Optimized first-order minimization methods

Donghwan Kim & Jeffrey A. Fessler

EECS Dept., BME Dept., Dept. of Radiology
University of Michigan

web.eecs.umich.edu/~fessler

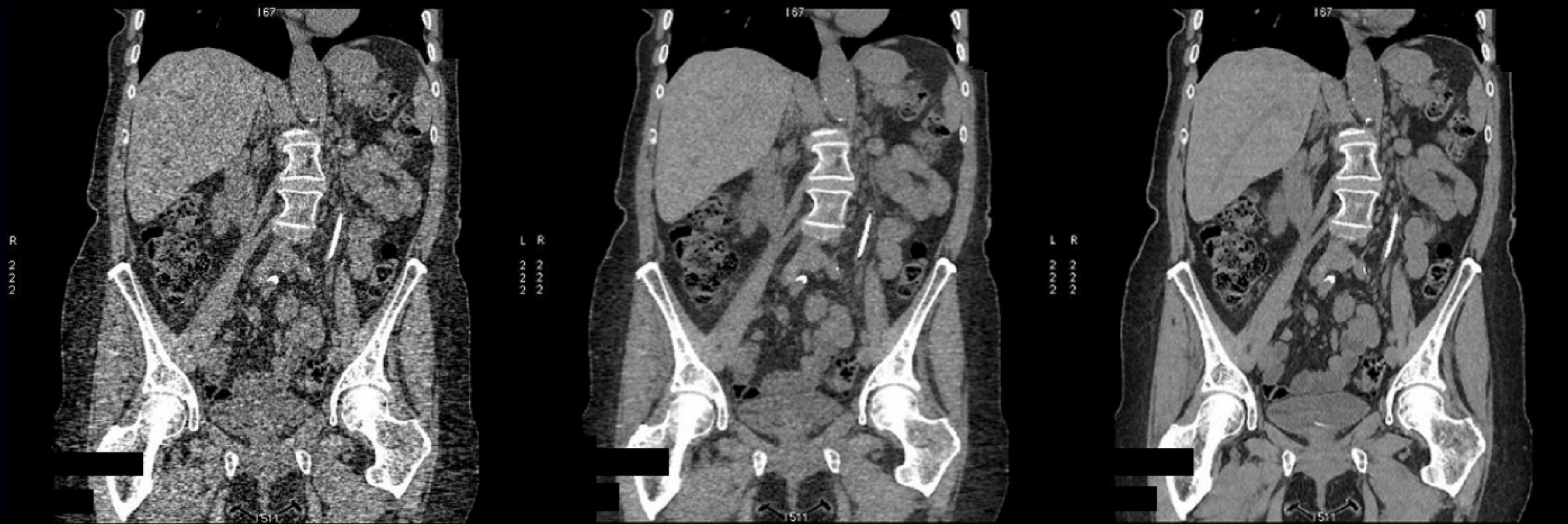
UM CSP Seminar

17 Apr 2014

Disclosure

- Research support from GE Healthcare
- Research support to GE Global Research
- Supported in part by NIH grants R01 HL-098686 and P01 CA-87634
- Equipment support from Intel Corporation

Low-dose X-ray CT image reconstruction



Thin-slice FBP	ASIR	Statistical
Seconds	A bit longer	Much longer

Image reconstruction as an optimization problem:

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x} \succeq \boldsymbol{0}} \frac{1}{2} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_W^2 + R(\boldsymbol{x})$$

(Same sinogram, so all at same dose)

Outline

- Motivation (done)
- Problem definition
- Existing algorithms
  - Gradient descent
  - Nesterov's "optimal" first-order methods
  - General first-order methods
- Optimizing first-order minimization methods
- Drori & Teboulle's numerical bounds
- Donghwan Kim's analytically optimized ("more optimal") first-order methods
- Example (logistic regression for machine learning)
- Summary / Future work

Problem setting

Optimization problem setting

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} f(\boldsymbol{x})$$

- Unconstrained
- Large-scale (Hessian too big to store)
  - image reconstruction
  - big-data / machine learning
  - ...
- Cost function assumptions (throughout)
  - $f : \mathbb{R}^M \mapsto \mathbb{R}$
  - convex (need not be strictly convex)
  - non-empty set of global minimizers:

$$\hat{\boldsymbol{x}} \in \mathscr{X}^* = \left\{ \boldsymbol{x}_\star \in \mathbb{R}^M : f(\boldsymbol{x}_\star) \leq f(\boldsymbol{x}), \ \forall \boldsymbol{x} \in \mathbb{R}^M \right\}$$

  - smooth (differentiable with L-Lipshitz gradient)

