Mathematics of Data: From Theory to Computation

Prof. Volkan Cevher
volkan.cevher@epfl.ch

Lecture 9: Composite convex minimization II

Laboratory for Information and Inference Systems (LIONS)
École Polytechnique Fédérale de Lausanne (EPFL)

EE-556 (Fall 2017)

Fonds national suisse
Schweizerischer Nationalfonds
Fondo nazionale svizzero
Swiss National Science Foundation

License Information for Mathematics of Data Slides

- This work is released under a Creative Commons License with the following terms:
- **Attribution**
  - The licensor permits others to copy, distribute, display, and perform the work. In return, licensees must give the original authors credit.
- **Non-Commercial**
  - The licensor permits others to copy, distribute, display, and perform the work. In return, licensees may not use the work for commercial purposes – unless they get the licensor's permission.
- **Share Alike**
  - The licensor permits others to distribute derivative works only under a license identical to the one that governs the licensor's work.
- Full Text of the License

Outline

- Today
  1. Proximal Newton-type method.
  2. Stochastic proximal gradient method.
  3. Stochastic proximal gradient method with progressive variance reduction.
- Next week
  1. Constrained convex minimization I: The primal-dual approach.
  2. Smoothing approaches for non-smooth convex minimization.

Recommended reading material

- A. Beck and M. Tebulle, A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems, SIAM J. Imaging Sciences, 2(1), 183–202, 2009.
- Y. Nesterov, Smooth minimization of non-smooth functions, Math. Program, 103(1), 127–152, 2005.
- Q. Tran-Dinh, A. Kyrillidis, and V. Cevher. Composite self-concordant minimization. Journal of Machine Learning Research (16), 371-416, 2015
- L. Xiao and T. Zhang, A Proximal Stochastic Gradient Method with Progressive Variance Reduction, SIAM J. Optim., 24(4), 2057-2075, 2014.
- N. Parikh and S. Boyd, Proximal Algorithms, Foundations and Trends in Optimization, 1(3):123-231, 2014.
- L. Rosacso, S. Villa, B. C. Vũ, Stochastic Forward-Backward Splitting for Monotone Inclusions, J. Optim. Theory Appl. (169), 388-406, 2016.

Motivation

Motivation

Data analytics problems in various disciplines can often be simplified to nonsmooth **composite convex minimization** problems. To this end, this lecture provides **efficient numerical solution methods** for such problems.

Intriguingly, composite minimization problems are far from generic nonsmooth problems and we can exploit individual function structures to obtain numerical solutions nearly as efficiently as if they are smooth problems.

Composite convex minimization

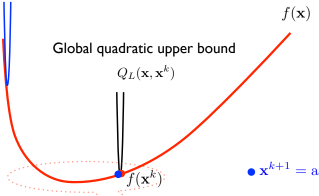
Problem (Unconstrained composite convex minimization)

$$\boxed{F^\star := \min_{\mathbf{x}\in\mathbb{R}^p} \{F(\mathbf{x}) := f(\mathbf{x}) + g(\mathbf{x})\}} \tag{1}$$

- *f and g are both proper, closed, and convex.*
- *dom(F) := dom$(f) \cap$ dom$(g) \neq \emptyset$ and $-\infty < F^\star < +\infty$.*
- *The solution set $\mathcal{S}^\star := \{\mathbf{x}^\star \in$ dom$(F) : F(\mathbf{x}^\star) = F^\star\}$ is nonempty.*

How can we better adapt to the local geometry?

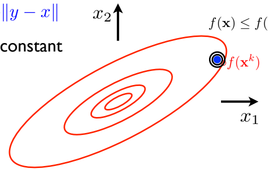
Non-adaptive:



- $\mathbf{x}^{k+1} = \arg\min_{\mathbf{x}} \left\{ f(\mathbf{x}^k) + \langle \nabla f(\mathbf{x}^k), \mathbf{x} - \mathbf{x}^k \rangle + \frac{L}{2} \|\mathbf{x} - \mathbf{x}^k\|_2^2 \right\}$

$\|\nabla f(x) - \nabla f(y)\| \leq L\|y - x\|$

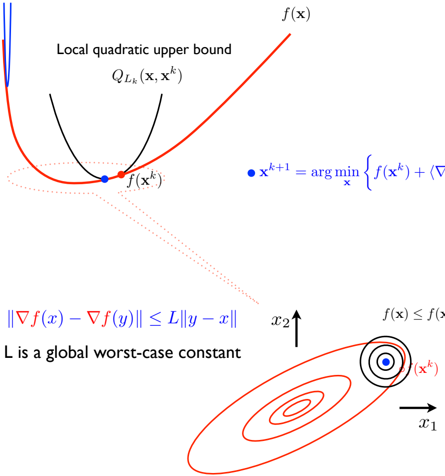
L is a global worst-case constant



$f(\mathbf{x}) \leq f(\mathbf{x}^k) + \nabla f(\mathbf{x}^k)^T(\mathbf{x} - \mathbf{x}^k) + \frac{L}{2}\|\mathbf{x} - \mathbf{x}^k\|_2^2$

How can we better adapt to the local geometry?

Line-search:


$f(\mathbf{x})$
Local quadratic upper bound
$Q_{L_k}(\mathbf{x}, \mathbf{x}^k)$
$f(\mathbf{x}^k)$
x_2
$f(\mathbf{x}^k)$
x_1

● $\mathbf{x}^{k+1} = \arg\min_{\mathbf{x}} \left\{ f(\mathbf{x}^k) + \langle \nabla f(\mathbf{x}^k), \mathbf{x} - \mathbf{x}^k \rangle + \frac{L_k}{2} \|\mathbf{x} - \mathbf{x}^k\|_2^2 \right\}$

$\|\nabla f(x) - \nabla f(y)\| \leq L\|y - x\|$

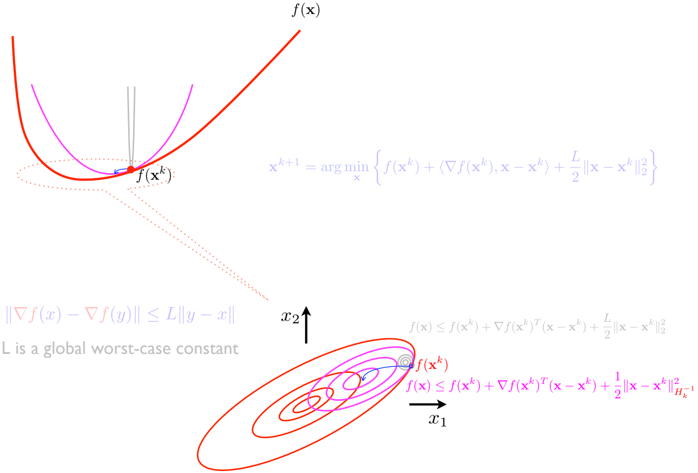
L is a global worst-case constant

$f(\mathbf{x}) \leq f(\mathbf{x}^k) + \nabla f(\mathbf{x}^k)^T(\mathbf{x} - \mathbf{x}^k) + \frac{L_k}{2}\|\mathbf{x} - \mathbf{x}^k\|_2^2$

applies only locally

How can we better adapt to the local geometry?

Variable metric:



$$\mathbf{x}^{k+1} = \arg\min_{\mathbf{x}} \left\{ f(\mathbf{x}^k) + \langle \nabla f(\mathbf{x}^k), \mathbf{x} - \mathbf{x}^k \rangle + \frac{L}{2} \|\mathbf{x} - \mathbf{x}^k\|_2^2 \right\}$$

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|y - x\|$$

L is a global worst-case constant

$$f(\mathbf{x}) \leq f(\mathbf{x}^k) + \nabla f(\mathbf{x}^k)^T(\mathbf{x} - \mathbf{x}^k) + \frac{L}{2}\|\mathbf{x} - \mathbf{x}^k\|_2^2$$

$$f(\mathbf{x}) \leq f(\mathbf{x}^k) + \nabla f(\mathbf{x}^k)^T(\mathbf{x} - \mathbf{x}^k) + \frac{1}{2}\|\mathbf{x} - \mathbf{x}^k\|_{H_k^{-1}}^2$$

The idea of the proximal-Newton method

Assumptions A.2

Assume that $f \in \mathcal{F}_{L,\mu}^{2,1}(\mathbb{R}^p)$ and $g \in \mathcal{F}_{\text{prox}}(\mathbb{R}^p)$.

