A fast incremental second-order optimization method with a superlinear rate of convergence

Anton Rodomanov Dmitry Kropotov

Bayesian methods research group (http://bayesgroup.ru)
Lomonosov Moscow State University

Solnechnogorsk, 2015

- Need to solve an ℓ_2-regularized empirical risk minimization problem:

$$\min_{\mathbf{w}\in\mathbb{R}^D} \left[F(\mathbf{w}) := \frac{1}{N}\sum_{i=1}^{N} f_i(\mathbf{w}) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2 \right]$$

  with $\lambda > 0$.
- E.g., logistic regression:

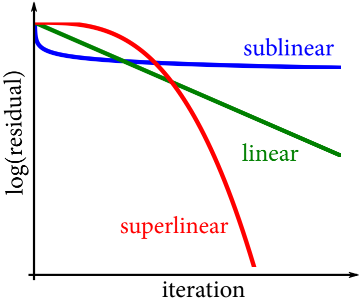
$$f_i(\mathbf{w}) := \ln(1 + \exp(-y_i \mathbf{w}^\top \mathbf{x}_i))$$

- Assumptions:
  - all f_i are twice continuously differentiable and convex
  - The Hessians $\nabla^2 f_i$ satisfy the Lipschitz condition:

$$\left\|\nabla^2 f_i(\mathbf{w}) - \nabla^2 f_i(\mathbf{u})\right\|_2 \le M \|\mathbf{w} - \mathbf{u}\|_2, \qquad \forall \mathbf{w}, \mathbf{u} \in \mathbb{R}^D.$$

Motivation

- Assume N is very large and D is small/moderate.
- Use methods whose iteration cost does not depend on N.
- They are called incremental methods [Bertsekas, 2011].
- All of them have either a sublinear or linear rate of convergence.
- We are interested in a very small error (say, 1e-8 or smaller).
- Goal: an incremental method with a superlinear rate of convergence.



NIM: a Newton-type incremental method

- Quadratic model of f_i with the center at $\mathbf{v}_i^k$:

$$q_i^k(\mathbf{w}) := f_i(\mathbf{v}_i^k) + \nabla f_i(\mathbf{v}_i^k)^\top(\mathbf{w} - \mathbf{v}_i^k) + \frac{1}{2}(\mathbf{w} - \mathbf{v}_i^k)^\top \nabla^2 f_i(\mathbf{v}_i^k)(\mathbf{w} - \mathbf{v}_i^k).$$

- Model of the full function F:

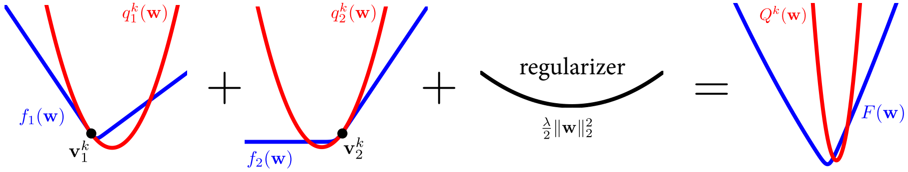
$$Q^k(\mathbf{w}) := \frac{1}{N}\sum_{i=1}^{N} q_i^k(\mathbf{w}) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2.$$

- Iteration:
  - Choose a number $i_k \in \{1, \ldots, N\}$.
  - Update only one component: $\mathbf{v}_{i_k}^k := \mathbf{w}_k, \quad \mathbf{v}_i^k := \mathbf{v}_i^{k-1},\ i \neq i_k$.
  - Find the model's minimum: $\bar{\mathbf{w}}_k := \operatorname{argmin}_{\mathbf{w} \in \mathbb{R}^D} Q^k(\mathbf{w})$.
  - Make a step in the direction of the model's minimum:

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \alpha_k(\bar{\mathbf{w}}_k - \mathbf{w}_k),$$

where $\alpha_k > 0$ is the step length.

