Importance Sampling for Minibatches

Dominik Csiba

School of Mathematics
University of Edinburgh

07.09.2016, Birmingham

Acknowledgements

This talk is based on [Csiba and Richtárik, 2016].

Co-author:

Peter Richtárik
University of Edinburgh

Introduction

Many supervised learning tasks can be written as optimization problems.

Introduction

Many supervised learning tasks can be written as optimization problems.

Regularized Loss Minimization (RLM)

Let $f_i : \mathbb{R}^d \to \mathbb{R}$ be a loss function associated with the i-th example and $R : \mathbb{R}^d \to \mathbb{R}$ be a regularizer. Then the RLM problem is defined as

$$\mathbf{w}^* \quad = \quad \arg\min_{\mathbf{w} \in \mathbb{R}^d} \left\{ F(\mathbf{w}) \triangleq \frac{1}{n} \sum_{i=1}^{n} f_i(\mathbf{w}) + R(\mathbf{w}) \right\}$$

Introduction

Many supervised learning tasks can be written as optimization problems.

Regularized Loss Minimization (RLM)

Let $f_i : \mathbb{R}^d \to \mathbb{R}$ be a loss function associated with the i-th example and $R : \mathbb{R}^d \to \mathbb{R}$ be a regularizer. Then the RLM problem is defined as

$$\mathbf{w}^* \quad = \quad \arg\min_{\mathbf{w}\in\mathbb{R}^d} \left\{ F(\mathbf{w}) \triangleq \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{w}) + R(\mathbf{w}) \right\}$$

Ridge Regression: training set $\{(\mathbf{x}^i, y_i)\}_{i=1}^n$, linear regression with square loss $f_i(\mathbf{w}) = \frac{1}{2}(\langle \mathbf{x}^i, \mathbf{w}\rangle - y_i)^2$, squared ℓ_2 regularizer $R(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|_2^2$

Introduction

Many supervised learning tasks can be written as optimization problems.

Regularized Loss Minimization (RLM)

Let $f_i : \mathbb{R}^d \to \mathbb{R}$ be a loss function associated with the i-th example and $R : \mathbb{R}^d \to \mathbb{R}$ be a regularizer. Then the RLM problem is defined as

$$\mathbf{w}^* \quad = \quad \arg\min_{\mathbf{w}\in\mathbb{R}^d} \left\{ F(\mathbf{w}) \triangleq \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{w}) + R(\mathbf{w}) \right\}$$

Ridge Regression: training set $\{(\mathbf{x}^i, y_i)\}_{i=1}^n$, linear regression with square loss $f_i(\mathbf{w}) = \frac{1}{2}(\langle \mathbf{x}^i, \mathbf{w}\rangle - y_i)^2$, squared ℓ_2 regularizer $R(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|_2^2$

Direct solution: usually not available $\to$ iterative methods

Introduction

Many supervised learning tasks can be written as optimization problems.

Regularized Loss Minimization (RLM)

Let $f_i : \mathbb{R}^d \to \mathbb{R}$ be a loss function associated with the i-th example and $R : \mathbb{R}^d \to \mathbb{R}$ be a regularizer. Then the RLM problem is defined as

$$\mathbf{w}^* \quad = \quad \arg\min_{\mathbf{w}\in\mathbb{R}^d} \left\{ F(\mathbf{w}) \triangleq \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{w}) + R(\mathbf{w}) \right\}$$

Ridge Regression: training set $\{(\mathbf{x}^i, y_i)\}_{i=1}^n$, linear regression with square loss $f_i(\mathbf{w}) = \frac{1}{2}(\langle \mathbf{x}^i, \mathbf{w}\rangle - y_i)^2$, squared ℓ_2 regularizer $R(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|_2^2$

Direct solution: usually not available $\to$ iterative methods

Batch methods: very expensive iterations $\to$ stochastic methods

Introduction

Many supervised learning tasks can be written as optimization problems.

Regularized Loss Minimization (RLM)

Let $f_i : \mathbb{R}^d \to \mathbb{R}$ be a loss function associated with the i-th example and $R : \mathbb{R}^d \to \mathbb{R}$ be a regularizer. Then the RLM problem is defined as

$$\mathbf{w}^* \quad = \quad \arg\min_{\mathbf{w}\in\mathbb{R}^d} \left\{ F(\mathbf{w}) \triangleq \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{w}) + R(\mathbf{w}) \right\}$$

Ridge Regression: training set $\{(\mathbf{x}^i, y_i)\}_{i=1}^n$, linear regression with square loss $f_i(\mathbf{w}) = \frac{1}{2}(\langle \mathbf{x}^i, \mathbf{w}\rangle - y_i)^2$, squared ℓ_2 regularizer $R(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|_2^2$