Optimality condition of (1):

$$0 \in \nabla f(\mathbf{x}^\star) + \partial g(\mathbf{x}^\star). \tag{2}$$

The idea of proximal-Newton method

- Given $\mathbf{x}^k$, under Assumptions A.2, we can linearize the smooth term of the (2):

$$0 \in \nabla f(\mathbf{x}^\star) + \partial g(\mathbf{x}^\star) \approx \nabla f(\mathbf{x}^k) + \nabla^2 f(\mathbf{x}^k)^T(\mathbf{x}^\star - \mathbf{x}^k) + \partial g(\mathbf{x}^\star).$$

- Similar to the classical Newton method: Solving

$$0 \in \nabla f(\mathbf{x}^k) + \nabla^2 f(\mathbf{x}^k)^T(\mathbf{x} - \mathbf{x}^k) + \partial g(\mathbf{x}) \tag{3}$$

to obtain $\mathbf{x}^{k+1}$.

- The last condition is equivalent to

$$\mathbf{x}^{k+1} := \arg\min_{\mathbf{x}\in\mathbb{R}^p}\left\{\frac{1}{2}(\mathbf{x}-\mathbf{x}^k)^T\nabla^2 f(\mathbf{x}^k)(\mathbf{x}-\mathbf{x}^k)+\nabla f(\mathbf{x}^k)^T(\mathbf{x}-\mathbf{x}^k)+g(\mathbf{x})\right\}. \tag{4}$$

Proximal-quasi-Newton-type algorithms

- Let $\mathbf{H}_k \approx \nabla^2 f(\mathbf{x}^k)$ be a symmetric positive definite (SDP) matrix. Then, we have

  $$\mathbf{x}^k - \mathbf{H}_k^{-1}\nabla f(\mathbf{x}^k) \in (\mathbb{I} + \mathbf{H}_k^{-1}\partial g)(\mathbf{x}^{k+1}),$$

  or

  $$\mathbf{x}^{k+1} := \mathrm{prox}_{\mathbf{H}_k^{-1} g}\Big(\mathbf{x}^k - \mathbf{H}_k^{-1}\nabla f(\mathbf{x}^k)\Big). \tag{5}$$

- By letting $\mathbf{d}^k := \mathbf{x}^{k+1} - \mathbf{x}^k$, (5) is equivalent to

  $$\mathbf{d}^k := \arg\min_{\mathbf{d}\in\mathbb{R}^p} \Big\{\frac{1}{2}\mathbf{d}^T\mathbf{H}_k\mathbf{d} + \nabla f(\mathbf{x}^k)^T\mathbf{d} + g(\mathbf{x}^k + \mathbf{d})\Big\}. \tag{6}$$

  Then $\mathbf{d}^k$ is called a proximal-Newton-type direction.

- Proximal-Newton-type algorithm generates a sequence $\{\mathbf{x}^k\}_{k\geq 0}$ starting from $\mathbf{x}^0 \in \mathbb{R}^p$ and update:

  $$\mathbf{x}^{k+1} := \mathbf{x}^k + \alpha_k\mathbf{d}^k, \tag{7}$$

  where $\mathbf{d}^k$ is given by (6) and $\alpha_k \in (0, 1]$ is a damped step-size.

How to find step size α_k?

Lemma (Descent lemma [6])

Let $\mathbf{x}^k(\alpha) := \mathbf{x}^k + \alpha \mathbf{d}^k$ *for sufficiently small* $\alpha \in (0, 1]$ *and* $\mathbf{H}_k \succ 0$. *Then, we have:*

$$F(\mathbf{x}^k(\alpha)) \leq F(\mathbf{x}^k) - (1/2)\alpha(\mathbf{d}^k)^T \mathbf{H}_k \mathbf{d}^k + \mathcal{O}(\alpha^2).$$

Since $\mathbf{H}_k \succ 0$, this lemma tells us that:

- If $\mathbf{d}^k \neq 0$, then there exists $\alpha > 0$ such that $F(\mathbf{x}^k(\alpha)) < F(\mathbf{x}^k)$.
- The value of α can be computed via **backtracking line search**.
- If $\mathbf{d}^k = 0$, then we can easily check that $\mathbf{x}^k$ is a solution of (1).

Backtracking line-search

- Let
$$r_k := \nabla f(\mathbf{x}^k)^T \mathbf{d}^k + g(\mathbf{x}^k + \mathbf{d}^k) - g(\mathbf{x}^k).$$
- Find the smallest integer number $j \geq 0$ such that $\alpha_k := \beta^j$ and
$$F(\mathbf{x}^k + \alpha_k \mathbf{d}^k) \leq F(\mathbf{x}^k) + c\alpha_k r_k, \tag{8}$$

where $c \in (0, 0.5]$ and $\beta \in (0, 1)$ are two given constants (e.g., $c = 0.1$ and $\beta = 0.5$).

The proximal-Newton-type algorithm

We can summary the **proximal-Newton-type method** as follows:

Proximal-Newton algorithm (PNA)

```
1. Given $\mathbf{x}^0 \in \mathbb{R}^p$ as a starting point. Choose $c := 0.1$ and $\beta := 0.5$
2. For $k = 0, 1, \cdots$, perform the following steps:
    2.1. Evaluate an SDP matrix $\mathbf{H}_k \approx \nabla^2 f(\mathbf{x}^k)$ and $\nabla f(\mathbf{x}^k)$.
    2.2. Compute $\mathbf{d}^k := \mathrm{prox}_{\mathbf{H}_k^{-1} g}\left(\mathbf{x}^k - \mathbf{H}_k^{-1}\nabla f(\mathbf{x}^k)\right) - \mathbf{x}^k$.
    2.3. Find the smallest integer number $j \geq 0$ such that
             $F(\mathbf{x}^k + \beta^j \mathbf{d}^k) \leq F(\mathbf{x}^k) + c\beta^j r_k$
         and set $\alpha_k := \beta^j$.
    2.4. Update $\mathbf{x}^{k+1} := \mathbf{x}^k + \alpha_k \mathbf{d}^k$.
```

- If $\mathbf{H}_k \equiv \nabla^2 f(\mathbf{x}^k)$, then **PNA** becomes a pure proximal-Newton algorithm.
- If $\mathbf{H}_k \approx \nabla^2 f(\mathbf{x}^k)$, then **PNA** becomes a proximal-quasi-Newton algorithm.
- Main computation is Step 2.2, which requires a generalized prox-operator: $\mathrm{prox}_{\mathbf{H}_k^{-1} g}\left(\mathbf{x}^k + \mathbf{H}_k^{-1}\nabla f(\mathbf{x}^k)\right)$.
- Let $g(\mathbf{x}) = \rho\|\mathbf{x}\|_1$. When $\mathbf{H}_k$ is not diagonal, the cost is the same as solving an ℓ_1-regularized least squares, otherwise it is simply soft thresholding.

Convergence analysis

Assumption A.3.

- The subproblem $\mathrm{prox}_{\mathbf{H}_k^{-1}g}\left(\mathbf{x}^k + \mathbf{H}_k^{-1}\nabla f(\mathbf{x}^k)\right)$ is solved exactly for all $k \geq 0$.