Illustration



- Minimum of the model:
$$\bar{\mathbf{w}}_k = (\mathbf{H}_k + \lambda \mathbf{I})^{-1}(\mathbf{p}_k - \mathbf{g}_k),$$
where
$$\mathbf{H}_k := \frac{1}{N}\sum_{i=1}^N \nabla^2 f_i(\mathbf{v}_i^k), \quad \mathbf{p}_k := \frac{1}{N}\sum_{i=1}^N \nabla^2 f_i(\mathbf{v}_i^k)\mathbf{v}_i^k, \quad \mathbf{g}_k := \frac{1}{N}\sum_{i=1}^N \nabla f_i(\mathbf{v}_i^k)$$
- Update using the "add-subtract" principle:
$$\mathbf{H}_k = \mathbf{H}_{k-1} + \frac{1}{N}\left(\nabla^2 f_{i_k}(\mathbf{w}_k) - \nabla^2 f_{i_k}(\mathbf{v}_{i_k}^{k-1})\right),$$
$$\mathbf{p}_k = \mathbf{p}_{k-1} + \frac{1}{N}\left(\nabla^2 f_{i_k}(\mathbf{w}_k)\mathbf{w}_k - \nabla^2 f_{i_k}(\mathbf{v}_{i_k}^{k-1})\mathbf{v}_{i_k}^{k-1}\right),$$
$$\mathbf{g}_k = \mathbf{g}_{k-1} + \frac{1}{N}\left(\nabla f_{i_k}(\mathbf{w}_k) - \nabla f_{i_k}(\mathbf{v}_{i_k}^{k-1})\right),$$
where $i_k \in \{1, \dots, N\}$ is the number of the component to update.
- Iteration complexity: $O(D^3)$ to solve the linear system.
- Memory: $O(ND + D^2)$ for storing $\mathbf{H}_k$ and all $\mathbf{v}_i^k$.

The algorithm

NIM: a Newton-type incremental method

Require: $\mathbf{w} \in \mathbb{R}^D$: initial point; $K \in \mathbb{N}$: number of iterations.

Initialize: $\mathbf{H} \leftarrow \mathbf{0}^{D\times D}$; $\mathbf{p} \leftarrow \mathbf{0}^D$; $\mathbf{g} \leftarrow \mathbf{0}^D$; $\mathbf{v}_i \leftarrow$ undefined, $i = 1, \ldots, N$
for $k = 0, 1, 2, \ldots, K-1$ **do**
    Choose an index (cyclic order): $i \leftarrow k \bmod N + 1$
    Update the average Hessian, scaled center and gradient:
        $\mathbf{H} \leftarrow \mathbf{H} + (1/N)[\nabla^2 f_i(\mathbf{w}) - \nabla^2 f_i(\mathbf{v}_i)]$
        $\mathbf{p} \leftarrow \mathbf{p} + (1/N)[\nabla^2 f_i(\mathbf{w})\mathbf{w} - \nabla^2 f_i(\mathbf{v}_i)\mathbf{v}_i]$
        $\mathbf{g} \leftarrow \mathbf{g} + (1/N)[\nabla f_i(\mathbf{w}) - \nabla f_i(\mathbf{v}_i)]$
    Move the ith center: $\mathbf{v}_i \leftarrow \mathbf{w}$
    Find the model's minimum: $\bar{\mathbf{w}} \leftarrow (\mathbf{H} + \lambda\mathbf{I})^{-1}(\mathbf{p} - \mathbf{g})$
    Make a step: $\mathbf{w} \leftarrow \mathbf{w} + \alpha(\bar{\mathbf{w}} - \mathbf{w})$ for some $\alpha > 0$
end for
return $\mathbf{w}$

Assume no subtraction is performed when $\mathbf{v}_i$ = undefined.

