# Mathematics of Data: From Theory to Computation

Prof. Volkan Cevher
*volkan.cevher@epfl.ch*

*Lecture 9: Composite minimization II*

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

**EE-556** (Fall 2019)

# License Information for Mathematics of Data Slides

# Outline

- Today
  1. Proximal gradient method - Nonconvex case
  2. Proximal-Newton method
  3. Stochastic proximal gradient method.
  4. Stochastic proximal gradient method with progressive variance reduction.
- Next week
  1. Constrained convex minimization I: The primal-dual approach.

# Recommended reading material


- A. Beck, First-order methods in optimization, SIAM, 2017.
- 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 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 nonconvex minimization

### Problem (Unconstrained composite minimization)

*We study following composite nonconvex minimization problem,*

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

*where*

- $g$: $\mathbb{R}^p \to \mathbb{R} \cup \{\infty\}$ *is proper, closed, convex, and (possibly) nonsmooth.*
- $f$: $\mathbb{R}^p \to \mathbb{R}$ *is proper, closed, nonconvex, dom*$(f)$ *is convex, and* $f \in \mathcal{F}^{1,1}_{L_f}$.

- Example: Neural network pruning with $\ell_1$-norm regularization

# Recall: Proximal-gradient algorithm

| **Basic proximal-gradient scheme** |
|---|
| **1.** Choose $\mathbf{x}^0 \in \mathrm{dom}\,(F)$ arbitrarily as a starting point. <br> **2.** For $k = 0, 1, \cdots$, generate a sequence $\{\mathbf{x}^k\}_{k\geq 0}$ as: <br> $$\mathbf{x}^{k+1} := \mathrm{prox}_{\alpha g}\left(\mathbf{x}^k - \alpha\nabla f(\mathbf{x}^k)\right),$$ |

# Recall: Proximal-gradient algorithm

| Basic proximal-gradient scheme |
| --- |
| **1.** Choose $\mathbf{x}^0 \in \mathrm{dom}\,(F)$ arbitrarily as a starting point.<br>**2.** For $k = 0, 1, \cdots$, generate a sequence $\{\mathbf{x}^k\}_{k \geq 0}$ as:<br>$$\mathbf{x}^{k+1} := \mathrm{prox}_{\alpha g}\left(\mathbf{x}^k - \alpha \nabla f(\mathbf{x}^k)\right),$$ |

Next:

▷ We analyze the same algorithm as in lecture $8$, but with nonconvex objective.

▷ Guarantees: To the first order stationary point of the composite problem.

▷ Metric for the convergence: Gradient Mapping.

## Sufficient decrease property

$$\boxed{\mathbf{x}^{k+1} := \text{prox}_{\alpha g}\left(\mathbf{x}^k - \alpha\nabla f(\mathbf{x}^k)\right)}$$

### Assumption

*$f$ is smooth with parameter $L$, i.e., $f \in \mathcal{F}_L^{1,1}(\mathbb{R}^p)$.*
*$g$: $\mathbb{R}^p \to \mathbb{R} \cup \{+\infty\}$ is proper, closed, convex, and (possibly) nonsmooth. $g$ is proximally tractable.*

### Lemma (**Sufficient decrease** )

$$F(\mathbf{x}^{k+1}) \leq F(\mathbf{x}^k) - \alpha\left(1 - \frac{\alpha L}{2}\right)\left\|\frac{1}{\alpha}(\mathbf{x}^k - \mathbf{x}^{k+1})\right\|_2^2. \tag{1}$$

▷ *Note:* $\frac{1}{\alpha}(\mathbf{x}^k - \mathbf{x}^{k+1}) = \frac{1}{\alpha}\left(\mathbf{x}^k - \text{prox}_{\alpha g}\left(\mathbf{x}^k - \alpha\nabla f(\mathbf{x}^k)\right)\right)$

### Corollary

$$F(\mathbf{x}^{k+1}) \leq F(\mathbf{x}^k) - \frac{1}{2L}\left\|L_f(\mathbf{x}^k - \mathbf{x}^{k+1})\right\|_2^2, \qquad \text{for } \alpha = 1/L$$

# A metric for $1$st order stationarity: Gradient mapping

## Definition (Gradient Mapping)

$$G_\alpha(\mathbf{x}^k) := \frac{1}{\alpha}(\mathbf{x}^k - \mathbf{x}^{k+1}) = \frac{1}{\alpha}\left(\mathbf{x}^k - \mathrm{prox}_{\alpha g}\left(\mathbf{x}^k - \alpha \nabla f(\mathbf{x}^k)\right)\right)$$

# A metric for $1$st order stationarity: Gradient mapping

## Definition (Gradient Mapping)

$$G_\alpha(\mathbf{x}^k) := \frac{1}{\alpha}(\mathbf{x}^k - \mathbf{x}^{k+1}) = \frac{1}{\alpha}\left(\mathbf{x}^k - \mathrm{prox}_{\alpha g}\left(\mathbf{x}^k - \alpha \nabla f(\mathbf{x}^k)\right)\right)$$

**Remarks:**

- $G_\alpha(\mathbf{x}) = \mathbf{0} \iff \mathbf{x}$ is a stationary point.
- If $g = 0 \quad \Rightarrow \quad G_\alpha(\mathbf{x}) = \nabla f(\mathbf{x}) \Rightarrow$ Reduces to same metric for gradient descent.