Theorem (Global convergence [6])

Assumptions:

- *The sequence $\{\mathbf{x}^k\}_{k\geq 0}$ is generated by PNA.*
- *Assumption A.3. is satisfied.*
- *There exists $\mu > 0$ such that $\mathbf{H}_k \succeq \mu\mathbb{I}$ for all $k \geq 0$.*

Conclusion:

- *$\{\mathbf{x}^k\}_{k\geq 0}$ globally converges to a solution $\mathbf{x}^\star$ of (1).*

- We have not yet obtained a **global convergence rate** of proximal-Newton methods.

Convergence analysis

Assumption A.3.

- The subproblem $\text{prox}_{\mathbf{H}_k^{-1}g}\left(\mathbf{x}^k + \mathbf{H}_k^{-1}\nabla f(\mathbf{x}^k)\right)$ is solved exactly for all $k \geq 0$.

Theorem (Local convergence [6])

Assumptions:

- *The sequence $\{\mathbf{x}^k\}_{k\geq 0}$ is generated by PNA.*
- *Assumption A.3. is satisfied.*
- *Exist $0 < \mu \leq L_2 < +\infty$ such that $\mu\mathbb{I} \preceq \mathbf{H}_k \preceq L_2\mathbb{I}$ for all sufficiently large k.*

Conclusion:

- *If $\mathbf{H}_k \equiv \nabla^2 f(\mathbf{x}^k)$, then $\alpha_k = 1$ for k sufficiently large (full-step).*
- *If $\mathbf{H}_k \equiv \nabla^2 f(\mathbf{x}^k)$, then $\{\mathbf{x}^k\}$ locally converges to $\mathbf{x}^\star$ at a quadratic rate.*
- *If $\mathbf{H}_k$ satisfies the Dennis-Moré condition:*

$$\lim_{k\to+\infty} \frac{\|(\mathbf{H}_k - \nabla^2 f(\mathbf{x}^\star))(\mathbf{x}^{k+1} - \mathbf{x}^k)\|}{\|\mathbf{x}^{k+1} - \mathbf{x}^k\|} = 0, \tag{9}$$

then $\{\mathbf{x}^k\}$ locally converges to $\mathbf{x}^\star$ at a super linear rate.

How to compute the approximation $\mathbf{H}_k$?

How to update $\mathbf{H}_k$?

Matrix $\mathbf{H}_k$ can be updated by using low-rank updates.

- **BFGS update**: maintain the **Dennis-Moré condition** and $\mathbf{H}_k \succ 0$.

$$\mathbf{H}_{k+1} := \mathbf{H}_k + \frac{\mathbf{y}_k\mathbf{y}_k^T}{\mathbf{s}_k^T\mathbf{y}_k} - \frac{\mathbf{H}_k\mathbf{s}_k\mathbf{s}_k^T\mathbf{H}_k}{\mathbf{s}_k^T\mathbf{H}_k\mathbf{s}_k}, \quad \mathbf{H}_0 := \gamma\mathbb{I}, \ (\gamma > 0).$$

  where $\mathbf{y}_k := \nabla f(\mathbf{x}^{k+1}) - \nabla f(\mathbf{x}^k)$ and $\mathbf{s}_k := \mathbf{x}^{k+1} - \mathbf{x}^k$.

- **Diagonal+Rank-1 [2]**: computing PN direction $\mathbf{d}^k$ is in polynomial time, but it does not maintain the Dennis-Moré condition:

$$\mathbf{H}_k := \mathbf{D}_k + \mathbf{u}_k\mathbf{u}_k^T, \quad \mathbf{u}_k := (\mathbf{s}_k - \mathbf{H}_0\mathbf{y}_k)/\sqrt{(\mathbf{s}_k - \mathbf{H}_0\mathbf{y}_k)^T\mathbf{y}_k},$$

  where $\mathbf{D}_k$ is a positive diagonal matrix.

Advantages and disadvantages

Advantages

- PNA has fast local convergence rate (super-linear or quadratic)
- Numerical robustness under the inexactness/noise (inexact proximal-Newton method [6]).
- Quasi-Newton method is useful if the evaluation of $\nabla^2 f$ is expensive.
- Suitable for problems with many data points but few parameters. For example, problems of the form:

$$F^* := \min_{\mathbf{x}\in\mathbb{R}^p} \left\{ \sum_{j=1}^{n} \ell_j(\mathbf{a}_j^T\mathbf{x} + b_j) + g(\mathbf{x}) \right\},$$

  where ℓ_j is twice continuously differentiable and convex, $g \in \mathcal{F}_{\mathrm{prox}}$, $p \ll n$.

Disadvantages

- Expensive iteration compared to proximal-gradient methods.
- Global convergence rate may be worse than accelerated proximal-gradient methods.
- Requires a good initial point to get fast local convergence, which is hard to find.
- Requires strict conditions for global/local convergence analysis.

Example 1: Sparse logistic regression

Problem (**Sparse logistic regression**)

Given a sample vector $\mathbf{a} \in \mathbb{R}^p$ and a binary class label vector $\mathbf{b} \in \{-1, +1\}^n$. The conditional probability of a label b given $\mathbf{a}$ is defined as:

$$\mathbb{P}(b|\mathbf{a}, \mathbf{x}, \mu) = 1/(1 + e^{-b(\mathbf{x}^T\mathbf{a}+\mu)}),$$

where $\mathbf{x} \in \mathbb{R}^p$ is a weight vector, μ is called the intercept.
Goal: *Find a sparse-weight vector $\mathbf{x}$ via the maximum likelihood principle.*

Optimization formulation

$$\boxed{\min_{\mathbf{x}\in\mathbb{R}^p} \Bigg\{ \underbrace{\frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(b_i(\mathbf{a}_i^T\mathbf{x} + \mu))}_{f(\mathbf{x})} + \underbrace{\rho\|\mathbf{x}\|_1}_{g(\mathbf{x})} \Bigg\},} \tag{10}$$

where $\mathbf{a}_i$ is the i-th row of data matrix $\mathbf{A}$ in $\mathbb{R}^{n\times p}$, $\rho > 0$ is a regularization parameter, and ℓ is the logistic loss function $\mathcal{L}(\tau) := \log(1 + e^{-\tau})$.

Example: Sparse logistic regression

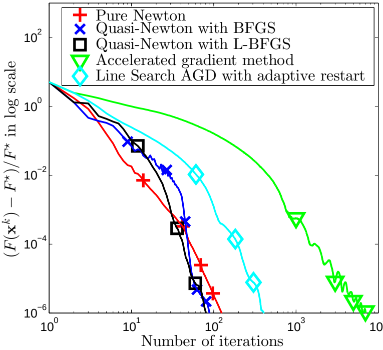
Real data

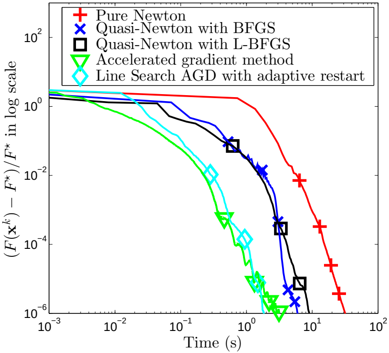
- Real data: `w2a` with $n = 3470$ data points, $p = 300$ features
- Available at `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html`.

Parameters

- Tolerance 10^{-6}.
- L-BFGS memory $m = 50$.
- Ground truth: Get a high accuracy approximation of $\mathbf{x}^\star$ and $f^\star$ by TFOCS with tolerance 10^{-12}.

