# First Order Algorithms for Convex Minimization


Marc Teboulle

School of Mathematical Sciences
Tel Aviv University



**Optimization Tutorials, September 14-17, 2010**
**IPAM - Institute for Pure and Applied Mathematics, UCLA, Los Angeles**




## Opening Remark and Credit

About more than 380 years ago.....In 1629..

- Solve for $x$ : $\left[\dfrac{f(x+d)-f(x)}{d}\right]_{d=0} = 0$

**...We can hardly expect to find a more general method to get the maximum or minimum points on a curve.....**

**Pierre de Fermat**

# First Order/Gradient Based Methods: Why?

- **A main drawback:** Can be very slow for producing high accuracy solutions....But... also **share many advantages**:
- Use minimal information, e.g., $(f, f')$ (as opposed to more sophisticated methods).
- Often lead to very simple and "cheap" iterative schemes.
- Complexity/iteration mildly dependent (e.g., linear) in problem's dimension.
- Suitable when high accuracy is not crucial [in many large scale applications, the data is anyway corrupted or known only roughly..]
- For very large scale problems with medium accuracy requirements, gradient based methods often remain the only practical alternative.



# Polynomial versus Gradient Methods

- Convex problems are polynomially solvable within $\varepsilon$ accuracy:

$$\text{Running Time} \leq Poly(\text{Problem's size}, \# \text{ of accuracy digits}).$$

- **Theoretically:** this means that large scale problems can be solved to high accuracy with polynomial methods, such as IPM.
- **Practically:** Running time is **dimension-dependent** and grows **nonlinearly** with problem's dimension. For IPM which are Newton's type methods: $\sim O(n^3)$.
- **Example:** reported on PET problem using best IPM (Ben-Tal, Nemirovsky, Margalit (2002)):
- $n = 250,000$, CPU /Iteration: $\sim 2.5$ Hours
- $n = 2,000,000$, CPU/Iteration: $\sim 2$ weeks!!
- **Thus, a "single iteration" can last forever!**

## Gradient-Based Algorithms

Widely used in applications....

- **Clustering Analysis:** *The k-means algorithm*
- **Neuro-computing:** *The backpropagation algorithm*
- **Statistical Estimation:** *The EM (Expectation-Maximization)* algorithm.
- **Machine Learning:** *SVM, Regularized regression, etc...*
- **Signal and Image Processing:** *Sparse Recovery, Denoising and Deblurring Schemes, Total Variation minimization...*
- **Matrix minimization Problems....and much more...**



## Objectives and Outline

1. **Convey basic ideas to Build and Analyze Gradient-Based Schemes**
2. **Exploit Structures for Various Classes of Smooth and Nonsmooth Convex Minimization Problems**

### Outline

**I. Gradient/Subgradient Algorithms: Basic Results**
**II. Mathematical Tools for Convergence Analysis**
**III. Fast Gradient-Based Methods**
**IV. Gradient Schemes based on Non-Euclidean Distances**

**Applications and examples illustrating ideas and methods**

# Quick Recalls on Convex Functions

- Throughout, $\mathbb{E}$ stands for a finite dimensional vector space.
- Let $f : \mathbb{E} \to (-\infty, +\infty]$ be proper, closed (lsc) convex function, with $\mathrm{dom}\, f = \{\mathbf{x} \,|\, f(\mathbf{x}) < +\infty\}$ its effective domain.
- Proper: $\mathrm{dom}\, f \neq \emptyset$ and $f(\mathbf{x}) > -\infty$, $\forall \mathbf{x} \in \mathbb{E}$.
- Closed and Convex: Its epigraph is a closed convex set

$$\mathrm{epi}\, f := \{(\mathbf{x}, \alpha) \in \mathbb{E} \times \mathbb{R} \,|\, \alpha \geq f(\mathbf{x})\}.$$

- Extended valued functions are useful for handling constraints:

$$\inf\{h(\mathbf{x}) : \ \mathbf{x} \in C\} \iff \inf\{f(\mathbf{x}) : \ \mathbf{x} \in \mathbb{E}\}, \ f := h + \delta_C$$

  where $\delta_C(\mathbf{x}) = 0$ if $\mathbf{x} \in C$ and $+\infty$ if $\mathbf{x} \notin C$ is the indicator of $C$.
- For any closed convex set $C \subset \mathbb{E}$, $(\mathrm{int}\, C), \mathrm{ri}\, C$ denotes its (interior) relative interior.



# Subdifferentiability of Convex Functions

- $\mathbf{g} \in \mathbb{E}$ is a subgradient of $f$ at $\mathbf{x}$ if:

$$f(\mathbf{z}) \geq f(\mathbf{x}) + \langle \mathbf{g}, \mathbf{z} - \mathbf{x} \rangle, \ \forall \mathbf{z}$$

- Subdifferential of $f$ at $\mathbf{x}$ = Set of all subgradients:

$$\partial f(\mathbf{x}) = \{\mathbf{g} \in \mathbb{E} \mid f(\mathbf{z}) \geq f(\mathbf{x}) + \langle \mathbf{g}, \mathbf{z} - \mathbf{x} \rangle, \ \forall \mathbf{z} \in \mathbb{E}\}.$$

- $\partial f(\mathbf{x})$ is a closed convex set (possibly empty) as an infinite intersection of closed half-spaces.
- If $\mathbf{x} \in \mathrm{int}\,\mathrm{dom}\, f$, $\partial f(\mathbf{x})$ is nonempty and bounded.
- When $f$ is differentiable, $\partial f(\mathbf{x}) \equiv \{\nabla f(\mathbf{x})\} \equiv \{f'(\mathbf{x})\}$.
- $f$ is $\sigma$-strongly convex iff $f(\cdot) - \sigma\|\cdot\|^2/2$ is convex, i.e.,

$$\langle \mathbf{u} - \mathbf{v}, \mathbf{x} - \mathbf{y} \rangle \geq \sigma\|\mathbf{x} - \mathbf{y}\|^2, \ \mathbf{u} \in \partial f(\mathbf{x}), \mathbf{v} \in \partial f(\mathbf{y}), \ (\sigma > 0).$$

- $f^*(\mathbf{y}) = \sup\{\langle \mathbf{x}, \mathbf{y} \rangle - f(\mathbf{x}) : \ \mathbf{x} \in \mathbb{E}\}$, its convex conjugate.

# A Generic Optimization Model

$$\text{(M)} \quad \min \{F(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x}) : \mathbf{x} \in \mathbb{E}\}$$

- $\mathbb{E}$ is a finite dimensional Euclidean space with inner product $\langle \cdot, \cdot \rangle$ and norm $\| \cdot \| = \langle \cdot, \cdot \rangle^{1/2}$.
- $g : \mathbb{E} \to (-\infty, \infty]$ is proper closed and convex, assumed subdifferentiable over $\operatorname{dom} g$ assumed closed.
- $f : \mathbb{E} \to \mathbb{R}$ is continuously differentiable on $\mathbb{E}$, with gradient $\nabla f \equiv f'$.
- We assume that (M) is solvable, i.e.,

$$X_* := \operatorname{argmin} f \neq \emptyset, \text{ and for } \mathbf{x}^* \in X_*, \text{ set } F_* := F(\mathbf{x}^*).$$

The model (M) is rich enough to recover various classes of smooth/nonsmooth convex minimization problems.



# Examples of (M) $\quad \min \{F(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x}) : \mathbf{x} \in \mathbb{E}\}$

- Differentiable Unconstrained Minimization: Pick $g \equiv 0$,

$$\min \{f(\mathbf{x}) : \mathbf{x} \in \mathbb{E}\}.$$

- Constrained Convex Minimization: Pick $g = \delta_C$,

$$\min \{f(\mathbf{x}) : \mathbf{x} \in C\}, \quad C \subseteq \mathbb{E} \text{ a closed convex set}$$

- Convex Program $\min\{h_0(\mathbf{x}) : h_i(\mathbf{x}) \leq 0, \; i = 1, \ldots, m\}$

$$f(x) := h_0(\mathbf{x}), g(\mathbf{x}) := \sum_{i=1}^{m} \delta_{(-\infty,0]}(h_i(\mathbf{x})).$$

- Nonsmooth Convex Minimization: Pick $f \equiv 0$, $\min \{g(\mathbf{x}) : \mathbf{x} \in \mathbb{E}\}$

More "specific" examples arising in various applications, later on ....

# The Gradient Method – Cauchy 1847..

- We begin with the simplest unconstrained minimization problem of a continuously differentiable function $f$ on $\mathbb{E}$ (set $g \equiv 0$ in (M)):

$$\text{(U)} \ \min\{f(\mathbf{x}) : \mathbf{x} \in \mathbb{E}\}.$$

- The basic gradient method generates a sequence $\{\mathbf{x}_k\}$ via

$$\mathbf{x}_0 \in \mathbb{E}, \quad \mathbf{x}_k = \mathbf{x}_{k-1} - t_k \nabla f(\mathbf{x}_{k-1}) \quad (k \geq 1)$$

with suitable step size $t_k > 0$: fixed; backtracking line search; exact line search; diminishing step-size: $t_k \to 0, \sum t_k = \infty$.

- This is a *descent method*:

$$\mathbf{x}^+ = \mathbf{x} + t\mathbf{d}; \quad \langle \mathbf{d}, \nabla f(\mathbf{x})\rangle < 0, \ \mathbf{d} := -\nabla f(\mathbf{x}) \neq 0.$$

- *Explicit discretization* of $d\mathbf{x}(t)/dt + \nabla f(\mathbf{x}(t)) = 0, \ \ \mathbf{x}(0) = \mathbf{x}_0$.

$$\frac{\mathbf{x}_k - \mathbf{x}_{k-1}}{h} = -\nabla f(\mathbf{x}_{k-1}), \ \text{(increment } h > 0).$$



# Backtracking Line Search – BLS

A simple inexact line search to find $t$ for descent methods:

$$\min_t \phi(t) := f(\mathbf{x} + t\mathbf{d}), \ \text{(e.g., here } \mathbf{d} := -\nabla f(\mathbf{x})).$$

Sufficient decrease + rules out too short steps.

1. Initialize: Choose $\bar{t} > 0$, (e.g., $\bar{t} = 1$), $\alpha, \beta \in (0, 1)$ Set $t = \bar{t}$
2. Until

$$(*) \qquad f(\mathbf{x} + t\mathbf{d}) \leq f(\mathbf{x}) + \alpha t \langle \mathbf{d}, \nabla f(\mathbf{x})\rangle$$

set $t \leftarrow \beta t$, (e.g., $\beta = 1/2$).

• BLS procedure warrants sufficient decrease.

• Not too short, since within factor $\beta$ of previous step $t/\beta$ which is rejected when violating (*), that is for being too long.

# Convergence of Algorithms: A Remark

- Traditionally, in numerical analysis of optimization algorithms the focus is on *pointwise* convergence of $\{\mathbf{x}_k\}$ and its *asymptotic* rate of convergence.
- Here, we depart from "tradition" and focus on **non-asymptotic** global rate of convergence and efficiency, measured in terms of function values, for all $k \geq 1$:

$$F(\mathbf{x}_k) - F_* \leq \frac{\Gamma}{k^\theta}, \qquad (\Gamma > 0, \theta > 0)$$

- We are interested in solving approximately a problem to a given accuracy $\varepsilon > 0$, i.e., to find an $\mathbf{x}_k$ s.t.

$$F(\mathbf{x}_k) - F_* \leq \varepsilon.$$

Thus, # iterations for such an approximation is $O(\varepsilon^{-1/\theta})$.



# Gradient Method: Classical Results

**Assumption** $f$ is $C^{1,1}_{L(f)}$ over $\mathbb{E}$, i.e., with gradient Lipschitz:

$$\exists L(f) > 0 : \; \|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \leq L(f)\|\mathbf{x} - \mathbf{y}\|, \; \forall \mathbf{x}, \mathbf{y}.$$

For $f \in C^{1,1}_{L(f)}$. The sequence generated by GM with either constant stepsize or via BLS satisfies:

$$\min_{1 \leq s \leq k} \|\nabla f(\mathbf{x}_{s-1})\| \leq \frac{1}{\sqrt{k}} \left( \frac{2\alpha^2 L(f)(f(\mathbf{x}_0) - f_*)}{\beta} \right)^{1/2}.$$

- In other words $\nabla f(\mathbf{x}_k) \to 0$ at a rate of $O(1/\sqrt{k})$.
- Mildly depends on *dimension*.
- No results for $\{\mathbf{x}_k\}$..or even.. $\{f(\mathbf{x}^k)\}$...
- Assuming that $f$ is also **convex**, we get more...