# A metric for $1$st order stationarity: Gradient mapping

## Definition (Gradient Mapping)

$$G_\alpha(\mathbf{x}^k) := \frac{1}{\alpha}(\mathbf{x}^k - \mathbf{x}^{k+1}) = \frac{1}{\alpha}\left(\mathbf{x}^k - \mathrm{prox}_{\alpha g}\left(\mathbf{x}^k - \alpha\nabla f(\mathbf{x}^k)\right)\right)$$

**Remarks:**

- $G_\alpha(\mathbf{x}) = \mathbf{0} \iff \mathbf{x}$ is a stationary point.
- If $g = 0 \quad \Rightarrow \quad G_\alpha(\mathbf{x}) = \nabla f(\mathbf{x}) \Rightarrow$ Reduces to same metric for gradient descent.

## Lipschitz continuity

$$\|G_\alpha(\mathbf{x}) - G_\alpha(\mathbf{y})\| \leq \left(\frac{2}{\alpha} + L\right)\|\mathbf{x} - \mathbf{y}\|, \qquad \forall \mathbf{x}, \mathbf{y} \in \mathrm{dom}(F)$$

$$\left\|G_{\frac{1}{L}}(\mathbf{x}) - G_{\frac{1}{L}}(\mathbf{y})\right\| \leq 3L\|\mathbf{x} - \mathbf{y}\|, \qquad \forall \mathbf{x}, \mathbf{y} \in \mathrm{dom}(F)$$

## Monotonicity

$\alpha_1 \geq \alpha_2 > 0$,

$$\|G_{\alpha_1}(\mathbf{x})\| \geq \|G_{\alpha_2}(\mathbf{x})\|, \qquad \forall \mathbf{x} \in \mathrm{dom}(F)$$

## Choosing the step-size

### Step size selection

**Constant.** $\alpha_k = \alpha \in \left(0, \frac{2}{L}\right]$

**Backtracking line search for estimating $L_f$:** Find the smallest integer $i_k$ which satisfies the sufficient decrease property for the set of parameters $(\gamma, \eta)$, where $\gamma \in (0,1)$, and $\eta > 1$, i.e.,

$$F(\mathbf{x}^{k+1}) \leq F(\mathbf{x}^k) - \frac{\gamma}{L_k} \|G_{\alpha_k}(\mathbf{x}^k)\|_2^2, \quad \text{where } \alpha_k = \frac{1}{L_k} = \frac{1}{L_{k-1}\eta^{i_k}}.$$

### Remarks

Line search procedure terminates in finite step under the given assumptions. See (1).

$$\text{If } \hat{L} \geq \frac{L}{2(1-\gamma)} \quad \Longrightarrow \quad \frac{\hat{L} - \frac{L}{2}}{\hat{L}} \geq \gamma \quad \Longrightarrow F(\mathbf{x}^{k+1}) \leq F(\mathbf{x}^k) - \frac{\gamma}{\hat{L}} \|G_{\alpha_k}(\mathbf{x}^k)\|_2^2.$$

This inequality implies that linesearch must stop when $L_k \geq \frac{L}{2(1-\gamma)}$.

$$\boxed{L_k \text{ is upper bounded: } L_k \leq \max\left\{L_{k-1}, \frac{\eta L}{2(1-\gamma)}\right\} \Longrightarrow \alpha_k \text{ is upper bounded}}$$

# Nonconvex case: Convergence

## Sufficient decrease property

- $\{\mathbf{x}^k\}_{k\geq 0}$: sequence generated by the proximal gradient method.
- Step size:
  - Constant step size: $\alpha$.
  - Step size with backtracking linesearch with parameters $(\gamma, \eta)$, $\alpha_k = 1/L_k$.
- Sufficient decrease property:
  - Constant step size:

$$\boxed{F(\mathbf{x}^{k+1}) \leq F(\mathbf{x}^k) - \alpha\left(1 - \frac{\alpha L}{2}\right)\|G_\alpha(\mathbf{x}^k)\|_2^2.}$$

  - Line search:

$$\boxed{F(\mathbf{x}^{k+1}) \leq F(\mathbf{x}^k) - \gamma \min\left\{\alpha_{k-1}, \frac{2(1-\gamma)}{\eta L}\right\}\|G_{\alpha_{k-1}}(\mathbf{x}^k)\|_2^2.}$$

# Nonconvex case: Convergence

**Theorem (Convergence of proximal-gradient method: Nonconvex [2])**

*Let $\{\mathbf{x}^k\}$ be generated by proximal-gradient method. Then:*

$$\min_{0,1,\cdots,k} \|G_\alpha(\mathbf{x}^k)\|_2 \leq \frac{\sqrt{F(\mathbf{x}^0) - F(\mathbf{x}^*)}}{\sqrt{M(k+1)}}$$

*The worst-case complexity to reach $\min_{0,1,\cdots,k} \|G_\alpha(\mathbf{x}^k)\|_2 \leq \varepsilon$, $\varepsilon$-first order stationary of the composite problem is $\mathcal{O}\left(\frac{1}{\varepsilon^2}\right)$.*

**Remark.**

Same convergence rate is preserved as in the unconstrained nonconvex setting.