Direct solution: usually not available $\to$ iterative methods

Batch methods: very expensive iterations $\to$ stochastic methods

Most popular methods $\to$ **stochastic iterative methods**

Randomized Coordinate Descent

NSync [Richtárik and Takáč, 2015]

Sampling $\hat{S}$ - random set-valued mapping with values subsets of $\{1,\dots,d\}$, such that $P(j \in \hat{S}) > 0$ for every $j \in \{1,\dots,d\}$

Stepsize parameters $v_1,\dots,v_d > 0$ computable from $(F, \hat{S})$

Algorithm: on each iteration $t > 0$ do

1. sample a random set S_t using $\hat{S}$
2. for each $j \in S_t$, update

$$w_j^t \leftarrow w_j^{t-1} - \frac{1}{v_j}\frac{\partial}{\partial w_j}F(\mathbf{w}^t)$$

Randomized Coordinate Descent

NSync [Richtárik and Takáč, 2015]

Sampling $\hat{S}$ - random set-valued mapping with values subsets of $\{1,\dots,d\}$, such that $P(j \in \hat{S}) > 0$ for every $j \in \{1,\dots,d\}$

Stepsize parameters $v_1,\dots,v_d > 0$ computable from $(F, \hat{S})$

Algorithm: on each iteration $t > 0$ do

1. sample a random set S_t using $\hat{S}$
2. for each $j \in S_t$, update

$$w_j^t \leftarrow w_j^{t-1} - \frac{1}{v_j}\frac{\partial}{\partial w_j}F(\mathbf{w}^t)$$

Serial sampling: $P(|\hat{S}| = 1) = 1$

Randomized Coordinate Descent

NSync [Richtárik and Takáč, 2015]

Sampling $\hat{S}$ - random set-valued mapping with values subsets of $\{1,\dots,d\}$, such that $P(j \in \hat{S}) > 0$ for every $j \in \{1,\dots,d\}$

Stepsize parameters $v_1,\dots,v_d > 0$ computable from $(F, \hat{S})$

Algorithm: on each iteration $t > 0$ do

1. sample a random set S_t using $\hat{S}$
2. for each $j \in S_t$, update

$$w_j^t \leftarrow w_j^{t-1} - \frac{1}{v_j}\frac{\partial}{\partial w_j}F(\mathbf{w}^t)$$

Serial sampling: $P(|\hat{S}| = 1) = 1$

Ridge Regression using serial sampling: $v_j = \frac{1}{n}\sum_{i=1}^n (x_j^i)^2 + \lambda$

Expected Separable Overapproximation

Assumption 1: ESO [Qu and Richtárik, 2014]

Let the objective F and the sampling $\hat{S}$ be given and let $p_j \triangleq P(j \in \hat{S})$. Let $\mathbf{h}_{[\hat{S}]}$ be a vector defined by the entries

$$\left(\mathbf{h}_{[\hat{S}]}\right)_j = \begin{cases} h_j & \text{if } j \in \hat{S} \\ 0 & \text{if } j \notin \hat{S} \end{cases}$$

The parameters $v_1, \ldots, v_d$ **satisfy the ESO** if for all $\mathbf{w}, \mathbf{h} \in \mathbb{R}^d$ it holds that

$$\mathbb{E}[F(\mathbf{w} + \mathbf{h}_{[\hat{S}]})] \leq F(\mathbf{w}) + \sum_{j=1}^{d} p_j h_j \frac{\partial}{\partial w_j} F(\mathbf{w}) + \sum_{j=1}^{d} p_j v_j h_j^2$$

and we say that $v_1, \ldots, v_d \sim ESO(F, \hat{S})$.

Expected Separable Overapproximation

Assumption 1: ESO [Qu and Richtárik, 2014]

Let the objective F and the sampling $\hat{S}$ be given and let $p_j \triangleq P(j \in \hat{S})$. Let $\mathbf{h}_{[\hat{S}]}$ be a vector defined by the entries

$$\left(\mathbf{h}_{[\hat{S}]}\right)_j = \begin{cases} h_j & \text{if } j \in \hat{S} \\ 0 & \text{if } j \notin \hat{S} \end{cases}$$

The parameters $v_1, \ldots, v_d$ **satisfy the ESO** if for all $\mathbf{w}, \mathbf{h} \in \mathbb{R}^d$ it holds that

$$\mathbb{E}[F(\mathbf{w} + \mathbf{h}_{[\hat{S}]})] \leq F(\mathbf{w}) + \sum_{j=1}^{d} p_j h_j \frac{\partial}{\partial w_j} F(\mathbf{w}) + \sum_{j=1}^{d} p_j v_j h_j^2$$

and we say that $v_1, \ldots, v_d \sim ESO(F, \hat{S})$.

