Non-smooth First-order Optimization Algorithms for Machine Learning

Peter Ochs

Mathematical Optimization Group

— 08.02.2021 —

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Optimization is not solvable!

- Efficiently finding solutions to the whole class of Lipschitz continuous problems is a hopeless case [Nesterov '04].
- Can take several million years for small problems with only 10 unknowns.

Optimization is not solvable!

- Efficiently finding solutions to the whole class of Lipschitz continuous problems is a hopeless case [Nesterov '04].
- Can take several million years for small problems with only 10 unknowns.

$\Rightarrow$ **Exploit the structure of optimization problems!**

Classic Optimization:

$$\min_{x \in C} f(x)\,, \qquad C := \{x \in \mathbb{R}^N \mid f_j(x) \le 0\,,\ j = 1, \ldots, M\}\,.$$

- **Linear Programming**: f, $f_1, \ldots, f_M$ are linear functions.
- **Smooth non-linear programming:** f is C^1-smooth, $C = \emptyset$.
- **Constrained non-linear programming:** f, f_j are C^1-smooth.

Traditional view on optimization (before 1990):

Linear vs. **non-linear**.

Why Non-smooth Optimization?

- Non-smoothness is **not just a deficiency**.
- Constraints lead to non-smoothness.
- Arises naturally in some applications, e.g. sparsity.
- Lagrangian Relaxation, functions with envelope representation, Penalty formulation, ...

$\rightsquigarrow$ Differentiability is replaced by "**structure**".

Why First-order Optimization?

- Problems in Data Science, Machine Learning, Computer Vision, Image Processing, ...
- Practical problems depend on data.

  $\rightsquigarrow$ rapidly growing.
- Problems are large / huge scale.

  $\rightsquigarrow$ computing / inverting Hessian matrix is too expensive.

  $\rightsquigarrow$ second order information can be unstable.
- Our goal is non-smooth optimization.

Sparsity Induced by Non-smooth Minimization

Least Absolute Shrinkage and Selection Operator:

$$\min_{x\in\mathbb{R}^N} \frac{1}{2}\|Ax-b\|^2 + \lambda\|x\|_1$$

- Sparse **linear regression**: ($A_i \in \mathbb{R}^M$ feature for b)

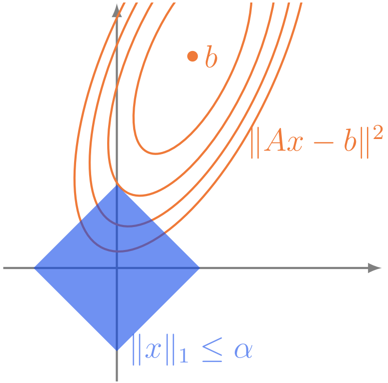
$$b \approx \sum_{i=1}^N A_i x_i\,, \quad A = (A_1, \dots, A_N) \in \mathbb{R}^{M\times N}\,,\ x = (x_1, \dots, x_N)^\top\,.$$

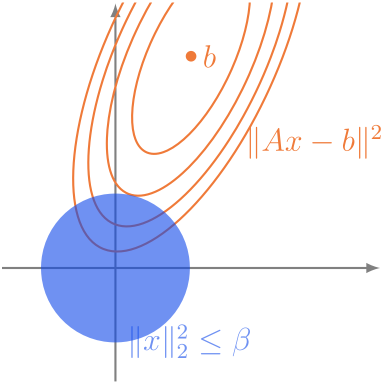
- $\|x\|_1$ used as a convex approximation to $\#\{i \,|\, x_i \neq 0\}$.
- **Motivation**: Interpretable model (many zero-coordinates)

$$b \approx \sum_{i=1}^N A_i x_i = \sum_{j\in\{i\,|\,x_i\neq 0\}} A_j x_j\,.$$

Sparsity Induced by Non-smooth Minimization

1-norm constrained minimization yields sparse solutions:





$\rightsquigarrow$ **solutions for the 1-norm constrained problem are likely to lie on coordinate axis**

Non-smooth Problems arise naturally

Matrix Completion:

$$\min_{X\in\mathbb{R}^{M\times N}} \frac{1}{2}\|\pi_S(A-X)\|_F^2 + \lambda\|X\|_*$$

- π_S projects onto a subspace S.
- $\|X\|_*$ is the Nuclear norm = sum of singular values
- $\|X\|_*$ used as convex relaxation of $\mathrm{rank}(X)$.
- Solution has low rank, i.e., exploits linear dependency of data.
- **Application:** e.g. Collaborative filtering

⇝ Dimensionality may be large / huge.

