Accelerated first-order methods

Geoff Gordon & Ryan Tibshirani
Optimization 10-725 / 36-725
Remember generalized gradient descent

We want to solve
\[
\min_{x \in \mathbb{R}^n} g(x) + h(x),
\]
for \(g\) convex and differentiable, \(h\) convex

**Generalized gradient descent:** choose initial \(x^{(0)} \in \mathbb{R}^n\), repeat:
\[
x^{(k)} = \text{prox}_{t_k}(x^{(k-1)} - t_k \cdot \nabla g(x^{(k-1)})), \quad k = 1, 2, 3, \ldots
\]
where the prox function is defined as
\[
\text{prox}_t(x) = \arg\min_{z \in \mathbb{R}^n} \frac{1}{2t} \|x - z\|^2 + h(z)
\]
If \(\nabla g\) is Lipschitz continuous, and prox function can be evaluated, then generalized gradient has rate \(O(1/k)\) (counts \# of iterations)

We can apply acceleration to achieve optimal \(O(1/k^2)\) rate!
Acceleration


- 1983: original acceleration idea for smooth functions
- 1988: another acceleration idea for smooth functions
- 2005: smoothing techniques for nonsmooth functions, coupled with original acceleration idea
- 2007: acceleration idea for composite functions


Tseng (2008): unified analysis of acceleration techniques (all of these, and more)

---

1 Each step uses entire history of previous steps and makes two prox calls
2 Each step uses only information from two last steps and makes one prox call
Today:

- Acceleration for composite functions (method of Beck and Teboulle (2008), presentation of Vandenberghe’s notes)
- Convergence rate
- FISTA
- Is acceleration always useful?
Accelerated generalized gradient method

Our problem

$$\min_{x \in \mathbb{R}^n} g(x) + h(x),$$

for $g$ convex and differentiable, $h$ convex

**Accelerated generalized gradient method:** choose any initial $x^{(0)} = x^{(-1)} \in \mathbb{R}^n$, repeat for $k = 1, 2, 3, \ldots$

$$y = x^{(k-1)} + \frac{k - 2}{k + 1} (x^{(k-1)} - x^{(k-2)})$$

$$x^{(k)} = \text{prox}_{t_k} (y - t_k \nabla g(y))$$

- First step $k = 1$ is just usual generalized gradient update
- After that, $y = x^{(k-1)} + \frac{k - 2}{k + 1} (x^{(k-1)} - x^{(k-2)})$ carries some “momentum” from previous iterations
- $h = 0$ gives accelerated gradient method
Consider minimizing

\[ f(x) = \sum_{i=1}^{n} \left( -y_i a_i^T x + \log(1 + \exp(a_i^T x)) \right) \]

i.e., **logistic regression** with predictors \( a_i \in \mathbb{R}^p \)

This is smooth, and

\[ \nabla f(x) = -A^T (y - p(x)) \]

where

\[ p_i(x) = \frac{\exp(a_i^T x)}{1 + \exp(a_i^T x)} \]

for \( i = 1, \ldots, n \)

No nonsmooth part here, so \( \text{prox}_t(x) = x \)
Example (with $n = 30$, $p = 10$):
Another example \((n = 30, p = 10)\):

![Graph showing descent methods](image)

Not a descent method!
Reformulation

Initialize $x^{(0)} = u^{(0)}$, and repeat for $k = 1, 2, 3, \ldots$

\[
    y = (1 - \theta_k)x^{(k-1)} + \theta_k u^{(k-1)}
\]

\[
    x^{(k)} = \text{prox}_{t_k}(y - t_k \nabla g(y))
\]

\[
    u^{(k)} = x^{(k-1)} + \frac{1}{\theta_k} (x^{(k)} - x^{(k-1)})
\]

with $\theta_k = 2/(k + 1)$

This is equivalent to the formulation of accelerated generalized gradient method presented earlier (slide 5). Makes convergence analysis easier

(Note: Beck and Teboulle (2008) use a choice $\theta_k < 2/(k + 1)$, but very close)
Convergence analysis

As usual, we are minimizing $f(x) = g(x) + h(x)$ assuming

- $g$ is convex, differentiable, $\nabla g$ is Lipschitz continuous with constant $L > 0$
- $h$ is convex, prox function can be evaluated

**Theorem:** Accelerated generalized gradient method with fixed step size $t \leq 1/L$ satisfies

$$f(x^{(k)}) - f(x^*) \leq \frac{2\|x^{(0)} - x^*\|^2}{t(k + 1)^2}$$

Achieves the optimal $O(1/k^2)$ rate for first-order methods!

