A Generic Quasi-Newton Algorithm for Faster Gradient-Based Optimization

Hongzhou Lin\textsuperscript{1}, Julien Mairal\textsuperscript{1}, Zaid Harchaoui\textsuperscript{2}

\textsuperscript{1}Inria, Grenoble \quad \textsuperscript{2}UofW

Les Houches, OSL 2017
An alternate title: Acceleration by Smoothing
Collaborators

Hongzhou Lin
Zaid Harchaoui
Dima Drusvyatskiy
Courtney Paquette

Publications and pre-prints


Focus of this work

Minimizing large finite sums

Consider the minimization of a large sum of convex functions

\[
\min_{x \in \mathbb{R}^d} \left \{ f(x) \triangleq \frac{1}{n} \sum_{i=1}^{n} f_i(x) + \psi(x) \right \},
\]

where each \( f_i \) is smooth and convex and \( \psi \) is a convex regularization penalty but not necessarily differentiable.

Motivation

<table>
<thead>
<tr>
<th></th>
<th>Composite</th>
<th>Finite sum</th>
<th>Exploit “curvature”</th>
</tr>
</thead>
<tbody>
<tr>
<td>First-order methods</td>
<td>✔️</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Quasi-Newton</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

[Nesterov, 2013, Wright et al., 2009, Beck and Teboulle, 2009],...
Focus of this work

Minimizing large finite sums

Consider the minimization of a large sum of convex functions

$$\min_{x \in \mathbb{R}^d} \left\{ f(x) \triangleq \frac{1}{n} \sum_{i=1}^{n} f_i(x) + \psi(x) \right\},$$

where each $f_i$ is smooth and convex and $\psi$ is a convex regularization penalty but not necessarily differentiable.

Motivation

<table>
<thead>
<tr>
<th></th>
<th>Composite</th>
<th>Finite sum</th>
<th>Exploit “curvature”</th>
</tr>
</thead>
<tbody>
<tr>
<td>First-order methods</td>
<td>✔</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>Quasi-Newton</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

[Schmidt et al., 2017, Xiao and Zhang, 2014, Defazio et al., 2014a,b, Shalev-Shwartz and Zhang, 2012, Mairal, 2015, Zhang and Xiao, 2015]
Focus of this work

Minimizing large finite sums

Consider the minimization of a large sum of convex functions

\[
\min_{x \in \mathbb{R}^d} \left\{ f(x) \triangleq \frac{1}{n} \sum_{i=1}^{n} f_i(x) + \psi(x) \right\},
\]

where each \( f_i \) is smooth and convex and \( \psi \) is a convex regularization penalty but not necessarily differentiable.

Motivation

<table>
<thead>
<tr>
<th></th>
<th>Composite</th>
<th>Finite sum</th>
<th>Exploit “curvature”</th>
</tr>
</thead>
<tbody>
<tr>
<td>First-order methods</td>
<td>✔</td>
<td>✔</td>
<td>✗</td>
</tr>
<tr>
<td>Quasi-Newton</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Focus of this work

Minimizing large finite sums
Consider the minimization of a large sum of convex functions

$$\min_{x \in \mathbb{R}^d} \left\{ f(x) \triangleq \frac{1}{n} \sum_{i=1}^{n} f_i(x) + \psi(x) \right\},$$

where each $f_i$ is smooth and convex and $\psi$ is a convex regularization penalty but not necessarily differentiable.

Motivation

<table>
<thead>
<tr>
<th></th>
<th>Composite</th>
<th>Finite sum</th>
<th>Exploit “curvature”</th>
</tr>
</thead>
<tbody>
<tr>
<td>First-order methods</td>
<td>✔</td>
<td>✔</td>
<td>X</td>
</tr>
<tr>
<td>Quasi-Newton</td>
<td>—</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

[Byrd et al., 2015, Lee et al., 2012, Scheinberg and Tang, 2016, Yu et al., 2008, Ghadimi et al., 2015],…
Focus of this work

Minimizing large finite sums
Consider the minimization of a large sum of convex functions

$$\min_{x \in \mathbb{R}^d} \left\{ f(x) \triangleq \frac{1}{n} \sum_{i=1}^{n} f_i(x) + \psi(x) \right\},$$

where each $f_i$ is smooth and convex and $\psi$ is a convex regularization penalty but not necessarily differentiable.