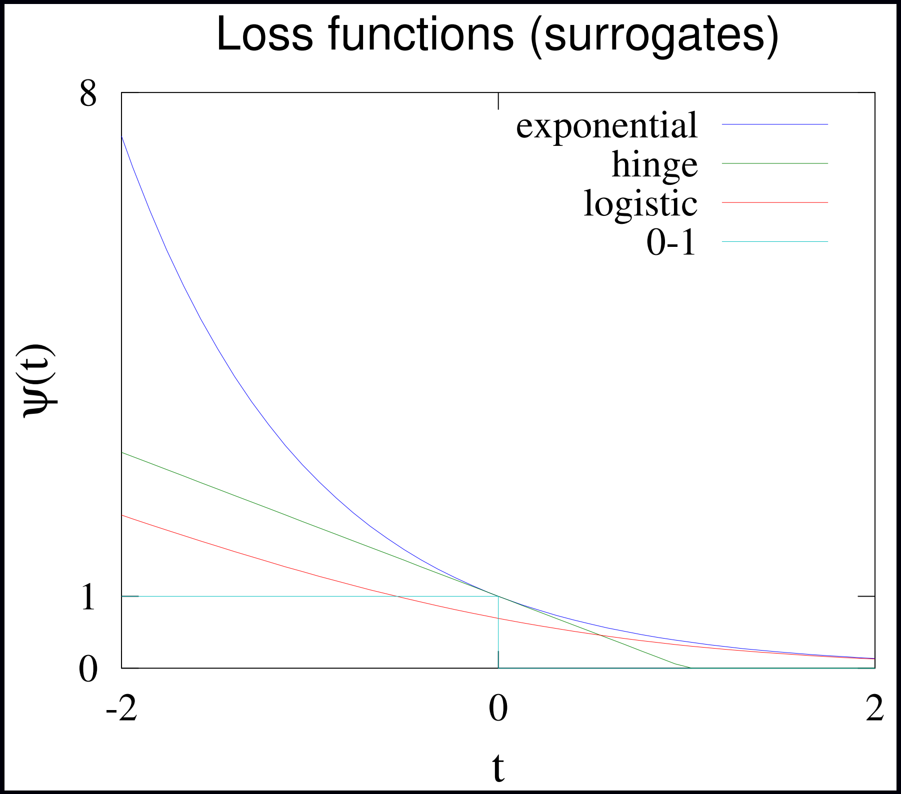
$$\|\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{z})\|_2 \leq L \|\boldsymbol{x} - \boldsymbol{z}\|_2, \quad \forall \boldsymbol{x}, \boldsymbol{z} \in \mathbb{R}^M$$

Example: Machine learning

To learn weights $\boldsymbol{x}$ of binary classifier given feature vectors $\{\boldsymbol{v}_i\}$ and labels $\{y_i\}$:

$$f(\boldsymbol{x}) = \sum_i \psi(y_i \langle \boldsymbol{x}, \boldsymbol{v}_i \rangle),$$

where $y_i = \pm 1$.



loss functions $\psi(t)$
0-1: $1_{\{t \leq 0\}}$
exponential: $\exp(-t)$
logistic: $\log(1+\exp(-t))$
hinge: $\max\{0, 1-t\}$

Which of these fit our conditions?

Algorithms

Gradient descent

iteration with step size $1/L$ ensures monotonic descent of f:

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n - \frac{1}{L}\nabla f(\boldsymbol{x}_n)$$

stacking:

$$\begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_{N-1} \\ \boldsymbol{x}_N \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_0 \\ \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_{N-2} \\ \boldsymbol{x}_{N-1} \end{bmatrix} - \frac{1}{L}\begin{bmatrix} \nabla f(\boldsymbol{x}_0) \\ \nabla f(\boldsymbol{x}_1) \\ \vdots \\ \nabla f(\boldsymbol{x}_{N-2}) \\ \nabla f(\boldsymbol{x}_{N-1}) \end{bmatrix}$$

i.e.:

$$\begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_{N-1} \\ \boldsymbol{x}_N \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_0 \\ \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_{N-2} \\ \boldsymbol{x}_{N-1} \end{bmatrix} - \frac{1}{L}\left(\underbrace{\begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ \vdots & & \ddots & & \\ 0 & \dots & 0 & 1 & 0 \\ 0 & \dots & 0 & 0 & 1 \end{bmatrix}}_{H_{\mathrm{GD}}} \otimes \boldsymbol{I} \right) \begin{bmatrix} \nabla f(\boldsymbol{x}_0) \\ \nabla f(\boldsymbol{x}_1) \\ \vdots \\ \nabla f(\boldsymbol{x}_{N-2}) \\ \nabla f(\boldsymbol{x}_{N-1}) \end{bmatrix}$$

Note: $N \times N$ coefficient matrix H_{GD} is diagonal (a special case of lower triangular).

Gradient descent convergence rate

Classic $O(1/n)$ convergence rate of cost function descent:

$$\underbrace{f(\boldsymbol{x}_n) - f(\boldsymbol{x}_\star)}_{\text{inaccuracy}} \leq \frac{L\,\|\boldsymbol{x}_0 - \boldsymbol{x}_\star\|_2^2}{2n}.$$

Drori & Teboulle (2013) derive tightest inaccuracy bound:

$$f(\boldsymbol{x}_n) - f(\boldsymbol{x}_\star) \leq \frac{L\,\|\boldsymbol{x}_0 - \boldsymbol{x}_\star\|_2^2}{4n+2}.$$

They construct a Huber-like function f for which GD achieves that bound.
Case closed for GD.

$O(1/n)$ rate is undesirably slow.

Heavy ball method

iteration (Polyak, 1987):

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n - \frac{\alpha}{L}\nabla f(\boldsymbol{x}_n) + \underbrace{\beta\,(\boldsymbol{x}_n - \boldsymbol{x}_{n-1})}_{\text{momentum!}} \qquad \text{(for implementation)}$$

$$= \boldsymbol{x}_n - \frac{1}{L}\sum_{k=0}^{n} \underbrace{\alpha\beta^{n-k}}_{\text{coefficients}} \nabla f(\boldsymbol{x}_i) \qquad \text{(for analysis)}$$

stacking:

$$\begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_{N-1} \\ \boldsymbol{x}_N \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_0 \\ \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_{N-2} \\ \boldsymbol{x}_{N-1} \end{bmatrix} - \frac{1}{L}\left(\underbrace{\begin{bmatrix} \alpha & 0 & 0 & \dots & 0 \\ \alpha\beta & \alpha & 0 & \dots & 0 \\ & & \ddots & & \\ \alpha\beta^{N-2} & \dots & \alpha\beta & \alpha & 0 \\ \alpha\beta^{N-1} & \dots & \alpha\beta^2 & \alpha\beta & \alpha \end{bmatrix}}_{H_{\mathrm{HB}}} \otimes \boldsymbol{I} \right) \begin{bmatrix} \nabla f(\boldsymbol{x}_0) \\ \nabla f(\boldsymbol{x}_1) \\ \vdots \\ \nabla f(\boldsymbol{x}_{N-2}) \\ \nabla f(\boldsymbol{x}_{N-1}) \end{bmatrix}$$

Here, $N \times N$ coefficient matrix H_{HB} is lower triangular.