Example: Sparse logistic regression-Numerical results





Example 2: ℓ_1-regularized least squares

Problem (ℓ_1-regularized least squares)

Given $\mathbf{A} \in \mathbb{R}^{n \times p}$ *and* $\mathbf{b} \in \mathbb{R}^n$, *solve:*

$$F^\star := \min_{\mathbf{x} \in \mathbb{R}^p} \left\{ F(\mathbf{x}) := \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 + \rho\|\mathbf{x}\|_1 \right\}, \tag{11}$$

where $\rho > 0$ *is a regularization parameter.*

Complexity per iterations

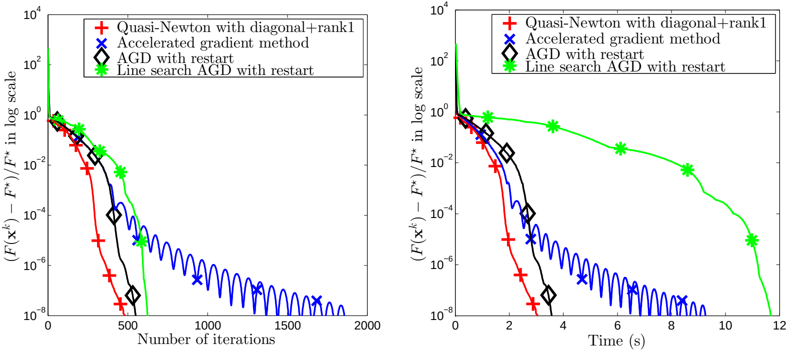
- Evaluating $\nabla f(\mathbf{x}^k) = \mathbf{A}^T(\mathbf{A}\mathbf{x}^k - \mathbf{b})$ requires one $\mathbf{A}\mathbf{x}$ and one $\mathbf{A}^T\mathbf{y}$.
- One soft-thresholding operator $\mathrm{prox}_{\lambda g}(\mathbf{x}) = \mathrm{sign}(\mathbf{x}) \otimes \max\{|\mathbf{x}| - \rho, 0\}$.
- **Optional**: Evaluating $L = \|\mathbf{A}^T\mathbf{A}\|$ (spectral norm) - via **power iterations** (e.g., 20 iterations, each iteration requires one $\mathbf{A}\mathbf{x}$ and one $\mathbf{A}^T\mathbf{y}$).

Synthetic data generation

- $\mathbf{A} := \mathrm{randn}(n, p)$ - standard Gaussian $\mathcal{N}(0, \mathbb{I})$.
- $\mathbf{x}^\star$ is a s-sparse vector generated randomly.
- $\mathbf{b} := \mathbf{A}\mathbf{x}^\star + \mathcal{N}(0, 10^{-3})$.

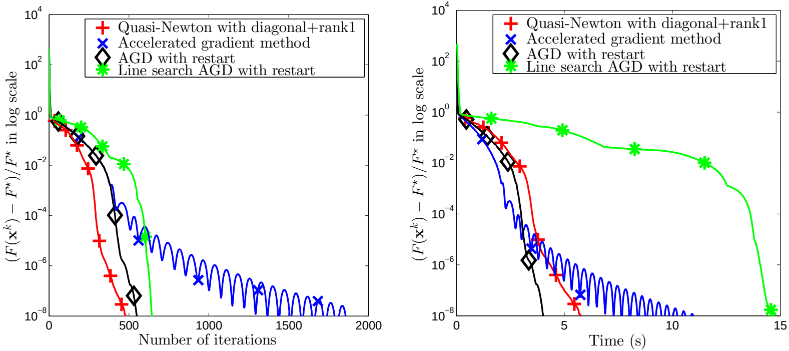
Example 2: ℓ_1-regularized least squares - Numerical results - Trial 1

Parameters: $n = 750, p = 2000, s = 200, \rho = 1$



Example 2: ℓ_1-regularized least squares - Numerical results - Trial 2

Parameters: $n = 750, p = 2000, s = 200, \rho = 1$



Outline

- Today
  1. Proximal Newton-type method.
  2. Stochastic proximal gradient method.
  3. Stochastic proximal gradient method with progressive variance reduction.
- Next week
  1. Constrained convex minimization I: The primal-dual approach.
  2. Smoothing approaches for non-smooth convex minimization.

Convex composite minimization

Problem (**Mathematical formulation**)

Consider the following constrained convex minimization problem:

$$F^\star = \min_{\mathbf{x}\in\mathbb{R}^p} \left\{ F(\mathbf{x}) := \mathbb{E}[h(\mathbf{x},\theta)] + g(\mathbf{x}) \right\}$$

- θ *is a random vector whose probability distribution is supported on set* Θ.
- *The solution set* $\mathcal{S}^\star := \{\mathbf{x}^\star \in$ *dom*$(F) : F(\mathbf{x}^\star) = F^\star\}$ *is nonempty.*
- $h(\mathbf{x},\theta) \in \mathcal{F}_{L_\theta}^{1,1}(\mathbb{R}^p)$ *and* $f = \mathbb{E}[h(\mathbf{x},\theta)] \in \mathcal{F}_L^{1,1}(\mathbb{R}^p)$.
- $g \in \mathcal{F}_{\mathrm{prox}}(\mathbb{R}^p)$.

• When $g = 0$, the problem reduces to the one considered in Lecture 7. Every example in Lecture 7 can be extended accordingly to the above formulation.
• The goal of this lecture is to extend the methods in Lecture 7 for a non-zero g.

Stochastic proximal gradient method

Stochastic proximal gradient method (SPG)

1. Choose $\mathbf{x}^0 \in \mathbb{R}^p$ and $(\gamma_k)_{k\in\mathbb{N}} \in]0, +\infty[^{\mathbb{N}}$.
2. For $k = 0, 1, \ldots$ perform:

$$\mathbf{x}^{k+1} = \mathrm{prox}_{\gamma_k g}(\mathbf{x}^k - \gamma_k G(\mathbf{x}^k, \theta_k)).$$

- We assume that $\{\theta_k\}$ are jointly independent and $G(\mathbf{x}^k, \theta_k)$ is an unbiased estimate of the full gradient, i.e., it satisfies

$$\mathbb{E}[G(\mathbf{x}^k, \theta_k)] = \nabla f(\mathbf{x}^k).$$

- We obtain the SG method in Lecture 7 when $g = 0$.

Stochastic proximal gradient method

Stochastic proximal gradient method (SPG)

1. Choose $\mathbf{x}^0 \in \mathbb{R}^p$ and $(\gamma_k)_{k\in\mathbb{N}} \in]0, +\infty[^{\mathbb{N}}$.
2. For $k = 0, 1, \ldots$ perform:

$$\mathbf{x}^{k+1} = \mathrm{prox}_{\gamma_k g}(\mathbf{x}^k - \gamma_k G(\mathbf{x}^k, \theta_k)).$$

- We assume that $\{\theta_k\}$ are jointly independent and $G(\mathbf{x}^k, \theta_k)$ is an unbiased estimate of the full gradient, i.e., it satisfies

$$\mathbb{E}[G(\mathbf{x}^k, \theta_k)] = \nabla f(\mathbf{x}^k).$$

- We obtain the SG method in Lecture 7 when $g = 0$.

Remark

- SPG shares the same structure as the proximal gradient method, but the gradient is replaced by an unbiased estimate in the 2nd step. The cost of computing this estimate is typically much cheaper than that of $\nabla f(\mathbf{x}^k)$.
- As $G(\mathbf{x}^k, \theta_k)$ is an unbiased estimate of the full gradient, we expect that SPG would also perform well.

Convergence analysis

Assumption A4.

(i) The variance is bounded: $\mathbb{E}_\theta[\|G(\mathbf{x},\theta) - \nabla f(x)\|^2] \le \sigma^2$

(ii) The step size $(\gamma_k)_{k\in\mathbb{N}} \in \ell^2(\mathbb{N})\backslash\ell^1(\mathbb{N})$, i.e.,

$$\sum_{k=0}^{\infty} \gamma_k = \infty \text{ and } \sum_{k=0}^{\infty} \gamma_k^2 < +\infty.$$

Theorem (Ergodic convergence)

Assumptions:

- *The sequence $\{\mathbf{x}^k\}_{k\ge 0}$ is generated by SPG.*
- *Assumption A4. is satisfied and the set of solutions is non-empty.*

Conclusion:

- *Define $\hat{\mathbf{x}}^s = \left(\sum_{k=0}^s \gamma_k \mathbf{x}^k\right) / \sum_{k=0}^s \gamma_k$, then*

$$\mathbb{E}F(\hat{\mathbf{x}}^s) - F(\mathbf{x}^\star) \le \left(0.5\|\mathbf{x}^0 - \mathbf{x}^\star\|^2 + \sigma^2 \sum_{k=0}^{\infty} \gamma_k^2\right) / \sum_{k=0}^{s} \gamma_k.$$

Convergence analysis

Assumption A4.