Intepretation: The expectation of F at the update behaves smoothly

Strong Convexity

Assumption 2: Strong Convexity

Function F is λ-**strongly convex** if for all $\mathbf{w}, \mathbf{h} \in \mathbb{R}^d$ it holds that

$$F(\mathbf{w}+\mathbf{h}) \geq F(\mathbf{w}) + \langle \nabla F(\mathbf{w}), \mathbf{h} \rangle + \frac{\lambda}{2}\|\mathbf{w}\|_2^2$$

Strong Convexity

Assumption 2: Strong Convexity

Function F is λ-**strongly convex** if for all $\mathbf{w}, \mathbf{h} \in \mathbb{R}^d$ it holds that

$$F(\mathbf{w}+\mathbf{h}) \geq F(\mathbf{w}) + \langle \nabla F(\mathbf{w}), \mathbf{h} \rangle + \frac{\lambda}{2}\|\mathbf{w}\|_2^2$$

Example: $R(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|_2^2$ is λ-strongly convex

Strong Convexity

Assumption 2: Strong Convexity

Function F is λ-**strongly convex** if for all $\mathbf{w}, \mathbf{h} \in \mathbb{R}^d$ it holds that

$$F(\mathbf{w}+\mathbf{h}) \geq F(\mathbf{w}) + \langle \nabla F(\mathbf{w}), \mathbf{h} \rangle + \frac{\lambda}{2}\|\mathbf{w}\|_2^2$$

Example: $R(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|_2^2$ is λ-strongly convex

Lemma: convex + strongly convex $\rightarrow$ strongly convex

Strong Convexity

Assumption 2: Strong Convexity

Function F is λ-**strongly convex** if for all $\mathbf{w}, \mathbf{h} \in \mathbb{R}^d$ it holds that

$$F(\mathbf{w}+\mathbf{h}) \geq F(\mathbf{w}) + \langle \nabla F(\mathbf{w}), \mathbf{h} \rangle + \frac{\lambda}{2}\|\mathbf{w}\|_2^2$$

Example: $R(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|_2^2$ is λ-strongly convex

Lemma: convex + strongly convex $\rightarrow$ strongly convex

Ridge logistic regression is λ-strongly convex

$$F(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} \log(1+\exp(-y_i\langle \mathbf{w}, \mathbf{x}^i \rangle)) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2$$

Strong Convexity

Assumption 2: Strong Convexity

Function F is λ-**strongly convex** if for all $\mathbf{w}, \mathbf{h} \in \mathbb{R}^d$ it holds that

$$F(\mathbf{w}+\mathbf{h}) \geq F(\mathbf{w}) + \langle \nabla F(\mathbf{w}), \mathbf{h} \rangle + \frac{\lambda}{2}\|\mathbf{w}\|_2^2$$

Example: $R(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|_2^2$ is λ-strongly convex

Lemma: convex + strongly convex $\rightarrow$ strongly convex

Ridge logistic regression is λ-strongly convex

$$F(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} \log(1+\exp(-y_i\langle \mathbf{w}, \mathbf{x}^i \rangle)) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2$$

Ridge regression is λ-strongly convex

Convergence Rate

Theorem [Richtárik and Takáč, 2015]

Let the objective F and the sampling $\hat{S}$ be such that F is λ-strongly convex and $v_1, \dots v_d \sim ESO(F, \hat{S})$ and let $\mathbf{v} \triangleq [v_1, \dots, v_d]$. Let $p_j = P(j \in \hat{S})$ and let $\mathbf{p} \triangleq [p_1, \dots p_d]$. Let

$$C(\mathbf{p}, \mathbf{v}) \triangleq \max_{j \in \{1,\dots,d\}} \left(\frac{v_j}{p_j \lambda} \right),$$

and let $\{\mathbf{w}^t\}_{t=1}^{\infty}$ be a sequence generated by **NSync**. Then

$$T \geq C(\mathbf{p}, \mathbf{v}) \log \left(\frac{C}{\epsilon} \right) \quad \Rightarrow \quad \mathbb{E}[F(\mathbf{w}^T) - F(\mathbf{w}^*)] \leq \epsilon,$$

where C is an absolute constant depending on $\mathbf{w}^0$ and $\mathbf{w}^*$.