- How to choose α and β?
- How to optimize $N \times N$ coefficient matrix H more generally?

General first-order method class

General "first-order" (FO) iteration:

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n - \frac{1}{L} \sum_{k=0}^{n} h_{n+1,k} \nabla f(\boldsymbol{x}_k)$$

stacking:

$$\begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_{N-1} \\ \boldsymbol{x}_N \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_0 \\ \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_{N-2} \\ \boldsymbol{x}_{N-1} \end{bmatrix} - \frac{1}{L} \left(\underbrace{\begin{bmatrix} h_{1,0} & 0 & 0 & \dots & 0 \\ h_{2,0} & h_{2,1} & 0 & \dots & 0 \\ & & \ddots & & \\ h_{N,0} & h_{N,1} & \dots & h_{N,N-2} & h_{N,N-1} \end{bmatrix}}_{H_{\mathrm{FO}}} \otimes \boldsymbol{I} \right) \begin{bmatrix} \nabla f(\boldsymbol{x}_0) \\ \nabla f(\boldsymbol{x}_1) \\ \vdots \\ \nabla f(\boldsymbol{x}_{N-2}) \\ \nabla f(\boldsymbol{x}_{N-1}) \end{bmatrix}$$

Primary goals:

- Analyze convergence rate of FO for any given H
- Optimize $N \times N$ lower-triangular ("causal") step-size coefficient matrix H.
  - fast convergence
  - efficient recursive implementation
  - universal (design *prior* to iterating)

Not: Barzilai-Borwein gradient method

Barzilai & Borwein, 1988

$$\boldsymbol{g}^{(n)} \triangleq \nabla f(\boldsymbol{x}_n)$$

$$\alpha_n = \frac{\|\boldsymbol{x}_n - \boldsymbol{x}_{n-1}\|^2}{\langle \boldsymbol{x}_n - \boldsymbol{x}_{n-1}, \boldsymbol{g}^{(n)} - \boldsymbol{g}^{(n-1)} \rangle}$$

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n - \alpha_n \nabla f(\boldsymbol{x}_n).$$

Not in "first-order" class FO.
Neither are methods like
- ◦ steepest descent (with line search),
- ◦ conjugate gradient,
- ◦ quasi-Newton ...

Nesterov's fast gradient method (FGM1)

Nesterov (1983) iteration: Initialize: $t_0 = 1$, $\boldsymbol{z}_0 = \boldsymbol{x}_0$

$$\begin{aligned}
\boldsymbol{z}_{n+1} &= \boldsymbol{x}_n - \frac{1}{L}\nabla f(\boldsymbol{x}_n) && \text{(usual GD update)} \\
t_{n+1} &= \frac{1}{2}\left(1 + \sqrt{1 + 4t_n^2}\right) && \text{(magic momentum factors)} \\
\boldsymbol{x}_{n+1} &= \boldsymbol{z}_{n+1} + \frac{t_n - 1}{t_{n+1}}(\boldsymbol{z}_{n+1} - \boldsymbol{z}_n) && \text{(update with momentum)} .
\end{aligned}$$

Reverts to GD if $t_n = 1, \forall n$.

FGM1 is in class FO: $$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n - \frac{1}{L}\sum_{k=0}^{n} h_{n+1,k} \nabla f(\boldsymbol{x}_k)$$

$$h_{n+1,k} = \begin{cases} \frac{t_n - 1}{t_{n+1}} h_{n,k}, & k = 0, \ldots, n-2 \\ \frac{t_n - 1}{t_{n+1}} (h_{n,n-1} - 1), & k = n-1 \\ 1 + \frac{t_n - 1}{t_{n+1}}, & k = n. \end{cases}$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1.25 & 0 & 0 & 0 & 0 \\ 0 & 0.10 & 1.40 & 0 & 0 & 0 \\ 0 & 0.05 & 0.20 & 1.50 & 0 & 0 \\ 0 & 0.03 & 0.11 & 0.29 & 1.57 & 0 \\ 0 & 0.02 & 0.07 & 0.18 & 0.36 & 1.62 \end{bmatrix}$$

Nesterov FGM1 optimal convergence rate

Shown by Nesterov to be $O(1/n^2)$ for "auxiliary" sequence:

$$f(\boldsymbol{z}_n) - f(\boldsymbol{x}_\star) \le \frac{2L \|\boldsymbol{x}_0 - \boldsymbol{x}_\star\|_2^2}{(n+1)^2}.$$

Nesterov constructed a function f such that FGM1 achieves

$$\frac{\frac{3}{32} L \|\boldsymbol{x}_0 - \boldsymbol{x}_\star\|_2^2}{(n+1)^2} \le f(\boldsymbol{x}_n) - f(\boldsymbol{x}_\star).$$

Thus $O(1/n^2)$ rate of FGM1 is optimal.