Motivation

<table>
<thead>
<tr>
<th></th>
<th>Composite</th>
<th>Finite sum</th>
<th>Exploit “curvature”</th>
</tr>
</thead>
<tbody>
<tr>
<td>First-order methods</td>
<td>✔</td>
<td>✔</td>
<td>✗</td>
</tr>
<tr>
<td>Quasi-Newton</td>
<td>—</td>
<td>✗</td>
<td>✔</td>
</tr>
</tbody>
</table>

[Byrd et al., 2016, Gower et al., 2016]
Focus of this work

Minimizing large finite sums

Consider the minimization of a large sum of convex functions

\[
\min_{x \in \mathbb{R}^d} \left\{ f(x) \triangleq \frac{1}{n} \sum_{i=1}^{n} f_i(x) + \psi(x) \right\},
\]

where each \( f_i \) is \textbf{smooth and convex} and \( \psi \) is a convex regularization penalty but not necessarily differentiable.

Motivation

Our goal is to

- \textbf{accelerate first-order methods} with Quasi-Newton heuristics;
- design algorithms that can adapt to composite and finite-sum structures and that can also exploit curvature information.

[Byrd et al., 2016, Gower et al., 2016]
QuickeNing: main idea

Idea: Smooth the function and then apply Quasi-Newton.

- The strategy appears in early work about variable metric bundle methods. [Chen and Fukushima, 1999, Fukushima and Qi, 1996, Mifflin, 1996, Fuentes, Malick, and Lemaréchal, 2012, Burke and Qian, 2000] ...
QuickeNing: main idea

Idea: Smooth the function and then apply Quasi-Newton.

- The strategy appears in early work about variable metric bundle methods. [Chen and Fukushima, 1999, Fukushima and Qi, 1996, Mifflin, 1996, Fuentes, Malick, and Lemaréchal, 2012, Burke and Qian, 2000] ...

The Moreau-Yosida smoothing

Given $f : \mathbb{R}^d \to \mathbb{R}$ a convex function, the Moreau-Yosida smoothing of $f$ is the function $F : \mathbb{R}^d \to \mathbb{R}$ defined as

$$F(x) = \min_{w \in \mathbb{R}^d} \left\{ f(w) + \frac{\kappa}{2} \| w - x \|^2 \right\} .$$

The proximal operator $p(x)$ is the unique minimizer of the problem.
The Moreau-Yosida regularization

\[ F(x) = \min_{w \in \mathbb{R}^d} \left\{ f(w) + \frac{\kappa}{2} \| w - x \|^2 \right\}. \]

**Basic properties** [see Lemaréchal and Sagastizábal, 1997]

- Minimizing \( f \) and \( F \) is equivalent in the sense that
  \[ \min_{x \in \mathbb{R}^d} F(x) = \min_{x \in \mathbb{R}^d} f(x), \]
  and the solution set of the two problems coincide with each other.

- \( F \) is continuously differentiable even when \( f \) is not and
  \[ \nabla F(x) = \kappa(x - p(x)). \]

  In addition, \( \nabla F \) is Lipschitz continuous with parameter \( L_F = \kappa \).

- If \( f \) is \( \mu \)-strongly convex then \( F \) is also strongly convex with parameter \( \mu_F = \frac{\mu \kappa}{\mu + \kappa} \).
The Moreau-Yosida regularization

\[ F(x) = \min_{w \in \mathbb{R}^d} \left\{ f(w) + \frac{\kappa}{2} \| w - x \|^2 \right\}. \]

Basic properties [see Lemaréchal and Sagastizábal, 1997]

- Minimizing \( f \) and \( F \) is equivalent in the sense that

\[ \min_{x \in \mathbb{R}^d} F(x) = \min_{x \in \mathbb{R}^d} f(x), \]

and the solution set of the two problems coincide with each other.

- \( F \) is continuously differentiable even when \( f \) is not and

\[ \nabla F(x) = \kappa(x - p(x)). \]

In addition, \( \nabla F \) is Lipschitz continuous with parameter \( L_F = \kappa \).

\( F \) enjoys nice properties: smoothness, (strong) convexity and we can control its condition number \( 1 + \kappa/\mu \).
A fresh look at Catalyst
A fresh look at the proximal point algorithm

A naive approach consists of minimizing the smoothed objective $F$ instead of $f$ with a method designed for smooth optimization.