Convergence Rate

Theorem [Richtárik and Takáč, 2015]

Let the objective F and the sampling $\hat{S}$ be such that F is λ-strongly convex and $v_1, \dots v_d \sim ESO(F, \hat{S})$ and let $\mathbf{v} \triangleq [v_1, \dots, v_d]$. Let $p_j = P(j \in \hat{S})$ and let $\mathbf{p} \triangleq [p_1, \dots p_d]$. Let

$$C(\mathbf{p}, \mathbf{v}) \triangleq \max_{j \in \{1,\dots,d\}} \left(\frac{v_j}{p_j \lambda} \right),$$

and let $\{\mathbf{w}^t\}_{t=1}^{\infty}$ be a sequence generated by **NSync**. Then

$$T \geq C(\mathbf{p}, \mathbf{v}) \log \left(\frac{C}{\epsilon} \right) \quad \Rightarrow \quad \mathbb{E}[F(\mathbf{w}^T) - F(\mathbf{w}^*)] \leq \epsilon,$$

where C is an absolute constant depending on $\mathbf{w}^0$ and $\mathbf{w}^*$.

Key quantity: $C(\mathbf{p}, \mathbf{v})$

Serial Importance Sampling

Let $\hat{S}$ be a serial sampling defined by $P(\{j\} = \hat{S}) = p_j$, where $\sum_j p_j = 1$.

Serial Importance Sampling

Let $\hat{S}$ be a serial sampling defined by $P(\{j\} = \hat{S}) = p_j$, where $\sum_j p_j = 1$.

Ridge regression using serial sampling: $v_j = \frac{1}{n}\sum_{i=1}^{n}(x_j^i)^2 + \lambda$

Serial Importance Sampling

Let $\hat{S}$ be a serial sampling defined by $P(\{j\} = \hat{S}) = p_j$, where $\sum_j p_j = 1$.

Ridge regression using serial sampling: $v_j = \frac{1}{n}\sum_{i=1}^{n}(x_j^i)^2 + \lambda$

Question: How to choose $\mathbf{p}$, so that the complexity

$$C(\mathbf{p}, \mathbf{v}) \triangleq \max_{j \in \{1,\ldots,d\}} \left(\frac{v_j}{p_j \lambda} \right)$$

is minimized?

Serial Importance Sampling

Let $\hat{S}$ be a serial sampling defined by $P(\{j\} = \hat{S}) = p_j$, where $\sum_j p_j = 1$.

Ridge regression using serial sampling: $v_j = \frac{1}{n}\sum_{i=1}^{n}(x_j^i)^2 + \lambda$

Question: How to choose $\mathbf{p}$, so that the complexity

$$C(\mathbf{p}, \mathbf{v}) \triangleq \max_{j \in \{1,\ldots,d\}} \left(\frac{v_j}{p_j \lambda} \right)$$

is minimized?

sampling	p_j	$C(\mathbf{p}, \mathbf{v})$
uniform	$1/d$	$(d \cdot \max_{j \in \{1,\ldots,d\}} v_j)/\lambda$

Serial Importance Sampling

Let $\hat{S}$ be a serial sampling defined by $P(\{j\} = \hat{S}) = p_j$, where $\sum_j p_j = 1$.

Ridge regression using serial sampling: $v_j = \frac{1}{n}\sum_{i=1}^{n}(x_j^i)^2 + \lambda$

Question: How to choose $\mathbf{p}$, so that the complexity

$$C(\mathbf{p}, \mathbf{v}) \triangleq \max_{j \in \{1,\ldots,d\}} \left(\frac{v_j}{p_j \lambda} \right)$$

is minimized?

sampling	p_j	$C(\mathbf{p}, \mathbf{v})$
uniform	$1/d$	$(d \cdot \max_{j \in \{1,\ldots,d\}} v_j)/\lambda$
importance	$v_j/(\sum_k v_k)$	$(\sum_k v_k)/\lambda$

Serial Importance Sampling

Let $\hat{S}$ be a serial sampling defined by $P(\{j\} = \hat{S}) = p_j$, where $\sum_j p_j = 1$.

Ridge regression using serial sampling: $v_j = \frac{1}{n}\sum_{i=1}^{n}(x_j^i)^2 + \lambda$

Question: How to choose $\mathbf{p}$, so that the complexity

$$C(\mathbf{p}, \mathbf{v}) \triangleq \max_{j \in \{1,\ldots,d\}} \left(\frac{v_j}{p_j \lambda} \right)$$

is minimized?

sampling	p_j	$C(\mathbf{p}, \mathbf{v})$
uniform	$1/d$	$(d \cdot \max_{j\in\{1,\ldots,d\}} v_j)/\lambda$
importance	$v_j/(\sum_k v_k)$	$(\sum_k v_k)/\lambda$

The speedup is proportional to mean($\mathbf{v}$) vs. maximum($\mathbf{v}$)

Serial Importance Sampling

Let $\hat{S}$ be a serial sampling defined by $P(\{j\} = \hat{S}) = p_j$, where $\sum_j p_j = 1$.