## Recall: Composite convex minimization

**Problem (Unconstrained composite convex minimization)**

$$\boxed{F^\star := \min_{\mathbf{x}\in\mathbb{R}^p} \left\{F(\mathbf{x}) := f(\mathbf{x}) + g(\mathbf{x})\right\}} \tag{2}$$

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

## Recall: Composite convex minimization guarantees

Proximal gradient method(ISTA) vs. fast proximal gradient method (FISTA)

### Assumptions, step sizes and convergence rates

Proximal gradient method:

$$f \in \mathcal{F}_L^{1,1}, \quad \alpha = \frac{1}{L} \qquad\qquad F(\mathbf{x}^k) - F(\mathbf{x}^\star) \leq \epsilon, \quad \mathcal{O}\left(\frac{1}{\epsilon}\right).$$

Fast proximal gradient method:

$$f \in \mathcal{F}_L^{1,1}, \quad \alpha = \frac{1}{L} \qquad\qquad F(\mathbf{x}^k) - F(\mathbf{x}^\star) \leq \epsilon, \quad \mathcal{O}\left(\frac{1}{\sqrt{\epsilon}}\right).$$

# Recall: Composite convex minimization guarantees

Proximal gradient method(ISTA) vs. fast proximal gradient method (FISTA)

### Assumptions, step sizes and convergence rates

Proximal gradient method:

$$f \in \mathcal{F}_L^{1,1}, \quad \alpha = \frac{1}{L} \qquad\qquad F(\mathbf{x}^k) - F(\mathbf{x}^\star) \leq \epsilon, \quad \mathcal{O}\left(\frac{1}{\epsilon}\right).$$

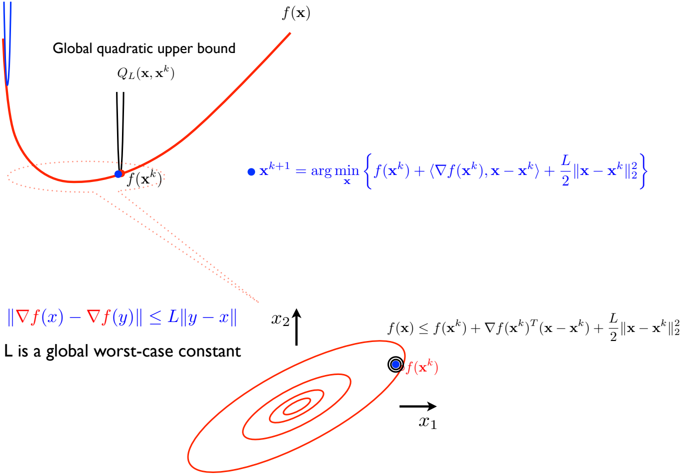
Fast proximal gradient method:

$$f \in \mathcal{F}_L^{1,1}, \quad \alpha = \frac{1}{L} \qquad\qquad F(\mathbf{x}^k) - F(\mathbf{x}^\star) \leq \epsilon, \quad \mathcal{O}\left(\frac{1}{\sqrt{\epsilon}}\right).$$

- We require $\alpha_k$ to be a function of $L$.
- It may not be possible to know exactly the Lipschitz constant. Line-search ?
- Adaptation to local geometry $\rightarrow$ may lead to larger steps.

# How can we better adapt to the local geometry?

Non-adaptive:



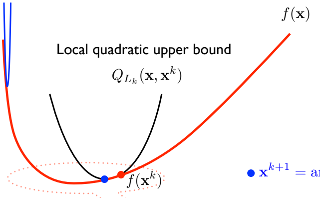

- $\mathbf{x}^{k+1} = \arg\min_{\mathbf{x}} \left\{ f(\mathbf{x}^k) + \langle \nabla f(\mathbf{x}^k), \mathbf{x} - \mathbf{x}^k \rangle + \frac{L}{2}\|\mathbf{x} - \mathbf{x}^k\|_2^2 \right\}$

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

L is a global worst-case constant

# How can we better adapt to the local geometry?

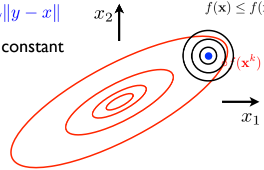
Line-search:




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

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

L is a global worst-case constant

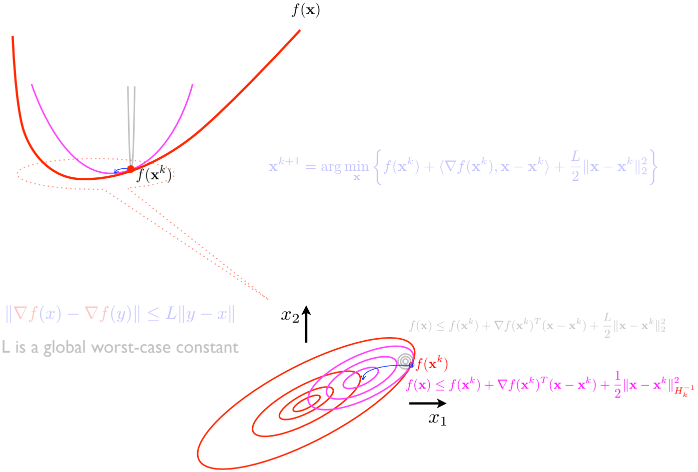



$f(\mathbf{x}) \leq f(\mathbf{x}^k) + \nabla f(\mathbf{x}^k)^T (\mathbf{x} - \mathbf{x}^k) + \frac{L_k}{2} \|\mathbf{x} - \mathbf{x}^k\|_2^2$

applies only locally

# How can we better adapt to the local geometry?

Variable metric:



# ⋆UniXGrad: A first order universal adaptive algorithm.

$$\min_{\mathbf{x} \in \mathcal{K}} f(\mathbf{x})$$

**Motivation behind UniXGrad**

Is it possible to get optimal rates for constrained convex optimization for $f \in \mathcal{F}_L^{2,1}$, without knowing the Lipschitz constant?

**UniXGrad [3]**

1. Set $\mathbf{y}^0 \in \mathcal{K}$, diameter $D$, weight $\alpha^k$, learning rate $\{\eta_k\}_{k \in [T]}$.
2. Define $d_\phi(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x}) - \phi(\mathbf{y}) - \langle \nabla \phi(\mathbf{y}), \mathbf{x} - \mathbf{y} \rangle$.
3. For $k = 0, 1, \ldots$, iterate

$$\begin{cases} \mathbf{x}^{k+1} = \arg\min_{\mathbf{x} \in \mathcal{K}} \alpha^k \left\langle \mathbf{x}, M^k \right\rangle + \frac{1}{\eta_k} d_\phi(\mathbf{x}, \mathbf{y}^k) & M^k = \nabla f(\tilde{z}^k) \\ \mathbf{y}^{k+1} = \arg\min_{\mathbf{y} \in \mathcal{K}} \alpha^k \left\langle \mathbf{y}, g^k \right\rangle + \frac{1}{\eta_k} d_\phi(\mathbf{y}, \mathbf{y}^k) & g^k = \nabla f(\tilde{\mathbf{x}}^k) \end{cases}$$

- This is essentially the **Mirror-Prox** scheme [4], with an adaptive step size!

# $^\star$UniXGrad - Properties and convergence

## The notion of averaging

Define $\alpha^k = k$ and following quantities:

$$\tilde{\mathbf{x}}^k = \frac{\alpha^k \mathbf{x}^k + \sum_{i=1}^{k-1} \alpha^i \mathbf{x}^i}{\sum_{i=1}^{k} \alpha^i}, \qquad \tilde{z}^k = \frac{\alpha^k \mathbf{y}^k + \sum_{i=1}^{k-1} \alpha^i \mathbf{x}^i}{\sum_{i=1}^{k} \alpha^i}$$

## Learning rate

UniXGrad uses the following weights and learning rate:

$$\eta_k = \frac{2D}{\sqrt{1 + \sum_{\tau=0}^{k} {\alpha^i}^2 \|g^i - M^i\|^2}},$$

where $D^2 = \sup_{\mathbf{x},\mathbf{y}\in\mathcal{K}} d_\phi(\mathbf{x},\mathbf{y})$ is diameter of the compact set $\mathcal{K}$ w.r.t. Bregman divergences.

# $^\star$UniXGrad - Convergence

## Convergence rate of AcceleGrad

Assume that f is convex and $f \in F_L^{1,1}$. Let $K$ be a convex set with bounded diameter D, and assume $x^\star \in K$. Define $\tilde{\mathbf{x}}^T = (\sum_{k=0}^{T-1} \alpha_k \mathbf{y}^{k+1})/(\sum_{k=0}^{T-1} \alpha_k)$. Then,

$$f(\tilde{\mathbf{x}}^T) - \min_{\mathbf{x} \in \mathbb{R}^d} f(x) \leq O\left(\frac{DG + LD^2}{T^2}\right)$$

If $f$ is only convex and $G$-Lipschitz, then

$$f(\overline{y}^T) - \min_{x \in \mathbb{R}^d} f(x) \leq O\left(GD/\sqrt{T}\right)$$

# The idea of the proximal-Newton method

## Assumptions A.2

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

## Proximal-Newton update

- Similar to classical newton, proximal-newton suggests the following update scheme using second order Taylor series expansion near $x_k$.

$$\mathbf{x}^{k+1} := \arg\min_{\mathbf{x}\in\mathbb{R}^p}\Big\{\underbrace{\frac{1}{2}(\mathbf{x}-\mathbf{x}^k)^T\nabla^2 f(\mathbf{x}^k)(\mathbf{x}-\mathbf{x}^k)+\nabla f(\mathbf{x}^k)^T(\mathbf{x}-\mathbf{x}^k)}_{\text{2nd-order Taylor expansion near } x^k}+g(\mathbf{x})\Big\}. \tag{3}$$

# The proximal-Newton-type algorithm

| Proximal-Newton algorithm (PNA) |
| --- |
| **1.** Given $\mathbf{x}^0 \in \mathbb{R}^p$ as a starting point. |
| **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. Update $\mathbf{x}^{k+1} := \mathbf{x}^k + \alpha_k \mathbf{d}^k$. |

# The proximal-Newton-type algorithm

**Proximal-Newton algorithm (PNA)**

**1.** Given $\mathbf{x}^0 \in \mathbb{R}^p$ as a starting point.

**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. Update $\mathbf{x}^{k+1} := \mathbf{x}^k + \alpha_k \mathbf{d}^k$.

