



Adaptive Probabilities in Stochastic Optimization Algorithms

Lei Zhong
Data Analytics Lab

Apr. 21, 2015

- 1 Warm-Up
- 2 Non-Uniform Sampling
- 3 Adaptive Sampling
- 4 Discussions and Experiments
- 5 Reference

Problem

Empirical Risk Minimization

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

$$f(\mathbf{w}) := \ell(\mathbf{w}) + \lambda r(\mathbf{w}) \quad (1)$$

where

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

and

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

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

Problem cont.

Partial Objective Function

$$f(\mathbf{w}, i) = \ell(\langle \mathbf{x}_i, \mathbf{w} \rangle, y_i) + \lambda r(\mathbf{w}).$$

- \mathbf{x}_i : feature vector of sample i
- y_i : label of sample i
- \mathbf{w} : solution of objective function
- η : stepsize for updating \mathbf{w}
- χ_i : subgradient of $f(\mathbf{w}, i)$

Assume f , ℓ and r are convex.

Part A

Non-Uniform Sampling Algorithms

Non-Uniform

Define p_i as the probability that sample i will be selected with $\sum_{j=1}^n p_j = 1$. We define \mathbf{g}_i as the weighted subgradient with $\mathbf{g}_i = \frac{\mathbf{x}_i}{np_i}$.

$$\mathbb{E}[\mathbf{g}(\mathbf{w})] = \sum_{i=1}^n \frac{\mathbf{x}_i}{n} = \nabla f(\mathbf{w})$$

NonUnifSGD

Algorithm 1: Non-Uniform Stochastic Gradient Descent

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

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

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

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

 Sample i_t from $\{1, \dots, n\}$ based on \mathbf{p} ;

 Set stepsize $\eta_t \leftarrow \frac{1}{\lambda t}$;

 Set $\chi_{i_t}^t(\mathbf{w}^t) \leftarrow \ell'(\langle \mathbf{w}^t, \mathbf{x}_{i_t} \rangle, y_{i_t}) \mathbf{x}_{i_t} + \lambda \nabla r(\mathbf{w}^t)$;

 Set $\mathbf{g}_{i_t}^t \leftarrow \frac{\chi_{i_t}^t(\mathbf{w}^t)}{np_{i_t}}$;

 Set $\mathbf{w}^{t+1} \leftarrow \mathbf{w}^t - \eta_t \mathbf{g}_{i_t}^t$;

end

Output: \mathbf{w}^{T+1}

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} .

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

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

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

Another Theorem for SGD

Theorem

Suppose f is a λ -strongly convex function. If we choose the stepsize $\eta_t = \frac{2}{\lambda(t+1)}$, then after T iterations of NonUnifSGD (Algorithm 1) with starting point $\mathbf{w}^1 = \mathbf{0}$, it holds that the weighted average of the iterates satisfies

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

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} .

Dual Problem

Dual Objective Function

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

The relationship between primal variable \mathbf{w} and dual variable α is

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

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

NonUnifSDCA

Algorithm 2: Non-Uniform Stochastic Dual Coordinate Ascent

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

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

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

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

Sample i_t from $\{1, \dots, n\}$ based on \mathbf{p} ;

Calculate $\Delta\alpha_{i_t}^t =$

$\arg \max_{\Delta\alpha_{i_t}^t} \left[-\frac{\lambda n}{2} \|\mathbf{w}^t + \frac{1}{\lambda n} \Delta\alpha_{i_t}^t \mathbf{x}_{i_t}\|^2 - \ell_{i_t}^*(-(\alpha_{i_t}^t + \Delta\alpha_{i_t}^t)) \right];$

Set $\alpha_{i_t}^{t+1} \leftarrow \alpha_{i_t}^t + \Delta\alpha_{i_t}^t;$

Set $\mathbf{w}^{t+1} \leftarrow \mathbf{w}^t + \frac{1}{\lambda n} \Delta\alpha_{i_t}^t \mathbf{x}_{i_t};$

end

Output: \mathbf{w}^{T+1}

Part B

Adaptive Sampling Algorithms

Idea behind SGD

According to the SGD theorem, we can reduce the convergence rate by solving the following optimization problem:

$$\min \mathbb{E}[\|\mathbf{g}_{i_t}^t\|^2].$$

By the Cauchy-Schwarz inequality and the fact that $\sum_{i=1}^n p_i = 1$,

$$\mathbb{E}[\|\mathbf{g}_{i_t}^t\|^2] = \sum_{i=1}^n \frac{\|\chi_i\|^2}{n^2 p_i} = \left(\sum_{i=1}^n \frac{\|\chi_i\|^2}{n^2 p_i}\right) \left(\sum_{i=1}^n p_i\right) \geq \left(\sum_{i=1}^n \frac{\|\chi_i\|}{n}\right)^2.$$