Sparsity Induced by Non-smooth Minimization

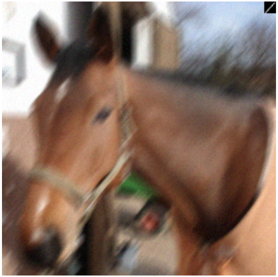
Image Deblurring: of an image $b \in \mathbb{R}^{M\times N}$

$$\min_{u\in\mathbb{R}^{M\times N}} \frac{1}{2}\|Ku - b\|^2 + \lambda\|Du\|_{2,1}$$

- K convolution / blurr operator; D discrete derivative operator

⇝ dimensionality u = number of pixel (gray-value image)


clean image


blurry image


reconstruction

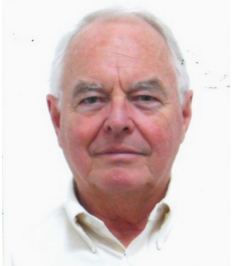
Classical Methods for Non-smooth Optimization?

- Constraint set C reformulated using **penalty terms**.
- **Generic non-linear solvers** (black box solvers):
  e.g. Steepest Descent, Conjugate Gradient, L-BFGS,
  Newton's Method, Interior Point Method, ...
- Hard to generalize to constraints or non-smooth functions.
- Line-search procedure can be time intensive.
- Maybe slow or memory intensive for large scale problems.

Modern view (after 1990):

"Watershed in optimization is
between **convexity** and **non-convexity**
instead of linearity and nonlinearity"

[Rockafellar, SIAM Review, 35 (1993)]



Subgradient Method

A Non-smooth Convex Optimization Problems:

- Lipschitz continuous, convex problem:

$$\min_{x\in\mathbb{R}^N} \; f(x) \; + \; g(x)$$

$\mathbb{R}^N \to \mathbb{R}$
smooth, convex
∇f Lipschitz

$\mathbb{R}^N \to \mathbb{R}$
non-smooth,
convex

- **Lower complexity estimate:** (ε: desired accuracy)

req. number of iterations to achieve accuracy of $\varepsilon = O(1/\varepsilon^2)$.

Proximal Gradient Method

Structured Optimization Problems: (**Splitting**)

- Additive Composite Setting:

$$\min_{x\in\mathbb{R}^N} \quad f(x) \quad + \quad g(x)$$

$\mathbb{R}^N \to \mathbb{R}$
smooth, convex
∇f Lipschitz

$\mathbb{R}^N \to \overline{\mathbb{R}}$
non-smooth,
convex
simple prox

Proximal Gradient Method

Structured Optimization Problems: **(Splitting)**

- Additive Composite Setting:

$$\min_{x\in\mathbb{R}^N} \; f(x) \; + \; g(x)$$

$\mathbb{R}^N \to \mathbb{R}$
smooth, convex
∇f Lipschitz

$\mathbb{R}^N \to \overline{\mathbb{R}}$
non-smooth,
convex
simple prox

Gradient Descent Step:

$$\bar{x} - \tau\nabla f(\bar{x})$$

Proximal Step: $\mathrm{prox}_{\tau g}(\bar{x})$, **i.e.**

$$\underset{x\in\mathbb{R}^N}{\mathrm{argmin}} \; g(x) + \frac{1}{2\tau}\|x-\bar{x}\|^2$$

Example:

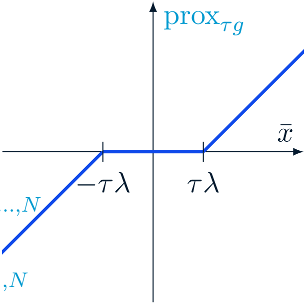
$$\min_{x\in\mathbb{R}^N} \underbrace{\frac{1}{2}\|Ax-b\|^2}_{=f(x)} + \underbrace{\lambda\|x\|_1}_{=g(x)}$$