I.e., to get $f(x^{(k)}) - f(x^*) \leq \epsilon$, need $O(1/\sqrt{\epsilon})$ iterations
Helpful inequalities

We will use

\[
\frac{1 - \theta_k}{\theta_k^2} \leq \frac{1}{\theta_{k-1}^2}, \quad k = 1, 2, 3, \ldots
\]

We will also use

\[
h(v) \leq h(z) + \frac{1}{t}(v - w)^T(z - v), \quad \text{all } z, w, v = \text{prox}_t(w)
\]

Why is this true? By definition of prox operator,

\[
v \text{ minimizes } \frac{1}{2t}\|w - v\|^2 + h(v) \iff 0 \in \frac{1}{t}(v - w) + \partial h(v)
\]

\[
\iff -\frac{1}{t}(v - w) \in \partial h(v)
\]

Now apply definition of subgradient
Convergence proof

We focus first on one iteration, and drop $k$ notation (so $x^+, u^+$ are updated versions of $x, u$). Key steps:

- $g$ Lipschitz with constant $L > 0$ and $t \leq 1/L \Rightarrow$

  \[ g(x^+) \leq g(y) + \nabla g(y)^T (x^+ - y) + \frac{1}{2t} \|x^+ - y\|^2 \]

- From our bound using prox operator,

  \[ h(x^+) \leq h(z) + \frac{1}{t} (x^+ - y)^T (z - x^+) + \nabla g(y)^T (z - x^+) \quad \text{all } z \]

- Adding these together and using convexity of $g$,

  \[ f(x^+) \leq f(z) + \frac{1}{t} (x^+ - y)^T (z - x^+) + \frac{1}{2t} \|x^+ - y\|^2 \quad \text{all } z \]
Using this bound at $z = x$ and $z = x^*$:

$$f(x^+) - f(x^*) - (1 - \theta)(f(x) - f(x^*))$$

$$\leq \frac{1}{t}(x^+ - y)^T(\theta x^* + (1 - \theta)x - x^+) + \frac{1}{2t}\|x^+ - y\|^2$$

$$= \frac{\theta^2}{2t}\left(\|u - x^*\|^2 - \|u^+ - x^*\|^2\right)$$

I.e., at iteration $k$,

$$\frac{t}{\theta_k^2}(f(x^{(k)}) - f(x^*)) + \frac{1}{2}\|u^{(k)} - x^*\|^2$$

$$\leq \frac{(1 - \theta_k)t}{\theta_k^2}(f(x^{(k-1)}) - f(x^*)) + \frac{1}{2}\|u^{(k-1)} - x^*\|^2$$
Using $(1 - \theta_i)/\theta_i^2 \leq 1/\theta_{i-1}^2$, and iterating this inequality,

$$
\frac{t}{\theta_k^2}(f(x^{(k)}) - f(x^*)) + \frac{1}{2}\|u^{(k)} - x^*\|^2 \\
\leq \frac{(1 - \theta_1)t}{\theta_1^2}(f(x^{(0)}) - f(x^*)) + \frac{1}{2}\|u^{(0)} - x^*\|^2 \\
= \frac{1}{2}\|x^{(0)} - x^*\|^2
$$

Therefore

$$
f(x^{(k)}) - f(x^*) \leq \frac{\theta_k^2}{2t}\|x^{(0)} - x^*\|^2 = \frac{2}{t(k + 1)^2}\|x^{(0)} - x^*\|^2
$$
Backtracking line search

A few ways to do this with acceleration ... here’s a simple method (more complicated strategies exist)

First think: what do we need $t$ to satisfy? Looking back at proof with $t_k = t \leq 1/L$,

- We used

$$g(x^+) \leq g(y) + \nabla g(y)^T (x^+ - y) + \frac{1}{2t} \|x^+ - y\|^2$$

- We also used

$$\frac{(1 - \theta_k)t_k}{\theta_k^2} \leq \frac{t_{k-1}}{\theta_{k-1}^2},$$

so it suffices to have $t_k \leq t_{k-1}$, i.e., decreasing step sizes
Backtracking algorithm: fix $\beta < 1$, $t_0 = 1$. At iteration $k$, replace $x$ update (i.e., computation of $x^+$) with:

- Start with $t_k = t_{k-1}$ and $x^+ = \text{prox}_{t_k}(y - t_k \nabla g(y))$
- While $g(x^+) > g(y) + \nabla g(y)^T(x^+ - y) + \frac{1}{2t_k} \|x^+ - y\|^2$, repeat:
  - $t_k = \beta t_k$ and $x^+ = \text{prox}_{t_k}(y - t_k \nabla g(y))$

Note this achieves both requirements. So under same conditions ($\nabla g$ Lipschitz, prox function evaluable), we get same rate

**Theorem:** Accelerated generalized gradient method with backtracking line search satisfies

$$f(x^{(k)}) - f(x^*) \leq \frac{2\|x^{(0)} - x^*\|^2}{t_{\min}(k + 1)^2}$$

where $t_{\min} = \min\{1, \beta/L\}$
Recall lasso problem,
\[
\min_x \frac{1}{2}\|y - Ax\|^2 + \lambda\|x\|_1
\]
and ISTA (Iterative Soft-thresholding Algorithm):
\[
x^{(k)} = S_{\lambda t_k}(x^{(k-1)} + t_k A^T(y - Ax^{(k-1)})), \quad k = 1, 2, 3, \ldots
\]
\(S_{\lambda}(\cdot)\) being matrix soft-thresholding. Applying acceleration gives us **FISTA** (F is for Fast):\(^3\)
\[
v = x^{(k-1)} + \frac{k - 2}{k + 1}(x^{(k-1)} - x^{(k-2)})
\]
\[
x^{(k)} = S_{\lambda t_k}(v + t_k A^T(y - Av)), \quad k = 1, 2, 3, \ldots
\]

\(^3\)Beck and Teboulle (2008) actually call their general acceleration technique (for general \(g, h\)) FISTA, which may be somewhat confusing
Lasso regression: 100 instances (with $n = 100$, $p = 500$):
Lasso logistic regression: 100 instances \((n = 100, p = 500)\):
Is acceleration always useful?