## Gradient Method for Convex $f$

For $f \in C^{1,1}_{L(f)}$ and convex,the sequence generated by GM with either constant step size or BLS satisfies for all $k \geq 1$:

$$f(\mathbf{x}_k) - f(\mathbf{x}^*) \leq \frac{\alpha L(f) \|\mathbf{x}^* - \mathbf{x}_0\|^2}{2k}.$$

- Thus, $\#$ iterations for $f(\mathbf{x}_k) - f(\mathbf{x}^*) \leq \epsilon$ is $O(1/\epsilon)$...
- Can be very slow even for low accuracy requirements...



## Constrained Problem: Gradient Projection Method

For the constrained problem (e.g., $g := \delta_C$ in (M)):

$$(P) \quad \min \{f(\mathbf{x}) : \mathbf{x} \in C\}, \; C \subseteq \mathbb{E} \text{ closed convex}$$

**The gradient projection method (GPM)**

$$\mathbf{x}_0 \in \mathbb{E}, \; \mathbf{x}_k = \Pi_C(\mathbf{x}_{k-1} - t_k \nabla f(\mathbf{x}_{k-1})), \; k \geq 1$$

orthogonal projection operator $\Pi_C(\mathbf{x}) = \operatorname*{argmin}_{\mathbf{z} \in C} \|\mathbf{z} - \mathbf{x}\|^2$.

- In the convex case, under same assumptions as (GM), ($f \in C^{1,1}$) we have the same convergence result.
- $\#$ iterations for $f(\mathbf{x}_k) - f(\mathbf{x}^*) \leq \epsilon$ is $O(1/\epsilon)$

# Simplest Method for NSO: Subgradient Method

Nondifferentiable Convex $(P)$ $\qquad \inf\{g(x): \quad x \in C\} = g_*$

**Subgradient Scheme: Shor (63), Polyak (65)**

$$\gamma^{k-1} \in \partial g(x^{k-1}),\ x^k = \Pi_C(x^{k-1} - t_k\gamma^{k-1}),\ (t_k > 0,\ \text{a stepsize})$$

- Subgradient scheme is **not a descent method**.
- Assuming that $g$ is Lipschitz, with constant $M > 0$,i.e.,

$$\|g(\mathbf{x}) - g(\mathbf{y})\| \le M\|\mathbf{x} - \mathbf{y}\|, \forall \mathbf{x}, \mathbf{y}\ (\Leftrightarrow\ \|\gamma\| \le M,\ \gamma \in \partial g(\mathbf{x}))$$

  For diminishing step size $t_s \to 0, \sum t_s = \infty$ we have

$$g_{\text{best}}(\mathbf{x}) := \min_{1 \le s \le k} g(\mathbf{x}_s) \to g_*.$$

- What about the rate of convergence in the nonsmooth case?



# Rate of Convergence of SM

**A typical result:** assume $C$ convex compact. Take

$$t_k = \frac{\text{Diam}(C)}{\sqrt{k}};\ \text{Diam}(C) := \max_{\mathbf{x},\mathbf{y} \in C} \|\mathbf{x} - \mathbf{y}\| < \infty,$$

$$\text{Then,} \qquad \min_{1 \le s \le k} g(\mathbf{x}_s) - g_* \le O(1)M\frac{\text{Diam}(C)}{\sqrt{k}}$$

- Thus, to find an approximate $\varepsilon$ solution: $\ O(1/\epsilon^2)$
- **Key Advantages:** rate nearly *independent* of problem's dimension. Simple, when projections are easy to compute...
- **Main Drawback of SM:** too slow...needs $k \ge \epsilon^{-2}$ iterations.
- Can we improve the situation of SM?...Later on by exploiting the structure/geometry of the constraint set $C$...

# Building Gradient-Based Schemes

Our objective is to solve

$$(\mathrm{M}) \quad \min\left\{F(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x}) : \mathbf{x} \in \mathbb{E}\right\}, \ f \text{ smooth}, \ g \text{ nonsmooth}$$

**Initial interpretation of GM:** *go towards the direction of the negative gradient of the objective.*

This cannot be extended to $F := f + g$, since $g$ is nonsmooth.

- Good approximation models for solving (M)
- Fixed point methods on corresponding optimality conditions
- The Proximal Framework
- Majorization-Minimization approach



# A Quadratic Approximation Model

- Simplest case of (M), unconstrained minimization of $f \in C^1$:

$$(U) \qquad \min\{f(\mathbf{x}) : \mathbf{x} \in \mathbb{E}\}.$$

- **Simplest idea**: Use the quadratic model

$$q_t(\mathbf{x}, \mathbf{y}) := f(\mathbf{y}) + \langle \mathbf{x} - \mathbf{y}, \nabla f(\mathbf{y}) \rangle + \frac{1}{2t}\|\mathbf{x} - \mathbf{y}\|^2, \ t > 0.$$

  Namely, use the linearized part of $f$ at some given point $\mathbf{y}$.
  Regularized with a quadratic proximity term that would measure the "local error" in the approximation.
- This leads to a (strongly) convex approximation for (U):

$$(\hat{U}_t) \quad \min\left\{q_t(\mathbf{x}, \mathbf{y}) : \mathbf{x} \in \mathbb{E}\right\}.$$

- Now, fixing $\mathbf{y} := \mathbf{x}_{k-1} \in \mathbb{E}$, the unique minimizer $\mathbf{x}_k$ solving $(\hat{U}_{t_k})$

$$\mathbf{x}_k = \operatorname{argmin}\left\{q_{t_k}(\mathbf{x}, \mathbf{x}_{k-1}) : \mathbf{x} \in \mathbb{E}\right\}.$$

- Therefore, optimality condition yields exactly the gradient method:

$$\nabla q_{t_k}(\mathbf{x}_k, \mathbf{x}_{k-1}) = 0 \implies \mathbf{x}_k = \mathbf{x}_{k-1} - t_k \nabla f(\mathbf{x}_{k-1}).$$

# Gradient Projection Method

- Simple algebra $\Longrightarrow$

$$\begin{aligned} q_t(\mathbf{x},\mathbf{y}) &= f(\mathbf{y}) + \langle \mathbf{x}-\mathbf{y}, \nabla f(\mathbf{y})\rangle + \frac{1}{2t}\|\mathbf{x}-\mathbf{y}\|^2, \\ &= \frac{1}{2t}\|\mathbf{x}-(\mathbf{y}-t\nabla f(\mathbf{y}))\|^2 - \frac{t}{2}\|\nabla f(\mathbf{y})\|^2 + f(\mathbf{y}). \end{aligned}$$

- Allows to easily pass from the unconstrained minimization problem (U) to constrained model:

$$(P) \quad \min\left\{f(\mathbf{x}) : \mathbf{x} \in C\right\},$$

- Ignoring the constant terms (in $\mathbf{y} := \mathbf{x}_{k-1}$) leads to solve (P) via:

$$\mathbf{x}_k = \operatorname*{argmin}_{\mathbf{x}\in C} \frac{1}{2}\left\|\mathbf{x}-(\mathbf{x}_{k-1}-t_k\nabla f(\mathbf{x}_{k-1}))\right\|^2, \; k = 1, \ldots$$

which recovers the Gradient Projection Method (GPM):

$$\mathbf{x}_k = \Pi_C(\mathbf{x}_{k-1} - t_k \nabla f(\mathbf{x}_{k-1})).$$



# Back to general Model(M): Smooth+Nonsmooth

- Naturally suggest to consider the following approximation in place of $f(\mathbf{x}) + g(\mathbf{x})$:

$$q(\mathbf{x},\mathbf{y}) = f(\mathbf{y}) + \langle \mathbf{x}-\mathbf{y}, \nabla f(\mathbf{y})\rangle + \frac{1}{2t}\|\mathbf{x}-\mathbf{y}\|^2 + g(\mathbf{x}).$$

That is, **leaving the nonsmooth part $g(\cdot)$ untouched**.
- In accordance with previous framework, the scheme reads:

$$\mathbf{x}_k = \operatorname*{argmin}_{\mathbf{x}\in\mathbb{E}} \left\{ g(\mathbf{x}) + \frac{1}{2t_k}\|\mathbf{x}-(\mathbf{x}_{k-1}-t_k\nabla f(\mathbf{x}_{k-1}))\|^2 \right\}$$

- This reveals the fundamental **proximal operator.** For any $t > 0$, the proximal map associated with $g$ at $\mathbf{z}$ is defined by

$$\operatorname{prox}_t(g)(\mathbf{z}) = \operatorname*{argmin}_{\mathbf{u}\in\mathbb{E}} \left\{ g(\mathbf{u}) + \frac{1}{2t}\|\mathbf{u}-\mathbf{z}\|^2 \right\}.$$

- Thus, the scheme is *a proximal step at a gradient iteration* for $f$ will be called the *proximal gradient* method, and reads as:

$$\mathbf{x}_k = \operatorname{prox}_{t_k}(g)(\mathbf{x}_{k-1} - t_k\nabla f(\mathbf{x}_{k-1})).$$

# The Fixed Point Approach for (M)

- Alternative derivation of the prox-grad via the optimality condition: $\mathbf{x}^* \in \operatorname{argmin}\{f(\mathbf{x}) + g(\mathbf{x})\}$ iff $\mathbf{0} \in \nabla f(\mathbf{x}^*) + \partial g(\mathbf{x}^*)$.
- Fix any $t > 0$, then the following equivalent statements hold:

$$\begin{aligned} \mathbf{0} &\in t\nabla f(\mathbf{x}^*) - \mathbf{x}^* + \mathbf{x}^* + t\partial g(\mathbf{x}^*), \\ (I + t\partial g)(\mathbf{x}^*) &\in (I - t\nabla f)(\mathbf{x}^*), \\ \mathbf{x}^* &\in (I + t\partial g)^{-1}(I - t\nabla f)(\mathbf{x}^*), \end{aligned}$$

- Last equation naturally calls for a *fixed point scheme*:

$$\mathbf{x}_0 \in \mathbb{E}, \quad \mathbf{x}_k = (I + t_k\partial g)^{-1}(I - t_k\nabla f)(\mathbf{x}_{k-1}) \quad (t_k > 0).$$

  But $(I + t_k\partial g)^{-1} = \operatorname{prox}_{t_k}(g)$ i.e., this is the prox-grad.
- **Note:** A special case of the **proximal backward-forward** scheme,(Passty 77), devised for solving the general inclusion:

$$\text{Find } \mathbf{x}^* \text{ s.t. } \mathbf{0} \in T_1(\mathbf{x}^*) + T_2(\mathbf{x}^*)$$

  $T_1, T_2$ are *maximal monotone set valued maps*
  (with $f, g$ convex $T_1 := \nabla f$, $T_2 := \partial g$).



# Majorization-Minimization - MM Approach

- A popular technique in statistical-engineering literature (other names: surrogate/transfer function, and bound optimization technique..)
- In fact MM follows the same previous approximation idea, except that the approximation **needs not to be quadratic.**
- Find a "relevant" approximation to the objective function $F$ s.t.
  **(i)** $M(\mathbf{x}, \mathbf{x}) = F(\mathbf{x})$ for every $\mathbf{x} \in \mathbb{E}$.
  **(ii)** $M(\mathbf{x}, \mathbf{y}) \geq F(\mathbf{x})$ for every $\mathbf{x}, \mathbf{y} \in \mathbb{E}$.
- From here a natural and simple minimization scheme is

$$\mathbf{x}_k \in \operatorname*{argmin}_{\mathbf{x}\in\mathbb{E}} M(\mathbf{x}, \mathbf{x}_{k-1}) \;\Rightarrow\; M(\mathbf{x}_k, \mathbf{x}_{k-1}) \leq M(\mathbf{x}, \mathbf{x}_{k-1}), \;\forall \mathbf{x}$$

  Easy to see that this scheme produces a descent scheme for $F$:

$$F(\mathbf{x}_k) \overset{(ii)}{\leq} M(\mathbf{x}_k, \mathbf{x}_{k-1}) \leq M(\mathbf{x}_{k-1}, \mathbf{x}_{k-1}) \overset{(i)}{=} F(\mathbf{x}_{k-1}).$$

- Key question: how to generate/find a "good" $M(\cdot, \cdot)$?
- There does not exist a universal rule to determine $M$. Most often structure of the problem at hand provides helpful hints.