Consider indeed

$$x_{k+1} = x_k - \frac{1}{\kappa} \nabla F(x_k).$$

By rewriting the gradient $\nabla F(x_k)$ as $\kappa (x_k - p(x_k))$, we obtain

$$x_{k+1} = p(x_k) = \arg \min_{w \in \mathbb{R}^p} \left\{ f(w) + \frac{\kappa}{2} \| w - x_k \|^2 \right\}.$$

This is exactly the **proximal point algorithm** [Rockafellar, 1976].
A fresh look at the accelerated proximal point algorithm

Consider now

\[ x_{k+1} = y_k - \frac{1}{\kappa} \nabla F(y_k) \quad \text{and} \quad y_{k+1} = x_{k+1} + \beta_{k+1}(x_{k+1} - x_k), \]

where \( \beta_{k+1} \) is a Nesterov-like extrapolation parameter. We may now rewrite the update using the value of \( \nabla F \), which gives:

\[ x_{k+1} = p(y_k) \quad \text{and} \quad y_{k+1} = x_{k+1} + \beta_{k+1}(x_{k+1} - x_k) \]

This is the \textit{accelerated proximal point algorithm} of Güler [1992].
A fresh look at the accelerated proximal point algorithm

Consider now

\[ x_{k+1} = y_k - \frac{1}{\kappa} \nabla F(y_k) \quad \text{and} \quad y_{k+1} = x_{k+1} + \beta_{k+1}(x_{k+1} - x_k), \]

where \( \beta_{k+1} \) is a Nesterov-like extrapolation parameter. We may now rewrite the update using the value of \( \nabla F \), which gives:

\[ x_{k+1} = p(y_k) \quad \text{and} \quad y_{k+1} = x_{k+1} + \beta_{k+1}(x_{k+1} - x_k) \]

This is the \text{accelerated proximal point algorithm} of Güler [1992].

Remarks

- \( F \) may be \text{better conditioned} than \( f \) when \( 1 + \kappa/\mu \leq L/\mu \);
- Computing \( p(y_k) \) has a cost!
A fresh look at Catalyst [Lin, Mairal, and Harchaoui, 2015]

Catalyst is a particular accelerated proximal point algorithm with inexact gradients [Güler, 1992].

\[ x_{k+1} \approx p(y_k) \quad \text{and} \quad y_{k+1} = x_{k+1} + \beta_{k+1}(x_{k+1} - x_k) \]

The quantity \( x_{k+1} \) is obtained by using an optimization method \( \mathcal{M} \) for approximately solving:

\[
\begin{align*}
  x_{k+1} & \approx \arg \min_{w \in \mathbb{R}^p} \left\{ f(w) + \frac{\kappa}{2} \| w - y_k \|^2 \right\},
\end{align*}
\]

Catalyst provides Nesterov’s acceleration to \( \mathcal{M} \) with...

- **restart strategies** for solving the sub-problems;
- **global complexity analysis** resulting in theoretical acceleration.
- **parameter choices** (as a consequence of the complexity analysis);

see also [Frostig et al., 2015] and [Paquette, Lin, Drusvyatskiy, Mairal, and Harchaoui, 2017]
Quasi-Newton methods work with the parameter and gradient differences between successive iterations:

\[ s_k \triangleq x_{k+1} - x_k, \quad y_k \triangleq \nabla f(x_{k+1}) - \nabla f(x_k). \]
Quasi-Newton and L-BFGS
Presentation borrowed from Mark Schmidt, NIPS OPT 2010

- **Quasi-Newton** methods work with the parameter and gradient differences between successive iterations:

\[ s_k \triangleq x_{k+1} - x_k, \quad y_k \triangleq \nabla f(x_{k+1}) - \nabla f(x_k). \]

- They start with an initial approximation \( B_0 \triangleq \sigma I \), and choose \( B_{k+1} \) to **interpolate the gradient difference**:

\[ B_{k+1}s_k = y_k. \]
Quasi-Newton and L-BFGS

Presentation borrowed from Mark Schmidt, NIPS OPT 2010

- **Quasi-Newton** methods work with the parameter and gradient differences between successive iterations:

\[ s_k \triangleq x_{k+1} - x_k, \quad y_k \triangleq \nabla f(x_{k+1}) - \nabla f(x_k). \]

- They start with an initial approximation \( B_0 \triangleq \sigma I \), and choose \( B_{k+1} \) to **interpolate the gradient difference**:

\[ B_{k+1} s_k = y_k. \]

- Since \( B_{k+1} \) is not unique, the Broyden-Fletcher-Goldfarb-Shanno (BFGS) method chooses the symmetric matrix whose difference with \( B_k \) is minimal:

\[
B_{k+1} = B_k - \frac{B_k s_k s_k B_k}{s_k B_k s_k} + \frac{y_k y_k^\top}{y_k^\top s_k}.
\]
Update skipping/damping or a sophisticated line search (Wolfe conditions) can keep $B_{k+1}$ positive-definite.
Quasi-Newton and L-BFGS
Presentation borrowed from Mark Schmidt, NIPS OPT 2010

