ETH zürich

Adaptive Probabilities in Stochastic Optimization Algorithms

Lei Zhong
Data Analytics Lab

Apr. 21, 2015

Part A

Non-Uniform Sampling Algorithms

Problem

$$f(\boldsymbol{w}) := \ell(\boldsymbol{w}) + \lambda r(\boldsymbol{w}) \tag{1}$$

where

$$\ell(\boldsymbol{w}) := \frac{1}{n}\sum_{i=1}^{n} \ell(\langle \boldsymbol{x}_i, \boldsymbol{w}\rangle, y_i). \tag{2}$$

and

$$r(\boldsymbol{w}) := \frac{1}{2}\|\boldsymbol{w}\|_2^2 \tag{3}$$

Here, $\ell(., y_i) : \mathbb{R} \to \mathbb{R}$ is a loss function and $r(.)$ takes the role of a regularizer.

Empirical Risk Minimization

$$\min_{\boldsymbol{w}\in\mathbb{R}^n} f(\boldsymbol{w})$$

Notations

- $\boldsymbol{x}_i$: feature vector of sample i
- y_i: label of sample i
- p_i: probability that sample i will be selected with $\sum_{j=1}^{n} p_j = 1$
- χ_i: subgradient of sample i
- $\boldsymbol{g}_i$: subgradient of sample i with $\boldsymbol{g}_i = \frac{\chi_i}{np_i}$
- $\boldsymbol{w}$: solution of objective function
- η: stepsize for updating $\boldsymbol{w}$

NonUnifSGD

Algorithm 1: Non-Uniform Stochastic Gradient Discent

Input: $\lambda > 0$, $p_i = \frac{\|\boldsymbol{x}_i\|}{\sum_{j=1}^{n} \|\boldsymbol{x}_j\|}, \forall i \in \{1, \dots, n\}$.

Data: $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$.

Initialize: $\boldsymbol{w}^1 = \boldsymbol{0}$.

for $t = 1, 2, \dots, T$

- Sample i_t from $\{1, \dots, n\}$ based on $\boldsymbol{p}$;
- Set stepsize $\eta_t \leftarrow \frac{1}{\lambda t}$;
- Set $\chi_{i_t}^t(\boldsymbol{w}^t) \leftarrow \ell'(\langle \boldsymbol{w}^t, \boldsymbol{x}_{i_t} \rangle, y_{i_t})\boldsymbol{x}_{i_t} + \lambda \nabla r(\boldsymbol{w}^t)$;
- Set $\boldsymbol{g}_{i_t}^t \leftarrow \frac{\chi_{i_t}^t(\boldsymbol{w}^t)}{n p_{i_t}}$;
- Set $\boldsymbol{w}^{t+1} \leftarrow \boldsymbol{w}^t - \eta_t \boldsymbol{g}_{i_t}^t$;

end

Output: $\boldsymbol{w}^{T+1}$

Key inequality

Taking the expectation over the random sampling at each step, we have the following bound:

$$\begin{aligned}\mathbb{E}[f(\boldsymbol{w}^t)] - f(\boldsymbol{w}^*) \leq \frac{\eta_t}{2}\mathbb{E}[\|\boldsymbol{g}_{i_t}^t\|^2] \\ + \frac{1-\lambda\eta_t}{2\eta_t}\mathbb{E}[\|\boldsymbol{w}^t - \boldsymbol{w}^*\|^2] - \frac{1}{2\eta_t}\mathbb{E}[\|\boldsymbol{w}^{t+1} - \boldsymbol{w}^*\|^2]\end{aligned}$$

Convergence Theorem

Theorem

Suppose f is a λ-strongly convex function. If we choose the stepsize $\eta_t = \frac{1}{\lambda t}$, then after T iterations of NonUnifSGD (Algorithm 1) starting with $\mathbf{w}^1 = \mathbf{0}$, it holds that

$$\mathbb{E}\left[f\left(\frac{1}{T}\sum_{t=1}^{T}\mathbf{w}^t\right)\right] - f(\mathbf{w}^*) \leq \frac{1}{2\lambda T}\sum_{t=1}^{T}\frac{\mathbb{E}[\|\mathbf{g}_{i_t}^t\|^2]}{t}$$

where $\mathbf{g}_{i_t}^t = \frac{\chi_{i_t}^t(\mathbf{w}^t)}{np_{i_t}}$ and the expectation is taken with respect to the distribution $\mathbf{p}$.