# II. Mathematical Tools

- Properties of main computational objects
- Some key generic inequalities
- Serve as main vehicle to establish:
  – convergence rate results of the proximal gradient method
  – its special cases just discussed
  – the improved versions and extensions..later on.



# The Proximal Map (Moreau - (1964))

**Theorem [Moreau-(64)]** Let $g : \mathbb{E} \to (-\infty, \infty]$ be closed proper convex. For any $t > 0$, let

$$g_t(\mathbf{z}) = \min_{\mathbf{u}} \left\{ g(\mathbf{u}) + \frac{1}{2t}\|\mathbf{u} - \mathbf{z}\|^2 \right\}. \tag{1}$$

1. $\min\{g_t(\mathbf{z}) : \ z \in \mathbb{E}\} = \min\{g(\mathbf{u}) : \ u \in \mathbb{E}\}$.
2. The minimum in (1) is attained at the *unique* point

$$\text{prox}_t(g)(\mathbf{z}) = (I + t\partial g)^{-1}(\mathbf{z}) \text{ for every } \mathbf{z} \in \mathbb{E},$$

   and the map $(I + t\partial g)^{-1}$ is single valued from $\mathbb{E}$ into itself.
3. The function $g_t(\cdot)$ is $C^{1,1}$ convex on $\mathbb{E}$ with a $\frac{1}{t}$-Lipschitz gradient:

$$\nabla g_t(\mathbf{z}) = \frac{1}{t}(I - \text{prox}_t(g)(\mathbf{z})) \text{ for every } \mathbf{z} \in \mathbb{E}.$$

## Examples

- **Computing $\text{prox}_t(g)$ can be very hard..If at all possible..!.?..**
- But, for many useful special cases can be easy...
- If $g \equiv \delta_C$, ($C \subseteq \mathbb{E}$ closed and convex), then

$$\begin{aligned}
\text{prox}_t(g)(\mathbf{x}) &= \underset{\mathbf{u}}{\text{argmin}}\{\delta_C(\mathbf{u}) + \frac{1}{2t}\|\mathbf{u}-\mathbf{x}\|^2\} \\
&= \text{argmin}\{\frac{1}{2t}\|\mathbf{u}-\mathbf{x}\|^2 : \ \mathbf{u} \in C\} \\
&= (I + t\partial g)^{-1}(\mathbf{x}) = \Pi_C(\mathbf{x}), \text{ the ortho projection on } C \\
\Longrightarrow & \quad \mathbf{g_t}(\mathbf{x}) = \|\mathbf{x} - \mathbf{\Pi_C}(\mathbf{x})\|^2, \text{ convex and } \mathbf{C^{1,1}}.
\end{aligned}$$

For some useful sets $C$ easy to compute $\Pi_C$:

- Affine sets, Simple Polyhedral Sets (halfspace, $\mathbb{R}^n_+$, $[l,u]^n$),
- $l_2, l_1, l_\infty$ - Balls,
- Ice Cream Cone, Semidefinite Cone $S^n_+$,
- Simplex and Spectrahedron (Simplex in $S^n$).



## Some Calculus Rules for Computing $\text{prox}_t(g)$

$$\text{prox}_t(g)(\mathbf{x}) = \underset{\mathbf{u}}{\text{argmin}}\left\{g(\mathbf{u}) + \frac{1}{2t}\|\mathbf{u}-\mathbf{x}\|^2\right\}.$$

| $g(\mathbf{u})$ | $\text{prox}_t(g)(\mathbf{x})$ |
|---|---|
| $\delta_C(\mathbf{u})$ | $\Pi_C(\mathbf{x})$ |
| $\delta^*_C(\mathbf{u})$ -support function- | $\mathbf{x} - \Pi_C(\mathbf{x})$ |
| $d_C(\mathbf{u})$ | $\begin{cases} \mathbf{x} + \frac{(\Pi_C(\mathbf{x})-\mathbf{x})}{t d_C(\mathbf{x})} & \text{if } d_C(\mathbf{x}) > 1/t \\ \mathbf{x} & \text{otherwise} \end{cases}$ |
| $\|\mathbf{Ax}-\mathbf{b}\|^2/2, \mathbf{A} \in \mathbb{R}^{m\times n}$ | $(I + t^{-1}\mathbf{A}^T\mathbf{A})^{-1}(\mathbf{x} + t^{-1}\mathbf{A}^T\mathbf{b})$ |
| $\|\mathbf{u}\|_1$ | (-shrinkage-) $\text{sgn}(x_j)\max\{\lvert x_j\rvert - t, 0\}$ |
| $\|\mathbf{u}\|$ | $\begin{cases} \|\mathbf{x}\|^2/2t & \text{if } \|\mathbf{x}\| \le t \\ \|\mathbf{x}\| - t/2 & \text{otherwise} \end{cases}$ |
| $\|\mathbf{U}\|_*$, $\mathbf{U} \in \mathbb{R}^{m\times n}$, $(m \ge n)$ | $\mathbf{P}\,\text{diag}(\mathbf{s})\mathbf{Q}^T$ |

- $\sigma_1(\mathbf{U}) \ge \sigma_2(\mathbf{U}) \ge \ldots$ singular values of $\mathbf{U}$
- Nuclear norm $\|\mathbf{U}\|_* = \sum_j \sigma_j(\mathbf{U})$
- Singular value decomposition

$$\mathbf{U} = \mathbf{P}\,\text{diag}(\sigma)\mathbf{Q}^T, \text{ then shrinkage } s_j = \text{sgn}(\sigma_j)\max\{|\sigma_j| - t, 0\}.$$

## The Prox-Grad Map

- We adopt the following approximation model for $F$. For any $L > 0$, and any $\mathbf{x}, \mathbf{y} \in \mathbb{E}$, define

$$Q_L(\mathbf{x}, \mathbf{y}) := f(\mathbf{y}) + \langle \mathbf{x} - \mathbf{y}, \nabla f(\mathbf{y}) \rangle + \frac{L}{2}\|\mathbf{x} - \mathbf{y}\|^2 + g(\mathbf{x}),$$

and

$$p_L^{f,g}(\mathbf{y}) := \text{argmin}\,\{Q_L(\mathbf{x}, \mathbf{y}) : \mathbf{x} \in \mathbb{E}\} \equiv p_L(\mathbf{y})$$

- Ignoring the constant terms in $\mathbf{y}$, this reduces to :

$$\begin{aligned} p_L(\mathbf{y}) &= \underset{\mathbf{x}\in\mathbb{E}}{\text{argmin}} \left\{ g(\mathbf{x}) + \frac{L}{2}\left\|\mathbf{x} - \left(\mathbf{y} - \frac{1}{L}\nabla f(\mathbf{y})\right)\right\|^2 \right\} \\ &= \text{prox}_{\frac{1}{L}}(g)\left(\mathbf{y} - \frac{1}{L}\nabla f(\mathbf{y})\right) \qquad (2) \end{aligned}$$

- **Blanket assumption:** $\nabla f$ is Lipschitz on $\mathbb{E}$, $(f \in C^{1,1}_{L(f)})$, namely:

$$\exists\, L(f) > 0 \;:\; \|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \le L(f)\|\mathbf{x} - \mathbf{y}\| \text{ for every } \mathbf{x}, \mathbf{y} \in \mathbb{E}.$$



## Key Inequalities–Lemma 1

**Lemma 1 - [Descent Lemma]** Let $f : \mathbb{E} \to (-\infty, \infty)$ be $C^{1,1}_{L(f)}$. Then for any $L \ge L(f)$,

$$f(\mathbf{x}) \le f(\mathbf{y}) + \langle \mathbf{x} - \mathbf{y}, \nabla f(\mathbf{y}) \rangle + \frac{L}{2}\|\mathbf{x} - \mathbf{y}\|^2 \text{ for every } \mathbf{x}, \mathbf{y} \in \mathbb{E}.$$

**Proof.** Mean value integral Theorem + Gradient Lipschitz. $\square$

# Key Inequalities–Lemma 2

**Lemma 2 - Prox Inequality** Let $\boldsymbol{\xi} = \text{prox}_{1/t}(g)(\mathbf{z})$ for some $\mathbf{z} \in \mathbb{E}$ and let $t > 0$. Then for any $\mathbf{u} \in \text{dom}\, g$,

$$\begin{aligned} 2t(g(\boldsymbol{\xi}) - g(\mathbf{u})) &\leq 2\langle \mathbf{u} - \boldsymbol{\xi}, \boldsymbol{\xi} - \mathbf{z} \rangle \\ &= \|\mathbf{u} - \mathbf{z}\|^2 - \|\mathbf{u} - \boldsymbol{\xi}\|^2 - \|\boldsymbol{\xi} - \mathbf{z}\|^2. \end{aligned}$$

**Proof.** Use optimality + convexity of $g$. □



# Key Inequalities - for prox-grad $p_L$-Lemma 3

Since $p_L(\mathbf{y}) = \text{prox}_{1/L}(g)\left(\mathbf{y} - \frac{1}{L}\nabla f(\mathbf{y})\right)$, invoking previous Lemma 2, we now obtain a useful inequality for $p_L$.
For further reference we denote for any $\mathbf{y} \in \mathbb{E}$:

$$\xi_L(\mathbf{y}) := \mathbf{y} - \frac{1}{L}\nabla f(\mathbf{y}). \tag{3}$$

**Lemma 3-[prox-grad]** For any $\mathbf{x} \in \text{dom}\, g, \mathbf{y} \in \mathbb{E}$, the prox-grad map $p_L$ satisfies

$$\frac{2}{L}\left[g(p_L(\mathbf{y})) - g(\mathbf{x})\right] \leq \|\mathbf{x} - \boldsymbol{\xi}_L(\mathbf{y})\|^2 - \|\mathbf{x} - p_L(\mathbf{y})\|^2 - \|p_L(\mathbf{y}) - \boldsymbol{\xi}_L(\mathbf{y})\|^2, \tag{4}$$

where $\xi_L(\mathbf{y})$ is given in (3).

**Proof.** Follows from Lemma 2:

$$2t(g(\boldsymbol{\xi}) - g(\mathbf{u})) \leq \|\mathbf{u} - \mathbf{z}\|^2 - \|\mathbf{u} - \boldsymbol{\xi}\|^2 - \|\boldsymbol{\xi} - \mathbf{z}\|^2,$$

with $t := \frac{1}{L}, ; \boldsymbol{\xi} := p_L(\mathbf{y}),\ \mathbf{u} := \mathbf{x};\ \mathbf{z} := \boldsymbol{\xi}_L(\mathbf{y})$. □

# Main Pillar I in Analysis - Proposition I

Our last result combines all the above to produce one of the main pillar of the analysis.

**Proposition I** Let $\mathbf{x} \in \operatorname{dom} g, \mathbf{y} \in \mathbb{E}$ and let $L > 0$ be such that the inequality

$$F(p_L(\mathbf{y})) \leq Q(p_L(\mathbf{y}), \mathbf{y}). \tag{5}$$

is satisfied. Then

$$\frac{2}{L}(F(\mathbf{x}) - F(p_L(\mathbf{y})) \geq \|\mathbf{x} - p_L(\mathbf{y})\|^2 - \|\mathbf{x} - \mathbf{y}\|^2.$$

**Note:** Thanks to the descent lemma condition (5) is always satisfied for $p_L(\mathbf{y})$ with $L \geq L(f)$.



# The Proximal Gradient Method

The proximal gradient method with a constant stepsize rule.