New results (Donghwan Kim, 2014):

- Bound on convergence rate of primary sequence $\{\boldsymbol{x}_n\}$:

$$f(\boldsymbol{x}_n) - f(\boldsymbol{x}_\star) \le \frac{2L \|\boldsymbol{x}_0 - \boldsymbol{x}_\star\|_2^2}{(n+2)^2}.$$

- Verifies (numerically inspired) conjecture of Drori & Teboulle (2013).

Overview

General first-order (FO) iteration:

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n - \frac{1}{L} \sum_{k=0}^{n} h_{n+1,k} \nabla f(\boldsymbol{x}_k)$$

- Analyze (*i.e.*, bound) convergence rate as a function of
  - number of iterations N
  - Lipschitz constant L
  - step-size coefficients $H = \{h_{n+1,k}\}$
  - Distance to a solution: $R = \|\boldsymbol{x}_0 - \boldsymbol{x}_\star\|$
- Optimize H by minimizing the bound

Ideal "universal" bound for first-order methods

For given

- number of iterations N
- Lipschitz constant L
- step-size coefficients $H = \{h_{n+1,k}\}$
- distance to a solution: $R = \|\boldsymbol{x}_0 - \boldsymbol{x}_\star\|$

bound the worst-case convergence rate of FO algorithm:

$$B_1(H,R,L,N) \triangleq \max_{f \in \mathscr{F}_L} \max_{\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathbb{R}^M} \max_{\substack{\boldsymbol{x}_\star \in \mathscr{X}^*(f) \\ \|\boldsymbol{x}_0 - \boldsymbol{x}_\star\| \leq R}} f(\boldsymbol{x}_N) - f(\boldsymbol{x}_\star)$$

such that $$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n - \frac{1}{L} \sum_{k=0}^{n} h_{n+1,k} \nabla f(\boldsymbol{x}_k), \quad n = 0, \ldots, N-1.$$

Clearly for any FO method:

$$f(\boldsymbol{x}_N) - f(\boldsymbol{x}_\star) \leq B_1(H,R,L,N)$$

Towards practical bounds for first-order methods

For convex functions with L-Lipschitz gradients

$$\frac{1}{2L}\|\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{z})\|^2 \le f(\boldsymbol{x}) - f(\boldsymbol{z}) - \langle \nabla f(\boldsymbol{z}), \boldsymbol{x} - \boldsymbol{z}\rangle, \quad \forall \boldsymbol{x}, \boldsymbol{z} \in \mathbb{R}^M.$$

Drori & Teboulle (2013) use this inequality to propose a “more tractable” bound:

$$B_2(H,R,L,N) \triangleq \max_{\boldsymbol{g}_0,\dots,\boldsymbol{g}_N \in \mathbb{R}^M} \max_{\delta_0,\dots,\delta_N \in \mathbb{R}} \max_{\boldsymbol{x}_0,\boldsymbol{x}_1,\dots,\boldsymbol{x}_N \in \mathbb{R}^M} \max_{\boldsymbol{x}_\star : \|\boldsymbol{x}_0 - \boldsymbol{x}_\star\| \le R} LR\delta_N^2$$

$$\text{such that} \quad \boldsymbol{x}_{n+1} = \boldsymbol{x}_n - \frac{1}{L}\sum_{k=0}^{n} h_{n+1,k} R\, \boldsymbol{g}_k, \quad n = 0,\dots,N-1,$$

$$\frac{1}{2}\|\boldsymbol{g}_i - \boldsymbol{g}_j\|^2 \le \delta_i - \delta_j - \frac{1}{R}\langle \boldsymbol{g}_j, \boldsymbol{x}_i - \boldsymbol{x}_j\rangle, \quad i,j = 0,\dots,N,*$$

where $\boldsymbol{g}_n = \frac{1}{LR}\nabla f(\boldsymbol{x}_n)$ and $\delta_n = \frac{1}{LR}(f(\boldsymbol{x}_n) - f(\boldsymbol{x}_\star))$.

For any FO method:

$$f(\boldsymbol{x}_N) - f(\boldsymbol{x}_\star) \le B_1(H,R,L,N) \le B_2(H,R,L,N)$$

However, even B_2 is as of yet unsolved.

Numerical bounds for first-order methods

Drori & Teboulle (2013) further relax the bound leading eventually to a still simpler optimization problem (with no known closed-form solution):

$$f(\boldsymbol{x}_N) - f(\boldsymbol{x}_\star) \leq B_1(H,R,L,N) \leq B_2(H,R,L,N) \leq B_3(H,R,L,N).$$

For given step-size coefficients H, and given number of iterations N, they use a semi-definite program (SDP) to compute B_3 numerically.

The find numerically that for the FGM1 choice of H, the convergence bound B_3 is slightly tighter than $\frac{2L\|\boldsymbol{x}_0 - \boldsymbol{x}_\star\|_2^2}{(N+1)^2}$.