Proof Snapshot

Proof With stepsize $\eta_t = \frac{1}{\lambda t}$ plugged into (4.6), we have

$$\mathbb{E}[f(\boldsymbol{w}^t)] - f(\boldsymbol{w}^*) \leq \frac{1}{2\lambda t}\mathbb{E}[\|\boldsymbol{g}_{i_t}^t\|^2] + \frac{\lambda(t-1)}{2}\mathbb{E}[\|\boldsymbol{w}^t - \boldsymbol{w}^*\|^2] - \frac{\lambda t}{2}\mathbb{E}[\|\boldsymbol{w}^{t+1} - \boldsymbol{w}^*\|^2] \tag{4.7}$$

We use convexity of the function f, as given by Jensen's inequality:

$$\mathbb{E}\left[f\left(\frac{1}{T}\sum_{t=1}^{T}\boldsymbol{w}^t\right)\right] - f(\boldsymbol{w}^*) \overset{Jensen}{\leq} \mathbb{E}\left[\frac{1}{T}\sum_{t=1}^{T} f(\boldsymbol{w}^t)\right] - f(\boldsymbol{w}^*)$$

Summing up (4.7) over all steps $t = 1 \dots T$, we can bound the right hand side of the above inequality

$$= \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}[f(\boldsymbol{w}^t)] - f(\boldsymbol{w}^*) \leq \frac{1}{T}\sum_{t=1}^{T}\frac{\mathbb{E}[\|\boldsymbol{g}_{i_t}^t\|^2]}{2\lambda}\frac{1}{t} - \frac{\lambda}{2}\mathbb{E}[\|\boldsymbol{w}^{T+1} - \boldsymbol{w}^*\|^2]$$

(where we have used the telescoping sum of the norm terms.)

Re-arranging terms, and trivially bounding the left hand side of Jensen's inequality by $0 \leq \mathbb{E}\left[f\left(\frac{1}{T}\sum_{t=1}^{T}\boldsymbol{w}^t\right)\right] - f(\boldsymbol{w}^*)$, we obtain the claimed bound

$$\mathbb{E}[\|\boldsymbol{w}^{T+1} - \boldsymbol{w}^*\|^2] \leq \frac{1}{\lambda^2 T}\sum_{t=1}^{T}\frac{\mathbb{E}[\|\boldsymbol{g}_{i_t}^t\|]^2}{t}. \qquad \square$$

Two corollaries

Definition

Define $G := \max_{i,t}\{\|\chi_i^t(\mathbf{w}^t)\|^2\}$ $(i = 1 \dots n,\ t = 1 \dots T)$. Define $W := \max_{i,t}\{\mathbb{E}[\|\chi_i^t(\mathbf{w}^t)\|^2]\}$ $(i = 1 \dots n,\ t = 1 \dots T)$.

Corollary

Assume that $\max_t\{\|\chi_{i_t}^t(\mathbf{w}^t)\|^2\} \leq G$ *for all* t. $\mathbb{E}[\|\chi_{i_t}^t(\mathbf{w}^t)\|^2] \leq W$ *for all* t *and* $p_i > \epsilon$ *for all* $i = \{1 \dots, n\}$,

$$\mathbb{E}\left[f\left(\frac{1}{T}\sum_{t=1}^{T}\mathbf{w}^t\right)\right] - f(\mathbf{w}^*) \leq \frac{1}{2\lambda T}\sum_{t=1}^{T}\frac{G}{\epsilon n t} \leq \frac{G(\ln T + 1)}{2\lambda \epsilon n T}$$

$$\mathbb{E}\left[f\left(\frac{1}{T}\sum_{t=1}^{T}\mathbf{w}^t\right)\right] - f(\mathbf{w}^*) \leq \frac{1}{2\lambda T}\sum_{t=1}^{T}\frac{W}{n^2\epsilon^2 t} \leq \frac{W(\ln T + 1)}{2\lambda T n^2 \epsilon^2}$$