Ridge regression using serial sampling: $v_j = \frac{1}{n}\sum_{i=1}^{n}(x_j^i)^2 + \lambda$

Question: How to choose $\mathbf{p}$, so that the complexity

$$C(\mathbf{p}, \mathbf{v}) \triangleq \max_{j \in \{1,\ldots,d\}} \left(\frac{v_j}{p_j \lambda} \right)$$

is minimized?

sampling	p_j	$C(\mathbf{p}, \mathbf{v})$
uniform	$1/d$	$(d \cdot \max_{j \in \{1,\ldots,d\}} v_j)/\lambda$
importance	$v_j/(\sum_k v_k)$	$(\sum_k v_k)/\lambda$

The speedup is proportional to mean($\mathbf{v}$) vs. maximum($\mathbf{v}$)

Note: importance sampling is the optimal fixed serial sampling

Minibatch Coordinate Descent

Assume we have τ cores and we want to make use of all of them.

Minibatch Coordinate Descent

Assume we have τ cores and we want to make use of all of them.

Parallel sampling: $P(|\hat{S}| = \tau) = 1$, each core processes one coordinate

Minibatch Coordinate Descent

Assume we have τ cores and we want to make use of all of them.

Parallel sampling: $P(|\hat{S}| = \tau) = 1$, each core processes one coordinate

Question: How to design importance sampling for minibatches?

Minibatch Coordinate Descent

Assume we have τ cores and we want to make use of all of them.

Parallel sampling: $P(|\hat{S}| = \tau) = 1$, each core processes one coordinate

Question: How to design importance sampling for minibatches?

Complications:

Minibatch Coordinate Descent

Assume we have τ cores and we want to make use of all of them.

Parallel sampling: $P(|\hat{S}| = \tau) = 1$, each core processes one coordinate

Question: How to design importance sampling for minibatches?

Complications:

- for serial sampling we had $\underbrace{P(\{j\} = \hat{S})}_{\text{parameter}} = P(j \in \hat{S})$, which is not the case for parallel

Minibatch Coordinate Descent

Assume we have τ cores and we want to make use of all of them.

Parallel sampling: $P(|\hat{S}| = \tau) = 1$, each core processes one coordinate

Question: How to design importance sampling for minibatches?

Complications:

- for serial sampling we had $\underbrace{P(\{j\} = \hat{S})}_{\text{parameter}} = P(j \in \hat{S})$, which is not the case for parallel
- We need to choose $\binom{d}{\tau}$ set-probabilities, instead of just $\binom{d}{1} = \text{d}$

Minibatch Coordinate Descent

Assume we have τ cores and we want to make use of all of them.

Parallel sampling: $P(|\hat{S}| = \tau) = 1$, each core processes one coordinate

Question: How to design importance sampling for minibatches?

Complications:

- for serial sampling we had $\underbrace{P(\{j\} = \hat{S})}_{\text{parameter}} = P(j \in \hat{S})$, which is not the case for parallel
- We need to choose $\binom{d}{\tau}$ set-probabilities, instead of just $\binom{d}{1} = \text{d}$
- All of them influence the values $P(j \in \hat{S})$

Minibatch Coordinate Descent

Assume we have τ cores and we want to make use of all of them.

Parallel sampling: $P(|\hat{S}| = \tau) = 1$, each core processes one coordinate

Question: How to design importance sampling for minibatches?

Complications:

- for serial sampling we had $\underbrace{P(\{j\} = \hat{S})}_{\text{parameter}} = P(j \in \hat{S})$, which is not the case for parallel
- We need to choose $\binom{d}{\tau}$ set-probabilities, instead of just $\binom{d}{1} = \mathrm{d}$
- All of them influence the values $P(j \in \hat{S})$

Goal: Find a parallel sampling with direct control over $p_j = P(j \in \hat{S})$

Minibatch Coordinate Descent

Assume we have τ cores and we want to make use of all of them.

Parallel sampling: $P(|\hat{S}| = \tau) = 1$, each core processes one coordinate

Question: How to design importance sampling for minibatches?