Optimizing step-size coefficients numerically

Drori & Teboulle (2013) also compute numerically the minimizer over H of their relaxed bound for given N using a semi-definite program (SDP):

$$H^* = \arg\min_H B_3(H, R, L, N).$$

Numerical solution for H^* for $N = 5$ iterations: [Fig. from Drori & Teboulle (2013)]

0. Input: $f \in C_L^{1,1}(\mathbb{R}^d)$, $x_0 \in \mathbb{R}^d$,
1. $x_1 = x_0 - \frac{1.6180}{L} f'(x_0)$,
2. $x_2 = x_1 - \frac{0.1741}{L} f'(x_0) - \frac{2.0194}{L} f'(x_1)$,
3. $x_3 = x_2 - \frac{0.0756}{L} f'(x_0) - \frac{0.4425}{L} f'(x_1) - \frac{2.2317}{L} f'(x_2)$,
4. $x_4 = x_3 - \frac{0.0401}{L} f'(x_0) - \frac{0.2350}{L} f'(x_1) - \frac{0.6541}{L} f'(x_2) - \frac{2.3656}{L} f'(x_3)$,
5. $x_5 = x_4 - \frac{0.0178}{L} f'(x_0) - \frac{0.1040}{L} f'(x_1) - \frac{0.2894}{L} f'(x_2) - \frac{0.6043}{L} f'(x_3) - \frac{2.0778}{L} f'(x_4)$.

Drawbacks
- Must choose N in advance
- Requires $O(N)$ memory for all gradient vectors $\{\nabla f(\boldsymbol{x}_n)\}_{n=1}^N$
- $O(N^2)$ computation for N iterations

Benefit: convergence bound (for specific N) $\approx 2\times$ lower than for Nesterov's FGM1.

New analytical solution

- Analytical solution for optimized step-size coefficients (Donghwan Kim, 2014):

$$H^* : \quad h_{n+1,k} = \begin{cases} \frac{\theta_n - 1}{\theta_{n+1}} h_{n,k}, & k = 0, \dots, n-2 \\ \frac{\theta_n - 1}{\theta_{n+1}} (h_{n,n-1} - 1), & k = n-1 \\ 1 + \frac{2\theta_n - 1}{\theta_{n+1}}, & k = n. \end{cases}$$

$$\theta_n = \begin{cases} 1, & n = 0 \\ \frac{1}{2}\left(1 + \sqrt{1 + 4\theta_{n-1}^2}\right), & n = 1, \dots, N-1 \\ \frac{1}{2}\left(1 + \sqrt{1 + 8\theta_{n-1}^2}\right), & n = N. \end{cases}$$

- Analytical convergence bound for these optimized step-size coefficients:

$$f(\boldsymbol{x}_N) - f(\boldsymbol{x}_\star) \le B_3(H^*, R, L, N) = \frac{1L \|\boldsymbol{x}_0 - \boldsymbol{x}_\star\|_2^2}{(N+1)(N+1+\sqrt{2})}.$$

  Of course bound is $O(1/N^2)$, but constant is twice better than that of Nesterov.
  No numerical SDP needed $\Longrightarrow$ feasible for large N.

(History: sought banded / structured lower-triangular form)

Optimized gradient method (OGM1)

Donghwan Kim (2014) found efficient recursive iteration:

Initialize: $\theta_0 = 1$, $\boldsymbol{z}_0 = \boldsymbol{x}_0$

$$\boldsymbol{z}_{n+1} = \boldsymbol{x}_n - \frac{1}{L}\nabla f(\boldsymbol{x}_n) \qquad \text{(usual GD update)}$$

$$\theta_n = \begin{cases} \frac{1}{2}\left(1 + \sqrt{1 + 4\theta_{n-1}^2}\right), & n = 1, \ldots, N-1 \\ \frac{1}{2}\left(1 + \sqrt{1 + 8\theta_{n-1}^2}\right), & n = N \end{cases} \qquad \text{(momentum factors)}$$

$$\boldsymbol{x}_{n+1} = \boldsymbol{z}_{n+1} + \frac{\theta_n - 1}{\theta_{n+1}}(\boldsymbol{z}_{n+1} - \boldsymbol{z}_n) + \underbrace{\frac{\theta_n}{\theta_{n+1}}(\boldsymbol{z}_{n+1} - \boldsymbol{x}_n)}_{\text{new momentum}}.$$

Reverts to Nesterov's FGM1 if the new terms are removed.

- Very simple modification of existing Nesterov code
- No need to choose N in advance (or solve SDP);
  use favorite stopping rule then run one last "decreased momentum" step.
- Factor of 2 better upper bound than Nesterov's "optimal" FGM1.

(Proofs omitted.)

Numerical Example(s)

Machine learning (logistic regression)

To learn weights $\boldsymbol{x}$ of binary classifier given feature vectors $\{\boldsymbol{v}_i\}$ and labels $\{y_i\}$:

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} f(\boldsymbol{x}), \qquad f(\boldsymbol{x}) = \sum_i \psi(y_i \langle \boldsymbol{x}, \boldsymbol{v}_i \rangle) + \beta \frac{1}{2} \|\boldsymbol{x}\|_2^2,$$

where $y_i = \pm 1$.

logistic: $\psi(t) = \log(1 + e^{-t}), \quad \dot{\psi}(t) = \frac{-1}{e^t + 1}, \quad \ddot{\psi}(t) = \frac{e^t}{(e^t + 1)^2} \in \left(0, \frac{1}{4}\right]$