## Remark

- $\mathbf{H}_k \equiv \nabla^2 f(\mathbf{x}^k) \Longrightarrow$ proximal-Newton algorithm.
- $\mathbf{H}_k \approx \nabla^2 f(\mathbf{x}^k) \Longrightarrow$ proximal-quasi-Newton algorithm.
- A generalized prox-operator: $\mathrm{prox}_{\mathbf{H}_k^{-1} g}\left(\mathbf{x}^k + \mathbf{H}_k^{-1} \nabla f(\mathbf{x}^k)\right)$.

# Convergence analysis

## Theorem (Global convergence [6])

*Assume generalized-prox subproblem is solved exactly for the algorithm and there exists $\mu > 0$ such that $\mathbf{H}_k \succeq \mu\mathbb{I}$ for all $k \geq 0$. Then;*

$$\boxed{\{\mathbf{x}^k\}_{k\geq 0} \text{ globally converges to a solution } \mathbf{x}^\star \text{ of (2).}}$$

# Convergence analysis

### Theorem (Global convergence [6])

*Assume generalized-prox subproblem is solved exactly for the algorithm and there exists $\mu > 0$ such that $\mathbf{H}_k \succeq \mu\mathbb{I}$ for all $k \geq 0$. Then;*

$$\boxed{\{\mathbf{x}^k\}_{k\geq 0} \text{ globally converges to a solution } \mathbf{x}^\star \text{ of (2).}}$$

### Theorem (Local convergence [6])

*Assume generalized-prox subproblem is solved exactly for the algorithm there exists $0 < \mu \leq L_2 < +\infty$ such that $\mu\mathbb{I} \preceq \mathbf{H}_k \preceq L_2\mathbb{I}$ for all sufficiently large $k$. Then;*

- *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:*

$$\lim_{k\to+\infty} \frac{\|(\mathbf{H}_k - \nabla^2 f(\mathbf{x}^\star))(\mathbf{x}^{k+1} - \mathbf{x}^k)\|}{\|\mathbf{x}^{k+1} - \mathbf{x}^k\|} = 0, \tag{4}$$

*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.

## Pros and cons

### Pros

- Fast local convergence rate (super-linear or quadratic)
- Numerical robustness under the inexactness/noise ([6]).
- Well-suited for problems with many data points but few parameters. For example,

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

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

## Pros and cons

### Pros

- Fast local convergence rate (super-linear or quadratic)
- Numerical robustness under the inexactness/noise ([6]).
- Well-suited for problems with many data points but few parameters. For example,

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

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

### Cons

- 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.
- 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,* $\mu$ *is called the intercept.*
***Goal:*** *Find a sparse-weight vector* $\mathbf{x}$ *via the maximum likelihood principle.*

## Optimization formulation

$$\boxed{\min_{\mathbf{x}\in\mathbb{R}^p} \left\{ \underbrace{\frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(b_i(\mathbf{a}_i^T\mathbf{x} + \mu))}_{f(\mathbf{x})} + \underbrace{\rho\|\mathbf{x}\|_1}_{g(\mathbf{x})} \right\},} \tag{5}$$

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 $\ell$ is the logistic loss function $\mathcal{L}(\tau) := \log(1 + e^{-\tau})$.

## Example: Sparse logistic regression

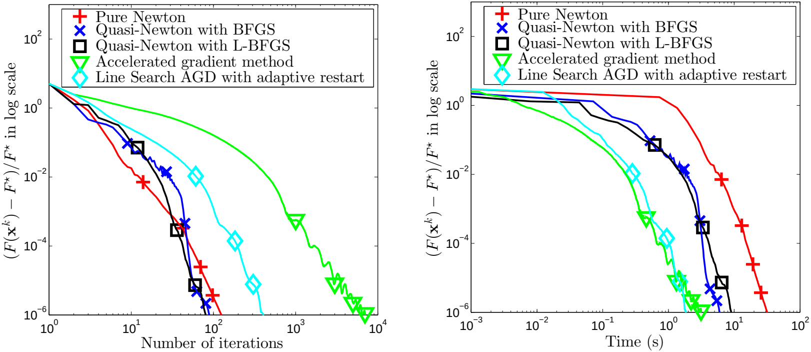
### Real data

- 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: $\ell_1$-regularized least squares

### Problem ($\ell_1$-regularized least squares)

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

$$F^\star := \min_{\mathbf{x} \in \mathbb{R}^p} \left\{ F(\mathbf{x}) := \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 + \rho\|\mathbf{x}\|_1 \right\}, \tag{6}$$