- Update skipping/damping or a sophisticated line search (Wolfe conditions) can keep $B_{k+1}$ positive-definite.
- They perform updates of the form

$$x_{k+1} \leftarrow x_k - \eta_k B_k^{-1} \nabla f(x_k).$$
Quasi-Newton and L-BFGS

Presentation borrowed from Mark Schmidt, NIPS OPT 2010

- Update skipping/damping or a sophisticated line search (Wolfe conditions) can keep $B_{k+1}$ positive-definite.
- They perform updates of the form

$$x_{k+1} \leftarrow x_k - \eta_k B_k^{-1} \nabla f(x_k).$$

- The BFGS method has a superlinear convergence rate.
Quasi-Newton and L-BFGS
Presentation borrowed from Mark Schmidt, NIPS OPT 2010

- Update skipping/damping or a sophisticated line search (Wolfe conditions) can keep $B_{k+1}$ positive-definite.
- They perform updates of the form
  \[ x_{k+1} \leftarrow x_k - \eta_k B_k^{-1} \nabla f(x_k). \]
- The BFGS method has a superlinear convergence rate.
- But, it still uses a dense $p \times p$ matrix $B_k$. 
Quasi-Newton and L-BFGS
Presentation borrowed from Mark Schmidt, NIPS OPT 2010

- Update skipping/damping or a sophisticated line search (Wolfe conditions) can keep $B_{k+1}$ positive-definite.
- They perform updates of the form

$$x_{k+1} \leftarrow x_k - \eta_k B_k^{-1} \nabla f(x_k).$$

- The BFGS method has a superlinear convergence rate.
- But, it still uses a dense $p \times p$ matrix $B_k$.
- Instead of storing $B_k$, the limited-memory BFGS (L-BFGS) method stores the previous $l$ differences $s_k$ and $y_k$. 
Quasi-Newton and L-BFGS
Presentation borrowed from Mark Schmidt, NIPS OPT 2010

- Update skipping/damping or a sophisticated line search (Wolfe conditions) can keep \( B_{k+1} \) positive-definite.
- They perform updates of the form

\[
x_{k+1} \leftarrow x_k - \eta_k B_k^{-1} \nabla f(x_k).
\]

- The BFGS method has a superlinear convergence rate.
- But, it still uses a dense \( p \times p \) matrix \( B_k \).
- Instead of storing \( B_k \), the limited-memory BFGS (L-BFGS) method stores the previous \( l \) differences \( s_k \) and \( y_k \).
- We can solve a linear system involving these updates when \( B_0 \) is diagonal in \( O(dl) \) [Nocedal, 1980].
Limited-Memory BFGS (L-BFGS)

Remarks

- using the right initialization $B_0$ is crucial.
- the calibration of the line-search is also an art.
Limited-Memory BFGS (L-BFGS)

Remarks

- using the right initialization $B_0$ is crucial.
- the calibration of the line-search is also an art.

Pros

- a big practical success of smooth optimization.
Limited-Memory BFGS (L-BFGS)

Remarks
- using the right initialization $B_0$ is crucial.
- the calibration of the line-search is also an art.

Pros
- a big practical success of smooth optimization.

Cons
- worst-case convergence rates for strongly-convex functions are linear, but no better than the gradient descent method.
- proximal variants typically requires solving many times
  $\min_{x \in \mathbb{R}^d} \frac{1}{2}(x - z)B_k(z - z) + \psi(x)$.
- no guarantee of approximating the Hessian.
Main recipe

- L-BFGS applied to the smoothed objective $F$ with inexact gradients [see Friedlander and Schmidt, 2012].
- Inexact gradients are obtained by solving sub-problems using a first-order optimization method $\mathcal{M}$;
- Ideally, $\mathcal{M}$ is able to adapt to the problem structure (finite sum, composite regularization).
- Replace L-BFGS steps by proximal point steps if no sufficient decrease is estimated $\Rightarrow$ no line search on $F$;
Obtaining inexact gradients

**Algorithm** Procedure ApproxGradient

**input** Current point $x$ in $\mathbb{R}^d$; smoothing parameter $\kappa > 0$.

1. Compute the approximate mapping using an optimization method $\mathcal{M}$:

   $$ z \approx \arg\min_{w \in \mathbb{R}^d} \left\{ h(w) \triangleq f(w) + \frac{\kappa}{2} \|w - x\|^2 \right\}, $$