Complications:

- for serial sampling we had $\underbrace{P(\{j\} = \hat{S})}_{\text{parameter}} = P(j \in \hat{S})$, which is not the case for parallel
- We need to choose $\binom{d}{\tau}$ set-probabilities, instead of just $\binom{d}{1} = \text{d}$
- All of them influence the values $P(j \in \hat{S})$

Goal: Find a parallel sampling with direct control over $p_j = P(j \in \hat{S})$

Solution: We introduce the **bucket sampling**

Bucket sampling

Idea: Divide the coordinates $\{1, \ldots, d\}$ into τ *buckets* and choose one coordinate **non-uniformly** from each bucket

Bucket sampling

Idea: Divide the coordinates $\{1, \ldots, d\}$ into τ *buckets* and choose one coordinate **non-uniformly** from each bucket

Bucket sampling

Let $C_1, \ldots, C_\tau$ be an arbitrary partition of $\{1, \ldots, d\}$. Choose the probabilities p_j, such that $\sum_{j \in C_k} p_j = 1$ for each $k \in \{1, \ldots, \tau\}$. The **bucket sampling** is a procedure, which outputs a single coordinate j from each C_k with probabilities p_j.

Bucket sampling

Idea: Divide the coordinates $\{1, \ldots, d\}$ into τ *buckets* and choose one coordinate **non-uniformly** from each bucket

Bucket sampling

Let $C_1, \ldots, C_\tau$ be an arbitrary partition of $\{1, \ldots, d\}$. Choose the probabilities p_j, such that $\sum_{j \in C_k} p_j = 1$ for each $k \in \{1, \ldots, \tau\}$. The **bucket sampling** is a procedure, which outputs a single coordinate j from each C_k with probabilities p_j.

Example: Let $\tau = 4$ and $d = 12$,

$$\mathbf{p} = \Big[\underbrace{0.1, 0.4, 0.2, 0.3}_{C_1}, \underbrace{1.0}_{C_2}, \underbrace{0.1, 0.1, 0.1, 0.1, 0.6}_{C_3}, \underbrace{0.5, 0.5}_{C_4}\Big]$$

Bucket sampling

Idea: Divide the coordinates $\{1, \ldots, d\}$ into τ *buckets* and choose one coordinate **non-uniformly** from each bucket

Bucket sampling

Let $C_1, \ldots, C_\tau$ be an arbitrary partition of $\{1, \ldots, d\}$. Choose the probabilities p_j, such that $\sum_{j \in C_k} p_j = 1$ for each $k \in \{1, \ldots, \tau\}$. The **bucket sampling** is a procedure, which outputs a single coordinate j from each C_k with probabilities p_j.

Example: Let $\tau = 4$ and $d = 12$,

$$\mathbf{p} = \Big[\underbrace{0.1, 0.4, 0.2, 0.3}_{C_1}, \underbrace{1.0}_{C_2}, \underbrace{0.1, 0.1, 0.1, 0.1, 0.6}_{C_3}, \underbrace{0.5, 0.5}_{C_4}\Big]$$

Observe: It holds, that $P(j \in \hat{S}) = p_j$.

Bucket sampling

Idea: Divide the coordinates $\{1, \ldots, d\}$ into τ *buckets* and choose one coordinate **non-uniformly** from each bucket

Bucket sampling

Let $C_1, \ldots, C_\tau$ be an arbitrary partition of $\{1, \ldots, d\}$. Choose the probabilities p_j, such that $\sum_{j \in C_k} p_j = 1$ for each $k \in \{1, \ldots, \tau\}$. The **bucket sampling** is a procedure, which outputs a single coordinate j from each C_k with probabilities p_j.

Example: Let $\tau = 4$ and $d = 12$,

$$\mathbf{p} = \Big[\underbrace{0.1, 0.4, 0.2, 0.3}_{C_1}, \underbrace{1.0}_{C_2}, \underbrace{0.1, 0.1, 0.1, 0.1, 0.6}_{C_3}, \underbrace{0.5, 0.5}_{C_4}\Big]$$

Observe: It holds, that $P(j \in \hat{S}) = p_j$.

Note: Both $p_1, \ldots, p_d$ and $C_1, \ldots, C_\tau$ are parameters of the sampling

Importance Sampling for Minibatches

Goal: We aim to minimize

$$C(\mathbf{p}^{bucket}, \mathbf{v}^{bucket}) = \max_{j \in \{1,\dots,d\}} \left(\frac{{v_j}^{bucket}}{\lambda {p_j}^{bucket}} \right)$$

Importance Sampling for Minibatches

Goal: We aim to minimize

$$C(\mathbf{p}^{bucket}, \mathbf{v}^{bucket}) = \max_{j \in \{1,\dots,d\}} \left(\frac{{v_j}^{bucket}}{\lambda {p_j}^{bucket}} \right)$$

Issue: The values $\mathbf{v}^{bucket}$ depend on $\mathbf{p}^{bucket}$,
i.e., ${v_j}^{bucket} = (\dots \mathbf{p}^{bucket} \dots)$