Dual Problem

Dual Objective Function

$$\max_{\boldsymbol{\alpha} \in \mathbb{R}^n} D(\boldsymbol{\alpha}) := \frac{1}{n} \sum_{i=1}^{n} -\ell_i^*(-\alpha_i) - \lambda r^*(\boldsymbol{v}(\boldsymbol{\alpha})).$$

The relationship between primal variable $\boldsymbol{w}$ and dual variable $\boldsymbol{\alpha}$ is

$$\boldsymbol{w}(\boldsymbol{\alpha}) := \nabla r^*(\boldsymbol{v}(\boldsymbol{\alpha})), \boldsymbol{v}(\boldsymbol{\alpha}) := \frac{1}{\lambda n} \sum_{i=1}^{n} \alpha_i \boldsymbol{x}_i$$

where $\boldsymbol{\alpha} \in \mathbb{R}^n$.

NonUnifSDCA

Algorithm 2: Non-Uniform Stochastic Dual Coordinate Ascent

Input: $\lambda > 0$, $p_i = \frac{\|\boldsymbol{x}_i\|}{\sum_{j=1}^{n} \|\boldsymbol{x}_j\|}$, $\forall i \in \{1, \ldots, n\}$.

Data: $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$

Initialize: $\boldsymbol{\alpha}^1 = \mathbf{0}$, $\boldsymbol{w}^1 = \mathbf{0}$.

for $t = 1, 2, \ldots, T$

- Sample i_t from $\{1, \ldots, n\}$ based on $\boldsymbol{p}$;
- Calculate $\Delta\alpha_{i_t}^t =$ $\arg\max_{\Delta\alpha_{i_t}^t} [-\frac{\lambda n}{2} \|\boldsymbol{w}^t + \frac{1}{\lambda n} \Delta\alpha_{i_t}^t \boldsymbol{x}_{i_t}\|^2 - \ell_{i_t}^*(-(\alpha_{i_t}^t + \Delta\alpha_{i_t}^t))]$;
- Set $\alpha_{i_t}^{t+1} \leftarrow \alpha_{i_t}^t + \Delta\alpha_{i_t}^t$;
- Set $\boldsymbol{w}^{t+1} \leftarrow \boldsymbol{w}^t + \frac{1}{\lambda n} \Delta\alpha_{i_t}^t \boldsymbol{x}_{i_t}$;

end

Output: $\boldsymbol{w}^{T+1}$

Part B

Adaptive Sampling Algorithms

Idea behind SGD

$$\begin{aligned}&\mathbb{E}[f(\boldsymbol{w}^{t+1})] - \mathbb{E}[f(\boldsymbol{w}^{t})]\\ &= \frac{\eta_t}{2}(1+\lambda\eta_t)\mathbb{E}[\|\boldsymbol{g}_{i_t}^t\|^2] - \eta_t\langle \boldsymbol{\chi}_{i_t}^t, \nabla\ell(\boldsymbol{w}^t)\rangle + \lambda\eta_t\langle \boldsymbol{w}^t, \boldsymbol{\chi}_{i_t}^t\rangle.\end{aligned}$$

$$\min \mathbb{E}[\|\boldsymbol{g}_{i_t}^t\|^2] \xrightarrow{solution} p_i = \frac{\|\boldsymbol{\chi}_i\|}{\sum_{j=1}^n \|\boldsymbol{\chi}_i\|}.$$

Two Updates

Algorithm 3: Aggressive Probability Update

for $j = 1, \ldots, n$
  Set $p_j \leftarrow \frac{c_j}{\sum_{k=1}^{n} c_k}$;
end

Algorithm 4: Conservative Probability Update

Set $s \leftarrow \sum_{j=1,\ldots,n,\mathbf{1}_i=0} c_j$;
Set $c \leftarrow |S|$ where $S \leftarrow \{j|\mathbf{1}_i = 1\}$;
for $j = 1, \ldots, n$
  $p_j > 0 \; ? \; p_j \leftarrow \frac{c_j}{s+c} \; : \; p_j \leftarrow \frac{1}{s+c}$;
end

$\mathbf{1}_i$ is a indicator function which returns 1 if point i is correctly classified during all the k iterations, otherwise returns 0.

