Second-Order Stochastic Optimization for Machine Learning in Linear Time

Final Presentation for OPT

Yue Zhao
201611130148

June 9, 2019

Content

1. Background
2. **LiSSA**
3. LiSSA-Sample
4. Results

1. Background

- What kind of method do we usually use?
  - First order methods, e.g. GD, SGD.
  - Second order method, e.g. Newton Method.
- Why SO methods aren't often used in ML?
  - $\mathbf{x}_{t+1} = \mathbf{x}_t - \nabla^{-2} f\left(\mathbf{x}_t\right) \nabla f\left(\mathbf{x}_t\right).$
  - Hessian, $O(md^2)$.
  - Inversion of the Hessian, $O(d^w)$.

1. Background

- Baseline:
  - Empirical risk minimization (ERM) problem:

$$\min_{\mathbf{x}\in\mathbb{R}^d} f(\mathbf{x}) = \min_{\mathbf{x}\in\mathbb{R}^d} \left\{ \frac{1}{m} \sum_{k=1}^{m} f_k(\mathbf{x}) + R(\mathbf{x}) \right\}.$$

  - Newton method:

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \nabla^{-2} f\left(\mathbf{x}_t\right) \nabla f\left(\mathbf{x}_t\right).$$

  - Condition number, $\kappa_l \leq \kappa$:
    - $\kappa \triangleq \frac{\max_x \lambda_{\max}\left(\nabla^2 f\right)}{\min_x \lambda_{\min}\left(\nabla^2 f\right)}$.
    - $\kappa_l \triangleq \max_{\mathbf{x}} \frac{\lambda_{\max}\left(\nabla^2 f(\mathbf{x})\right)}{\lambda_{\min}\left(\nabla^2 f(\mathbf{x})\right)}$.

2.1. LiSSA —— Main Idea

- Alternative for $\nabla^{-2} f$:
  - Tylor Expansion: $A^{-1} = \sum_{i=0}^{\infty} (I - A)^i$.
  - $A_j^{-1} \triangleq \sum_{i=0}^{j} (I - A)^i$, or equivalently $A_j^{-1} \triangleq I + (I - A) A_{j-1}^{-1}$.
  - Estimator: $\tilde{\nabla}^{-2} f_0 = I$ and $\tilde{\nabla}^{-2} f_t = I + (I - X_t)\, \tilde{\nabla}^{-2} f_{t-1}$.

2.2. LiSSA —— Algorithm

Algorithm 1 **LiSSA: Linear (time) Stochastic Second-Order Algorithm**

Input: T, $f(\mathbf{x}) = \sum_{k=1}^{m} f_k(\mathbf{x})$, S_1, S_2, T_1
$\mathbf{x}_1 = FO(f(\mathbf{x}), T_1)$
for $t = 1$ to T **do**
  for $i = 1$ to S_1 **do**
    $X_{[i,0]} = \nabla f(\mathbf{x}_t)$
    for $j = 1$ to S_2 **do**
      Sample $\tilde{\nabla}^2 f_{[i,j]}(\mathbf{x}_t)$ uniformly from $\{\nabla^2 f_k(\mathbf{x}_t) \mid k \in [m]\}$
      $X_{[i,j]} = \nabla f(\mathbf{x}_t) + (I - \tilde{\nabla}^2 f_{[i,j]}(\mathbf{x}_t))X_{[i,j-1]}$
    end for
    $\cancel{X_{[i]} = X_{[i,S_2]}}$
  end for
  $X_t = 1/S_1 \left(\sum_{i=1}^{S_1} X_{[i,S_2]} \right)$
  $\mathbf{x}_{t+1} = \mathbf{x}_t - X_t$
end for
return $\mathbf{x}_{T+1}$

Theorem 3.3

Consider Algorithm 1, and set the parameters as follows:

$$T_1 = FO\left(M, \hat{\kappa}_l\right), S_1 = O\left(\left(\hat{\kappa}_l^{\max}\right)^2 \ln\left(\frac{d}{\delta}\right)\right), S_2 \geq 2\hat{\kappa}_l \ln\left(4\hat{\kappa}_l\right).$$

The following guarantee holds for every $t \geq T_1$ with probability $1-\delta$

$$\|\mathbf{x}_{t+1} - \mathbf{x}^*\| \leq \frac{\|\mathbf{x}_t - \mathbf{x}^*\|}{2}.$$

Moreover, we have that each step of the algorithm takes at most $\tilde{O}\left(md + \left(\hat{\kappa}_l^{\max}\right)^2 \hat{\kappa}_l d^2\right)$ time. Additionally, if f is GLM, then each step of the algorithm can be run in time $O\left(md + \left(\hat{\kappa}_l^{\max}\right)^2 \hat{\kappa}_l d\right)$.