Importance Sampling for Minibatches

Goal: We aim to minimize

$$C(\mathbf{p}^{bucket}, \mathbf{v}^{bucket}) = \max_{j \in \{1,\dots,d\}} \left(\frac{{v_j}^{bucket}}{\lambda {p_j}^{bucket}} \right)$$

Issue: The values $\mathbf{v}^{bucket}$ depend on $\mathbf{p}^{bucket}$,
i.e., ${v_j}^{bucket} = (\dots \mathbf{p}^{bucket} \dots)$

Optimal probabilities: $\rightarrow$ joint optimization $\rightarrow$ difficult

Importance Sampling for Minibatches

Goal: We aim to minimize

$$C(\mathbf{p}^{bucket}, \mathbf{v}^{bucket}) = \max_{j \in \{1,\dots,d\}} \left(\frac{{v_j}^{bucket}}{\lambda {p_j}^{bucket}} \right)$$

Issue: The values $\mathbf{v}^{bucket}$ depend on $\mathbf{p}^{bucket}$,
i.e., ${v_j}^{bucket} = (\dots \mathbf{p}^{bucket} \dots)$

Optimal probabilities: $\rightarrow$ joint optimization $\rightarrow$ difficult

Strategy: Choose a *reasonable* $\mathbf{v}^{bucket} \rightarrow$ pick ${p_j}^{bucket} \sim {v_j}^{bucket}$

Importance Sampling for Minibatches

Goal: We aim to minimize

$$C(\mathbf{p}^{bucket}, \mathbf{v}^{bucket}) = \max_{j \in \{1,\dots,d\}} \left(\frac{{v_j}^{bucket}}{\lambda {p_j}^{bucket}} \right)$$

Issue: The values $\mathbf{v}^{bucket}$ depend on $\mathbf{p}^{bucket}$,
i.e., ${v_j}^{bucket} = (\dots \mathbf{p}^{bucket} \dots)$

Optimal probabilities: $\rightarrow$ joint optimization $\rightarrow$ difficult

Strategy: Choose a *reasonable* $\mathbf{v}^{bucket} \rightarrow$ pick ${p_j}^{bucket} \sim {v_j}^{bucket}$

In theory: We cannot guarantee a speedup, just convergence

Importance Sampling for Minibatches

Goal: We aim to minimize

$$C(\mathbf{p}^{bucket}, \mathbf{v}^{bucket}) = \max_{j \in \{1,\dots,d\}} \left(\frac{{v_j}^{bucket}}{\lambda {p_j}^{bucket}} \right)$$

Issue: The values $\mathbf{v}^{bucket}$ depend on $\mathbf{p}^{bucket}$,
i.e., ${v_j}^{bucket} = (\dots \mathbf{p}^{bucket} \dots)$

Optimal probabilities: $\rightarrow$ joint optimization $\rightarrow$ difficult

Strategy: Choose a *reasonable* $\mathbf{v}^{bucket} \rightarrow$ pick ${p_j}^{bucket} \sim {v_j}^{bucket}$

In theory: We cannot guarantee a speedup, just convergence

In practice: We will see.

Experiments: Convergence

Ridge Logistic Regression using **Quartz** [Qu et al., 2015]

Experiments: Convergence

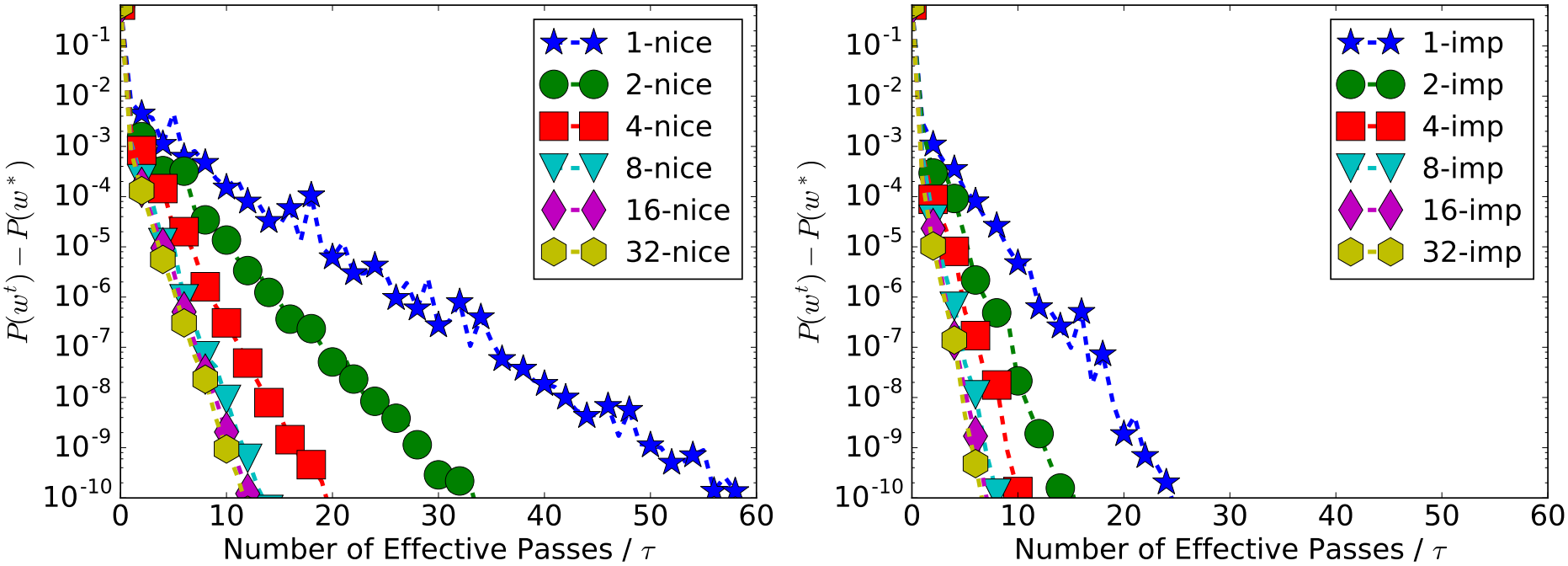
Ridge Logistic Regression using **Quartz** [Qu et al., 2015]