(i) The variance is bounded: $\mathbb{E}_\theta[\|G(\mathbf{x},\theta) - \nabla f(x)\|^2] \leq \sigma^2$

(ii) The step size $(\gamma_k)_{k\in\mathbb{N}} \in \ell^2(\mathbb{N})\backslash\ell^1(\mathbb{N})$, i.e.,

$$\sum_{k=0}^{\infty} \gamma_k = \infty \text{ and } \sum_{k=0}^{\infty} \gamma_k^2 < +\infty.$$

Theorem (Non-ergodic convergence [14])

Assumptions:

- *The sequence $\{\mathbf{x}^k\}_{k\geq 0}$ is generated by SPG.*
- *Assumption A4(i). is satisfied and $\gamma_k \sim 1/(k+1)$.*
- *f is μ-strongly convex.*

Conclusion:

- *$1/k$ rate is obtained:*

$$\mathbb{E}\|\mathbf{x}^k - \mathbf{x}^\star\|^2 = \mathcal{O}(1/k).$$

- *If F is R-smooth, i.e. $F(\mathbf{x}) - F(\mathbf{x}^\star) \leq R\|\mathbf{x} - \mathbf{x}^\star\|^2$, then*

$$\mathbb{E}F(\mathbf{x}^k) - F(\mathbf{x}^\star) = \mathcal{O}(1/k).$$

Composite optimization with finite sums

Composite optimization with finite sums

$$F^\star := \min_{\mathbf{x}\in\mathrm{dom}(F)} \left\{ F(\mathbf{x}) := \frac{1}{m}\sum_{k=1}^{m} f_k(\mathbf{x}) + g(\mathbf{x}) \right\}, \tag{12}$$

- $f_k \in \mathcal{F}^{1,1}_{L_k}(\mathbb{R}^p)$ and $f = \frac{1}{k}\sum_{k=1}^m f_k \in \mathcal{F}^{1,1}_{L}(\mathbb{R}^p)$.
- $g \in \mathcal{F}_{\mathrm{prox}}(\mathbb{R}^p)$.

Why is stochastic minimization?

- $f(\mathbf{x}) = \mathbb{E}_j f_j(\mathbf{x})$ where $\mathbb{P}(j = k) = 1/m$.
- Computation $\nabla f(x) = \frac{1}{m}\sum_{k=1}^m \nabla f_k(\mathbf{x})$ is expensive when $m \ggg 1$.
- Covers many well-known examples in maching learning, portfolio optimization, SVM.

Large scale problems

Definition (Recall)

- ∇f_j is called thestochastic gradient of f, and it is unbiased estimate, i.e.

$$\mathbb{E}_j \nabla f_j(\mathbf{x}) = \sum_{i=1}^{m} \mathbb{P}(j = k) \nabla f_k(\mathbf{x}) = \frac{1}{m} \sum_{i=1}^{m} \nabla f_k(\mathbf{x}) = \nabla f(\mathbf{x})$$

- $\mathbb{E}_j \|\nabla f_j(\mathbf{x}) - \nabla f(\mathbf{x})\|^2$ is called variance.

Example

Define $f(\mathbf{x}) = \frac{1}{2m}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2$ with $\mathbf{b} \in \mathbb{R}^m$. To find stochastic gradient, observe:

$$f(\mathbf{x}) = \frac{1}{2m} \sum_{k=1}^{m} |\mathbf{a}_k^T \mathbf{x} - \mathbf{b}_k|^2$$

Thus,

$$f_j(\mathbf{x}) = \frac{1}{2}|\mathbf{a}_j^T \mathbf{x} - \mathbf{b}_j|^2 \text{ with } \nabla f_j(\mathbf{x}) = (\mathbf{a}_j^T \mathbf{x} - \mathbf{b}_j)\mathbf{a}_j$$

Large scale problems

Definition (Recall)

- ∇f_j is called thestochastic gradient of f, and it is unbiased estimate, i.e.

$$\mathbb{E}_j \nabla f_j(\mathbf{x}) = \sum_{i=1}^{m} \mathbb{P}(j = k) \nabla f_k(\mathbf{x}) = \frac{1}{m} \sum_{i=1}^{m} \nabla f_k(\mathbf{x}) = \nabla f(\mathbf{x})$$

- $\mathbb{E}_j \|\nabla f_j(\mathbf{x}) - \nabla f(\mathbf{x})\|^2$ is called variance.

Example

Similarly, one can find stochastic gradient of

1. $f(\mathbf{x}) = \frac{1}{m} \sum_{i=1}^{m} \log(1 + \exp(-\mathbf{b}_i \mathbf{a}_i^T \mathbf{x}))$ where $\mathbf{a}_i \in \mathbb{R}^p, \mathbf{b}_i = \pm 1$.
2. $f(\mathbf{x}) = \frac{1}{2} \mathbf{x}^T \mathbf{Q} \mathbf{x}$ where $\mathbf{Q}$ is positive semidefinite matrix.
3. $f(\mathbf{x}) = \frac{1}{m} \sum_{i=1}^{m} (\mathbf{a}_i^T \mathbf{x} - \bar{\mathbf{b}})^2$ where $\mathbf{a}_i \in \mathbb{R}^p$, $\bar{\mathbf{b}} \in \mathbb{R}$.

Stochastic proximal gradient algorithm for the finite sum problem

Stochastic proximal gradient algorithm (SPG) [4, 14]

1. Choose $\mathbf{x}^0 \in \mathbb{R}^p$ as a starting point and $\gamma_0 > 0$.
2. For $k = 0, 1, \cdots,$ perform:

$$\begin{cases} \text{Pick } i_k \in \{1, \ldots, m\} \text{ uniformly at random} \\ \mathbf{x}^{k+1} := \mathrm{prox}_{\gamma_k g}\left(\mathbf{x}^k - \gamma_k \nabla f_{i_k}(\mathbf{x}^k)\right), \end{cases} \tag{13}$$

where $\gamma_k \in (0, 1/L]$ is a given step size aka learning rate

Common features

- SPG shares the same structure as PG (proximal gradient) where the full gradient is replaced by stochastic gradient. Thus, SPG only evaluates gradient of a single component function.
- The computational cost per iteration is only $1/m$ that of the PG method.
- $L \leq m^{-1} \sum_{k=1}^m L_k \leq L_{\max} = \max_{1 \leq k \leq m} L_k$.
- To ensure convergnce, the step size is often chosen such that $(\gamma_k)_{k \in \mathbb{N}} \in \ell^2(\mathbb{N}) \backslash \ell^1(\mathbb{N})$.

Stochastic proximal gradient algorithm for the finite sum problem

Stochastic proximal gradient algorithm (SPG) [4, 14]

1. Choose $\mathbf{x}^0 \in \mathbb{R}^p$ as a starting point and $\gamma_0 > 0$.
2. For $k = 0, 1, \cdots,$ perform:

$$\begin{cases} \text{Pick } i_k \in \{1, \ldots, m\} \text{ uniformly at random} \\ \mathbf{x}^{k+1} := \mathrm{prox}_{\gamma_k g}\left(\mathbf{x}^k - \gamma_k \nabla f_{i_k}(\mathbf{x}^k)\right), \end{cases} \tag{14}$$

where $\gamma_k \in (0, 1/L]$ is a given step size aka learning rate

Complexity per iteration: One gradient and prox

- Evaluation of $\nabla f_{i_k}(\mathbf{x}^k)$ is much cheaper than $\nabla f(\mathbf{x}^k)$.
- Closed-form expressions for $\mathrm{prox}_{\gamma_k g}$ are given in previous lecture for several g.

