# Implicit Stochastic Average Gradient

Vladislav Chabanenko

Faculty of Computational Mathematics and Cybernetics
Lomonosov Moscow State Universuty

April 10, 2015
Seminar "Bayesian methods in machine learning "

# Outline

# Stochastic and Full Gradient Descent

- We want to solve the following optimization problem:

$$Q(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\theta) \to \min_{\theta\in\mathbb{R}^d}$$

  Suppose all $f_i$ are differentiable and we know their gradients. What methods do we know for solving this problem?

- Full Gradient Descent

$$\theta_{k+1} = \theta_k - \gamma \nabla Q(\theta_k)$$

- Stochastic Gradient Descent

$$\theta_{k+1} = \theta_k - \gamma_k \nabla f_{i_k}(\theta_k), \quad \gamma_k = \frac{\alpha}{k+1}$$

- What is the difference?

# Outline

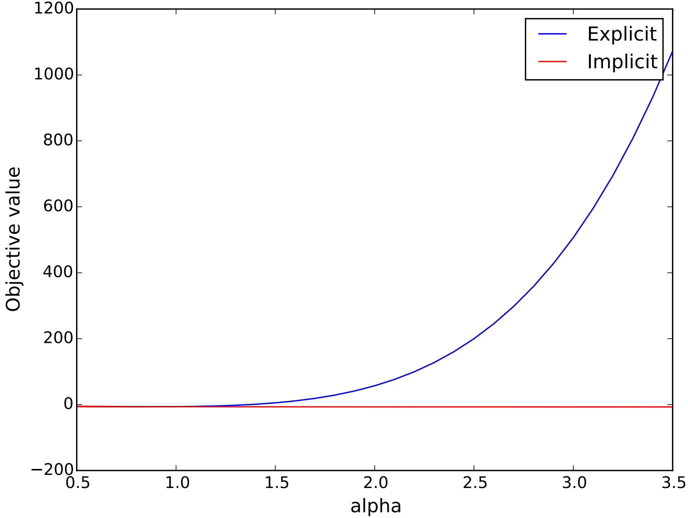
Theory
- Stochastic and Full Gradient Descent
- Explicit and Implicit methods
- Stochastic Average Gradient
- SAG + Implicitness

Experiments
- Models
- Results

# Explicit and Implicit methods

- We can rewrite FG and SGD schemes in **implicit** style
- Implicit FG
$$\theta_{k+1} = \theta_k - \gamma \nabla Q(\boldsymbol{\theta_{k+1}})$$
- Implicit SGD
$$\theta_{k+1} = \theta_k - \gamma_k \nabla f_i(\boldsymbol{\theta_{k+1}})$$
- Advantages: stability for learning rate setting and usually better results
- Drawbacks: more complicated implementation, more time-consuming iterations

# Learning rate: example



# Outline

Theory
- Stochastic and Full Gradient Descent
- Explicit and Implicit methods
- Stochastic Average Gradient
- SAG + Implicitness

Experiments
- Models
- Results

# Stochastic Average Gradient (SAG)

- The SAG method incorporates both SGD and FG: it has the low iteration cost of SGD, but makes gradient step with respect to the approximation of the full gradient
- The SAG iterations take the following form

$$\theta_{k+1} = \theta_k - \frac{\gamma}{n}\sum_{i=1}^{n} g_i^k,$$

where at each iteration a random index $i_k$ is selected and we set

$$g_i^k = \begin{cases} f'_{i_k}(\theta_k) & \text{if } i = i_k, \\ g_i^{k-1} & \text{otherwise} \end{cases}$$

- To achieve low iteration cost we just need to store the table of gradients $g_i^k$ and their sum

# Learning rate for SAG

- If the following inequality holds

$$||h(y) - h(x)|| \leqslant L||y - x||, \quad \forall x, y$$

  then $L$ is called Lipschitz constant for a function $h$
- If $L$ is Lipschitz constant for all $f_i'$ then it claims that SAG achieves FG convergence rates with $\gamma = \frac{1}{16L}$. But in practice authors use $\gamma = \frac{1}{L}$ that gives even better results (higher $\gamma$ may be better, but not always)
- In general $L$ will not be known, but we can use a basic line-search: we start with an initial estimate $L_0$, and at each iteration we double this estimate while the following inequality is not satisfied

$$f_{i_k}\left(\theta_k - \frac{1}{L_k} f_{i_k}'(\theta_k)\right) \leqslant f_{i_k}'(\theta_k) - \frac{1}{2L_k}||f_{i_k}'(\theta_k)||^2,$$

  which must be true if $L_k$ is valid.