2. Estimate the gradient $\nabla F(x)$

   $$ g = \kappa (x - z). $$

**output** approximate gradient estimate $g$, objective value $F_a \triangleq h(z)$, proximal mapping mapping $z$. 

Julien Mairal

QuickeNing
Algorithm QuickeNing

**input** $x_0$ in $\mathbb{R}^p$; number of iterations $K$; $\kappa > 0$; minimization algorithm $\mathcal{M}$.

1. Initialization: $(g_0, F_0, z_0) = \text{ApproxGradient}(x_0, \mathcal{M})$; $B_0 = \kappa I$.
2. for $k = 0, \ldots, K - 1$ do
3.   Perform the Quasi-Newton step
4.     \[ x_{\text{test}} = x_k - B_k^{-1}g_k \]
5.     \[ (g_{\text{test}}, F_{\text{test}}, z_{\text{test}}) = \text{ApproxGradient}(x_{\text{test}}, \mathcal{M}). \]
6.     if $F_{\text{test}} \leq F_k - \frac{1}{2\kappa} \|g_k\|^2$, then
7.       $(x_{k+1}, g_{k+1}, F_{k+1}, z_{k+1}) = (x_{\text{test}}, g_{\text{test}}, F_{\text{test}}, z_{\text{test}})$.
8.     else
9.       Update the current iterate with the last proximal mapping:
10.      \[ x_{k+1} = z_k = x_k - \frac{1}{\kappa}g_k \]
11.      \[ (g_{k+1}, F_{k+1}, z_{k+1}) = \text{ApproxGradient}(x_{k+1}, \mathcal{M}). \]
12. end if
13. update $B_{k+1} = \text{L-BFGS}(B_k, x_{k+1} - x_k, g_{k+1} - g_k)$.
14. end for

**output** last proximal mapping $z_K$ (solution).
Algorithm QuickeNing

**input** $x_0$ in $\mathbb{R}^p$; number of iterations $K$; $\kappa > 0$; minimization algorithm $\mathcal{M}$.

1: Initialization: $(g_0, F_0, z_0) = \text{ApproxGradient}(x_0, \mathcal{M})$; $B_0 = \kappa I$.
2: for $k = 0, \ldots, K - 1$ do
3: Perform the Quasi-Newton step
   
   $$x_{\text{test}} = x_k - B_k^{-1}g_k$$
   
   $$(g_{\text{test}}, F_{\text{test}}, z_{\text{test}}) = \text{ApproxGradient}(x_{\text{test}}, \mathcal{M}).$$

4: if $F_{\text{test}} \leq F_{k-1} - \frac{2}{\kappa} \|g_k\|^2$ then
5: $(x_{k+1}, g_{k+1}, F_{k+1}, z_{k+1}) = (x_{\text{test}}, g_{\text{test}}, F_{\text{test}}, z_{\text{test}})$.
6: else
7: Update the current iterate with the last proximal mapping:
   
   $$x_{k+1} = x_k - \frac{1}{\kappa}g_k$$
   
   $$(g_{k+1}, F_{k+1}, z_{k+1}) = \text{ApproxGradient}(x_{k+1}, \mathcal{M}).$$
8: end if
9: update $B_{k+1} = \text{L-BFGS}(B_k, x_{k+1} - x_k, g_{k+1} - g_k)$.
10: end for

**output** last proximal mapping $z_K$ (solution).

The main characters:

- the sequence $(x_k)_{k \geq 0}$ that minimizes $F$;
- the sequence $(z_k)_{k \geq 0}$ produced by $\mathcal{M}$ that minimizes $f$;
- the gradient approximations $g_k \approx \nabla F(x_k)$;
- the function value approximations $F_k \approx \nabla F(x_k)$;
- an L-BFGS update with inexact gradients;
- an approximate sufficient descent condition.
Requirements on $\mathcal{M}$ and restarts

Method $\mathcal{M}$

- Say a sub-problem consists of minimizing $h$; we want $\mathcal{M}$ to produce a sequence of iterates $(w_t)_{t \geq 0}$ with linear convergence rate

$$h(w_t) - h^* \leq C_\mathcal{M}(1 - \tau_\mathcal{M})^t(h(w_0) - h^*).$$

Restarts

- When $f$ is smooth, we initialize $w_0 = x$ when solving

$$\min_{w \in \mathbb{R}^d} \left\{ f(w) + \frac{\kappa}{2} \|w - x\|^2 \right\}.$$

- When $f = f_0 + \psi$ is composite, we use the initialization

$$w_0 = \arg \min_{w \in \mathbb{R}^d} \left\{ f_0(x) + \langle \nabla f_0(x), w - x \rangle + \frac{L + \kappa}{2} \|w - x\|^2 + \psi(w) \right\}.$$
When do we stop the method $\mathcal{M}$?

Three strategies

(a) use a **pre-defined sequence** $(\epsilon_k)_{k \geq 0}$ and stop the optimization method $\mathcal{M}$ when the approximate proximal mapping is $\epsilon_k$-accurate.