**Proximal Gradient Method with Constant Stepsize**
**Input:** $L = L(f)$ - A Lipschitz constant of $\nabla f$.
**Step 0.** Take $\mathbf{x}_0 \in \mathbb{E}$.
**Step k.** $(k \geq 1)$ Compute

$$\mathbf{x}_k = p_L(\mathbf{x}_{k-1}) = \operatorname*{argmin}_{\mathbf{x} \in \mathbb{E}} \left\{ g(\mathbf{x}) + \frac{L}{2}\|\mathbf{x} - (\mathbf{x}_{k-1} - \frac{1}{L}\nabla f(\mathbf{x}_{k-1}))\|^2 \right\}$$

- An evident possible drawback of the above scheme is that the Lipschitz constant $L(f)$ is not always known or not easily computable.
- This issue can be resolved with an easy backtracking stepsize rule.

# Prox-Grad Method with Backtracking Step Rule

**Proximal Gradient Method with Backtracking**
**Step 0.** Take $L_0 > 0$, some $\eta > 1$ and $\mathbf{x}_0 \in \mathbb{E}$.
**Step k.** $(k \geq 1)$ Find the smallest nonnegative integer $i_k$ such that with, $\bar{L} = \eta^{i_k} L_{k-1}$:

$$F(p_{\bar{L}}(\mathbf{x}_{k-1})) \leq Q_{\bar{L}}(p_{\bar{L}}(\mathbf{x}_{k-1}), \mathbf{x}_{k-1})). \tag{6}$$

Set $L_k = \eta^{i_k} L_{k-1}$ and compute

$$\mathbf{x}_k = p_{L_k}(\mathbf{x}_{k-1}) = \operatorname*{argmin}_{\mathbf{x} \in \mathbb{E}} \left\{ g(\mathbf{x}) + \frac{L_k}{2} \|\mathbf{x} - (\mathbf{x}_{k-1} - \frac{1}{L_k} \nabla f(\mathbf{x}_{k-1}))\|^2 \right\}.$$

.



# Rate of Convergence of Prox-Grad

**Theorem - [Rate of Convergence of Prox-Grad]**
Let $\{\mathbf{x}_k\}$ be the sequence generated by the proximal gradient method with either a constant $(\alpha = 1)$ or a backtracking stepsize rule $(\alpha = \eta)$. Then for every $k \geq 1$:

$$F(\mathbf{x}_k) - F(\mathbf{x}^*) \leq \frac{\alpha L(f) \|\mathbf{x}_0 - \mathbf{x}^*\|^2}{2k}$$

for every optimal solution $\mathbf{x}^*$.

- Thus, to solve (M), the proximal gradient method converges at a *sublinear rate* in function values.
- # iterations for $F(\mathbf{x}_k) - F(\mathbf{x}^*) \leq \epsilon$ is $O(1/\epsilon)$.
- **Note:** The sequence $\{\mathbf{x}_k\}$ can be proven to *converge* to solution $\mathbf{x}^*$ provided a step size is in $(0, 2/L)$.

# Special Cases

- With $g \equiv 0$ and $g = \delta_C$, our model (M) recovers the basic gradient and gradient projection methods respectively.
- With $f = 0$ in (M), this is the *Proximal Minimization Algorithm* described next.



# Proximal Minimization Algorithm-PMA

Set $f \equiv 0$, in (M), i.e., we solve the convex nonsmooth problem

$$\min\{g(\mathbf{x}) : \mathbf{x} \in \mathbb{E}\}.$$

PG reduces to **Proximal Minimization Algorithm (Martinet (70))**:

$$\mathbf{x}_0 \in \mathbb{E}, \ \mathbf{x}_k = \underset{\mathbf{x} \in \mathbb{E}}{\operatorname{argmin}} \left\{ g(\mathbf{x}) + \frac{1}{2t_k} \|\mathbf{x} - \mathbf{x}_{k-1}\|^2 \right\}.$$

This an *implicit* discretization of $0 \in d\mathbf{x}(t)/dt + \partial g(\mathbf{x}(t)), \ \mathbf{x}(0) = \mathbf{x}_0$.

**Theorem** Let $\mathbf{x}_k$ be the sequence generated by PMA, and set $\sigma_k = \sum_{s=1}^{k} t_s$.

$$\text{Then,} \quad g(\mathbf{x}_k) - g(\mathbf{x}) \leq \|\mathbf{x}_0 - \mathbf{x}\|^2 / 2\sigma_k, \forall \mathbf{x} \in \mathbb{E}.$$

In particular, if $\sigma_k \to \infty$ then $g(\mathbf{x}_k) \downarrow g_* = \inf_{\mathbf{x}} g(\mathbf{x})$ and if $X_* \neq \emptyset$, then $\mathbf{x}_k$ converges to some point in $X_*$.

This algorithm is "better" than SM...But is non-implementable, unless $g$ is "simple". Nevertheless, very useful when combined with duality:
$\longrightarrow$ Augmented Lagrangians Methods.

# III-Fast Gradient Schemes – Improving Complexity

Previous **explicit** methods are simple but are often too slow.

- For Prox-Grad and Gradient methods: a complexity rate of $O(1/k)$
- For Subgradient Methods: complexity rate of $O(1/\sqrt{k})$.
- Can we do better to solve the nonsmooth problem (M)?

$$(M) \qquad \min\{F(\mathbf{x}) := f(\mathbf{x}) + g(\mathbf{x}) : \ \mathbf{x} \in \mathbb{E}\}.$$

- Can we devise a method with:
  ♠ the *same computational effort/simplicty as Prox-Grad* .
  ♠ a *Faster* global rate of convergence.



# Yes we Can...

- **Answer: Yes**, through an "equally simple" scheme

$$\clubsuit \ \mathbf{x}_{k+1} = \operatorname*{argmin}_{\mathbf{x}} Q_L(\mathbf{x}, \mathbf{y}_k), \hookleftarrow \ \mathbf{y}_k \text{ instead of } \mathbf{x}_k$$

  The new point $\mathbf{y}_k$ will be smartly chosen and **easy** to compute.
- **Idea:** From an old algorithm of Nesterov (1983) designed for minimizing **a smooth** convex function, and proven to be an *"optimal"* first order method (Yudin-Nemirovsky (80)).
- But, here our problem (M) is **nonsmooth**. Yet, we can derive a faster algorithm than PG for the general NSO problem (M).

Y. Nesterov. A method for solving the convex programming problem with convergence rate $O(1/k^2)$. *Dokl. Akad. Nauk SSSR*, 269(3):543–547, (1983).

# A Fast Prox-Grad Algorithm - [BT09]

An equally simple algorithm as prox-grad. (Here $L(f)$ is known).

**FPG with constant stepsize**
**Input:** $L = L(f)$ - A Lipschitz constant of $\nabla f$.
**Step 0.** Take $\mathbf{y}_1 = \mathbf{x}_0 \in \mathbb{E}$, $t_1 = 1$.
**Step k.** $(k \geq 1)$ Compute

$$\begin{aligned}
\mathbf{x}_k &= \operatorname*{argmin}_{\mathbf{x}\in\mathbb{E}} \left\{ g(\mathbf{x}) + \frac{L}{2}\|\mathbf{x} - (\mathbf{y}_k - \frac{1}{L}\nabla f(\mathbf{y}_k))\|^2 \right\} \\
\mathbf{x}_k &\equiv p_L(\mathbf{y}_k), \quad \hookleftarrow \text{ main computation as Prox-Grad} \\
\bullet \quad t_{k+1} &= \frac{1 + \sqrt{1 + 4t_k^2}}{2}, \\
\bullet\bullet \quad \mathbf{y}_{k+1} &= \mathbf{x}_k + \left(\frac{t_k - 1}{t_{k+1}}\right)(\mathbf{x}_k - \mathbf{x}_{k-1}).
\end{aligned}$$

Additional computation for FPG in $(\bullet)$ and $(\bullet\bullet)$ is clearly marginal.

With $g = 0$, this is the smooth Fast Gradient of Nesterov (83);
With $t_k \equiv 1, \forall k$ we recover ProxGrag (PG).



# Knowledge of $L(f)$ is not Necessary

**FPG with backtracking**
**Step 0.** Take $L_0 > 0$, some $\eta > 1$ and $\mathbf{x}_0 \in \mathbb{E}$. Set $\mathbf{y}_1 = \mathbf{x}_0$, $t_1 = 1$.
**Step k.** $(k \geq 1)$ Find the smallest nonnegative integers $i_k$ such that with $i = i_k$, $\bar{L} = \eta^{i_k} L_{k-1}$:

$$F(p_{\bar{L}}(\mathbf{y}_k)) \leq Q_{\bar{L}}(p_{\bar{L}}(\mathbf{y}_k), \mathbf{y}_k).$$

Set $L_k = \eta^{i_k} L_{k-1}$ and compute

$$\begin{aligned}
\mathbf{x}_k &= p_{L_k}(\mathbf{y}_k), \\
t_{k+1} &= \frac{1 + \sqrt{1 + 4t_k^2}}{2}, \\
\mathbf{y}_{k+1} &= \mathbf{x}_k + \left(\frac{t_k - 1}{t_{k+1}}\right)(\mathbf{x}_k - \mathbf{x}_{k-1}).
\end{aligned}$$

# Theorem - Global Rate of Convergence FPG

**Theorem – [BT09]** Let $\{\mathbf{x}_k\}$ be generated by FPG. Then for any $k \geq 1$

$$F(\mathbf{x}_k) - F(\mathbf{x}^*) \leq \frac{2\alpha L(f)\|\mathbf{x}_0 - \mathbf{x}^*\|^2}{(k+1)^2},$$

where $\alpha = 1$ for the constant stepsize setting and $\alpha = \eta$ for the backtracking stepsize setting.

- # of iterations to reach $F(\tilde{\mathbf{x}}) - F_* \leq \varepsilon$ is $\sim O(1/\sqrt{\varepsilon})$.
- Clearly improves PG by **a square root factor**.
- **Do we practically achieve this theoretical rate?..Example Soon**



# Main Pillar II in Analysis - Proposition II

**Proposition II-Recursion** The sequences $\{\mathbf{x}_k, \mathbf{y}_k\}$ generated via the fast proximal gradient method with either a constant or backtracking stepsize rule satisfy for every $k \geq 1$

$$\frac{2}{L_k} t_k^2 v_k - \frac{2}{L_{k+1}} t_{k+1}^2 v_{k+1} \geq \|\mathbf{u}_{k+1}\|^2 - \|\mathbf{u}_k\|^2,$$

where

$$\begin{aligned} v_k &:= F(\mathbf{x}_k) - F(\mathbf{x}^*), \\ \mathbf{u}_k &:= t_k \mathbf{x}_k - (t_k - 1)\mathbf{x}_{k-1} - \mathbf{x}^*. \end{aligned}$$

Proof relies on Proposition I and the recursion for $\{t_k\}$.

# A Different $O(1/k^2)$ algorithm for solving (M)

**Nesterov (2007)**: Gradient methods for minimizing composite objective function. CORE Report. Available at http://www.ecore.beDPs/dp1191313936.pdf.

◦ Same iteration complexity bound $O(1/k^2)$ like FPG.
◦ Depends **on the accumulated history of past gradient iterates**
◦ **Requires two prox** operations at each iteration.
◦ Totally different nontrivial convergence analysis.



# Application: Linear Inverse Problems

**Problem: Find $\mathbf{x} \in C \subset \mathbb{E}$ which "best" solves $\mathcal{A}(\mathbf{x}) \approx \mathbf{b}$,** $\mathcal{A} : \mathbb{E} \to \mathbb{F}$, where $\mathbf{b}$ (observable output), and $\mathcal{A}$ (Blurring matrix) are known.

**Approach: via Regularization Models**

- $g(\mathbf{x})$ is a "regularizer" (one – or sum of functions)
- $d(\mathbf{b}, \mathcal{A}(\mathbf{x}))$ some "proximity" measure from $\mathbf{b}$ to $\mathcal{A}(\mathbf{x})$

$$\begin{aligned}
&\min \quad \{g(\mathbf{x}) : \ \mathcal{A}(\mathbf{x}) = \mathbf{b}, \ \mathbf{x} \in C\} \\
&\min \quad \{g(\mathbf{x}) : \ d(\mathbf{b}, \mathcal{A}(\mathbf{x})) \leq \epsilon, \ \mathbf{x} \in C\} \\
&\min \quad \{d(\mathbf{b}, \mathcal{A}(\mathbf{x})) : \ g(\mathbf{x}) \leq \delta, \ \mathbf{x} \in C\} \\
&\min \quad \{d(\mathbf{b}, \mathcal{A}(\mathbf{x})) + \lambda g(\mathbf{x}) : \ \mathbf{x} \in C\} \ (\lambda > 0) \longleftarrow
\end{aligned}$$

• Intensive research activities over the last 50 years...Now, more...with Sparse Optimization problems..
• Choices for $g(\cdot)$, $d(\cdot, \cdot)$ depends on the application at hand.
**Nonsmooth** regularizers are particularly useful.