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

### Complexity per iterations

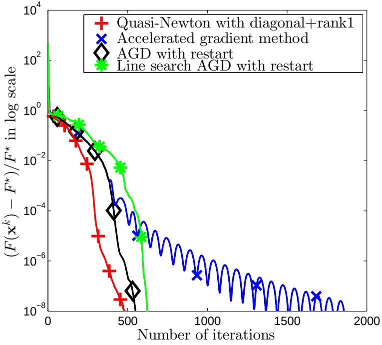
- Evaluating $\nabla f(\mathbf{x}^k) = \mathbf{A}^T(\mathbf{A}\mathbf{x}^k - \mathbf{b})$ requires one $\mathbf{A}\mathbf{x}$ and one $\mathbf{A}^T\mathbf{y}$.
- One soft-thresholding operator $\mathrm{prox}_{\lambda g}(\mathbf{x}) = \mathrm{sign}(\mathbf{x}) \otimes \max\{|\mathbf{x}| - \rho, 0\}$.
- **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 $\mathbf{A}^T\mathbf{y}$).

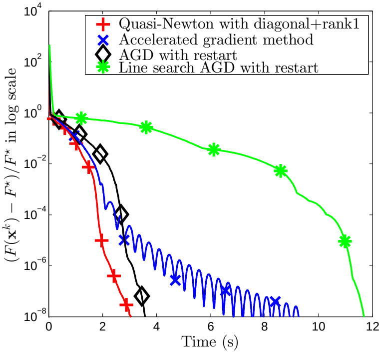
### Synthetic data generation

- $\mathbf{A} := \mathrm{randn}(n, p)$ - 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: $\ell_1$-regularized least squares - Numerical results - Trial 1

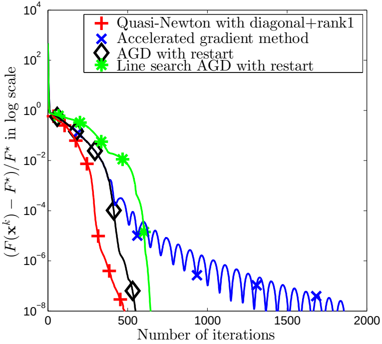
**Parameters:** $n = 750, p = 2000, s = 200, \rho = 1$

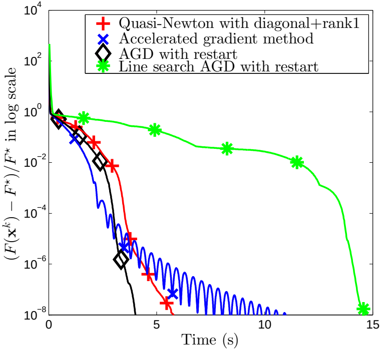


# Example 2: $\ell_1$-regularized least squares - Numerical results - Trial 2

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



# Stochastic convex composite minimization

## Problem (Mathematical formulation)

*Consider the following constrained convex minimization problem:*

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

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

- The goal is to extend the methods to proximal case where $g \neq 0$.

# 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:

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

- $\{\theta_k\}$: jointly independent random variables.
- $G(\mathbf{x}^k, \theta_k)$: an unbiased estimate of the full gradient:

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

## 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:

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

- $\{\theta_k\}$: jointly independent random variables.
- $G(\mathbf{x}^k, \theta_k)$: an unbiased estimate of the full gradient:

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

### Remark

- Cost of computing $G(\mathbf{x}^k, \theta_k)$ is usually much cheaper than $\nabla f(\mathbf{x}^k)$.

## Convergence analysis

### Assumption A4.

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

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

(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 $\mu$-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

$$F^\star := \min_{\mathbf{x}\in\mathrm{dom}\,(F)} \left\{ F(\mathbf{x}) := \frac{1}{m}\sum_{k=1}^{m} f_k(\mathbf{x}) + g(\mathbf{x}) \right\}, \tag{7}$$

- $f_k \in \mathcal{F}^{1,1}_{L_k}(\mathbb{R}^p)$ 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 $m$ is large.
- Examples: Portfolio optimization, SVM.

## Large scale problems

### Definition (Recall)

- $\nabla f_j$ is called the stochastic gradient of $f$, and it is unbiased estimate, i.e.

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

### 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)

- $\nabla f_j$ is called the stochastic gradient of $f$, and it is unbiased estimate, i.e.

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

### 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:

$$\begin{cases} \text{Pick } i_k \in \{1, \ldots, m\} \text{ uniformly at random} \\ \mathbf{x}^{k+1} := \mathrm{prox}_{\gamma_k g}\left(\mathbf{x}^k - \gamma_k \nabla f_{i_k}(\mathbf{x}^k)\right), \end{cases} \quad (8)$$

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

## Properties

- Cheap per iteration complexity (one component gradient and one $\mathrm{prox}$).
- Decreasing stepsize such that $(\gamma_k)_{k\in\mathbb{N}} \in \ell^2(\mathbb{N})\backslash\ell^1(\mathbb{N})$.

## 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:***

- *$f$ is $\mu$-strongly convex and $\gamma_k \sim 1/(k+1)$.*

***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*

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

# Comparisons

- 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.