The above inequality holds when

$$p_i = \frac{\|\chi_i\|}{\sum_{j=1}^n \|\chi_j\|}.$$

AdaSGD

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

Input: $\lambda > 0$

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

Initialize: $w^0 = 0$, 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 $\eta_t \leftarrow$

 Calculate $c_i \leftarrow \max\{c_i, \|\chi_i\|\};$

 Set $\chi_{i_t}^t(w^t) \leftarrow \ell'(x_{i_t}, w^t) + \lambda \nabla r(w^t);$

if $(t-1) \bmod n \geq n-k$ **then**

for $i = 1, 2, \dots, n$

 Calculate $\ell'(x_i, w^t, y_i);$

 Set $\chi_i \leftarrow \ell'(x_i, w^t, y_i)x_i + \lambda \nabla r(w^t);$

 Set $c_i \leftarrow \max\{c_i, \|\chi_i\|\};$

end

end

if $t \bmod n = 0$ **then**

Option I: Run Algorithm 3 (Aggressive Update);

Option II: Run Algorithm 4 (Conservative Update);

end

 Set $g_{i_t}^t \leftarrow \frac{\chi_{i_t}^t(w^t)}{np_{i_t}};$

 Set $w^{t+1} \leftarrow w^t - \eta_t g_{i_t}^t;$

end

Output: w^{T+1}

Two Updates

Algorithm 3: Aggressive Probability Update

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

Algorithm 4: Conservative Probability Update

```
for  $j = 1, \dots, n$   
  | Set  $p_j \leftarrow \frac{\tilde{c}_j}{\sum_{k=1}^n \tilde{c}_k}$ ;  
end
```

For all $i \in \{1, \dots, n\}$, $\tilde{c}_i := \max\{1, c_i\}$.

AdaSVRG (variant of AdaSGD)

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

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

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

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

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 α^t , that is $\mathbf{w}^t(\alpha^t) := \frac{1}{\lambda n} \sum_{i=1}^n \alpha_i \mathbf{x}_i^t$.

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

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

AdaSDCA (Duality Gap)

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

Input: $\lambda > 0$

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

Initialize: $\alpha^1 = 0, w^1 = 0$, probabilities $p_i = \frac{1 + \frac{1}{\lambda w_{y_i}}}{n + \sum_{j=1}^n \frac{1}{\lambda w_{y_j}}}$ or

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

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

 Sample i_t from $\{1, \dots, n\}$ based on p ;

 Calculate $\Delta \alpha_{i_t}^t$ using following formulas:

$\Delta \alpha_{i_t}^t = \arg \max_{\alpha} \text{Set } c_i \leftarrow \max\{c_i, \sigma_i^t\}; \quad 1) \};$

 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 $(t-1) \bmod n \geq n-k$ **then**

for $i = 1, 2, \dots, n$

 Calculate $\sigma_i^t \leftarrow \ell(x_i^T w^t) + l^*(-\alpha_i^t) + \alpha_i^t \langle x_i, w^t \rangle$;

 Set $c_i \leftarrow \max\{c_i, \sigma_i^t\}$;

end

end

if $t \bmod n = 0$ **then**

Option I: Run Algorithm 3 (Aggressive Update);

Option II: Run Algorithm 4 (Conservative Update);

end

end

Output: w^{T+1}

AdaSDCAS (Subgradient)

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

Input: $\lambda > 0$

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

Initialize: $\alpha^1 = 0, w^1 = 0$, probabilities $p_i = \frac{1 + \frac{1}{\lambda n \gamma_i}}{n + \sum_{j=1}^n \frac{1}{\lambda n \gamma_j}}$ or

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

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

 Sample i_t from $\{1, \dots, n\}$ based on p ;

 Calculate $\Delta \alpha_{i_t}^t$ using following formulas:

$\Delta \alpha_{i_t}^t = \arg \max_{\alpha} \{c_{i_t} \leftarrow \max\{c_{i_t}, \|\chi_{i_t}^t\|\}; \alpha\};$