- Linear models: $f_i(\mathbf{w}) := \phi_i(\mathbf{x}_i^\top \mathbf{w})$ for some $\mathbf{x}_i \in \mathbb{R}^D$
- The gradients and Hessians have a special structure:
$$\nabla f_i(\mathbf{w}) = \phi_i'(\mathbf{x}_i^\top \mathbf{w})\mathbf{x}_i,$$
$$\nabla^2 f_i(\mathbf{w}) = \phi_i''(\mathbf{x}_i^\top \mathbf{w})\mathbf{x}_i\mathbf{x}_i^\top.$$
- Instead of $\mathbf{v}_i^k$ we can store only the dot produts:
$$\mu_i^k := \mathbf{x}_i^\top \mathbf{v}_i^k.$$
- No need for solving the linear system, update $\mathbf{B}_k := (\mathbf{H}_k + \lambda \mathbf{I})^{-1}$:
$$\mathbf{B}_k = \mathbf{B}_{k-1} - \frac{\delta_k \mathbf{B}_{k-1}\mathbf{x}_{i_k}\mathbf{x}_{i_k}^\top \mathbf{B}_{k-1}}{N + \delta_k \mathbf{x}_{i_k}^\top \mathbf{B}_{k-1}\mathbf{x}_{i_k}},$$
where $\delta_k := \phi_{i_k}''(\mu_{i_k}^k) - \phi_{i_k}''(\mu_{i_k}^{k-1})$.
- Iteration complexity: $O(D^2)$ instead of $O(D^3)$.
- Memory: $O(N + D^2)$ instead of $O(ND + D^2)$.

Rate of convergence

Theorem (local rate of convergence)

- *Let all the centers be initialized close enough to the optimum* $\mathbf{w}_*$*:*

$$\left\|\mathbf{v}_i^0 - \mathbf{w}_*\right\|_2 \le \frac{2\lambda}{M\sqrt{N}}.$$

- *Assume the unit step length* $\alpha_k \equiv 1$ *is used.*

Then $\{\mathbf{w}_k\}$ *converges to* $\mathbf{w}_*$ *at an R-superlinear rate:*

$$\left\|\mathbf{w}_k - \mathbf{w}_*\right\|_2 \le r_k \qquad \textit{and} \qquad \lim_{k\to\infty} \frac{r_{k+1}}{r_k} = 0.$$

Moreover, $\{\mathbf{w}_k\}$ *also has an* N*-step R-quadratic rate of convergence:*

$$r_{k+N} \le \frac{M}{2\lambda} r_k^2, \qquad k = 2N, 2N+1, \ldots.$$

Theoretical comparison with other methods

Function: $F(\mathbf{w}) := (1/N)\sum_{i=1}^{N} \phi_i(\mathbf{x}_i^\top \mathbf{w}) + (\lambda/2)\,\|\mathbf{w}\|_2^2$.

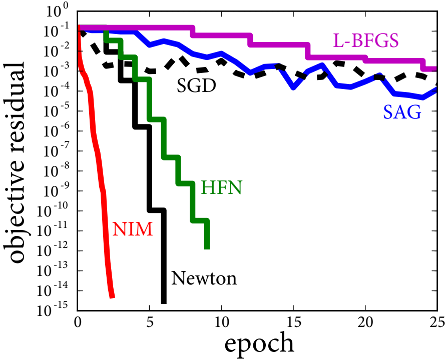
Method	Iteration cost	Memory	Rate of convergence	
			In iterations	**In epochs**
SGD	$O(D)$	$O(D)$	Sublinear	Sublinear
SAG	$O(D)$	$O(N+D)$	Linear	Linear
NIM	$O(D^2)$	$O(N+D^2)$	Superlinear	Quadratic

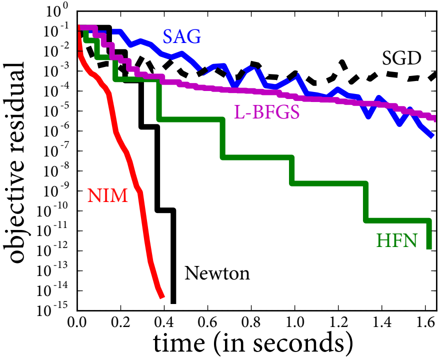
Notation:

- N = number of functions;
- D = number of variables;
- One epoch = N iterations.
- SGD = stochastic gradient method.
- SAG = stochastic average gradient of [Schmidt et al., 2013].