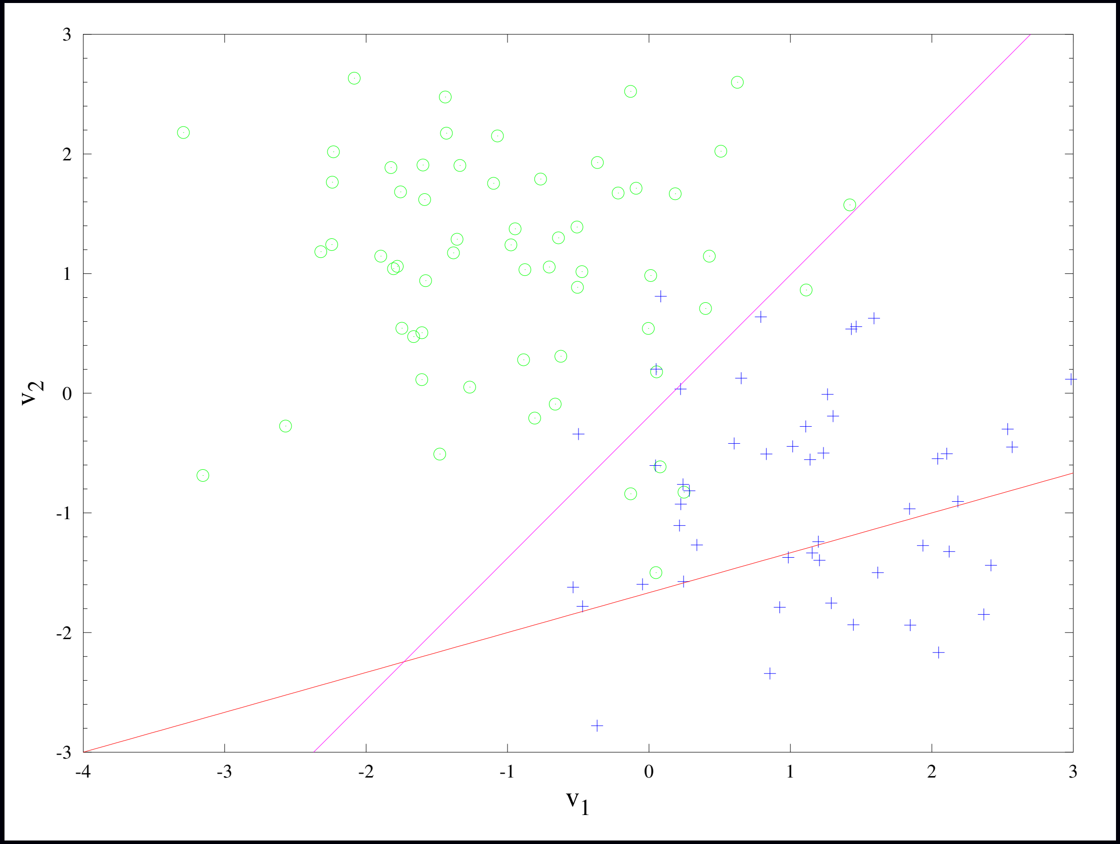
Gradient $\nabla f(\boldsymbol{x}) = \sum_i y_i \boldsymbol{v}_i \dot{\psi}(y_i \langle \boldsymbol{x}, \boldsymbol{v}_i \rangle) + \beta \boldsymbol{x}$

Hessian is positive definite so strictly convex:

$$\nabla^2 f(\boldsymbol{x}) = \sum_i \boldsymbol{v}_i \ddot{\psi}(y_i \langle \boldsymbol{x}, \boldsymbol{v}_i \rangle) \boldsymbol{v}_i' + \beta \boldsymbol{I} \preceq \frac{1}{4} \sum_i \boldsymbol{v}_i \boldsymbol{v}_i' + \beta \boldsymbol{I}$$

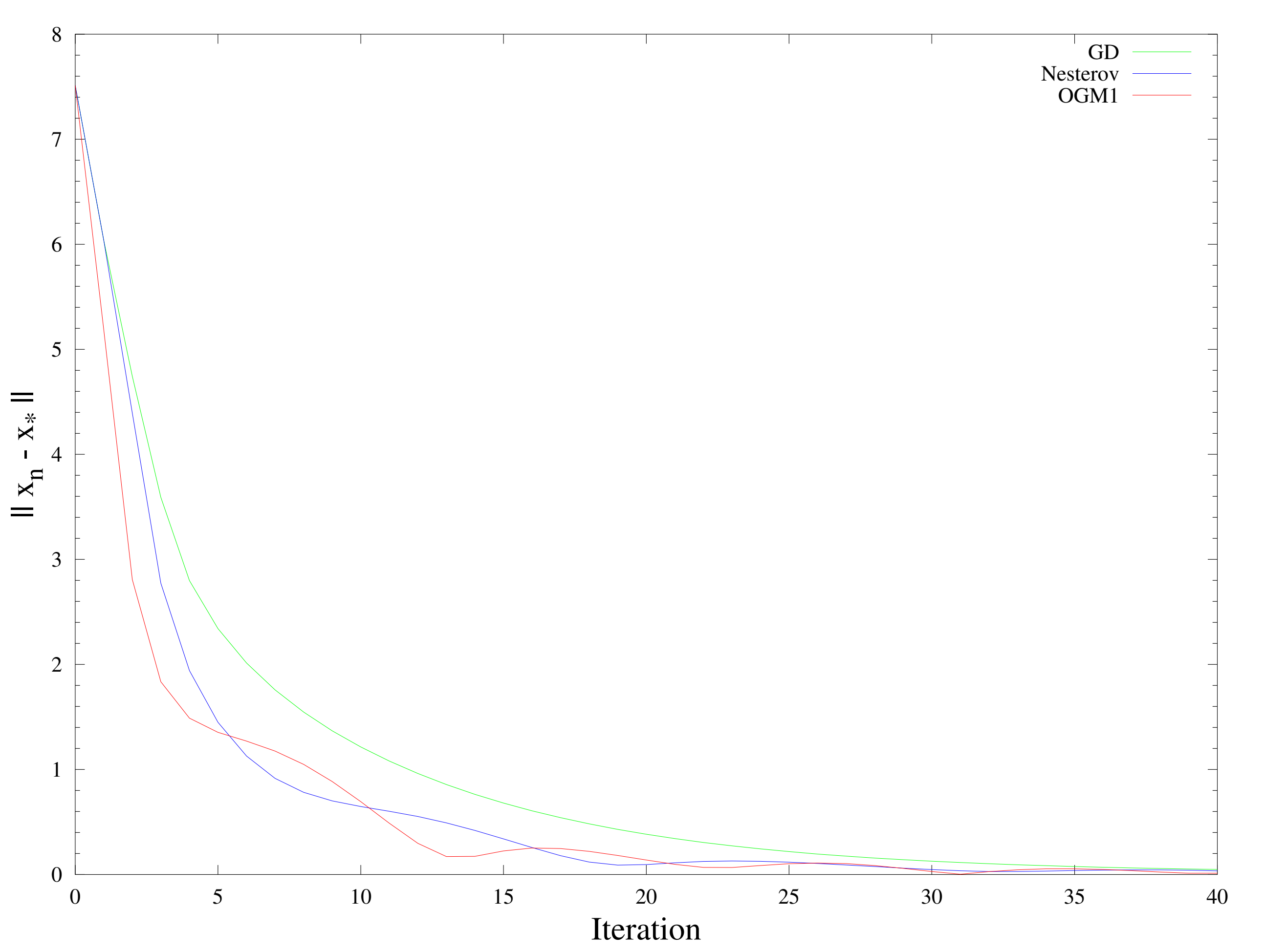
$$\Longrightarrow L \triangleq \frac{1}{4} \rho \left(\sum_i \boldsymbol{v}_i \boldsymbol{v}_i' \right) + \beta \geq \max_{\boldsymbol{x}} \rho \left(\nabla^2 f(\boldsymbol{x}) \right)$$