 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 $(t-1) \bmod n \geq n-k$ **then**

for $i = 1, 2, \dots, n$

 Calculate $\ell'(\langle x_i, w^t \rangle, y_i)$;

 Set $\chi_i^t \leftarrow \ell'(\langle x_i, w^t \rangle, y_i) x_i + \lambda \nabla r(w^t)$;

 Record $c_i \leftarrow \max\{c_i, \|\chi_i^t\|\}$;

end

end

if $t \bmod n = 0$ **then**

Option I: Run Algorithm 3 (Aggressive Update);

Option II: Run Algorithm 4 (Conservative Update);

end

end

Output: w^{T+1}

Part C

Discussions and Experiments

Datasets for empirical study

Dataset	Training(n)	Test	Features (d)	Sparsity($\frac{nnz}{nd}$)
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	$(k + 1) \text{ nnz}$	✓	✓
AdaSVRG	$nd + k \text{ nnz}$	✓	✓
AdaSDCA	$(k + 1) \text{ nnz}$	✓	✓
AdaSDCAS	$(k + 1) \text{ nnz}$	✓	✓
AdaGrad (by Duchi)	$2nd$	✗	✗
AdaSDCA (by Csiba)	$n \text{ nnz}$	✓	✓
AdaSDCA+ (by Csiba)	2 nnz	✓	✓

nnz: is the number of nonzero elements of the matrix consisting of all the samples in the dataset.

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 $f(\mathbf{w}) - f(\mathbf{w}^*)$.

Definition

Test error is the error rate on test dataset.

We calculate the value by

$$\ln(f(\mathbf{w}) - f(\mathbf{w}^*) + \epsilon).$$

Performance of Two Updating Algorithms

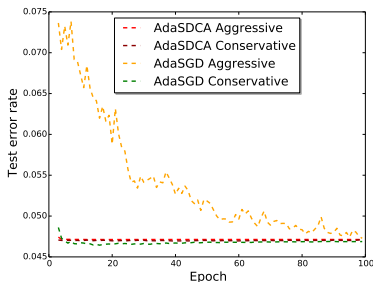
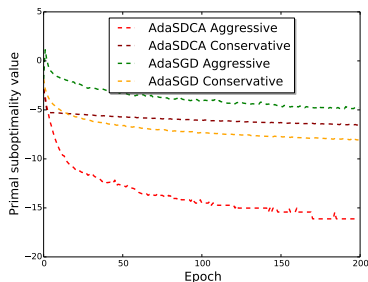


Figure: Comparison of two updating algorithms for AdaSGD and AdaSDCA on rcv1

Different Adaptive Strategies for AdaSDCA

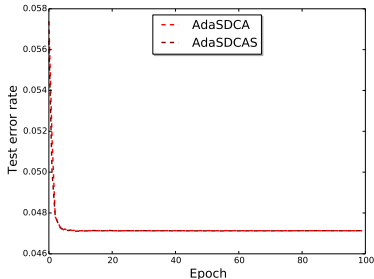
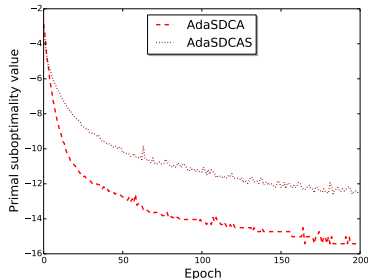


Figure: Comparison of AdaSDCA and AdaSDCAS on rcv1

Comparison of Adaptive Algorithms

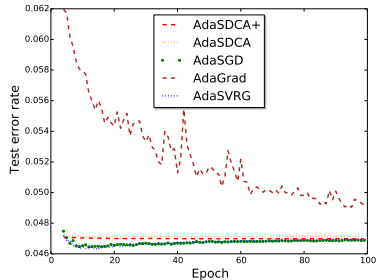
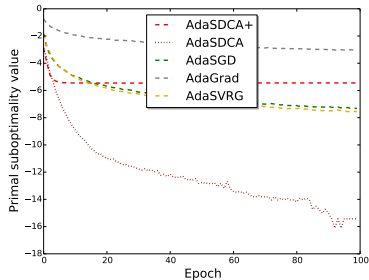


Figure: Comparison of five adaptive algorithms on rcv1

Comparison of Adaptive Algorithms cont.