## Special Cases: $f(\mathbf{x}) = d(\mathbf{b}, \mathcal{A}(\mathbf{x})) := \|\mathcal{A}(\mathbf{x}) - \mathbf{b}\|^2$

- $g = \lambda\|\cdot\|_1$ - $l_1$-regularized convex problem.

$$\min_{\mathbf{x}}\{f(\mathbf{x}) + \lambda\|\mathbf{Lx}\|_1\}$$

  L - identity, differential operator, wavelet.
- $g = TV(\cdot)$ - Total Variation-based regularization (Rudin-Osher-Fatemi (92)).

$$\min_{\mathbf{x}}\{f(\mathbf{x}) + \lambda TV(\mathbf{x})\}$$

  1-dim: $TV(x) = \sum_i |x_i - x_{i+1}|$
  2-dim:
  isotropic: $\mathrm{TV}(\mathbf{x}) = \sum_i \sum_j \sqrt{(x_{i,j} - x_{i+1,j})^2 + (x_{i,j} - x_{i,j+1})^2}$
  anisotropic: $\mathrm{TV}(\mathbf{x}) = \sum_i \sum_j (|x_{i,j} - x_{i+1,j}| + |x_{i,j} - x_{i,j+1}|)$
- In Image Processing:
  When $\mathcal{A} = I$, this is called *image denoising*=prox
  When $\mathcal{A} \neq I$, this is *Image Deblurring*.



## Example $l_1$ regularization -PG = ISTA

$$\min_{\mathbf{x}}\{\|\mathbf{Ax} - \mathbf{b}\|^2 + \lambda\|\mathbf{x}\|_1\} \equiv \min_{\mathbf{x}}\{f(\mathbf{x}) + g(\mathbf{x})\}$$

The proximal map of $g(\mathbf{x}) = \lambda\|\mathbf{x}\|_1$ is simply:

$$\mathrm{prox}_t(g)(\mathbf{y}) = \operatorname*{argmin}_{\mathbf{u}}\left\{\frac{1}{2t}\|\mathbf{u} - \mathbf{y}\|^2 + \lambda\|\mathbf{u}\|_1\right\} = \mathcal{T}_{\lambda t}(\mathbf{y}),$$

where $\mathcal{T}_\alpha : \mathbb{R}^n \to \mathbb{R}^n$ is the shrinkage or soft threshold operator:

$$\mathcal{T}_\alpha(\mathbf{x})_i = (|x_i| - \alpha)_+ \mathrm{sgn}\,(x_i). \tag{7}$$

The Prox Grad method is the so-called *Iterative Shrinkage/Thresholding Algorithm* (ISTA).

Other names in the signal processing literature include for example: threshold Landweber method, iterative denoising, deconvolution algorithms...

# PG=ISTA and FPG=FISTA

**ISTA** with Constant Stepsize $L = L(f) = 2\lambda_{\max}(\mathbf{A}^T\mathbf{A})$. Lipschitz constant of $\nabla f$

$$\mathbf{x}_0 \in \mathbb{E}, \mathbf{x}_k = \mathcal{T}_{\lambda/L}\left(\mathbf{x}_{k-1} - \frac{2}{L}\mathbf{A}^T(\mathbf{A}\mathbf{x}_{k-1} - \mathbf{b})\right)$$

**FISTA** with constant stepsize $L = \lambda_{\max}(\mathbf{A}^T\mathbf{A})$.
$\mathbf{y}_1 = \mathbf{x}_0 \in \mathbb{E}, \; t_1 = 1.$

$$\begin{aligned}
\mathbf{x}_k &= \mathcal{T}_{\lambda/L}\left(\mathbf{y}_k - \frac{2}{L}\mathbf{A}^T(\mathbf{A}\mathbf{y}_k - \mathbf{b})\right), \\
t_{k+1} &= \frac{1 + \sqrt{1 + 4t_k^2}}{2}, \\
\mathbf{y}_{k+1} &= \mathbf{x}_k + \left(\frac{t_k - 1}{t_{k+1}}\right)(\mathbf{x}_k - \mathbf{x}_{k-1}).
\end{aligned}$$



# A Numerical Example: $l_1$-Image Deblurring

$$\min_{\mathbf{x}}\{\|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2 + \lambda\|\mathbf{x}\|_1\}$$

Comparing ISTA versus FISTA on Problems
• dimension $d$ like
$d = 256 \times 256 = 65,536,$ or/and $512 \times 512 = 262,144.$
• The $d \times d$ matrix $\mathbf{A}$ is **dense** (Gaussian blurring times inverse of two-stage Haar wavelet transform).
• All problems solved with **fixed** $\lambda$ and Gaussian noise.

# Deblurring of the Cameraman

original

blurred and noisy



# 1000 Iterations of ISTA versus 200 of FISTA

ISTA: **1000 Iterations**

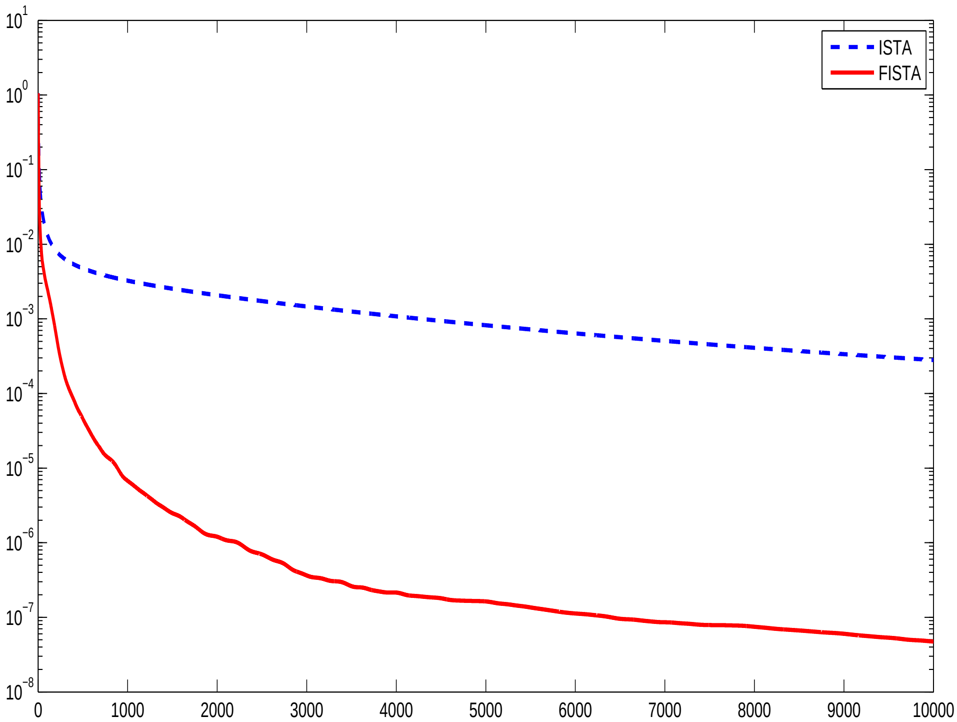
FISTA: **200 Iterations**

# Original Versus Deblurring via FISTA

Original FISTA:1000 Iterations



# Function Values errors $F(\mathbf{x}_k) - F(\mathbf{x}^*)$



# Example 2: $l_1$ versus TV Regularization

Main difference between $l_1$ and TV regularization:

- prox of $l_1$ - simple and explicit (shrinkage/soft threshold).
- prox of TV - TV-denoising problem requires an iterative method:
- **g=TV**

$$\mathbf{x}_{k+1} = D\left(\mathbf{x}_k - \frac{2}{L}\mathbf{A}^T(\mathbf{A}\mathbf{x}_k - \mathbf{b}), \frac{2\lambda}{L}\right).$$

where $D(\mathbf{w}, t) = \underset{\mathbf{x}}{\text{argmin}} \left\{\|\mathbf{x} - \mathbf{w}\|^2 + 2t\text{TV}(\mathbf{x})\right\}$

Here:

- Prox operation $\Leftrightarrow$ TV-based denoising
- **No analytic expression in this case.** Still can be solved very efficiently by solving a *smooth dual* formulation by a fast gradient method.



# Total Variation-Based Denoising via Dual

$$\text{(DenP)} \quad \min_{\mathbf{x}\in C}\{\|\mathbf{x} - \mathbf{b}\|_F^2 + 2\lambda\text{TV}(\mathbf{x})\}, \ \mathbf{A} \equiv I$$

**Nonsmootness** handled via the **dual approach – Chambolle (04).**
**Result:** Let $(\mathbf{p}, \mathbf{q}) \in \mathcal{P}$ be the optimal solution of the dual problem

$$\min\left\{h(\mathbf{p}, \mathbf{q}) \equiv -\|H_C(\mathbf{b} - \lambda\mathcal{L}(\mathbf{p}, \mathbf{q}))\|_F^2 + \|\mathbf{b} - \lambda\mathcal{L}(\mathbf{p}, \mathbf{q})\|_F^2 : \ (\mathbf{p}, \mathbf{q}) \in \mathcal{P}\right\}$$

where $H_C(\mathbf{x}) = \mathbf{x} - P_C(\mathbf{x})$ for every $\mathbf{x} \in \mathbb{R}^{m\times n}$.

- **Optimal solution of (DenP):** $\mathbf{x} = P_C(\mathbf{b} - \lambda\mathcal{L}(\mathbf{p}, \mathbf{q}))$.
- **The dual $h \in C^{1,1}$ is convex:**
  $\nabla h(\mathbf{p}, \mathbf{q}) = -2\lambda\mathcal{L}^T P_C(\mathbf{b} - \lambda\mathcal{L}(\mathbf{p}, \mathbf{q})), \ L_h \leq 16\lambda^2$.
- Gradient Projection can be applied on dual $h$ (Chambolle (04), (05)).
- **Here we can thus apply a "Fast Gradient Projection" (FGP) (FISTA with $g = 0$)**

# A Fast Denoising Method – Algorithm FGP($\mathbf{b}, \lambda, N$)

**Input:** $\mathbf{b}$ - observed image, $\lambda$ - reg. param., $N$ - Number of iterations.
**Output:** $\mathbf{x}^*$ - An optimal solution of DenP (up to a tolerance).

**Step 0.** Take $(\mathbf{r}_1, \mathbf{s}_1) = (\mathbf{p}_0, \mathbf{q}_0) = (\mathbf{0}_{(m-1)\times n}, \mathbf{0}_{m\times(n-1)})$, $t_1 = 1$.
**Step k.** ($k = 1, \ldots, N$) Compute

$$\begin{aligned}
(\mathbf{p}_k, \mathbf{q}_k) &= P_{\mathcal{P}}\left[(\mathbf{r}_k, \mathbf{s}_k) + \frac{1}{8\lambda}\mathcal{L}^T\left(P_C[\mathbf{b} - \lambda\mathcal{L}(\mathbf{r}_k, \mathbf{s}_k)]\right)\right], \\
t_{k+1} &= \frac{1 + \sqrt{1 + 4t_k^2}}{2}, \\
(\mathbf{r}_{k+1}, \mathbf{s}_{k+1}) &= (\mathbf{p}_k, \mathbf{q}_k) + \left(\frac{t_k - 1}{t_{k+1}}\right)(\mathbf{p}_k - \mathbf{p}_{k-1}, \mathbf{q}_k - \mathbf{q}_{k-1}).
\end{aligned}$$

**Set** $\mathbf{x}^* = P_C[\mathbf{b} - \lambda\mathcal{L}(\mathbf{p}_N, \mathbf{q}_N)]$

**Projections on $\mathcal{P}$ are exact formula. For $C$ as usual when "simple".**



# Total Variation-Based Deblurring

- $\min_{\mathbf{x}\in C} \|\mathcal{A}(\mathbf{x}) - \mathbf{b}\|_F^2 + 2\lambda \mathrm{TV}(\mathbf{x})$
- $f(\mathbf{x}) \equiv \|\mathcal{A}(\mathbf{x}) - \mathbf{b}\|^2$, $\quad g(\mathbf{x}) \equiv 2\lambda\mathrm{TV}(\mathbf{x}) + \delta_C(\mathbf{x})$, $\ \mathbb{E} = \mathbb{R}^{m\times n}$.
- Deblurring is of course more challenging than denoising.
- An equivalent smooth optimization problem via its dual needs to invert the operator $\mathcal{A}^T\mathcal{A}$...In general not viable.
- No analytical expression for "prox" step in FISTA...But again duality helps..