Acceleration is generally a very effective speedup tool ... but should it always be used?

In practice the speedup of using acceleration is diminished in the presence of **warm starts**. I.e., suppose want to solve lasso problem for tuning parameters values

\[
\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_r
\]

- When solving for \(\lambda_1\), initialize \(x^{(0)} = 0\), record solution \(\hat{x}(\lambda_1)\)
- When solving for \(\lambda_j\), initialize \(x^{(0)} = \hat{x}(\lambda_{j-1})\), the recorded solution for \(\lambda_{j-1}\)

Over a fine enough grid of \(\lambda\) values, generalized gradient descent perform can perform just as well without acceleration
Sometimes acceleration and even backtracking can be harmful!

Recall matrix completion problem: observe some only entries of $A$, $(i, j) \in \Omega$, we want to fill in the rest, so we solve

$$
\min_X \frac{1}{2} \| P_\Omega(A) - P_\Omega(X) \|_F^2 + \lambda \| X \|_*
$$

where $\| X \|_* = \sum_{i=1}^r \sigma_i(X)$, nuclear norm, and

$$
[P_\Omega(X)]_{ij} = \begin{cases}
X_{ij} & (i, j) \in \Omega \\
0 & (i, j) \notin \Omega
\end{cases}
$$

Generalized gradient descent with $t = 1$ (soft-impute algorithm): updates are

$$
X^+ = S_\lambda(P_\Omega(A) + P_{\|\Omega}^\perp(X))
$$

where $S_\lambda$ is the matrix soft-thresholding operator ... requires SVD
Backtracking line search with generalized gradient:

- Each backtracking loop evaluates generalized gradient $G_t(x)$ at various values of $t$
- Hence requires multiple evaluations of $\text{prox}_t(x)$
- For matrix completion, can’t afford this!

Acceleration with generalized gradient:

- Changes argument we pass to prox function: $y - t\nabla g(y)$ instead of $x - t\nabla g(x)$
- For matrix completion (and $t = 1$),

$$X - \nabla g(X) = P_\Omega(A) + P_\Omega^\perp(X) \Rightarrow \text{fast SVD}$$

$$Y - \nabla g(Y) = P_\Omega(A) + P_\Omega^\perp(Y) \Rightarrow \text{slow SVD}$$
Soft-impute uses $L = 1$ and exploits special structure ... so it can outperform fancier methods. E.g., soft-impute (solid blue line) vs accelerated generalized gradient (dashed black line):

(From Mazumder et al. (2011), *Spectral regularization algorithms for learning large incomplete matrices*)
Optimization for well-behaved problems

For statistical learning problems, “well-behaved” means:

- signal to noise ratio is decently high
- correlations between predictor variables are under control
- number of predictors \( p \) can be larger than number of observations \( n \), but not absurdly so

For well-behaved learning problems, people have observed that gradient or generalized gradient descent can converge extremely quickly (much more so than predicted by \( O(1/k) \) rate)

Largely unexplained by theory, topic of current research. E.g., very recent work\(^4\) shows that for some well-behaved problems, w.h.p.:

\[
\| x^{(k)} - x^* \|^2 \leq c^k \| x^{(0)} - x^* \|^2 + o(\| x^* - x^{\text{true}} \|^2)
\]

\(^4\) Agarwal et al. (2012), *Fast global convergence of gradient methods for high-dimensional statistical recovery*
Nesterov’s four ideas (three acceleration methods):

- Y. Nesterov (1983), *A method for solving a convex programming problem with convergence rate $O(1/k^2)$*
- Y. Nesterov (1988) *On an approach to the construction of optimal methods of minimization of smooth convex functions*
- Y. Nesterov (2005), *Smooth minimization of non-smooth functions*
- Y. Nesterov (2007), *Gradient methods for minimizing composite objective function*
Extensions and/or analyses:

- A. Beck and M. Teboulle (2008), *A fast iterative shrinkage-thresholding algorithm for linear inverse problems*
- S. Becker and J. Bobin and E. Candes (2009), *NESTA: A fast and accurate first-order method for sparse recovery*
- P. Tseng (2008), *On accelerated proximal gradient methods for convex-concave optimization*

and there are many more ...

Helpful lecture notes/books:

- E. Candes, Lecture Notes for Math 301, Stanford University, Winter 2010-2011
- L. Vandenberghe, Lecture Notes for EE 236C, UCLA, Spring 2011-2012