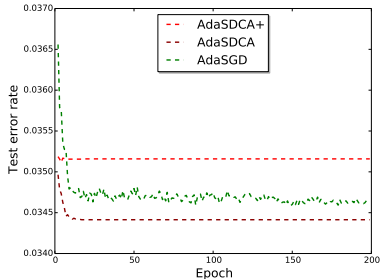
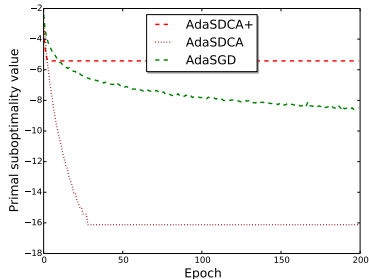


Figure: Comparison of three adaptive algorithms on astro-ph

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 Time

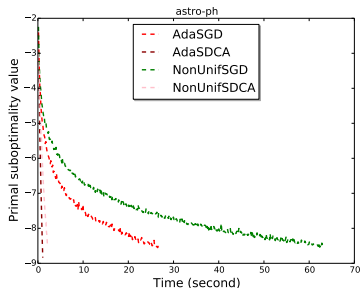
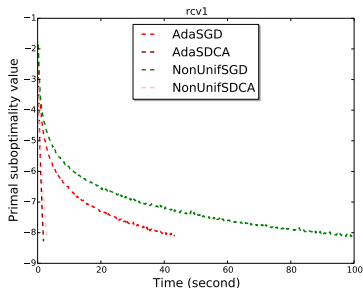


Figure: Comparison of the total running time to reach the same optimality

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 Vector Operation

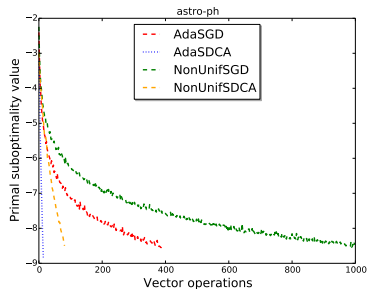
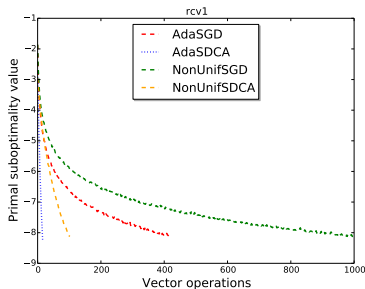


Figure: Comparison of the vector operations taken to reach the same optimality

Adaptive vs. Non-Uniform Algorithms

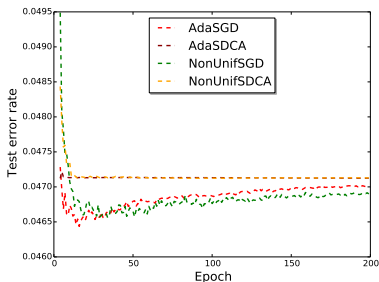
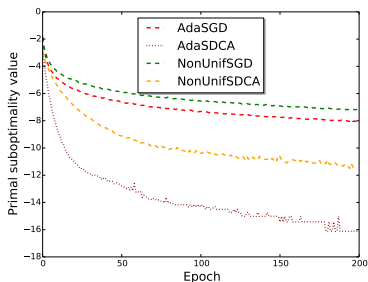


Figure: Comparison of adaptive algorithms with non-adaptive algorithms on rcv1

Adaptive vs. Non-Uniform Algorithms cont.

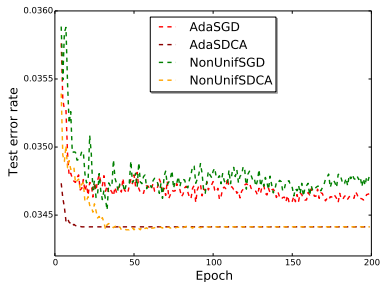
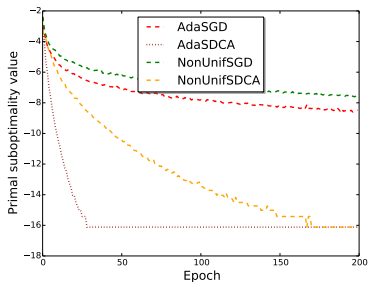


Figure: Comparison of adaptive algorithms with non-adaptive algorithms on astro-ph

Summary

- 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 subgradient).
- AdaSDCA has the best performance among all the adaptive algorithms (AdaSDCA, AdaSGD, AdaSVRG, AdaGrad and AdaSDCA+) and AdaSGD is the second best.
- AdaSVRG achieves a slightly better performance per epoch than AdaSGD but sacrifices the running time on sparse datasets.

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!