice" convex function $f(w)$. SGD $w^{t+1} \leftarrow w^t - \eta_t \hat{g}(w^t)$ outputs an ε-optimal solution in $O(1/\varepsilon^2)$ iterations.



THEOREM

Assume an unbiased stochastic gradient oracle $\hat{g}(w)$ for "nice" convex function $f(w)$. SGD $w^{t+1} \leftarrow w^t - \eta_t \hat{g}(w^t)$ outputs an ε-optimal solution in $O(1/\varepsilon^2)$ iterations.



- τ-margin hinge loss $\ell(w; x) = \max(0, 2 - \frac{2y^\star}{\tau} \cdot (w^\top x))$ is "nice"

Table of Contents

Step II: Dimensionality Reduction

- Repeat Step I with disjoint data to obtain solutions $\hat{w}^{(1)}, \ldots, \hat{w}^{(B)}$ and compute average solution $\bar{w} = \frac{1}{B} \sum_i \hat{w}^{(i)}$

Step II: Dimensionality Reduction

- Repeat Step I with disjoint data to obtain solutions $\hat{w}^{(1)}, \ldots, \hat{w}^{(B)}$ and compute average solution $\bar{w} = \frac{1}{B} \sum_i \hat{w}^{(i)}$
  - By convexity, $\mathbb{E}_{x \sim \mathcal{D}}[\ell(\bar{w}; x)] \leq \varepsilon$ as well!

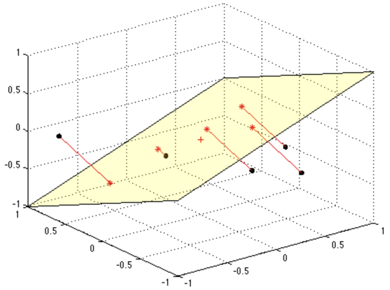
Yale

Step II: Dimensionality Reduction

- Repeat Step I with disjoint data to obtain solutions $\hat{w}^{(1)}, \ldots, \hat{w}^{(B)}$ and compute average solution $\bar{w} = \frac{1}{B} \sum_i \hat{w}^{(i)}$
  - By convexity, $\mathbb{E}_{x \sim \mathcal{D}}[\ell(\bar{w}; x)] \leq \varepsilon$ as well!
  - By independence, $\bar{w}$ concentrates about its mean!

Step II: Dimensionality Reduction

- Repeat Step I with disjoint data to obtain solutions $\hat{w}^{(1)}, \ldots, \hat{w}^{(B)}$ and compute average solution $\bar{w} = \frac{1}{B} \sum_i \hat{w}^{(i)}$
  - By convexity, $\mathbb{E}_{x \sim \mathcal{D}}[\ell(\bar{w}; x)] \leq \varepsilon$ as well!
  - By independence, $\bar{w}$ concentrates about its mean!
- Random orthogonal projection $A : \mathbb{R}^d \to \mathbb{R}^k$ for $k(\tau) << d$



Step II: Dimensionality Reduction

- Repeat Step I with disjoint data to obtain solutions $\hat{w}^{(1)}, \ldots, \hat{w}^{(B)}$ and compute average solution $\bar{w} = \frac{1}{B}\sum_i \hat{w}^{(i)}$
  - By convexity, $\mathbb{E}_{x\sim\mathcal{D}}[\ell(\bar{w}; x)] \leq \varepsilon$ as well!
  - By independence, $\bar{w}$ concentrates about its mean!
- Random orthogonal projection $A : \mathbb{R}^d \to \mathbb{R}^k$ for $k(\tau) << d$
  - projection preserves distances and inner products (solution quality)

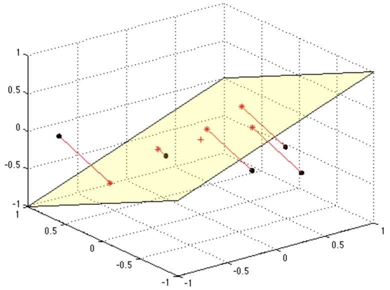


Table of Contents

Step III: Replicable Rounding

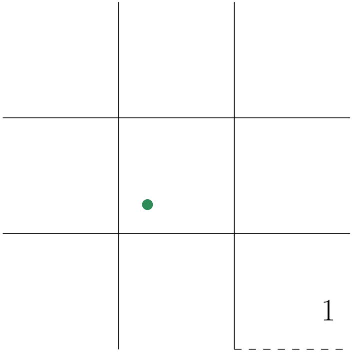
- Assume now the ambient dimension is $k << d$

Step III: Replicable Rounding

- Assume now the ambient dimension is $k << d$
- Suppose there are two vectors $w, \tilde{w} \in \mathbb{R}^k$ with $\|w - \tilde{w}\|_1$ small

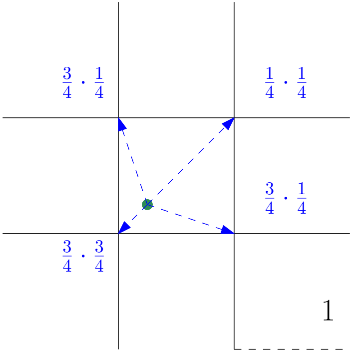
Step III: Replicable Rounding

- Assume now the ambient dimension is $k << d$
- Suppose there are two vectors $w, \tilde{w} \in \mathbb{R}^k$ with $\|w - \tilde{w}\|_1$ small
- Implicitly impose a grid of length L on $\mathbb{R}^k$