(b) define a **stopping criterion** that depends on quantities that are available at iteration $k$.

(c) use a **pre-defined budget** $T_{\mathcal{M}}$ of iterations of the method $\mathcal{M}$ for solving each sub-problem.
When do we stop the method $\mathcal{M}$?

Three strategies

(a) use a **pre-defined sequence** $(\epsilon_k)_{k \geq 0}$ and stop the optimization method $\mathcal{M}$ when the approximate proximal mapping is $\epsilon_k$-accurate.

(b) define a **stopping criterion** that depends on quantities that are available at iteration $k$.

(c) use a **pre-defined budget** $T_\mathcal{M}$ of iterations of the method $\mathcal{M}$ for solving each sub-problem.

Remarks

- (a) is the **less practical** strategy.
- (b) is **simpler to use and conservative** (compatible with theory).
- (c) requires $T_\mathcal{M}$ to be large enough in theory. The **aggressive** strategy $T_\mathcal{M} = n$ for an incremental method is **extremely simple to use and effective in practice**.
When do we stop the method $\mathcal{M}$?

Three strategies in details for $\mu$-strongly convex $f$

(a) use a **pre-defined sequence** $(\epsilon_k)_{k \geq 0}$ and stop the optimization method $\mathcal{M}$ when the approximate proximal mapping is $\epsilon_k$-accurate.

$$
\epsilon_k = \frac{1}{2} C (1 - \rho)^{k+1} \quad \text{with} \quad C \geq f(x_0) - f^* \quad \text{and} \quad \rho = \frac{\mu}{4(\mu + \kappa)}.
$$

(b) For minimizing $h(w) = f(w) + (\kappa/2)\|w - x\|^2$, stop when

$$
h(w_t) - h^* \leq \frac{\kappa}{36} \|w_t - x\|^2.
$$

(c) use a **pre-defined budget** $T_{\mathcal{M}}$ of iterations of the method $\mathcal{M}$ for solving each sub-problem with

$$
T_{\mathcal{M}} = \frac{1}{\tau_{\mathcal{M}}} \log \left( 19 C_{\mathcal{M}} \frac{L + \kappa}{\kappa} \right). \quad \text{(be more aggressive in practice)}
$$
Remarks and global complexity

Composite objectives and sparsity
Consider a composite problem with a sparse solution (e.g., $\psi = \ell_1$). The method produces two sequences $(x_k)_{k \geq 0}$ and $(z_k)_{k \geq 0}$;

- $F(x_k) \to F^*$, minimizes the \textit{smoothed objective} $\Rightarrow$ no sparsity;
- $f(z_k) \to f^*$, minimizes the \textit{true objective} $\Rightarrow$ the iterates may be sparse if $\mathcal{M}$ handles composite optimization problems;

Global complexity
The number of iterations of $\mathcal{M}$ to guarantee $f(z_k) - f^* \leq \varepsilon$ is at most

- $\tilde{O}(\frac{\mu + \kappa}{\tau \mathcal{M} \mu} \log(1/\varepsilon))$ for $\mu$-strongly convex problems.
- $\tilde{O}(\frac{\kappa R^2}{\tau \mathcal{M} \varepsilon})$ for convex problems.
Global Complexity and choice of $\kappa$

Example for gradient descent

With the right step-size, we have $\tau_M = (\mu + \kappa)/(L + \kappa)$ and the complexity for $\mu > 0$ becomes

$$\tilde{O}\left(\frac{L + \kappa}{\mu} \log(1/\varepsilon)\right).$$

Example for SVRG for minimizing the sum of $n$ functions

$\tau_M = \min(1/n, (\mu + \kappa)/(L + \kappa))$ and the complexity for $\mu > 0$ is

$$\tilde{O}\left(\max\left(\frac{\mu + \kappa}{\mu} n, \frac{L + \kappa}{\mu}\right) \log(1/\varepsilon)\right).$$
Global Complexity and choice of \( \kappa \)

**Example for gradient descent**

With the right step-size, we have \( \tau_M = (\mu + \kappa)/(L + \kappa) \) and the complexity for \( \mu > 0 \) becomes

\[
\tilde{O} \left( \frac{L + \kappa}{\mu} \log\left(\frac{1}{\varepsilon}\right) \right).
\]

**Example for SVRG for minimizing the sum of \( n \) functions**

\( \tau_M = \min(1/n, (\mu + \kappa)/(L + \kappa)) \) and the complexity for \( \mu > 0 \) is

\[
\tilde{O} \left( \max \left( \frac{\mu + \kappa}{\mu} n, \frac{L + \kappa}{\mu} \right) \log\left(\frac{1}{\varepsilon}\right) \right).
\]