AdaSGD

Algorithm 5: AdaSGD (Adaptive Non-Uniform Stochastic Gradient Descent)

Input: $\lambda > 0$
Data: $\{(x_i, y_i)\}_{i=1}^n$
Initialize: $w^0 = \mathbf{0}$, probabilities

$$p_i = \frac{\|x_i\|^2 + \sqrt{\lambda}}{\sum_{j=1}^n \|x_j\|^2 + \sqrt{\lambda}}, c_i = 0, \forall i \in \{1, \dots, n\}.$$

for $t = 1, 2, \dots, T$
  Sample i_t from $\{1, \dots, n\}$ based on p;
  Set $\eta_t \leftarrow \frac{1}{\lambda t}$;
  Calculate $\ell' \leftarrow \ell'(\langle x_{i_t}, w^t \rangle, y_i)$;
  Set $\chi^t_{i_t}(w^t) \leftarrow \ell' x_{i_t} + \lambda \nabla r(w^t)$;
  if *(t-1) mod* $n \geq n$*-k* **then**
    for $i = 1, 2, \dots, n$
      Calculate $\ell'(\langle x_i, w^t \rangle, y_i)$;
      Set $\chi_i \leftarrow \ell'(\langle x_i, w^t \rangle, y_i) x_i + \lambda \nabla r(w^t)$;
      Set $c_i \leftarrow \max\{c_i, \|\chi_i\|\}$;
    end
  end
  if t *mod* $n = 0$ **then**
    Option I: Run Algorithm 3 (Aggressive Update);
    Option II: Run Algorithm 4 (Conservative Update);
  end
  Set $g^t_{i_t} \leftarrow \frac{\chi^t_{i_t}(w^t)}{n p_{i_t}}$;
  Set $w^{t+1} \leftarrow w^t - \eta_t g^t_{i_t}$;
end
Output: w^{T+1}

AdaSVRG

We add a $\tilde{\boldsymbol{w}}$ (which denotes the $\boldsymbol{w}$ of last epoch) for a new update equation. Therefore, we get

$$\boldsymbol{w}^{t+1} := \boldsymbol{w}^t - \eta_t[\boldsymbol{g}^t_{i_t}(\boldsymbol{w}^t) - \boldsymbol{g}^t_{i_t}(\tilde{\boldsymbol{w}}) - \nabla f(\tilde{\boldsymbol{w}})]$$

The expectation of the update function is still the same as before, because

$$\mathbb{E}[\boldsymbol{g}(\boldsymbol{w}) - \boldsymbol{g}(\tilde{\boldsymbol{w}}) + \nabla f(\tilde{\boldsymbol{w}})] = \mathbb{E}[\boldsymbol{g}(\boldsymbol{w})] - \mathbb{E}[\boldsymbol{g}(\tilde{\boldsymbol{w}})] + \nabla f(\tilde{\boldsymbol{w}}) = \nabla f(\boldsymbol{w}).$$

Idea behind SDCA

Definition

Define the gap of point i as

$$\sigma_i^t = \ell(\mathbf{x}_i^\top \mathbf{w}^t) + \ell^*(-\alpha_i^t) + \alpha_i^t \mathbf{x}_i^\top \mathbf{w}^t$$

where $\mathbf{w}^t$ here is assumed to be the corresponding primal vector for the current $\boldsymbol{\alpha}^t$, that is $\mathbf{w}^t(\boldsymbol{\alpha}^t) := \frac{1}{\lambda n} \sum_{i=1}^n \alpha_i \mathbf{x}_i^t$.