Randomly chosen equally sized buckets $C_1, \ldots, C_\tau$

Experiments: Convergence

Ridge Logistic Regression using **Quartz** [Qu et al., 2015]

Randomly chosen equally sized buckets $C_1, \ldots, C_\tau$



url dataset: $d = 2,396,130, n = 3,231,962$, sparsity $= 0.04\%$

Experiments: Theory vs. Practice

Ridge Logistic Regression using **Quartz** [Qu et al., 2015]

Randomly chosen equally sized buckets $C_1, \ldots, C_\tau$

Data	$\tau = 1$	$\tau = 2$	$\tau = 4$	$\tau = 8$	$\tau = 16$	$\tau = 32$
ijcnn1	1.2 : 1.1	1.4 : 1.1	1.6 : 1.3	1.9 : 1.6	2.2 : 1.6	2.3 : 1.8
protein	1.3 : 1.2	1.4 : 1.2	1.5 : 1.4	1.7 : 1.4	1.8 : 1.5	1.9 : 1.5
w8a	2.8 : 2.0	2.9 : 1.9	2.9 : 1.9	3.0 : 1.9	3.0 : 1.8	3.0 : 1.8
url	3.0 : 2.3	2.6 : 2.1	2.0 : 1.8	1.7 : 1.6	1.8 : 1.6	1.8 : 1.7
aloi	13 : 7.8	12 : 8.0	11 : 7.7	9.9 : 7.4	9.3 : 7.0	8.8 : 6.7

Table: The **theoretical** : **empirical** ratios $\theta^{(\tau\text{-imp})}/\theta^{(\tau\text{-nice})}$.

Conclusion

Conclusion

- we introduced a flexible parallel sampling - the **bucket sampling**

Conclusion

- we introduced a flexible parallel sampling - the **bucket sampling**
- we used it to get an **importance sampling for minibatches**

Conclusion

- we introduced a flexible parallel sampling - the **bucket sampling**
- we used it to get an **importance sampling for minibatches**
- can be used for various stochastic iterative methods

Conclusion

- we introduced a flexible parallel sampling - the **bucket sampling**
- we used it to get an **importance sampling for minibatches**
- can be used for various stochastic iterative methods
- we empirically **match the improvement gained by serial importance sampling**

Conclusion

- we introduced a flexible parallel sampling - the **bucket sampling**
- we used it to get an **importance sampling for minibatches**
- can be used for various stochastic iterative methods
- we empirically **match the improvement gained by serial importance sampling**
- in theory, we cannot in general guarantee that importance sampling for minibatches will be better than the uniform parallel sampling

References

Csiba, Dominik and Richtárik, Peter

Importance Sampling for Minibatches

arXiv e-prints (2016): 1602.02283

Richtárik, Peter and Takáč, Martin

On optimal probabilities in stochastic coordinate descent methods

Optimization Letters (2015): 1-11.

Qu, Zheng and Richtárik, Peter

Coordinate descent with arbitrary sampling II: Expected separable overapproximation

arXiv e-prints (2014): 1412.8063

Qu, Zheng and Richtárik, Peter and Zhang, Tong

Quartz: Randomized dual coordinate ascent with arbitrary sampling

Advances in Neural Information Processing Systems, 2015