Numerical Results: logistic regression

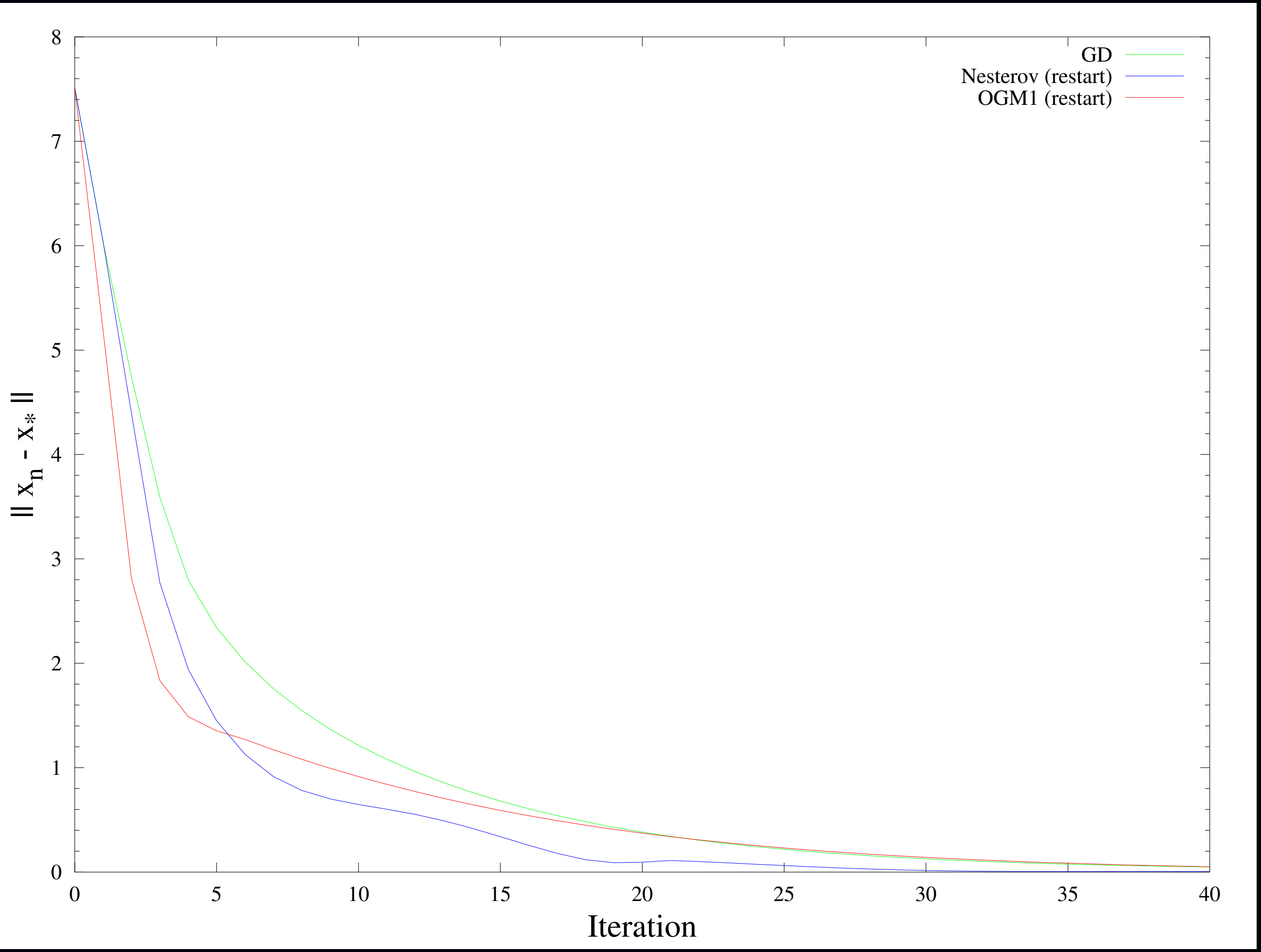


Training data, initial decision boundary (red), final decision boundary (magenta)