The **duality gap** between the primal objective and dual objective at the t-th iteration is defined as

$$f(\mathbf{w}^t) - D(\boldsymbol{\alpha}^t) = \frac{1}{n} \sum_{i=1}^n \sigma_i^t.$$

AdaSDCA

Algorithm 7: AdaSDCA (Adaptive Non-uniform Stochastic Dual Coordinate Ascent)

Input: $\lambda > 0$
Data: $\{(x_i, y_i)\}_{i=1}^n$
Initialize: $\alpha^1 = \mathbf{0}$, $w^1 = \mathbf{0}$, probabilities $p_i = \frac{1+\frac{1}{\lambda n \gamma_i}}{n+\sum_{j=1}^n \frac{1}{\lambda n \gamma_j}}$ or

$p_i = \frac{\|x_i\|}{\sum_{j=1}^n \|x_j\|}$, $c_i = 0$, $\forall i \in \{1, \ldots, n\}$.

for $t = 1, 2, \ldots, T$
  Sample i_t from $\{1, \ldots, n\}$ based on p;
  Calculate $\Delta\alpha_{i_t}^t$ using following formulas:
  $\Delta\alpha_{i_t}^t = \arg\max_{\Delta\alpha_{i_t}^t} [-\frac{\lambda n}{2}\|w^t + \frac{1}{\lambda n}\Delta\alpha_{i_t}^t x_{i_t}\|^2 - \ell_{i_t}^*(-(\alpha_{i_t}^t + \Delta\alpha_{i_t}^t))]$;
  Set $\alpha_{i_t}^{t+1} \leftarrow \alpha_{i_t}^t + \Delta\alpha_{i_t}^t$;
  Set $w^{t+1} \leftarrow w^t + \frac{1}{\lambda n}\Delta\alpha_{i_t}^t x_{i_t}$;
  if *(t-1) mod n* $\geq$ *n-k* **then**
    for $i = 1, 2, \ldots, n$
      Calculate $\sigma_i^t \leftarrow \ell(x_i^\top w^t) + l^*(-\alpha_i^t) + \alpha_i^t \langle x_i, w^t \rangle$;
      Set $c_i \leftarrow \max\{c_i, \sigma_i^t\}$;
    end
  end
  if *t mod n* = 0 **then**
    Option I: Run Algorithm 3 (Aggressive Update);
    Option II: Run Algorithm 4 (Conservative Update);
  end
end
Output: w^{T+1}

AdaSDCAS

Algorithm 8: AdaSDCAS (Adaptive Non-uniform Stochastic Dual Coordinate Ascent by Subgradient)

Input: $\lambda > 0$
Data: $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$
Initialize: $\boldsymbol{\alpha}^1 = \mathbf{0}$, $\boldsymbol{w}^1 = \mathbf{0}$, probabilities $p_i = \frac{1+\frac{1}{\lambda n \gamma_i}}{n+\sum_{j=1}^n \frac{1}{\lambda n \gamma_j}}$ or

$p_i = \frac{\|\boldsymbol{x}_i\|}{\sum_{j=1}^n \|\boldsymbol{x}_j\|}$, $c_i = 0, \forall i \in \{1, \ldots, n\}$.

for $t = 1, 2, \ldots, T$
  Sample i_t from $\{1, \ldots, n\}$ based on $\boldsymbol{p}$;
  Calculate $\Delta\alpha_{i_t}^t$ using following formulas:
  $\Delta\alpha_{i_t}^t = \arg\max_{\Delta\alpha_{i_t}^t} [-\frac{\lambda n}{2}\|\boldsymbol{w}^t + \frac{1}{\lambda n}\Delta\alpha_{i_t}^t \boldsymbol{x}_{i_t}\|^2 - \ell_{i_t}^*(-(\alpha_{i_t}^t + \Delta\alpha_{i_t}^t))]$;
  Set $\alpha_{i_t}^{t+1} \leftarrow \alpha_{i_t}^t + \Delta\alpha_{i_t}^t$;
  Set $\boldsymbol{w}^{t+1} \leftarrow \boldsymbol{w}^t + \frac{1}{\lambda n}\Delta\alpha_{i_t}^t \boldsymbol{x}_{i_t}$;
  if *(t-1) mod n* $\geq$ *n-k* **then**
    for $i = 1, 2, \ldots, n$
      Calculate $\ell'(\langle \boldsymbol{x}_i, \boldsymbol{w}^t\rangle, y_i)$;
      Set $\chi_i^t \leftarrow \ell'(\langle \boldsymbol{x}_i, \boldsymbol{w}^t\rangle, y_i)\boldsymbol{x}_i + \lambda\nabla r(\boldsymbol{w}^t)$;
      Record $c_i \leftarrow \max\{c_i, \|\chi_i^t\|\}$;
    end
  end
  if *t mod n* $= 0$ **then**
    Option I: Run Algorithm 3 (Aggressive Update);
    Option II: Run Algorithm 4 (Conservative Update);
  end
end
Output: $\boldsymbol{w}^{T+1}$