Convergence analysis

Assumption A4.

(i) The variance is bounded: $\mathbb{E}_j[\|\nabla f_j(x) - \nabla f(x)\|^2] \leq \sigma^2$

(ii) The step size $(\gamma_k)_{k\in\mathbb{N}} \in \ell^2(\mathbb{N})\backslash\ell^1(\mathbb{N})$, i.e.,

$$\sum_{k=0}^{\infty} \gamma_k = \infty \text{ and } \sum_{k=0}^{\infty} \gamma_k^2 < +\infty.$$

Theorem (Ergodic convergence)

Assumptions:

- *The sequence $\{\mathbf{x}^k\}_{k\geq 0}$ is generated by SPG.*
- *Assumption A4. is satisfied and the set of solutions is non-empty.*

Conclusion:

- *Define $\hat{\mathbf{x}}^s = \left(\sum_{k=0}^{s} \gamma_k \mathbf{x}^k\right) / \sum_{k=0}^{s} \gamma_k$, then*

$$\mathbb{E}F(\hat{\mathbf{x}}^s) - F(\mathbf{x}^\star) \leq \left(0.5\|\mathbf{x}^0 - \mathbf{x}^\star\|^2 + \sigma^2 \sum_{k=0}^{\infty} \gamma_k^2\right) / \sum_{k=0}^{s} \gamma_k.$$

Convergence analysis

Assumption A4.

(i) The variance is bounded: $\mathbb{E}_j[\|\nabla f_j(x) - \nabla f(x)\|^2] \leq \sigma^2$

(ii) The step size $(\gamma_k)_{k\in\mathbb{N}} \in \ell^2(\mathbb{N})\backslash\ell^1(\mathbb{N})$, i.e.,

$$\sum_{k=0}^{\infty} \gamma_k = \infty \text{ and } \sum_{k=0}^{\infty} \gamma_k^2 < +\infty.$$

Theorem (Non-ergodic convergence [14])

Assumptions:

- *The sequence $\{\mathbf{x}^k\}_{k\geq 0}$ is generated by SPG.*
- *Assumption A4(i). is satisfied and $\gamma_k \sim 1/(k+1)$.*
- *f is μ-strongly convex.*

Conclusion:

- *$1/k$ rate is obtained:*
$$\mathbb{E}\|\mathbf{x}^k - \mathbf{x}^\star\|^2 = \mathcal{O}(1/k).$$
- *If F is R-smooth, i.e. $F(\mathbf{x}) - F(\mathbf{x}^\star) \leq R\|\mathbf{x} - \mathbf{x}^\star\|^2$, then*
$$\mathbb{E}F(\mathbf{x}^k) - F(\mathbf{x}^\star) = \mathcal{O}(1/k).$$

Comparions

- SPG vs PG

Algorithm	$\gamma_k \to 0$	Strong convexity	Convergence	Rate	# grad/Iter.
PG	No	No	Ergodic	1/k	m
SPG	Yes	No	Ergodic	$1/\sum_{j=0}^{k-1}\gamma_j$	1
PG	No	Yes	Non-Ergodic	Linear	m
SPG	Yes	Yes	Non-Ergodic	$1/k$	1

  - PG= Proximal gradient, aka Forward-backward.
  - SPG= Stochastic Proximal gradient.

Comparions

- SPG vs PG

Algorithm	$\gamma_k \to 0$	Strong convexity	Convergence	Rate	# grad/Iter.
PG	No	No	Ergodic	1/k	m
SPG	Yes	No	Ergodic	$1/\sum_{j=0}^{k-1}\gamma_j$	1
PG	No	Yes	Non-Ergodic	Linear	m
SPG	Yes	Yes	Non-Ergodic	$1/k$	1

  - PG= Proximal gradient, aka Forward-backward.
  - SPG= Stochastic Proximal gradient.

- **Advantages and disadvantages of SPG**
  - Complexity per iteration is very low and hence SPG is suitable for large problems such as SVM, logistic regression.
  - The convergence rate is slower than PG and hence SPG is suitable when we require only low accurate of solution.

Conclusion: SPG $\longrightarrow$ Large scale problems + low accurate of solution

  - The bounded variance condition is standard but it is hard to evaluate and hence the learning rate need to go to 0 to cancel its growth.

Soltuion: Use the variance reduction technique (see Prox-SVRG)

Example: ℓ_1-regularized least squares revisited

Problem (ℓ_1-regularized least squares)

Given $\mathbf{A} \in \mathbb{R}^{m \times p}$ *and* $\mathbf{b} \in \mathbb{R}^m$, *solve:*

$$F^\star := \min_{\mathbf{x} \in \mathbb{R}^p} \left\{ F(\mathbf{x}) := \frac{1}{2m} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 + \rho \|\mathbf{x}\|_1 \right\}, \tag{15}$$

where $\rho > 0$ *is a regularization parameter.*

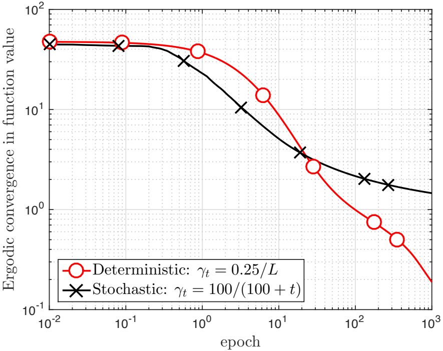
Complexity per iterations

- Evaluating $\nabla f_j(\mathbf{x}^k) = (a_j^T \mathbf{x}^k - \mathbf{b}) a_j$ requires one $a_j^T \mathbf{x}$ and one λa_j.
- One soft-thresholding operator $\mathrm{prox}_{\lambda g}(\mathbf{x}) = \mathrm{sign}(\mathbf{x}) \otimes \max\{|\mathbf{x}| - \kappa, 0\}$.
- **Optional**: Evaluating $L_{\max} = \max_{1 \leq k \leq m} \|a_k\|^2$ - via **iterations**.

Synthetic data generation

- $\mathbf{A} := \mathrm{randn}(m, p)$ - standard Gaussian $\mathcal{N}(0, \mathbb{I})$.
- $\mathbf{x}^\star$ is a sparse vector generated randomly.
- $\mathbf{b} := \mathbf{A}\mathbf{x}^\star + \mathcal{N}(0, 10^{-3})$.

Example: ℓ_1-regularized least squares revisited- Numerical test



*Accelerated SPG I

Accelerated SPG (AccSPG)

```
0.  $0 \leq \mu$-strong convexity of $F$.
1.  Choose $\mathbf{y}^0 = \mathbf{z}^0 = \mathbf{0}$, $(\gamma_k)_{k\in\mathbb{N}}$, $(\alpha_k)_{k\in\mathbb{N}} \in \,]0, +\infty[^{\mathbb{N}}$, $\alpha_0 = 1, \gamma_0 = L + \mu$.
2.  For $k = 0, 1, \ldots$ perform:
2a. $\mathbf{x}^{k+1} = (1 - \alpha_k)\mathbf{y}^k + \alpha_k \mathbf{z}^k$.
2b. $\mathbf{y}^{k+1} = \mathrm{prox}_{g/\gamma_k}\left(\mathbf{x}^{k+1} - \frac{1}{\gamma_k} G(\mathbf{x}^{k+1}, \theta_k)\right)$.
2c. $\mathbf{z}^{k+1} = \mathbf{z}^k - \frac{1}{\gamma_k \alpha_k + \mu}\left(\gamma_k(\mathbf{x}^{k+1} - \mathbf{y}^{k+1}) + \mu(\mathbf{z}^k - \mathbf{x}^{k+1})\right)$ .
```