Corollary 3.4

For a GLM function $f(\mathbf{x})$ Algorithm 1 returns a point $\mathbf{x}_t$ such that with probability at least $1-\delta$

$$f(\mathbf{x}_t) \le \min_{\mathbf{x}^*} f(\mathbf{x}^*) + \varepsilon$$

in total time $\tilde{O}\left(m + \left(\hat{\kappa}_l^{\max}\right)^2 \hat{\kappa}_l\right) d \ln\left(\frac{1}{\varepsilon}\right)$ for $\varepsilon \to 0$.

2.5. LiSSA —— Summary

- Main idea: Tylor Expansion Estimator.
- Iteration: in $O(d)$ time.
  - Sparsity: in $O(s)$ time.
- Convergence: Linear.
- Other details:
  - Better on condition number, $\kappa_l \leq \kappa$.
  - Better in high accuracy regime.

3.1. LiSSA-Sample —— Main Idea

- Much better when $m >> d$.
  - Utilize Matrix Sampling Techniques $[CLM + 15]$.

Algorithm 3 REPEATED HALVING

Input: $A = \sum_{i=1}^{m}(\mathbf{v}_i \mathbf{v}_i^T + \lambda I)$
Output: B **an** $O(dlog(d))$ **size weighted sample of** A **and** $B \preceq A \preceq 2B$
Take a uniformly random unweighted sample of size $\frac{m}{2}$ of A to form A'
if A' has size $> O(d\log(d))$ **then**
    Recursively compute an 2-spectral approximation $\tilde{A}'$ of A'
end if
Compute estimates γ_i of generalized leverage scores $\{\hat{\tau}_i^{A'}(A)\}$ s.t. the following are satisfied

$$\gamma_i \geq \hat{\tau}_i^{A'}(A)$$

$$\sum \gamma_i \leq \sum 16\hat{\tau}_i^{A'}(A) + 1$$

Use these estimates to sample matrices from A to form B

- **Time:** $\tilde{O}\left(md + d\sqrt{\kappa_{\text{sample}} d}\right)$.
  - Better condition number, κ_{sample}.

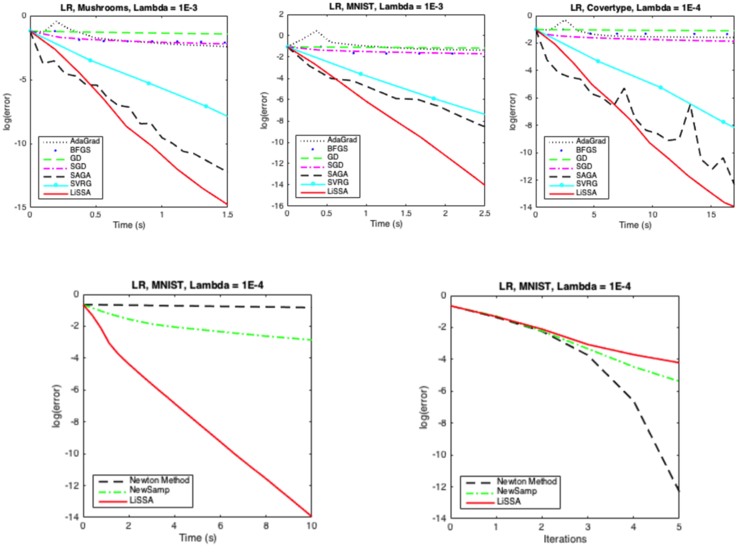
Algorithm 4 Fast Quadratic Solver (FQS)

1: **Input:** $A = \sum_{i=1}^{m}(\mathbf{v}_i\mathbf{v}_i^T + \lambda I)$, $\mathbf{b}$, ε
2: **Output :** $\tilde{\mathbf{v}}$ **s.t.** $\|A^{-1}\mathbf{b} - \tilde{\mathbf{v}}\| \leq \varepsilon$
3: Compute B s.t. $2B \succeq A \succeq B$ using REPEATED HALVING(Algorithm 3)
4: $Q(\mathbf{y}) = \frac{\mathbf{y}^T AB^{-1}\mathbf{y}}{2} + \mathbf{b}^T\mathbf{y}$
5: Compute $\hat{\mathbf{y}}$ such that $\|\hat{\mathbf{y}} - \operatorname{argmin} Q(\mathbf{y})\| \leq \frac{\varepsilon}{4\|B^{-1}\|}$
6: Output $\tilde{\mathbf{v}}$ such that $\|B^{-1}\hat{\mathbf{y}} - \tilde{\mathbf{v}}\| \leq \varepsilon/2$

4.1. Theoretical Results

Algorithm	Runtime
SVRG, SAGA, SDCA	$(md + O(\hat{\kappa} d)) \log(\frac{1}{\varepsilon})$
LiSSA	$(md + O(\hat{\kappa}_l) S_1) \log(\frac{1}{\varepsilon})$
AccSDCA, Catalyst, Katyusha	$\tilde{O}\left(md + d\sqrt{\hat{\kappa} m}\right) \log(\frac{1}{\varepsilon})$
LiSSA-Sample	$\tilde{O}\left(md + d\sqrt{\kappa_{sample} d}\right) \log^2(\frac{1}{\varepsilon})$

4.2. Empirical Results



Thanx :) !