QuickeNing does not provide any theoretical acceleration, but it does not degrade significantly the worst-case performance of \( \mathcal{M} \) (unlike L-BFGS vs gradient descent).
Global Complexity and choice of $\kappa$

Example for gradient descent

With the right step-size, we have $\tau_M = (\mu + \kappa)/(L + \kappa)$ and the complexity for $\mu > 0$ becomes

$$\tilde{O}\left(\frac{L + \kappa}{\mu} \log(1/\varepsilon)\right).$$

Example for SVRG for minimizing the sum of $n$ functions

$\tau_M = \min(1/n, (\mu + \kappa)/(L + \kappa))$ and the complexity for $\mu > 0$ is

$$\tilde{O}\left(\max\left(\frac{\mu + \kappa}{\mu} n, \frac{L + \kappa}{\mu}\right) \log(1/\varepsilon)\right).$$

Then, how to choose $\kappa$?

(i) assume that L-BFGS steps do as well as Nesterov

(ii) choose $\kappa$ as in Catalyst.
Experiments: formulations

- $\ell_2$-regularized Logistic Regression:
  \[
  \min_{x \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \log \left( 1 + \exp \left( -b_i a_i^T x \right) \right) + \frac{\mu}{2} \|x\|^2,
  \]

- $\ell_1$-regularized Linear Regression (LASSO):
  \[
  \min_{x \in \mathbb{R}^d} \frac{1}{2n} \sum_{i=1}^{n} (b_i - a_i^T x)^2 + \lambda \|x\|_1,
  \]

- $\ell_1 - \ell_2$-regularized Linear Regression (Elastic-Net):
  \[
  \min_{x \in \mathbb{R}^d} \frac{1}{2n} \sum_{i=1}^{n} (b_i - a_i^T x)^2 + \lambda \|x\|_1 + \frac{\mu}{2} \|x\|^2,
  \]
Experiments: Datasets

We consider four standard machine learning datasets with different characteristics in terms of size and dimension:

<table>
<thead>
<tr>
<th>name</th>
<th>covtype</th>
<th>alpha</th>
<th>real-sim</th>
<th>rcv1</th>
</tr>
</thead>
<tbody>
<tr>
<td>(n)</td>
<td>581 012</td>
<td>250 000</td>
<td>72 309</td>
<td>781 265</td>
</tr>
<tr>
<td>(d)</td>
<td>54</td>
<td>500</td>
<td>20 958</td>
<td>47 152</td>
</tr>
</tbody>
</table>

- we simulate the ill-conditioned regime \(\mu = 1/(100n)\);
- \(\lambda\) for the Lasso leads to about 10% non-zero coefficients.
Experiments: QuickeNing-SVRG

We consider the methods

- **SVRG**: the Prox-SVRG algorithm of Xiao and Zhang [2014].
- **Catalyst-SVRG**: Catalyst applied to SVRG;
- **L-BFGS** (for smooth objectives): Mark Schmidt’s implementation.
- **QuickeNing-SVRG1**: QuickeNing with aggressive strategy (c): one pass over the data in the inner loop.
- **QuickeNing-SVRG2**: strategy (b), compatible with theory.

We produce 12 figures (3 formulations, 4 datasets).
Experiments: QuickeNing-SVRG (log scale)

- covtype, logistic, $\mu = 1/100 \ n$
- covtype, lasso, $\lambda = 10 / n$
- rcv1, logistic, $\mu = 1/100 \ n$
- rcv1, lasso, $\lambda = 10 / n$

- QuickeNing-SVRG1 $\geq$ SVRG, QuickeNing-SVRG2;
- QuickeNing-SVRG2 $\geq$ SVRG;
- QuickeNing-SVRG1 $\geq$ Catalyst-SVRG in 10/12 cases.
Experiments: QuickeNing-ISTA

We consider the methods

- **ISTA**: the proximal gradient descent method with line search.
- **FISTA**: the accelerated ISTA of Beck and Teboulle [2009].
- **L-BFGS** (for smooth objectives): Mark Schmidt’s implementation.
- **QuickeNing-ISTA1**: QuickeNing with aggressive strategy (c): one pass over the data in the inner loop.
- **QuickeNing-ISTA2**: strategy (b), compatible with theory.
Experiments: QuickeNing-ISTA (log scale)

L-BFGS (for smooth $f$) is slightly better than QuickeNing-ISTA1;
QuickeNing-ISTA $\geq$ or $\gg$ FISTA in 11/12 cases.
QuickeNing-ISTA1 $\geq$ QuickeNing-ISTA2.
Experiments: Influence of $\kappa$