Part C

Discussions and Experiments

Datasets for empirical study

Dataset	Training(n)	Test	Features (d)	Sparsity($\frac{\text{nnz}}{nd}$)
rcv1	20,242	677,399	47,236	0.16%
astro-ph	29,882	32,487	99,757	0.08%

- **rcv1** is a corpus from Reuters news stories.
- **astro-ph** is astronomy data.

Cost per epoch and properties of algorithms

ALGORITHM	cost of an epoch	non-uniform	adaptive
NonUnifSGD	nnz	✓	✗
NonUnifSDCA	nnz	✓	✗
AdaSGD	$(k+1)$ nnz	✓	✓
AdaSVRG	$nd + k$ nnz	✓	✓
AdaSDCA	$(k+1)$ nnz	✓	✓
AdaSDCAS	$(k+1)$ nnz	✓	✓
AdaGrad (by Duchi)	$2nd$	✗	✗
AdaSDCA (by Csiba)	n nnz	✓	✓
AdaSDCA+ (by Csiba)	2 nnz	✓	✓

Test Error with Different Values of λ

rcv1	1e-2	5e-3	**1e-3**	5e-4	1e-4
Test Error	0.05160	0.04833	**0.04713**	0.04913	0.05693

astro-ph	1e-2	5e-3	**1e-3**	5e-4	1e-4
Test Error	0.04103	0.03715	**0.03441**	0.03586	0.04371

Verifying the Convergence of Duality Gap

Table: Average duality gap at different epochs for $\lambda = 0.001$

#epoch	duality gap on `rcv1`	duality gap on `astro-ph`
1	0.0863765	0.0883917
3	0.0105347	6.13163e-03
10	1.7485e-04	3.93673e-05
20	2.21547e-05	6.24779e-07
50	3.12797e-06	6.7474e-10
100	5.47897e-07	1.43083e-12

Performance Metrics

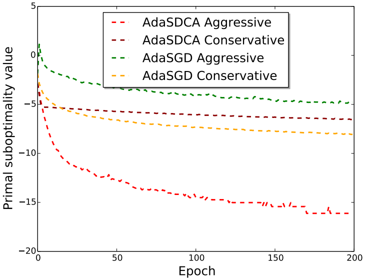
Definition

The **primal sub-optimality** of algorithm is defined as $P(\boldsymbol{w}(\alpha)) - P(\boldsymbol{w}^*)$.

Definition

Test error is the error rate on test dataset.

Performance of Two Updating Algorithms



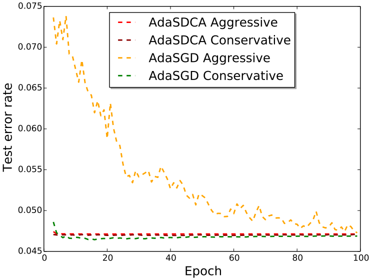
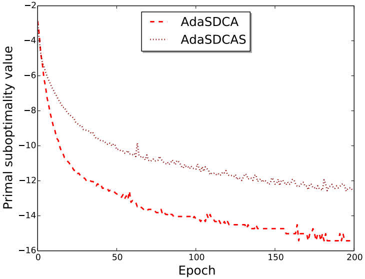