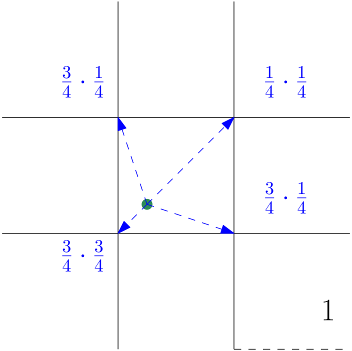
Step III: Replicable Rounding

- Assume now the ambient dimension is $k << d$
- Suppose there are two vectors $w, \tilde{w} \in \mathbb{R}^k$ with $\|w - \tilde{w}\|_1$ small
- Implicitly impose a grid of length L on $\mathbb{R}^k$
- Round to vertex $v = \texttt{round}(w)$ with prob. $\propto \prod_{j=1}^{k}(L - |w_i - v_i|)$



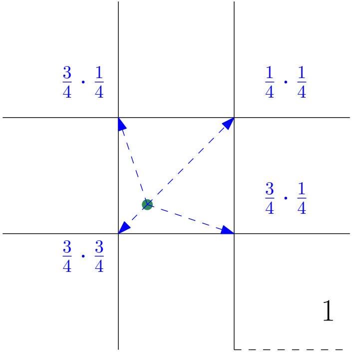
Alon-Klartag Rounding

- Round to vertex $v = \texttt{round}(w)$ with prob. $\propto \prod_{j=1}^{k}(L - |w_i - v_i|)$



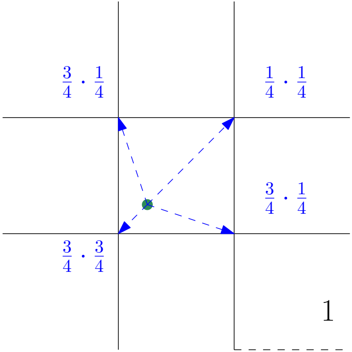
Alon-Klartag Rounding

- Round to vertex $v = \texttt{round}(w)$ with prob. $\propto \prod_{j=1}^{k} (L - |w_i - v_i|)$
- $\mathbb{E}[\texttt{round}(w)] = w$ so rounding preserves inner products



Alon-Klartag Rounding

- Round to vertex $v = \texttt{round}(w)$ with prob. $\propto \prod_{j=1}^{k}(L - |w_i - v_i|)$
- $\mathbb{E}[\texttt{round}(w)] = w$ so rounding preserves inner products
- $\texttt{round}(w) = \texttt{round}(\tilde{w})$ with probability $O(\|w - \tilde{w}\|_1)$



Summary

1. Solve non-replicable problem multiple times $\rightarrow w^{(1)}, \ldots, w^{(B)}$, take average $\bar{w}$

Summary

1. Solve non-replicable problem multiple times $\rightarrow w^{(1)}, \ldots, w^{(B)}$, take average $\bar{w}$
   - average remains good solution

Yale

Summary

1. Solve non-replicable problem multiple times $\rightarrow w^{(1)}, \ldots, w^{(B)}$, take average $\bar{w}$
   - average remains good solution
   - average will be close across executions

Yale

Summary

1. Solve non-replicable problem multiple times $\rightarrow w^{(1)}, \ldots, w^{(B)}$, take average $\bar{w}$
   - average remains good solution
   - average will be close across executions

2. Randomly project $\bar{w}$ to lower-dimension

Yale

Summary

1. Solve non-replicable problem multiple times $\rightarrow w^{(1)}, \ldots, w^{(B)}$, take average $\bar{w}$
   - average remains good solution
   - average will be close across executions

2. Randomly project $\bar{w}$ to lower-dimension
   - projection preserves inner products (and solution quality)

Yale

Summary

1. Solve non-replicable problem multiple times $\rightarrow w^{(1)}, \dots, w^{(B)}$, take average $\bar{w}$
   - average remains good solution
   - average will be close across executions
2. Randomly project $\bar{w}$ to lower-dimension
   - projection preserves inner products (and solution quality)
   - projection preserves distances
     (projected solutions remain close across executions)

Summary

1. Solve non-replicable problem multiple times $\rightarrow w^{(1)}, \ldots, w^{(B)}$, take average $\bar{w}$
   - average remains good solution
   - average will be close across executions
2. Randomly project $\bar{w}$ to lower-dimension
   - projection preserves inner products (and solution quality)
   - projection preserves distances
     (projected solutions remain close across executions)
3. Round to corners of grid

Summary

1. Solve non-replicable problem multiple times $\to w^{(1)}, \dots, w^{(B)}$, take average $\bar{w}$
   - average remains good solution
   - average will be close across executions
2. Randomly project $\bar{w}$ to lower-dimension
   - projection preserves inner products (and solution quality)
   - projection preserves distances
     (projected solutions remain close across executions)
3. Round to corners of grid
   - rounding is unbiased (preserves inner products and solution quality)

Summary

1. Solve non-replicable problem multiple times $\rightarrow w^{(1)}, \ldots, w^{(B)}$, take average $\bar{w}$
   - average remains good solution
   - average will be close across executions
2. Randomly project $\bar{w}$ to lower-dimension
   - projection preserves inner products (and solution quality)
   - projection preserves distances
     (projected solutions remain close across executions)
3. Round to corners of grid
   - rounding is unbiased (preserves inner products and solution quality)
   - rounds closeby solutions to the same point

Future Work

- Improve sample complexity of replicable algorithms

Future Work

- Improve sample complexity of replicable algorithms
- Sample complexity lower bounds specifically for replicability

Thank you!

arxiv.org/abs/2402.13857

felix-zhou.com

felix.zhou@yale.edu