# Outline

# Implicit SAG

- Now review our own research
- We have that IFG and ISGD outperform their explicit versions and are more stable for learning rate setting
- We try to introduce implicitness for SAG as follows

$$\theta_{k+1} = \theta_k - \frac{\gamma}{n} \sum_{i=1}^{n} g_i^k,$$

$$g_i^k = \begin{cases} f'_{i_k}(\boldsymbol{\theta_{k+1}}) & \text{if } i = i_k, \\ g_i^{k-1} & \text{otherwise} \end{cases}$$

# Outline

# Models

- **Linear regression**: We solve the following optimization problem:

$$\frac{1}{n}\sum_{i=1}^{n}\left(\frac{(x_i^T\theta)^2}{2} - y_i x_i^T\theta\right) \to \min_{\theta}$$

  where $x_i \in \mathbb{R}^d$ are features and $y_i \in \mathbb{R}$ is response. Here we generate synthetic data: $x_i \sim \mathcal{N}(0, V_x), y_i \sim \mathcal{N}(x_i^T\theta, 1), d = 20$. We generate $n = 10000$ objects

- **Logistic regression**: We solve the following optimization problem:

$$\frac{\lambda}{2}||\theta||^2 + \frac{1}{n}\sum_{i=1}^{n}\log(1+\exp(-y_i x_i^T\theta)) \to \min_{\theta},$$

  where $x_i \in \mathbb{R}^d$ are features and $y_i \in \{-1, 1\}$ is a label for binary classification. We use the *quantum* dataset obtained from the KDD Cup 2004 website. [1] It contains $n = 50000$ objects with $d = 78$

[1] http://osmot.cs.cornell.edu/kddcup

# Considered methods

For aforemetioned models we will compare the following optimization methods:

- SGD and Implicit SGD
- FG and Implicit FG
- SAG and Implicit SAG
- Moreover we will compare our methods to the state-of-the-art method BFGS

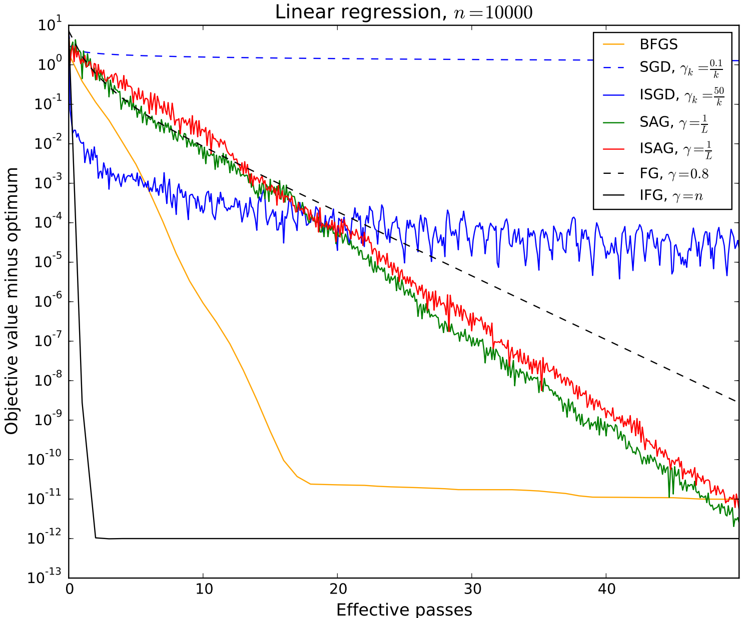
For all the methods we tune a learning rate (where it is required)

# Implementation remarks

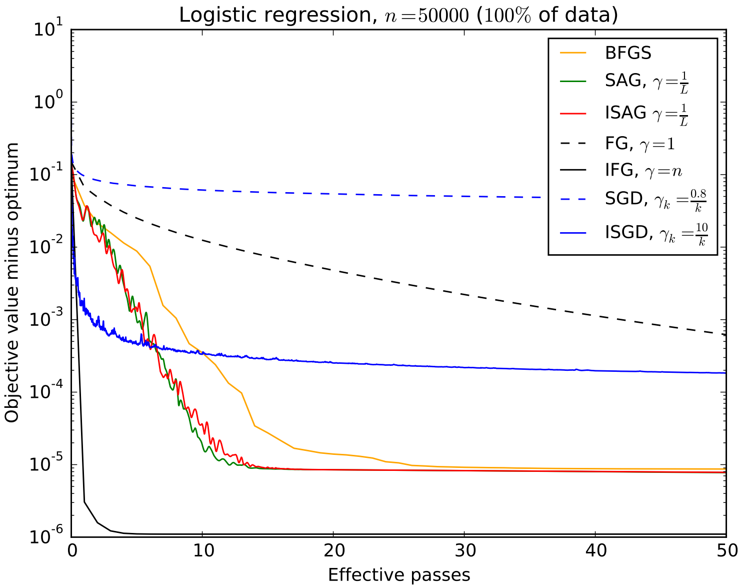
- In linear regression for every method we can derive all the formulae analytically
- In logistic regression we can't do this. Therefore, we need to solve additional optimization problem at each step. ISGD and ISAG require solving an one-dimensional equation that we solve with Newton method; IFG requires solving a system of nonlinear equations that we solve with Newton-Krylov method

# Outline

# Experiments, Linear regression



# Experiments, Logistic regression



# Conclusion

- Implicit methods have a big advantage over their explicit antagonists except SAG
- Implicit FG shows very impressive results, but it can be applied only in the case of small $n$ and $d$
- ISAG and SAG show similiar results

# Future work

- We will try to change our intuition of implicit SAG to make it closer to implicit FG
- We will try to apply optimization scheme with mini-batches for ISGD and SAG/ISAG