To avoid this difficulty, the TV deblurring problem can be treated in two steps through the denoising problem solved via **dual** with FGP:

$$\begin{aligned}
D_C(\mathbf{z}, \alpha) &:= \operatorname{argmin}\{\|\mathbf{x} - \mathbf{z}\|^2 + 2\alpha\mathrm{TV}(\mathbf{x}) : \mathbf{x} \in C\} \ (Denoising step) \\
p_L(\mathbf{Y}) &= D_C\left(\mathbf{Y} - \frac{2}{L}\mathcal{A}^T(\mathcal{A}(\mathbf{Y}) - \mathbf{b}), \frac{2\lambda}{L}\right) \ (FISTA step).
\end{aligned}$$

# FPG=FISTA is NOT a Monotone Method!

- **FISTA is not a monotone method.**
- In practice, "almost always" monotone.
- No effect on the convergence properties when the prox operation can be computed exactly.
- Might have severe effects when the prox-subproblems **cannot** be solved exactly, e.g., for TV based deblurring.



# MFISTA: Monotone FISTA

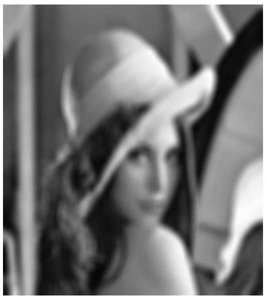
**Input:** $L \geq L(f)$ - An upper bound on the Lipschitz constant of $\nabla f$.
**Step 0.** Take $\mathbf{y}_1 = \mathbf{x}_0 \in \mathbb{E},\ t_1 = 1$.
**Step k.** $(k \geq 1)$ Compute

$$\begin{aligned}
\mathbf{z}_k &= p_L(\mathbf{y}_k), \\
t_{k+1} &= \frac{1+\sqrt{1+4t_k^2}}{2}, \\
\mathbf{x}_k &= \operatorname{argmin}\{F(\mathbf{x}) :\ \mathbf{x} = \mathbf{z}_k, \mathbf{x}_{k-1}\} \\
\mathbf{y}_{k+1} &= \mathbf{x}_k + \left(\frac{t_k}{t_{k+1}}\right)(\mathbf{z}_k - \mathbf{x}_k) + \left(\frac{t_k - 1}{t_{k+1}}\right)(\mathbf{x}_k - \mathbf{x}_{k-1}).
\end{aligned}$$

**With Same Rate of Convergence as FPG!**

# Lena and 3 Reconstructions – N=100 Iterations

Blurred and Noisy



ISTA($F_{100} = 0.606$)

MFISTA($F_{100} = 0.466$)



# Applications/Limitations of FISTA for (M)

$$(M) \min\{f(\mathbf{x}) + g(\mathbf{x}) : \mathbf{x} \in \mathbb{E}\}$$

The smooth convex function can be of any type $f \in C^{1,1}$ with available gradient.

As long as the **prox** of the nonsmooth function $g$

$$p_L(\mathbf{y}) = \operatorname*{argmin}_{\mathbf{x}\in\mathbb{E}} \left\{ g(\mathbf{x}) + \frac{L}{2}\|\mathbf{x} - (\mathbf{y} - \frac{1}{L}\nabla f(\mathbf{y}))\|^2 \right\}$$

can be computed analytically or easily/efficiently, via some other approach (e.g., dual for TV),
FISTA (MFISTA) is useful and efficient.

As seen previously, (see Prox-Calculus Table) FISTA covers some interesting models in

- Signal/image recovery problems
- Matrix minimization problems arising in many machine learning models, (e.g., nuclear matrix norm regularization, multi-task learning, matrix classification, matrix completion problems.)

# IV-Gradient Schemes with Non-Euclidean Distances

- All previous schemes were based on using the squared Euclidean distance for measuring proximity of two points in $\mathbb{E}$
- It is useful to exploit the *geometry of the constraints*
- This is done by selecting a "distance-like" function that sometimes can reduce computational costs or even improve the rate of convergence.

1. Mirror Descent Algorithms
2. More on Fast Gradient Schemes
3. Building Gradient Schemes via Algorithms for Variational Inequalities



# A Proximal Distance-Like Function

Exploiting the Geometry of $C$

- Usual gradient method reads:

$$y = \operatorname*{argmin}_{\boldsymbol{\xi} \in C} \{ t \langle \boldsymbol{\xi}, \nabla f(\mathbf{x}) \rangle + \frac{1}{2} \|\boldsymbol{\xi} - \mathbf{x}\|^2 \}, \; t > 0.$$

- Replace $\|\cdot\|^2$ by some **distance-like** $d(\cdot,\cdot)$ that better exploits $C$ (e.g., allows for deriving **explicit and simple** formula) through a **Projection-Like Map:**

$$p(\mathbf{g}, \mathbf{x}) := \operatorname*{argmin}_{\mathbf{v}} \{ \langle \mathbf{v}, \mathbf{g} \rangle + d(\mathbf{v}, \mathbf{x}) \}.$$

- **Minimal required properties for $d$:**

  $d(\cdot, \mathbf{v})$ is a convex function, $\forall \mathbf{v}$
  $d(\cdot,\cdot) \geq 0$, and $d(\mathbf{u}, \mathbf{v}) = 0$ iff $\mathbf{u} = \mathbf{v}$ $\forall \mathbf{u}, \mathbf{v}$.
  • $d$ **is not a distance**: no symmetry or/and triangle inequality

# Two Generic Families for Proximal Distances $d$

- Bregman type distances - based on kernel $\psi$:

$$D_\psi(\mathbf{x},\mathbf{y}) = \psi(\mathbf{x}) - \psi(\mathbf{y}) - \langle \mathbf{x}-\mathbf{y}, \nabla\psi(\mathbf{y})\rangle, \ \psi \text{ strongly convex}$$

- $\Phi$-divergence type distances - based on 1-d kernel $\phi$ convex

$$d_\varphi(\mathbf{x},\mathbf{y}) := \sum_{j=1}^n y_j^r \varphi(\frac{x_j}{y_j}) + \frac{\sigma}{2}\|\mathbf{x}-\mathbf{y}\|^2, \ r = 1, 2; \ \varphi \text{ convex on } \mathbb{R}.$$

The choice of $d$ should be dictated to

♠ best match the constraints of a given problem
♠ to simplify the projection-like computation.



# Examples

- **Example 1** The choice $\psi(\mathbf{z}) = \frac{1}{2}\|\mathbf{z}\|^2$ yields the usual squared Euclidean norm distance $D_\psi(\mathbf{x},\mathbf{y}) = \frac{1}{2}\|\mathbf{x}-\mathbf{y}\|^2$.
- **Example 2** The entropy-like distance defined on the simplex,

$$\psi(\mathbf{z}) = \sum_{j=1}^d z_j \ln z_j, \ \text{for } \mathbf{z} \in \Delta_d = \{\mathbf{z} \in \mathbb{R}^d : \sum_{j=1}^d z_j = 1, \mathbf{z} > \mathbf{0}\}.$$

- In that case, $D_\psi(\mathbf{x},\mathbf{y}) = \sum_{j=1}^d x_j \ln \frac{x_j}{y_j}$ and the following holds:

$$D_\psi(\mathbf{x},\mathbf{y}) \geq \frac{1}{2}\|\mathbf{x}-\mathbf{y}\|_1^2 \text{ for every } \mathbf{x},\mathbf{y} \in \Delta_d,$$

namely, $D_\psi$ is 1-strongly convex with respect to the $l_1$ norm.

More examples soon...

# Pythagoras...Without Squares...

A very simple but key property of *Bregman distances*.
Plays a crucial role in the analysis of any optimization method based on Bregman distances.

**Lemma (The three points identity - C.-T(93))**

*For any three points* $\mathbf{x}, \mathbf{y} \in int(dom\,\psi)$ *and* $\mathbf{z} \in dom\,\psi$*, the following three points identity holds true*

$$D_\psi(\mathbf{z}, \mathbf{y}) - D_\psi(\mathbf{z}, \mathbf{x}) - D_\psi(\mathbf{x}, \mathbf{y}) = \langle \mathbf{z} - \mathbf{x}, \nabla\psi(\mathbf{x}) - \nabla\psi(\mathbf{y}) \rangle.$$

With $\psi(\mathbf{u}) = \|\mathbf{u}\|^2/2$ we recover the classical identity:

$$\|\mathbf{z} - \mathbf{y}\|^2 - \|\mathbf{z} - \mathbf{x}\|^2 - \|\mathbf{x} - \mathbf{y}\|^2 = 2\langle \mathbf{z} - \mathbf{x}, \mathbf{x} - \mathbf{y} \rangle.$$



# The Mirror Descent Algorithm-MDA

$$\min\{g(\mathbf{x}) : \ \mathbf{x} \in C\} \quad \text{Convex Nonsmooth}$$

- Originated from functional analytic arguments in infinite dimensional setting between primal-dual spaces.
  A. S. Nemirovsky and D. B. Yudin. *Problem complexity and method efficiency in optimization* Wiley-Interscience Publication, (1983).
- In (Beck-Teboulle-2003) we have shown that the (MDA) can be simply viewed as a **subgradient method** with a strongly convex Bregman proximal distance:

$$\mathbf{x}_{k+1} = \operatorname*{argmin}_{\mathbf{x}}\{\langle \mathbf{x}, \mathbf{v}_k \rangle + \frac{1}{t_k} D_\psi(\mathbf{x}, \mathbf{x}_k)\}, \ \mathbf{v}_k \in \partial g(\mathbf{x}_k), \ t_k > 0.$$

- **Example: Convex Minimization over the Unit Simplex $\Delta_n$.** Use the *entropy kernel* defined on $\Delta_n$ (is 1-strongly convex w.r.t $\|\cdot\|_1$). Exploiting *geometry* of constraints can improve performance of SM.

# Convex Minimization over the Unit Simplex $\Delta_n$

$$\inf\{g(\mathbf{x}):\ x \in \Delta_n\},\ \Delta_n = \{\mathbf{x} \in \mathbb{R}^n:\ e^T\mathbf{x} = 1, \mathbf{x} \geq 0\}$$

- **EMDA:** Start with $\mathbf{x}^0 = n^{-1}e$. For $k \geq 1$ generate

$$x_j^k = \frac{x_j^{k-1}\exp(-t_k v_j^{k-1})}{\sum_{i=1}^n x_i^{k-1}\exp(-t_k v_i^{k-1})},\ j = 1, \ldots, n\ t_k := \frac{\sqrt{2\log n}}{L_g\sqrt{k}},$$

where $\mathbf{v}^{k-1} := (v_1^{k-1}, \ldots, v_n^{k-1}) \in \partial g(\mathbf{x}_{k-1})$.

**Theorem** The sequence generated by EMDA satisfies for all $k \geq 1$

$$\min_{1\leq s\leq k} f(\mathbf{x}^s) - \min_{\mathbf{x}\in\Delta} f(\mathbf{x}) \leq \sqrt{2\log n}\frac{\max_{1\leq s\leq k}||\mathbf{v}^s||_\infty}{\sqrt{k}}$$

**This outperforms the classical subgradient (based on $\|\cdot\|^2$), by a factor of $(n/\log n)^{1/2}$, which for large $n$ can make a huge difference!....**



# A Fast Non-Euclidean Gradient Method

For the smooth convex case $\min\{f(\mathbf{x}):\ \mathbf{x} \in C\},\ f \in C^{1,1}$ [Auslender-Teboulle (06)].