- $\kappa_0$ is the parameter (same as in Catalyst) used in all experiments;
- QuickeNing slows down when using $\kappa > \kappa_0$;
- here, for SVRG, QuickeNing is robust to small values of $\kappa$!
Experiments: Influence of $l$

- $l = 100$ in all previous experiments;
- $l = 5$ seems to be a reasonable choice in many cases, especially for sparse problems.
Conclusions and perspectives

- QuickeNing has been a safe heuristic so far.
- It may be the first L-BFGS algorithm for composite objectives with reasonable known complexity for solving the sub-problems.
- We also have a variant for dual approaches.
- the gap between theory and practice is significant.

Perspectives

- QuickeNing-BCD, QuickeNing-SAG,SAGA,SDCA...
- Other types of smoothing techniques?
Outer-loop convergence analysis

Lemma: approximate descent property

\[ F(x_{k+1}) \leq f(z_k) \leq F(x_k) - \frac{1}{4\kappa} \|\nabla F(x_k)\|_2^2 + 2\varepsilon_k. \]

Then, \( \varepsilon_k \) should be smaller than \( \frac{1}{4\kappa} \|\nabla F(x_k)\|_2^2 \), and indeed
Outer-loop convergence analysis

**Lemma:** approximate descent property

\[ F(x_{k+1}) \leq f(z_k) \leq F(x_k) - \frac{1}{4\kappa} \| \nabla F(x_k) \|_2^2 + 2\varepsilon_k. \]

Then, \( \varepsilon_k \) should be smaller than \( \frac{1}{4\kappa} \| \nabla F(x_k) \|_2^2 \), and indeed

**Proposition:** convergence with impractical \( \varepsilon_k \) and \( \mu > 0 \)

If \( \varepsilon_k \leq \frac{1}{16\kappa} \| \nabla F(x_k) \|_2^2 \), define \( \rho = \frac{\mu}{4(\mu+\kappa)} \), then

\[ F(x_{k+1}) - F^* \leq f(z_k) - f^* \leq (1 - \rho)^{k+1} (f(x_0) - f^*). \]

Unfortunately, \( \| \nabla F(x_k) \| \) is unknown.
Outer-loop convergence analysis

Lemma: approximate descent property

\[ F(x_{k+1}) \leq f(z_k) \leq F(x_k) - \frac{1}{4\kappa} \| \nabla F(x_k) \|_2^2 + 2\varepsilon_k. \]

Then, \( \varepsilon_k \) should be smaller than \( \frac{1}{4\kappa} \| \nabla F(x_k) \|_2^2 \), and indeed

Proposition: convergence with impractical \( \varepsilon_k \) and \( \mu > 0 \)

If \( \varepsilon_k \leq \frac{1}{16\kappa} \| \nabla F(x_k) \|_2^2 \), define \( \rho = \frac{\mu}{4(\mu + \kappa)} \), then

\[ F(x_{k+1}) - F^* \leq f(z_k) - f^* \leq (1 - \rho)^{k+1} (f(x_0) - f^*). \]

Unfortunately, \( \| \nabla F(x_k) \| \) is unknown.

Lemma: convergence with adaptive \( \varepsilon_k \) and \( \mu > 0 \)

If \( \varepsilon_k \leq \frac{1}{36\kappa} \| g_k \|_2^2 \), then \( \varepsilon_k \leq \frac{1}{16} \| \nabla F(x_k) \|_2^2 \).

This is strategy (b). \( g_k \) is known and easy to compute.
Inner-loop complexity analysis

Restart for $L$-smooth functions

For minimizing $h$, initialize the method $\mathcal{M}$ with $w_0 = x$. Then,

$$h(w_0) - h^* \leq \frac{L + \kappa}{2\kappa^2} \| \nabla F(x) \|^2. \tag{1}$$

Proof.

We have the optimality condition $\nabla f(w^*) + \kappa (w^* - x) = 0$. As a result,

$$h(w_0) - h^* = f(x) - \left( f(w^*) + \frac{\kappa}{2} \| w^* - x \|^2 \right)$$

$$\leq f(w^*) + \langle \nabla f(w^*), x - w^* \rangle + \frac{L}{2} \| x - w^* \|^2 - \left( f(w^*) + \frac{\kappa}{2} \| w^* - x \|^2 \right)$$

$$= \frac{L + \kappa}{2} \| w^* - x \|^2 = \frac{L + \kappa}{2\kappa^2} \| \nabla F(x) \|^2.$$
References I


References II


References V


References VI