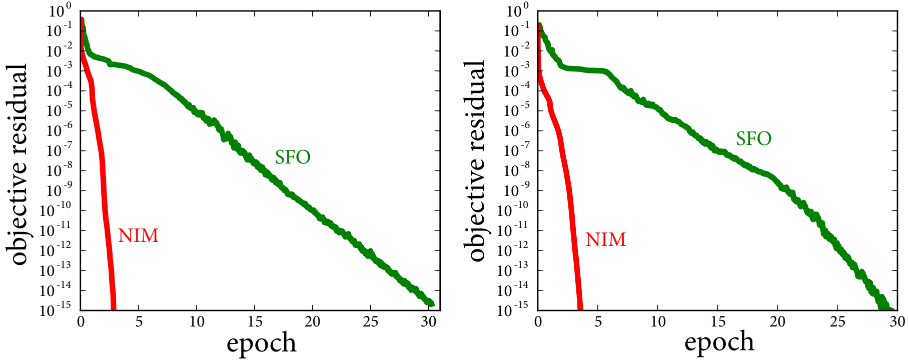
Experimental evaluation: moderate N

- Objective: ℓ_2-regularized logistic regression.
- Dataset *quantum* (25 MB; $N = 50\,000$, $D = 65$):

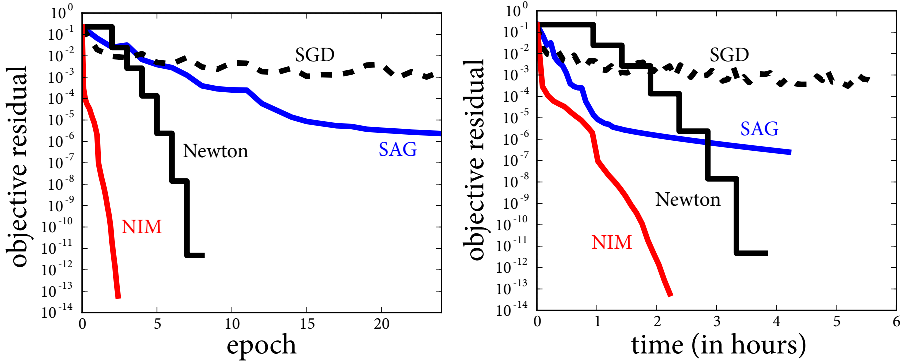




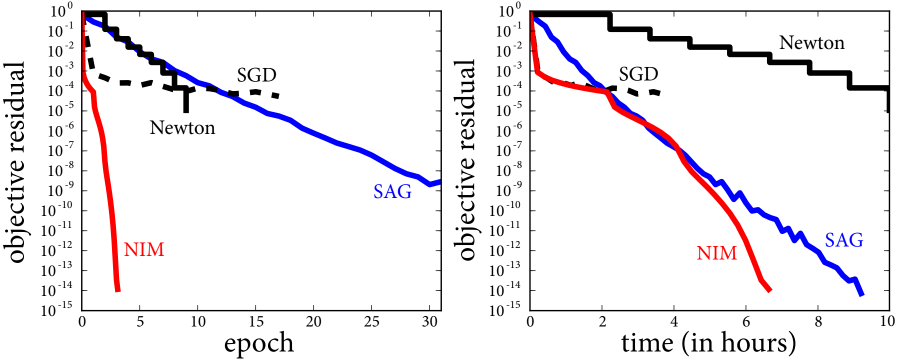
- Datasets *a9a* ($N = 32\,561$, $D = 125$) and *covtype* ($N = 581\,012$, $D = 54$).
- Compare with SFO [Sohl-Dickstein et al., 2014]:



- Dataset *mnist8m* (47 GB; $N = 8\,100\,000$, $D = 784$):



- Dataset *dna18m* (107 GB; $N = 18\,000\,000$, $D = 800$):



Conclusion and discussion

- New incremental second-order Newton-type method.
- Superlinear rate of convergence.
- Can be efficiently applied for linear models.
- Works better than other methods for a small number of variables.
- Does not work for problems with a lot of variables.