**A Fast Non-Euclidean Gradient Method**
**Input:** $L = L(f), \sigma > 0, \psi, \sigma$-strongly convex.
**Step 0:** Take $\mathbf{x}_0, \mathbf{z}_0 \in \mathrm{ri}(\mathrm{dom}\,\psi), t_0 = 1$

$$\begin{aligned}
\textbf{Step k: } \text{Compute } \mathbf{y}_k &= (1 - t_k^{-1})\mathbf{x}_k + t_k^{-1}\mathbf{z}_k \\
\mathbf{z}_{k+1} &= \operatorname*{argmin}_{\mathbf{x}} \left\{\langle \mathbf{x}, \nabla f(\mathbf{y}_k)\rangle + \frac{L}{\sigma t_k} D_\psi(\mathbf{x}, \mathbf{z}_k)\right\}, \\
\mathbf{x}_{k+1} &= (1 - t_k^{-1})\mathbf{x}_k + t_k^{-1}\mathbf{z}_{k+1}, \\
t_{k+1} &= \frac{1 + \sqrt{1 + 4t_k^2}}{2},
\end{aligned}$$

Extension of this algorithm for the general model (M) to produce FPG with Bregman distance can be obtained along the same methodology developed for FPG.

# Complexity of Non-Euclidean FPG

### Theorem

*Let $\{\mathbf{x}_k, \mathbf{y}_k, \mathbf{z}_k\}$ be generated by the previous algorithm. Then for all $k \geq 1$,*

$$f(\mathbf{x}_k) - f(\mathbf{x}^*) \leq \frac{4LD_\psi(\mathbf{x}^*, \mathbf{x}_0)}{\sigma(k+1)^2},$$

**Two other schemes :**

- One requires past history of all gradients + 2 prox: one quadratic, and one based on $\psi$;
- the other also requires past history of all gradients, and 2 prox based on $\psi$.

  See, Nesterov. Smooth minimization of non-smooth functions. *Math. Program. Series A*, Vol. 103, 127–152, (2005).



# Gradient Schemes via Variational Inequalities

- $X \subset \mathbb{R}^n$ closed convex set
- $F : X \to \mathbb{R}^n$ monotone map on $X$, i.e.,

$$\langle F(\mathbf{x}) - F(\mathbf{y}), \mathbf{x} - \mathbf{y} \rangle \geq 0, \ \forall \mathbf{x}, \mathbf{y} \in X.$$

### VI Problem

$$\text{Find } \mathbf{x}^* \in X \text{ such that } \langle F(\mathbf{x}^*), \mathbf{x} - \mathbf{x}^* \rangle \geq 0 \quad \forall \mathbf{x} \in X.$$

- VI extend and encompass a broad spectrum of problems: Complementarity, Optimization, Saddle point, Equilibrium...
- Here, $X$ is assumed **"simple"** for the VI.
- This will be exploited to derive schemes **with explicit formulas** for general constrained smooth convex problems as well as some structured nonsmooth problems.
- So, what are "simple" constraints...?..

# Simple Constraints

"Simple" but also fundamental.. $X := \overline{C} \cap V$, $\overline{C}$ closure of $C$ with

$$C \text{ open convex}, V := \{\mathbf{x} \in \mathbb{R}^n : \mathcal{A}(\mathbf{x}) = \mathbf{b}\}, \ \mathcal{A} \text{ linear}, \ \mathbf{b} \in \mathbb{R}^m.$$

- $\mathbb{R}^n_+$,
- unit ball, box constraints,
- $\Delta_n$ the simplex in $\mathbb{R}^n$,
- $S^n_+$ (symmetric semidefinite positive matrices),
- $L^n_+$ the Lorentz cone,
- the Spectrahedron (Simplex in $S^n$)



# Starting Idea: The Extra-Gradient Method

Korpelevich, G. M. Extrapolation gradient methods and their relation to modified Lagrange functions. *Ekonom. i Mat. Metody*, **19** (1976), no. 4, 694–703.

- Provides a "simple cure" to difficulties, and strong assumptions needed in the usual *Projection methods for VI* (e.g., $F$ strongly monotone on $X$)

$$\mathbf{x}^k = \Pi_X(\mathbf{x}^{k-1} - t_k F(\mathbf{x}^{k-1})), \ t_k > 0.$$

- **Extragradient Method-Korpelevich (76):**

$$\mathbf{y}^{k-1} = \Pi_X(\mathbf{x}^{k-1} - \beta_k F(\mathbf{x}^{k-1})), \quad \mathbf{x}^k = \Pi_X(\mathbf{x}^{k-1} - \alpha_k F(\mathbf{y}^{k-1})),$$

  with $\beta_k = \alpha_k = \frac{1}{L}$ ($L$ is the Lipschtiz constant for $F$)

- **No complexity results.../or potential implications to solve NSO/constrained problems.**
- **Does not exploit the geometry of set $X$.**

# Basic Model Algorithm is Very Simple

- Pick some suitable prox-distance $d(\cdot,\cdot)$ and let

$$p(\mathbf{g},\mathbf{x}) = \operatorname*{argmin}_{\mathbf{v}}\{\langle \mathbf{v},\mathbf{g}\rangle + d(\mathbf{v},\mathbf{x})\}.$$

- **Extra-Gradient-Like: EGL**
  Given $x^1 \in C \cap V$, compute:

$$\begin{aligned}
\mathbf{y}^k &= p(\beta^k F(\mathbf{x}^k),\mathbf{x}^k)\\
\mathbf{x}^{k+1} &= p(\alpha^k F(\mathbf{y}^k),\mathbf{x}^k)\\
\mathbf{z}^k &= \sum_{l=1}^{k} \frac{\alpha^l \mathbf{y}^l}{\sum_{l=1}^k \alpha^l} \quad \leftarrow \text{average comp.}
\end{aligned}$$

  with $\alpha^k,\ \beta^k > 0$ determined within algorithm, or fixed in terms of $L$.
- **Main Computational Object: The Projection-Like Map $p(\cdot,\cdot)$ with respect to the choice of $d(\cdot,\cdot)$.**



# Main Tool for Analysis of EGL

Associate to given $d(\cdot,\cdot)$ an induced Prox Distance $H(\cdot,\cdot)$ s.t.:

$$\langle \mathbf{c}-\mathbf{b}, \nabla_1 d(\mathbf{b},\mathbf{a})\rangle \le H(\mathbf{c},\mathbf{a}) - H(\mathbf{c},\mathbf{b}) - \gamma H(\mathbf{b},\mathbf{a}) \quad \forall \mathbf{a},\mathbf{b},\mathbf{c} \in C \quad \clubsuit.$$

**Convergence Result** (Auslender-Teboulle (06)
Let $\{\mathbf{x}^k,\mathbf{y}^k,\mathbf{z}^k\}$ the sequences generated by EGL. Then,

1. The sequences $\{\mathbf{x}^k\}$, $\{\mathbf{z}^k\}$ are bounded and each limit point of $\{\mathbf{z}^k\}$ is a solution of (VI).
2. If $H(\mathbf{x},\mathbf{y}) = \frac{\sigma}{2}\|\mathbf{x}-\mathbf{y}\|^2$ (e.g., $\Phi$-div. distance) then the **whole sequence** $\{\mathbf{x}^k\}$ converges to a solution of (VI).
3. If $F$ is $L$-Lipschitz on $X$, we have the complexity estimate

$$\theta(\mathbf{z}^k) = O(\frac{1}{k}),$$

   - where $\theta(\mathbf{z}) = \sup\{\langle F(\xi), z-\xi\rangle : \xi \in X\}$ is the gap function.

# Examples of couple $(d, H)$

| $C \cap \mathcal{V}$ | $d(\mathbf{x},\mathbf{y})$ | $H(\mathbf{x},\mathbf{y})$ |
|---|---|---|
| $\mathbb{R}^n_{++}$ | $\sum_{j=1}^n -y_j^2 \log \frac{x_j}{y_j} + x_j y_j - y_j^2 + \frac{\sigma}{2}\|\mathbf{x}-\mathbf{y}\|^2$ | $\frac{1}{2}\|\mathbf{x}-\mathbf{y}\|^2$ |
| $S^n_{++}$ | $-\log\det(\mathbf{x}\mathbf{y}^{-1}) + \operatorname{tr}(\mathbf{x}\mathbf{y}^{-1}) + \sigma \operatorname{tr}(\mathbf{x}-\mathbf{y})^2 - n$ | $H = d$ |
| $L^n_{++}$ | $-\log \frac{\mathbf{x}^T D_n \mathbf{x}}{\mathbf{y}^T D_n \mathbf{y}} + \frac{2\mathbf{x}^T D_n \mathbf{y}}{\mathbf{y}^T D_n \mathbf{y}} - 2 + \frac{\sigma}{2}\|\mathbf{x}-\mathbf{y}\|^2$ | $H = d$ |
| $\Delta_n$ | $\sum_{j=1}^n x_j \log \frac{x_j}{y_j} + y_j - x_j$ | $H = d$ |
| $\Sigma_n$ | $\operatorname{tr}(\mathbf{x}\log\mathbf{x} - \mathbf{x}\log\mathbf{y} + \mathbf{y} - \mathbf{x})$ | $H = d$ |

$$\Delta_n := \{\mathbf{x} \in \mathbb{R}^n \mid \sum_{j=1}^n x_j = 1, x > 0\},\ \Sigma_n := \{\mathbf{x} \in S_n \mid \operatorname{tr}(x) = 1, \mathbf{x} \succ 0\}.$$

$$L^n_{++} := \{\mathbf{x} \in \mathbb{R}^n \mid x_n > (x_1^2 + \ldots + x_{n-1}^2)^{1/2}\},\ D_n \equiv \operatorname{diag}(-1,\ldots,-1,1).$$

$C_n = \{\mathbf{x} \in \mathbb{R}^n : \ a_j < x_j < b_j \quad j = 1\ldots n\}$ similar to $\mathbb{R}^n_{++}$ (log quad)



# Computing Explicit Projections $p(\mathbf{g},\mathbf{x})$

| $C \cap \mathcal{V}$ | $p(\mathbf{g},\mathbf{x})$ or $p_j(\mathbf{g},\mathbf{x}), j = 1,\ldots,n$ |
|---|---|
| $\mathbb{R}^n_{++}$ | $x_j(\varphi^*)'(-g_j x_j^{-1})$ |
| $S^n_{++}$ | $(2\sigma)^{-1}(A(\mathbf{g},\mathbf{x}) + \sqrt{A(\mathbf{g},\mathbf{x})^2 + 4\sigma I})$ |
| $L^n_{++}$ | $\frac{1}{2\sigma}\left((1+\frac{w_n}{\zeta})\bar{w}, (w_n+\zeta)\right)$ |
| $\Delta_n$ | $\frac{x_j \exp(-g_j)}{\sum_{i=1}^n x_i \exp(-g_i)}$ |
| $\Sigma_n$ | via eigenvalue decomp. reduces to similar comp. as $\Delta_n$ |

$$\begin{aligned}
(\varphi^*)'(s) &= (2\sigma)^{-1}\{(\sigma-1) + s + \sqrt{((\sigma-1)+s)^2 + 4\sigma}\} \\
A(\mathbf{g},\mathbf{x}) &= \sigma\mathbf{x} - \mathbf{g} - \mathbf{x}^{-1}, \tau(\mathbf{x}) = \mathbf{x}^T D_n \mathbf{x} \\
w &= (-2\tau(\mathbf{x})^{-1} D_n \mathbf{x} + 2\sigma\mathbf{x} - \mathbf{g})/2,\ \mathbf{w} = (\bar{\mathbf{w}}, w_n) \in \mathbb{R}^{n-1} \times \mathbb{R} \\
\zeta &= \left(\frac{\|\mathbf{w}\|^2 + 4\sigma + \sqrt{(\|\mathbf{w}\|^2 + 4\sigma)^2 - 4w_n^2\|\bar{\mathbf{w}}\|^2}}{2}\right)^{1/2}.
\end{aligned}$$