- Cheap per iteration complexity + slow convergence rate.

$$\boxed{\text{SPG} \longrightarrow \text{Large scale problems} + \text{low accurate of solution}}$$

- Decreasing learning rate due to variance, hence slower convergence.

$$\boxed{\text{Solution: Variance reduction technique (see Prox-SVRG)}}$$

# Example: $\ell_1$-regularized least squares revisited

## Problem ($\ell_1$-regularized least squares)

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

$$F^\star := \min_{\mathbf{x} \in \mathbb{R}^p} \left\{ F(\mathbf{x}) := \frac{1}{2m} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 + \rho \|\mathbf{x}\|_1 \right\}, \tag{9}$$

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

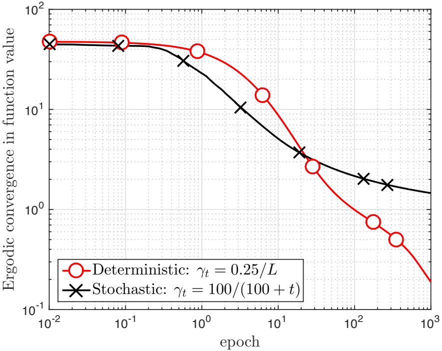
## Complexity per iterations

- 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 $\lambda a_j$.
- One soft-thresholding operator $\mathrm{prox}_{\lambda g}(\mathbf{x}) = \mathrm{sign}(\mathbf{x}) \otimes \max\{|\mathbf{x}| - \kappa, 0\}$.
- **Optional**: Evaluating $L_{\max} = \max_{1 \le k \le m} \|a_k\|^2$ - 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: $\ell_1$-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.** $\mathbf{z}^{k+1} = \mathbf{z}^k - \frac{1}{\gamma_k \alpha_k + \mu}\left(\gamma_k(\mathbf{x}^{k+1} - \mathbf{y}^{k+1}) + \mu(\mathbf{z}^k - \mathbf{x}^{k+1})\right)$ .

# $^\star$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* $\mu$*-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* $\mathbb{E}[\|\mathbf{z}^k - \mathbf{x}^\star\|^2]$.

# $^\star$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,*

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

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

Recall: Variance reduction techniques.

- Select the stochastic gradient $\nabla f_{i_k}$, and compute a gradient estimate

$$\mathbf{r}_k = \nabla f_{i_k}(\mathbf{x}^k) - \nabla f_{i_k}(\tilde{\mathbf{x}}) + \nabla f(\tilde{\mathbf{x}}),$$

where $\tilde{\mathbf{x}}$ is a good approximation of $\mathbf{x}^\star$.

- As $\tilde{\mathbf{x}} \to \mathbf{x}^\star$ and $\mathbf{x}^k \to \mathbf{x}^\star$,

$$\nabla f_{i_k}(\mathbf{x}^k) - \nabla f_{i_k}(\tilde{\mathbf{x}}) + \nabla f(\tilde{\mathbf{x}}) \to 0.$$

- Therefore,

$$\mathbb{E}\left[\|\nabla f_{i_k}(\mathbf{x}^k) - \nabla f_{i_k}(\tilde{\mathbf{x}}) + \nabla f(\tilde{\mathbf{x}})\|^2\right] \to 0.$$

# 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:

$$\begin{cases} \text{Pick } i_k \in \{1, \ldots, m\} \text{ uniformly at random} \\ \mathbf{r}_k = \nabla f_{i_k}(\mathbf{x}_k) - \nabla f_{i_k}(\widetilde{\mathbf{x}}) + \widetilde{\mathbf{v}} \\ \mathbf{x}^{k+1} := \mathrm{prox}_{\gamma g}\left(\mathbf{x}^k - \gamma \mathbf{r}_k\right), \end{cases} \tag{10}$$

   - **2c**. Update $\widetilde{\mathbf{x}}^s = \frac{1}{n}\sum_{j=0}^{n-1} \mathbf{x}^j$.

## Properties

- A multistage scheme to reduce the variance of the stochastic gradient.
- Possibility of constant learning rate.
- $m + 2n$ component gradient evaluations at each iteration.

## Convergence analysis

### Assumption A5.

(i) $F = f + g$ is $\mu$-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:*

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

# Choice of $\gamma$ and $n$, and complexity

**Chose $\gamma$ 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**

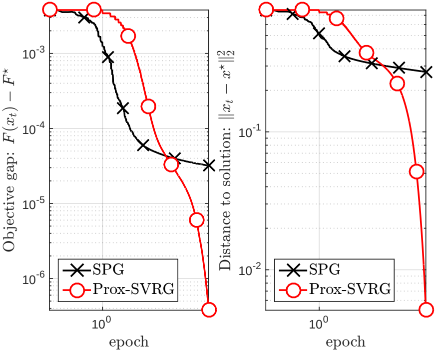
$$\mathbb{E}F(\widetilde{\mathbf{x}^s}) - F(\mathbf{x}^\star) \leq \varepsilon \ when\ s \geq \log(\kappa^{-1}) \log((F(\widetilde{\mathbf{x}^0}) - F(\mathbf{x}^\star))/\epsilon)$$

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: $\ell_1$-regularized least squares revisited- Numerical test

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



# SVRG$^{++}$ for nonstrongly convex objectives

Strong convexity assumption rules out many important applications.