Numerical Results: convergence rates

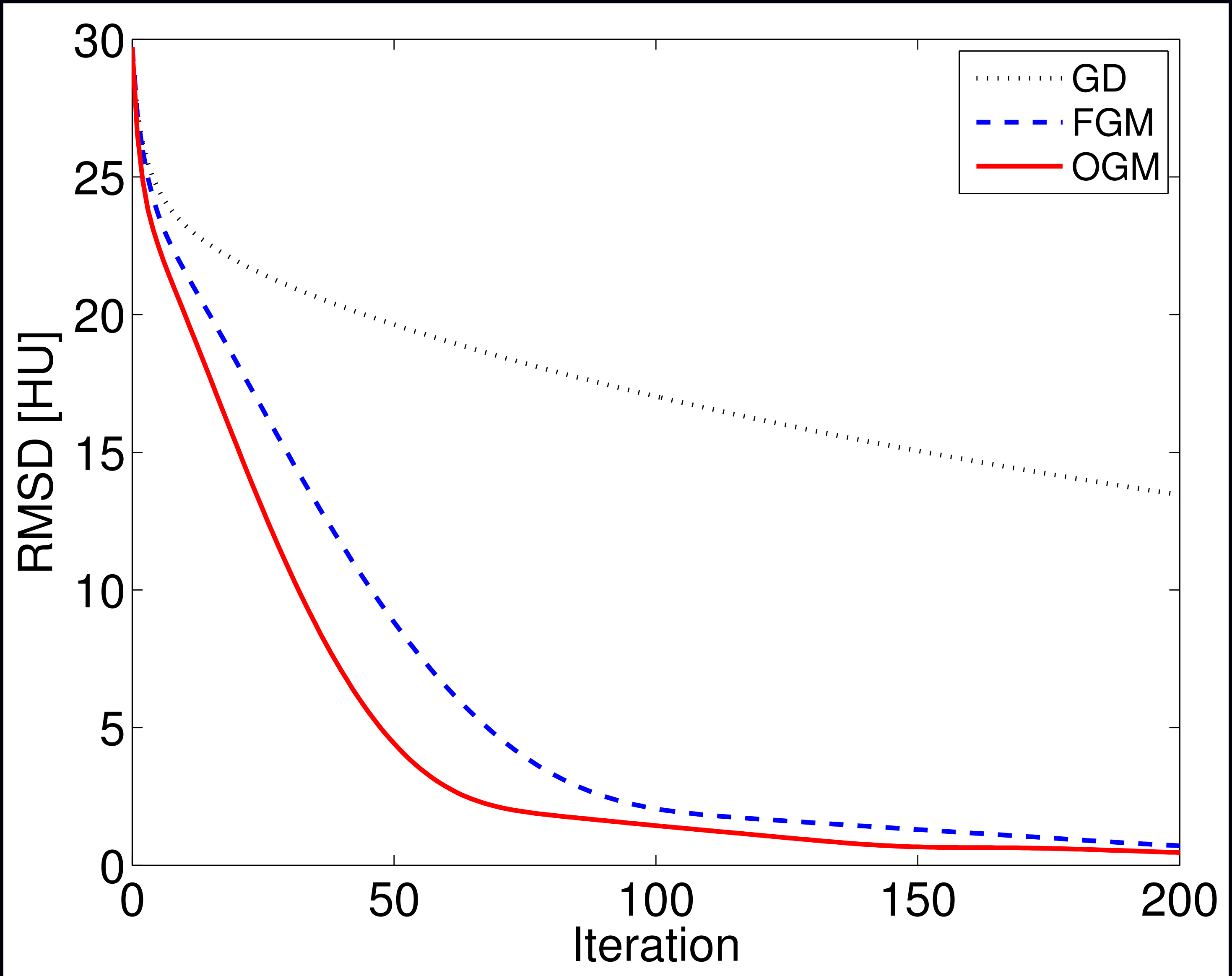


Numerical Results: adaptive restart



O'Donoghue & Candès, 2014

Low-dose 2D X-ray CT image reconstruction simulation



Summary

New optimized first-order minimization algorithm
Simple implementation akin to Nesterov's FGM
Analytical converge rate bound
Bound is $2\times$ better than Nesterov

Future work

- Constraints
- Non-smooth cost functions, *e.g.*, ℓ_1
- Tighter bounds
- Strongly convex case
- Asymptotic / local convergence rates
- Incremental gradients
- Stochastic gradient descent
- Adaptive restart
- Low-dose 3D X-ray CT image reconstruction

Bibliography

[1] Y. Drori and M. Teboulle. Performance of first-order methods for smooth convex minimization: A novel approach. Mathematical Programming, 2013.

[2] Y. Nesterov. A method for unconstrained convex minimization problem with the rate of convergence $O(1/k^2)$. Dokl. Akad. Nauk. USSR, 269(3):543–7, 1983.

[3] Y. Nesterov. Smooth minimization of non-smooth functions. Mathematical Programming, 103(1):127–52, May 2005.

[4] D. Kim and J. A. Fessler. Optimized first-order methods for smooth convex minimization, 2014. In preparation.

[5] D. Böhning and B. G. Lindsay. Monotonicity of quadratic approximation algorithms. Ann. Inst. Stat. Math., 40(4):641–63, December 1988.

[6] B. O'Donoghue and E. Candès. Adaptive restart for accelerated gradient schemes. Found. Computational Math., 2014.