Figure: Comparison of two updating algorithms for AdaSGD and AdaSDCA on `rcv1`

Different Adaptive Strategies for AdaSDCA



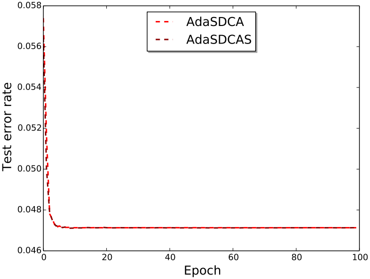


Figure: Comparison of AdaSDCA and AdaSDCAS on `rcv1`

Comparison of Average Time

Table: Detailed training time and total running time per epoch

rcv1	Training time(s)	Total running time(s)
AdaSGD	0.04765	0.2059
AdaSDCA	0.05042	0.2064
NonUnifSGD	0.04244	0.1988
NonUnifSDCA	0.04716	0.2037

astro-ph	Training time(s)	Total running time(s)
AdaSGD	0.07236	0.1363
AdaSDCA	0.07050	0.1343
NonUnifSGD	0.06284	0.1259
NonUnifSDCA	0.07054	0.1339

Comparison of Adaptive Algorithms

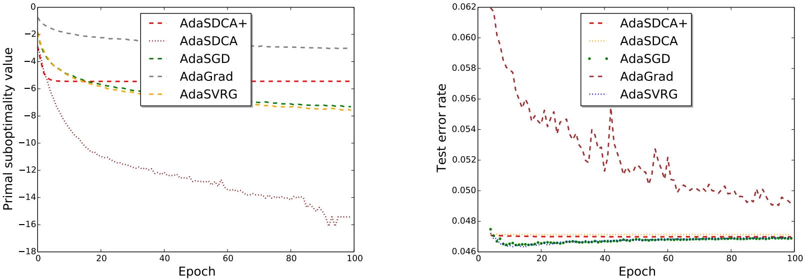


Figure: Comparison of five adaptive algorithms on `rcv1`

Comparison of Adaptive Algorithms cont.

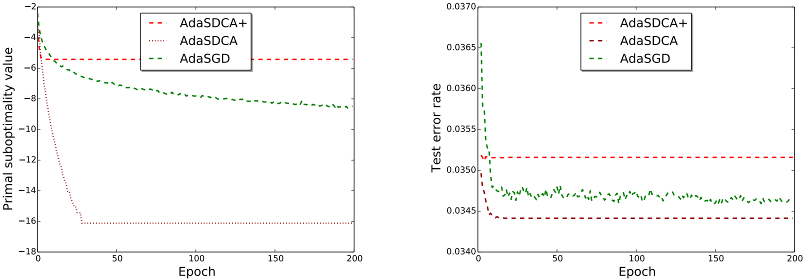


Figure: Comparison of three adaptive algorithms on `astro-ph`

Same Level of Optimality

Table: The number of epochs taken to reach the same level of optimality

rcv1	AdaSDCA	NonUnifSDCA	AdaSGD	NonUnifSGD
#epochs	9	35	210	500
astro-ph	AdaSDCA	NonUnifSDCA	AdaSGD	NonUnifSGD
#epochs	8	28	195	500