## Applying EGL to Convex Minimization

- Allows to easily handle general smooth convex constrained problems.
- Possible, thanks to the *theory of duality for variational inequalities*.
- Produce methods with explicit formulas at each iteration *that does not require the solution of any subproblem*.
- Naturally applied to Structured Nonsmooth Convex Problems: Saddle point/minimax



## Smooth Constrained Convex Optimization

- $\mathbb{R}^n$, $\mathbb{R}^m$, and $\mathbb{R}^p$ finite dim. v.s. with inner products, $\langle \cdot, \cdot \rangle_{n,m,p}$
- $(P) \quad f_* = \inf\{f(\mathbf{x}) : \mathbf{x} \in X \equiv S \cap Q\}$
- $X := S \cap Q$ closed convex with $S$ "simple"
- $Q = \{\mathbf{x} \in \mathbb{R}^n : \ -G(\mathbf{x}) \in K, \quad \mathbf{Ax} = \mathbf{a}\}$ $\mathbf{a} \in \mathbb{R}^p$, $\mathbf{A} : \mathbb{R}^n \to \mathbb{R}^p$.
- $K$ closed convex cone, int $K \neq \emptyset$; *e.g.*, $K = \mathbb{R}^m_+, S^m_+, L^m_+$

# Assumptions on Convex Model

- • $f : \mathbb{R}^n \to \mathbb{R}$ convex, $C^1$ with a gradient locally Lipschitz on $X$.
- • $G : \mathbb{R}^n \to \mathbb{R}^p$, $C^1$ with derivative $DG$ locally Lipschitz on $X$ and $K$- convex on $X$:

  $$\lambda G(\mathbf{x}) + (1-\lambda)G(\mathbf{y}) - G(\lambda\mathbf{x} + (1-\lambda)\mathbf{y}) \in K \ \forall \mathbf{x}, \mathbf{y} \in X, \ \forall \lambda \in [0,1].$$

- **Examples of $K$-convex G**
  1. $G(\mathbf{x}) = \mathbf{Bx} - \mathbf{b}$, $\mathbf{B} : \mathbb{R}^n \to \mathbb{R}^p$
  2. $G(\mathbf{x}) = \sum_{i=1}^m \mathbf{B}_i g_i(\mathbf{x})$, $\mathbf{B}_i \in S_+^m$, $g_i : \mathbb{R}^n \to \mathbb{R}$ convex; $K = S_+^m$.
  3. $G(\mathbf{x}) = (g_1(\mathbf{x}), \ldots, g_m(\mathbf{x}))$, $g_i$ convex, $K = \mathbb{R}_+^m$.



# Primal-Dual Variational Inequality Associated to (P)

$$(P) \quad f_* = \inf\{f(\mathbf{x}) : \ -G(\mathbf{x}) \in K, \ \mathbf{Ax} = \mathbf{a} \in S\}.$$

One can show: $\mathbf{x}^*$ solves (P) iff $\exists(\mathbf{u}^*, \mathbf{v}^*)$ s.t. $(\mathbf{x}^*, \mathbf{u}^*, \mathbf{v}^*)$ solves (PDVI):

$$\text{Find } \mathbf{z}^* = (\mathbf{x}^*, \mathbf{u}^*, \mathbf{v}^*) \in \Omega : \ \langle T(\mathbf{z}^*), \mathbf{z} - \mathbf{z}^* \rangle \geq 0, \forall \mathbf{z} \in \Omega$$

with

- $\Omega := S \times (K \times \mathbb{R}^p)$= "simple" $\times$ "Hard" $\times$ "Affine"
- The primal-dual operator is defined by

  $$\begin{aligned} T(\mathbf{z}) &:= (\nabla f(\mathbf{x}) + D_{\mathbf{u}} G(\mathbf{x})(\mathbf{u}) + \mathbf{A}^* \mathbf{v}, -G(\mathbf{x}), -(\mathbf{Ax} - \mathbf{a})) \\ &\equiv (T_1(\mathbf{z}), T_2(\mathbf{z}), T_3(\mathbf{z})). \end{aligned}$$

  with $D_{\mathbf{u}} G(\mathbf{x}) := \langle \mathbf{u}, \nabla G(\mathbf{x}) \rangle_m$.

# Projection-like Map for PDVI are Easy to Compute!

• Given $\mathbf{z} = (\mathbf{x}, \mathbf{u}, \mathbf{v}) \in \Omega, \quad \Omega \equiv S \times (K \times \mathbb{R}^p)$
• let $Z := (X, U, W) = T(\bar{\mathbf{z}})$ for some other given $\bar{\mathbf{z}} \in \Omega$.
To apply EGL for solving (PDVI), **all we need is to compute** $\mathbf{z}^+ := p(Z, \mathbf{z})$ for some chosen distance $d(\mathbf{z}', \mathbf{z})$.

We choose $d$ defined by:

$$d(\mathbf{z}', \mathbf{z}) := d_1(\mathbf{x}', \mathbf{x}) + d_2(\mathbf{u}', \mathbf{u}) + \frac{1}{2}\|\mathbf{v}' - \mathbf{v}\|^2,$$

- $d_1$ captures the "simple" constraints described by $S$
- $d_2$ captures the "hard" constraints through projections-like map on the cone $K$
- Last distance captures the affine equality constraints (if any).
- Since $d$ is *separable*, the computation of $p$ decomposed accordingly, and hence $\mathbf{z}^+ = (\mathbf{x}^+, \mathbf{u}^+, \mathbf{v}^+)$ are computed independently and easily as follows.



# Projection-Like Map Formulas

$$\begin{aligned}
\mathbf{x}^+ = p_1(T_1(\bar{\mathbf{z}}), \mathbf{x}) := p_1(X, \mathbf{x}) &= \operatorname{argmin}\{\langle \mathbf{w}, X\rangle + d_1(\mathbf{w}, \mathbf{x}) : \ \mathbf{w} \in S\}, \\
\mathbf{u}^+ = p_2(T_2(\bar{\mathbf{z}}), u) := p_2(U, \mathbf{u}) &= \operatorname{argmin}\{\langle \mathbf{w}, U\rangle + d_2(\mathbf{w}, \mathbf{u}) : \ \mathbf{w} \in K\}, \\
\mathbf{v}^+ = p_3(T_3(\bar{\mathbf{z}}), v) := p_3(W, \mathbf{v}) &= \operatorname{argmin}\{\langle \mathbf{w}, W\rangle + \frac{1}{2}\|\mathbf{w} - \mathbf{v}\|^2 : \ \mathbf{w} \in \mathbb{R}^p\}
\end{aligned}$$

In particular, note that one always has: $\mathbf{v}^+ = \mathbf{v} - W$.

- For computing $\mathbf{x}^+, \mathbf{u}^+$ we use the results given in the previous tables, e.g. for $S = \mathbb{R}^n, \mathbb{R}^n_+, \ S^n_+, L^n_+$. Similarly, for $K = \mathbb{R}^n_+, \ S^n_+$, and $L^n_+$.
- **No matter how complicated the constraints are in the ground set $S \cap Q$**, the resulting projections-like maps for (PDVI) are **given by analytical formulas.**

# Other Useful Applications of EGL

- Decomposition Methods :
  $f(\mathbf{x}) = \sum_{j=1}^{l} f_j(\mathbf{x}_j), \; g_i(\mathbf{x}) = \sum_{j=1}^{l} g_{ij}(\mathbf{x}_j), \; X = \prod_{j=1}^{l} X_j$
- Particularly useful and cheap for very large scale problems, since explicit formulas at each step are obtained.
- Semidefinite programming
- Second order cone programs
- Bilinear matrix games
- **Saddle point and minimax problems**



# EGL for Structured Nonsmooth Optimization

$$\min\{g(\mathbf{x}) : \; \mathbf{x} \in X\}, \quad \text{convex nonsmmooth}$$

- As seen, projected subgradient methods, have complexity estimate $O(\frac{1}{\sqrt{k}})$
- **Many nonsmooth convex problems admit a saddle pt structure**,

$$g(\mathbf{x}) = \max\{\Phi(\mathbf{x}, \mathbf{y}) | \; \mathbf{y} \in Y\}$$

  $Y$ convex compact "simple" in $\mathbb{R}^p$; $\Phi$ convex-concave on $X \times Y$ with a derivative $D\Phi$ Lipschitz on $X \times Y$.
- This Saddle Point Problem $\min_{x \in X} \max_{y \in Y} \Phi(x, y)$ can be written as a basic (VI) problem.
- Hence **EGL** can be applied with a complexity estimate $\sim O(\frac{1}{k})$.
- **Again, "structure" helps to get better complexity results for another class of NSO**.

# Structured Nonsmooth Optimization: Example 1

- Minimizing the maximum eigenvalue of a convex combination of $n \times n$ matrices $A_1, \dots, A_m$,

$$(Eig) \quad \min_{\mathbf{x}}\{g(\mathbf{x}) := \lambda_{\max}(\mathbf{A}(\mathbf{x})) : \mathbf{x} \in \Delta_m\}; \; \mathbf{A}(\mathbf{x}) := \sum_{j=1}^{m} x_j A_j.$$

  But, for any $\mathbf{B} \in S^n$, $\lambda_{max}(\mathbf{B}) = \max\{\mathrm{tr}(\mathbf{ZB}) : \; \mathrm{tr}(\mathbf{Z}) = 1, \mathbf{Z} \in S_+^n\}$

- Thus, (Eig) equivalent to

$$\min_{\mathbf{x}\in\Delta_m} \max_{\mathbf{y}\in\Sigma_n} \Phi(\mathbf{x}, \mathbf{y}) \equiv \mathrm{tr}(\mathbf{y}(\mathbf{Ax}))$$

  where $\Sigma_n = \{\mathbf{y} \in S_+^n | \; \mathrm{tr}(\mathbf{y}) = 1\}$ Spectrahedron.

  Here $D\Phi$ is globally Lipschitz with constant $L = \frac{1}{2||A||}$

  **EGL** can be easily applied using Entropy-like distances.



# Structured Nonsmooth Optimization: Example 2

Computing Lovasz capacity: $G$ graph, $n$ vertices, $m$ arcs $\mathcal{A}$. Define

- $d \in S^n : \; d_{ij} = 0 \; \forall (i,j) \in \mathcal{A}, \quad d_{i,j} = 1$ otherwise
- $X = \{x \in S^n : \; x_{ij} = 0, \; \forall (i,j) \notin \mathcal{A}\}$
- $Y = \Sigma_m = \{y \in S_+^n | \; tr(y) = 1\}$, Spectrahedron

The Lovasz capacity of $G$ is then modeled by:

$$\min_{x\in X} \max_{y\in Y} \Phi(x, y) := tr(y(d + x)) \quad \spadesuit$$

- **EGL** can then be applied to solve $\spadesuit$ and produces a simple explicit algorithm.
- No needs to solve any optimization at each iteration!

# Conclusions

- Gradient-Based Schemes can be efficiently applied to a broad class of problems ...**Old methods back alive and kicking!**
- Strong potential for designing simple and efficient algorithms in many applied areas with structured optimization models.
- Further needs for simple and efficient schemes that can cope with **curse of dimensionality and Nonconvex/Nonsmooth settings**.

**..........Optimizers are not (yet..) out of job.........!**



# For More Details, Results and References...


- A. Auslender, M. Teboulle Interior projection-like methods for monotone variational inequalities. *Mathematical Programming*, **104**, (2005), 39–68.
- A. Auslender and M. Teboulle. Interior gradient and proximal methods in convex and conic optimization. *SIAM J. Optimization*, **16**, (2006), 697–725.
- A. Beck and M. Teboulle, A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. *SIAM J. on Imaging Sciences*, vol. 2, 183–202, (2009).
- A. Beck and M. Teboulle, Fast Gradient-Based Algorithms for Constrained Total Variation Image Denoising and Deblurring Problems. *IEEE Trans. Image Processing*, **18**, 2419–2434 (2009).
- A. Beck and M. Teboulle, Gradient-Based Algorithms with Applications to Signal Recovery Problems. *In: Convex Optimization in Signal Processing and Communications*, Edited by Y. Eldar and D. Palomar, Cambridge University Press, (2010).

**Thank you for listening!**