*Accelerated SPG I

Theorem (Convergence of AccSPG with strong convexity [15])

Define $\lambda_k = \prod_{j=1}^{k}(1-\alpha_j)$ *and* $\lambda_0 = 1.$ *Le:t*

1. f *is* μ*-strongly convex,*
2. $\mathbb{E}[\|\mathbf{z}^k - \mathbf{x}^\star\|^2] \leq D^2$*,*
3. $\mathbb{E}[\|G(\mathbf{x}^k, \theta_k) - \nabla f(\mathbf{x}^k)\|^2] \leq M^2$ *.*
4. $\gamma_k = L + \frac{\mu}{\lambda_{k-1}}$ *and* $\alpha_k = \sqrt{\lambda_{k-1} + \frac{\lambda_{k-1}^2}{4}} - \frac{\lambda_{k-1}}{2}.$

Then,

$$\mathbb{E}[f(\mathbf{y}^{k+1}) - f(\mathbf{x}^\star)] \leq \frac{2(L+\mu)D^2}{k^2} + \frac{6M^2}{\mu k}.$$

The accelerated technique can be used to reduce the error term related to $\mathbb{E}[\|\mathbf{z}^k - \mathbf{x}^\star\|^2]$.

*Accelerated SPG II

Accelerated SPG (AccSPG)

```
1. Choose $\mathbf{y}^0 = \mathbf{z}^0 = \mathbf{0}$, $(\gamma_k)_{k\in\mathbb{N}}$, $(\alpha_k)_{k\in\mathbb{N}} \in ]0, +\infty[^{\mathbb{N}}$, $\alpha_0 = 1, \gamma_0 = L$.
2. For $k = 0, 1, \ldots$ perform:
2a. $\mathbf{x}^{k+1} = (1-\alpha_k)\mathbf{y}^k + \alpha_k \mathbf{z}^k$.
2b. $\mathbf{y}^{k+1} = \mathrm{prox}_{g/\gamma_k}\left(\mathbf{x}^{k+1} - \frac{1}{\gamma_k} G(\mathbf{x}^{k+1}, \theta_k)\right)$.
2c. $\mathbf{z}^{k+1} = \mathbf{z}^k - \frac{1}{\alpha_k}(\mathbf{x}^{k+1} - \mathbf{y}^{k+1})$ .
```

Theorem (Convergence of AccSPG without strong convexity [15])

Let:

1. $\mathbb{E}[\|\mathbf{z}^k - \mathbf{x}^\star\|^2] \leq D^2$,
2. $\mathbb{E}[\|G(\mathbf{x}^k, \theta_k) - \nabla f(\mathbf{x}^k)\|^2] \leq M^2$,
3. $\gamma_k = c(k+1)^{3/2} + L$ *for a fixed* $c > 0$, *and* $\alpha_k = 2/(k+2)$.

Then,

$$\mathbb{E}[f(\mathbf{y}^{k+1}) - f(\mathbf{x}^\star)] \leq \frac{3D^2L}{k^2} + \left(3D^2c + \frac{5M^2}{3c}\right)\frac{1}{\sqrt{k}}.$$

Stochastic proximal gradient algorithm with variance reduction (Prox-SVRG) [13]

Variance reduction

Lemma

Given $\widetilde{\mathbf{x}} \in \mathbb{R}^p$ *and* $\mathbb{P}(j = k) = 1/m$, *define*

$$\mathbf{r}^k = \nabla f_j(\mathbf{x}^k) - \nabla f_j(\widetilde{\mathbf{x}}) + \nabla f(\widetilde{\mathbf{x}}).$$

Then, conditioned on $\mathbf{x}^k$, *we have* $\mathbb{E}_j \mathbf{r}^k = \nabla f(\mathbf{x}^k)$ *and*

$$\mathbb{E}_j \|\mathbf{r}^k - \nabla f(\mathbf{x}^k)\|^2 \leq 4L_{\max}\Big(F(\mathbf{x}^k) - F(\mathbf{x}^\star) + F(\widetilde{\mathbf{x}}) - F(\mathbf{x}^\star)\Big).$$

Main idea of Prox-SVRG

- Step 1: Maintain an estimate $\widetilde{\mathbf{x}}$ of $\mathbf{x}^\star$ by updating $\widetilde{\mathbf{x}}$ periodically (after n-iteration of SPG).
- Step 2: When $\widetilde{\mathbf{x}}$ is updated, we also compute the full gradient:

$$\nabla f(\widetilde{\mathbf{x}}) = \frac{1}{n}\sum_{k=1}^{n} \nabla f_k(\widetilde{\mathbf{x}}).$$

- Step 3: Repeats Steps 1&2.

Stochastic proximal gradient algorithm with variance reduction

Stochastic proximal gradient algorithm with variance reduction (Prox-SVRG) [13]

```
1. Choose $\widetilde{\mathbf{x}}^0 \in \mathbb{R}^p$ as a starting point and $\gamma > 0$ and $n \in \mathbb{N}_+$.
2. For $s = 0, 1, 2 \cdots$, perform:
    2a. $\widetilde{\mathbf{x}} = \widetilde{\mathbf{x}}^s, \quad \widetilde{\mathbf{v}} = \nabla f(\widetilde{\mathbf{x}}), \quad \mathbf{x}_0 = \widetilde{\mathbf{x}}.$
    2b. For $k = 0, 1, \cdots n-1$, perform:
        $\begin{cases} \text{Pick } i_k \in \{1, \ldots, m\} \text{ uniformly at random} \\ \mathbf{r}_k = \nabla f_{i_k}(\mathbf{x}_k) - \nabla f_{i_k}(\widetilde{\mathbf{x}}) + \widetilde{\mathbf{v}} \\ \mathbf{x}^{k+1} := \mathrm{prox}_{\gamma g}\left(\mathbf{x}^k - \gamma \mathbf{r}_k\right), \end{cases}$    (16)
    2c. Update $\widetilde{\mathbf{x}}^s = \frac{1}{n}\sum_{j=0}^{n-1} \mathbf{x}^j$.
```

Common features

- The Prox-SVRG method uses a multistage scheme to reduce the variance of the stochastic gradient $\mathbf{r}_k$ where $\mathbf{x}_k$ and $\widetilde{\mathbf{x}}^s$ tend to $\mathbf{x}_\star$.
- Learning rate is not necessary tend to 0.
- Each stage, Prox-SVRG uses $m + 2n$ component gradient evaluations: m for the full gradient at the beginning of each stage, and $2n$ for each of the n proximal stochastic gradient steps.

Convergence analysis

Assumption A5.

(i) $F = f + g$ is μ-strongly convex

(ii) The learning rate $0 < \gamma < 1/(4L_{\max})$.

(iii) n is large enough such that

$$\kappa = \frac{1}{\mu\gamma(1 - 4\gamma L_{\max})n} + \frac{4\gamma L_{\max}(n+1)}{(1 - 4\gamma L_{\max})n} < 1.$$

Theorem

Assumptions:

- *The sequence $\{\widetilde{\mathbf{x}^s}\}_{k\geq 0}$ is generated by Prox-SVRG.*
- *Assumption A5. is satisfied.*

Conclusion: *Linear convergence is obtained:*

$$\mathbb{E}F(\widetilde{\mathbf{x}^s}) - F(\mathbf{x}^\star) \leq \kappa^s(F(\widetilde{\mathbf{x}^0}) - F(\mathbf{x}^\star)).$$

Convergence analysis

Assumption A5.

(i) $F = f + g$ is μ-strongly convex

(ii) The learning rate $0 < \gamma < 1/(4L_{\max})$.