Comparison of Time

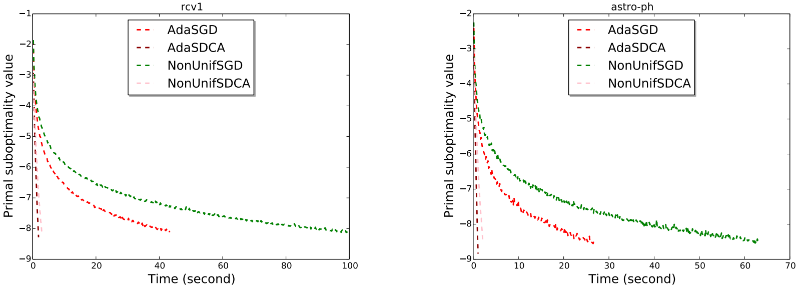


Figure: Comparison of the total running time to reach the same optimality

Comparison of Vector Operation

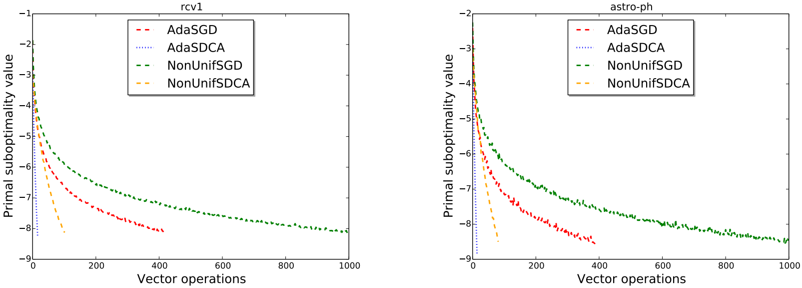


Figure: Comparison of the vector operations taken to reach the same optimality

Adaptive vs. Non-Uniform Algorithms

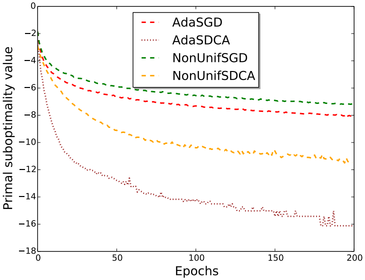


Figure: Comparison of adaptive algorithms with non-adaptive algorithms on `rcv1`

Adaptive vs. Non-Uniform Algorithms cont.

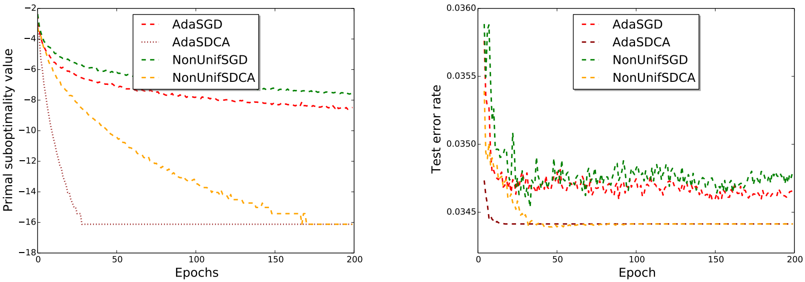


Figure: Comparison of adaptive algorithms with non-adaptive algorithms on `astro-ph`

Summary

- We chose $\lambda = 0.001$ for both `rcv1` and `astro-ph`.
- We compare the performance of Conservative Update and Aggressive Update on AdaSGD and AdaSDCA. Conservative Update works better on AdaSGD while Aggressive Update works better on AdaSDCA.
- AdaSDCA (adaptive algorithm with duality gap) performs better than AdaSDCAS (adaptive algorithm with subgradients).
- AdaSDCA has the best performance among all the adaptive algorithms (AdaSDCA, AdaSGD, AdaSVRG, AdaGrad and AdaSDCA+) and AdaSGD is the second best.

Summary cont.

- AdaSVRG achieves a slightly better performance per epoch than AdaSGD but sacrifices the running time on sparse datasets.
- To reach the same optimality given by 500 epochs run on NonUnifSGD, AdaSGD takes only around 200 epochs, whereas NonUnifSDCA takes around 30 which is three times more than AdaSDCA does.

Reference

[1] John Duchi, Elad Hazan, and Yoram Singer. "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization." JMLR, 12:21212159, August 2011.

[2] Dominik Csiba, COM Zheng Qu, and Peter Richtárik. "Stochastic dual coordinate ascent with adaptive probabilities."

[3] Peilin Zhao and Tong Zhang. "Stochastic Optimization with Importance Sampling." arXiv.org, January 2014.

[4] Zheng Qu, Peter Richtárik, and Tong Zhang. "Randomized dual coordinate ascent with arbitrary sampling." arXiv preprint arXiv:1411.5873, 2014.

[5] Rie Johnson and Tong Zhang. "Accelerating Stochastic Gradient Descent using Pre- dictive Variance Reduction." In NIPS, 2013.

Q&A

Thank You!

Acknowledgement:
Thanks to Martin for helpful discussions and suggestions!