$\triangleright$ Examples: *Lasso, ridge rigression,* $\ell_1$ *regularized logistic regression.*

# SVRG$^{++}$ for nonstrongly convex objectives

Strong convexity assumption rules out many important applications.
▷ Examples: *Lasso, ridge rigression, $\ell_1$ regularized logistic regression.*

**Idea:** Increase the number of inner iterations linearly.

# SVRG$^{++}$ for nonstrongly convex objectives

Strong convexity assumption rules out many important applications.
▷ Examples: *Lasso, ridge rigression, $\ell_1$ regularized logistic regression.*

**Idea:** Increase the number of inner iterations linearly.

**SVRG$^{++}$ [1]**

**1**. Choose $\widetilde{\mathbf{x}}^0 \in \mathbb{R}^p$ as a starting point ,$m_0 \in \mathbb{N}_+$ as the inner loop parameter, $\gamma > 0$ and $n \in \mathbb{N}_+$.
**2**. For $s = 0, 1, 2 \cdots n$, perform:
**2a**. $\widetilde{\mathbf{x}} = \widetilde{\mathbf{x}}^s, \quad \widetilde{\mathbf{v}} = \nabla f(\widetilde{\mathbf{x}}), \quad m_s = 2^s m_0.$
**2b**. For $k = 0, 1, \cdots m_s - 1$, perform:

$$\begin{cases} \text{Pick } i_k \in \{1, \ldots, m\} \text{ uniformly at random} \\ \mathbf{r}_k = \nabla f_{i_k}(\mathbf{x}_s^k) - \nabla f_{i_k}(\widetilde{\mathbf{x}}) + \widetilde{\mathbf{v}} \\ \mathbf{x}_s^{k+1} := \mathrm{prox}_{\gamma g}\left(\mathbf{x}_s^k - \gamma \mathbf{r}_k\right), \end{cases} \tag{11}$$

**2c**. Update $\widetilde{\mathbf{x}}^s = \frac{1}{m_s}\sum_{j=1}^{m_s} \mathbf{x}^j, \quad \mathbf{x}_{s+1}^0 = \mathbf{x}_s^{m_s}.$

# SVRG$^{++}$ for nonstrongly convex objectives

Strong convexity assumption rules out many important applications.
▷ Examples: *Lasso, ridge rigression, $\ell_1$ regularized logistic regression.*

**Idea:** Increase the number of inner iterations linearly.

**SVRG$^{++}$ [1]**

**1**. Choose $\widetilde{\mathbf{x}}^0 \in \mathbb{R}^p$ as a starting point ,$m_0 \in \mathbb{N}_+$ as the inner loop parameter, $\gamma > 0$ and $n \in \mathbb{N}_+$.
**2**. For $s = 0, 1, 2 \cdots n$, perform:
  **2a**. $\widetilde{\mathbf{x}} = \widetilde{\mathbf{x}}^s$, $\quad \widetilde{\mathbf{v}} = \nabla f(\widetilde{\mathbf{x}})$, $\quad m_s = 2^s m_0$.
  **2b**. For $k = 0, 1, \cdots m_s - 1$, perform:

$$\begin{cases} \text{Pick } i_k \in \{1, \ldots, m\} \text{ uniformly at random} \\ \mathbf{r}_k = \nabla f_{i_k}(\mathbf{x}_s^k) - \nabla f_{i_k}(\widetilde{\mathbf{x}}) + \widetilde{\mathbf{v}} \\ \mathbf{x}_s^{k+1} := \mathrm{prox}_{\gamma g}\left(\mathbf{x}_s^k - \gamma \mathbf{r}_k\right), \end{cases} \tag{11}$$

  **2c**. Update $\widetilde{\mathbf{x}}^s = \frac{1}{m_s}\sum_{j=1}^{m_s} \mathbf{x}^j$, $\quad \mathbf{x}_{s+1}^0 = \mathbf{x}_s^{m_s}$.

## Properties

- Learning rate $\gamma < 1/(7L_{\max})$.
- Gradient complexity $\mathcal{O}(S \cdot n + 2^S \cdot m_0)$.
- Linear convergence as in the strongly convex case [1].

# References I


[1] Zeyuan Allen-Zhu and Yang Yuan.
Improved svrg for non-strongly-convex or sum-of-non-convex objectives.
In *International conference on machine learning*, pages 1080–1089, 2016.

[2] Amir Beck.
*First-order methods in optimization*, volume 25.
SIAM, 2017.

[3] Ali Kavis, Kfir Y Levy, Francis Bach, and Volkan Cevher.
Unixgrad: A universal, adaptive algorithm with optimal guarantees for constrained optimization.
*arXiv preprint arXiv:1910.13857*, 2019.

[4] Arkadi Nemirovski.
Prox-method with rate of convergence o (1/t) for variational inequalities with lipschitz continuous monotone operators and smooth convex-concave saddle point problems.
*SIAM Journal on Optimization*, 15(1):229–251, 2004.

[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.

# References II


[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 III

[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 IV


[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.