(iii) n is large enough such that

$$\kappa = \frac{1}{\mu\gamma(1-4\gamma L_{\max})n} + \frac{4\gamma L_{\max}(n+1)}{(1-4\gamma L_{\max})n} < 1.$$

Theorem

Assumptions:

- *The sequence $\{\mathbf{x}^k\}_{k\geq 0}$ is generated by Prox-SVRG.*
- *Assumption A5. is satisfied.*

Conclusion: *For any $\epsilon > 0$ and $\delta \in (0,1)$: $P(F(\widetilde{\mathbf{x}^s}) - F(\mathbf{x}^\star) \leq \epsilon) \geq 1-\delta$ whenever*

$$s \geq \log\left(\frac{F(\widetilde{\mathbf{x}^0}) - F(\mathbf{x}^\star)}{\delta\epsilon}\right) / \log(1/\kappa)$$

Choice of γ and n, and complexity

Chose γ and n such that $\kappa \in (0,1)$:

For example

$$\gamma = 0.1/L\max, n = 100(L_{\max}/\mu) \Longrightarrow \kappa \approx 5/6.$$

Complexity

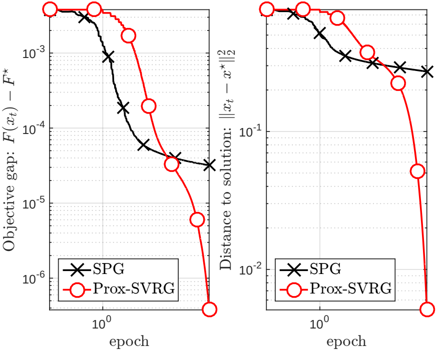
$$\mathbb{E}F(\widetilde{\mathbf{x}^s}) - F(\mathbf{x}^\star) \leq \varepsilon \; when \; s \geq \log(\kappa^{-1}) \log((F(\widetilde{\mathbf{x}^0}) - F(\mathbf{x}^\star))/\epsilon)$$

Since at each stage needs $m + 2n$ component gradient evaluations, with $n = \mathcal{O}(L_{\max}/\mu)$, we get the overall complexity is

$$\mathcal{O}\Big((m + L_{\max}/\mu) \log(1/\epsilon)\Big).$$

Example: ℓ_1-regularized least squares revisited- Numerical test

SPG: $\gamma_t = 100/(100+t)$ and Prox-SVRG: $\gamma = 0.1/L_{\max}$



*Variance reduction techniques: SAGA

Stochastic Average Gradient (SAGA) [16]

1a. Choose $\tilde{\mathbf{x}}_i^0 = \mathbf{x}^0 \in \mathbb{R}^p, \forall i,\ 0 \neq q \in \mathbb{N}$ and stepsize $\gamma > 0$.

1b. Store $\nabla f_i(\tilde{\mathbf{x}}_i^0)$ in a table data-structure with length m.

2. For $k = 0, 1 \ldots$ perform:

2a. pick $i_k \in \{1, \ldots, m\}$ uniformly at random

2b. Take $\tilde{\mathbf{x}}_{i_k}^{k+1} = \mathbf{x}^k$, store $\nabla f_{i_k}(\tilde{\mathbf{x}}_{i_k}^{k+1})$ in the table and leave other entries the same.

2c. $v_k = \nabla f_{i_k}(\mathbf{x}^k) - \nabla f_{i_k}(\tilde{\mathbf{x}}_{i_k}^k) + \frac{1}{m}\sum_{i=1}^m \nabla f_i(\tilde{\mathbf{x}}_i^k)$

3. $\mathbf{x}^{k+1} = \mathrm{prox}_{\gamma g}(\mathbf{x}^k - \gamma v_k)$

Recipe:

In a cycle of q iterations:

- Store last gradient evaluated at each datapoint.
- Previous gradient for datapoint j is $\nabla f_j(\tilde{\mathbf{x}}_j^k)$.
- Perform q SG-iterations with the following stochastic gradient

$$v_k = \nabla f_{i_k}(\mathbf{x}^k) - \nabla f_{i_k}(\tilde{\mathbf{x}}_{i_k}^k) + \sum_{i=1}^n \nabla f_i(\tilde{\mathbf{x}}_i^k).$$

*Convergence of SAGA

$$\boxed{F^\star := \min_{\mathbf{x}\in\mathbb{R}^p}\left\{(f(\mathbf{x}) + g(\mathbf{x})) := \frac{1}{m}\sum_{i=1}^{m} f_i(\mathbf{x}) + g(\mathbf{x})\right\}.}$$

Theorem (Convergence of SAGA [16])

Set $L_{max} = \max_{1\leq i\leq m} L_i$, where L_i is Lipschitz constant of ∇f_i. Suppose that f is μ-strongly convex and that the stepsize is $\gamma = \frac{1}{2(\mu n+L)}$ with

$$\rho = 1 - \frac{\mu}{2(\mu n + L)} < 1,$$

$$C = \|\mathbf{x}^0 - \mathbf{x}^\star\|^2 + \frac{n}{\mu n + L}[f(\mathbf{x}^0) - \langle\nabla f(\mathbf{x}^\star), \mathbf{x}^0 - \mathbf{x}^\star\rangle - f(\mathbf{x}^\star)]$$

Then

$$\mathbb{E}[\|\mathbf{x}^k - \mathbf{x}^\star\|^2] \leq \rho^k C.$$

- Allows the constant step-size.
- Obtains linear rate convergence.

Recall that in Lecture 7 we mentioned SARAH. However, how to extend SARAH to composite optimization is open.

References I

[1] A. Beck and M. Teboulle.
A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems.
SIAM J. Imaging Sciences, 2(1):183–202, 2009.

[2] S. Becker and M.J. Fadili.
A quasi-Newton proximal splitting method.
In *Proceedings of Neutral Information Processing Systems Foundation*, 2012.

[3] P. Combettes and Pesquet J.-C.
Signal recovery by proximal forward-backward splitting.
In *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pages 185–212. Springer-Verlag, 2011.

[4] J, Duchi and Y. Singer,
Efficient Online and Batch Learning Using Forward Backward Splitting
J. Mach. Learn. Res., (10), 2899-2934, 2009.

[5] O. Güler.
On the convergence of the proximal point algorithm for convex minimization.
SIAM J. Control Optim., 29(2):403–419, 1991.

[6] J.D. Lee, Y. Sun, and M.A. Saunders.
Proximal newton-type methods for convex optimization.
Tech. Report., pages 1–25, 2012.

References II

[7] Y. Nesterov.

Introductory lectures on convex optimization: a basic course, volume 87 of *Applied Optimization*.

Kluwer Academic Publishers, 2004.

[8] Y. Nesterov and A. Nemirovski.

Interior-point Polynomial Algorithms in Convex Programming.

Society for Industrial Mathematics, 1994.

[9] N. Parikh and S. Boyd.

Proximal algorithms.

Foundations and Trends in Optimization, 1(3):123–231, 2013.

[10] R. T. Rockafellar.

Convex Analysis, volume 28 of *Princeton Mathematics Series*.

Princeton University Press, 1970.

[11] R.T. Rockafellar.

Monotone operators and the proximal point algorithm.

SIAM Journal on Control and Optimization, 14:877–898, 1976.

[12] Q. Tran-Dinh, A. Kyrillidis, and V. Cevher.

Composite self-concordant minimization.

J. Mach. Learn. Res., 16 (2015) 371-416.

References III

[13] L. Xiao and T. Zhang,
A Proximal Stochastic Gradient Method with Progressive Variance Reduction,
SIAM J. Optim., 24(4), 2057-2075, 2014.

[14] L. Rosacso, S. Villa, B. C. Vũ,
Stochastic Forward-Backward Splitting for Monotone Inclusions,
J. Optim. Theory Appl. (169), 388-406, 2016.

[15] J. T. Kwok, C. Hu and W. Pan.
Accelerated gradient methods for stochastic optimization and online learning.
Advances in Neural Information Processing Systems, vol. 22, pp. 781–789, 2009.

[16] A. Defazio, F. Bach, and S. Lacoste-Julien.
SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives.
Advances in Neural Information Processing Systems, pp. 1646–1654, 2014.