- Compute gradient:

$$\nabla f(x) = A^\top(Ax-b)$$

- Compute proximal mapping:

$$\begin{aligned}\operatorname{prox}_{\tau g}(\bar{x}) &= \operatorname*{argmin}_{x\in\mathbb{R}^N} \lambda\|x\|_1 + \frac{1}{2\tau}\|x-\bar{x}\|_2^2 \\ &= \Big(\operatorname*{argmin}_{x_i\in\mathbb{R}} \lambda|x_i| + \frac{1}{2\tau}(x_i-\bar{x}_i)^2\Big)_{i=1,\dots,N} \\ &= \Big(\max(|\bar{x}_i|-\tau\lambda, 0)\cdot \operatorname{sign}(\bar{x}_i)\Big)_{i=1,\dots,N}\end{aligned}$$



Algorithm: (Proximal Gradient Method)

- **Optimization problem**: $\min_x f(x) + g(x)$
  - $f\colon \mathbb{R}^N \to \mathbb{R}$ convex with ∇f L-Lipschitz.
  - $g\colon \mathbb{R}^N \to \overline{\mathbb{R}}$ proper, lsc, convex with simple prox.
- **Iterations** $(k \geq 0)$: Update ($x^{(0)} \in \mathbb{R}^N$)

$$x^{(k+1)} = \operatorname{prox}_{\tau g}(x^{(k)} - \tau \nabla f(x^{(k)}))$$

Complexity estimate: $O(1/\varepsilon)$

Method traces back to:

[P. L. Lions and B. Mercier: *Splitting algorithms for the sum of two nonlinear operators*, SIAM J. Numer. Anal., 16 (1979), pp. 964–979.]

“Acceleration” Strategies

Acceleration Strategies:

- **Higher order Optimization?** (“non-smooth Newton method”)
  - Not tractable for large scale. (requires inverse Hessian)
- **“Accelerate” first order methods:**
  - In convex optimization, we can accelerate by *extrapolation/momentum*. (worst case $O(1/\sqrt{\varepsilon})$)
  - In non-convex optimization, this is an *effective* heuristic.
- **Alternative acceleration strategies:**
  - Quasi-Newton schemes. (**this talk**)
  - Subspace Optimization.

Quasi-Newton Methods:

- Can be interpreted as variable metric method:

$$x^{(k+1)} = x^{(k)} - \tau V_k^{-1} \nabla f(x^{(k)}) \,.$$

  with (positive definite) Hessian approximation $V_k \approx \nabla^2 f(x^{(k)})$.

- Quasi-Newton Methods successively improve V_k by **low-rank modifications** that approximate the "curvature" via secant equations.

- **SR1**: rank 1 updates; **BFGS**: rank 2 updates.

Non-smooth Quasi-Newton Methods:

- **Non-smooth setting:**

$$x^{(k+1)} = \mathrm{prox}_{\tau g}^{V_k}\Big(x^{(k)} - \tau V_k^{-1}\nabla f(x^{(k)})\Big)$$

where

$$\mathrm{prox}_{\tau g}^{V_k}(\bar{x}) := \operatorname*{argmin}_x \; g(x) + \frac{1}{2\tau}\langle x - \bar{x}, V_k(x - \bar{x})\rangle$$

$\rightsquigarrow$ **Couples the coordinates !** (unless V_k is diagonal)

Solving the rank-1 Proximal Mapping: (g convex)

- For general $\boldsymbol{V}$, the main algorithmic step is hard to solve:

$$\hat{x} = \operatorname{prox}_g^{\boldsymbol{V}} := \underset{x\in\mathbb{R}^N}{\operatorname{argmin}}\; g(x) + \frac{1}{2}\|x-\bar{x}\|_{\boldsymbol{V}}^2$$

- **Theorem: [Rank-1 Proximal Mapping]**
  $\boldsymbol{V} = \boldsymbol{D} \pm uu^\top \in \mathbb{S}_{++}$ for $u\in\mathbb{R}^N$ and $\boldsymbol{D}$ diagonal. Then

$$\operatorname{prox}_g^{\boldsymbol{V}}(\bar{x}) = \boldsymbol{D}^{-1/2}\circ \operatorname{prox}_{g\circ \boldsymbol{D}^{-1/2}} \circ \boldsymbol{D}^{1/2}(\bar{x} \mp \alpha^\star \boldsymbol{D}^{-1}u)$$

where $\alpha^\star$ is the unique root of

$$l(\alpha) = \langle u, \bar{x} - \boldsymbol{D}^{-1/2}\circ \operatorname{prox}_{g\circ \boldsymbol{D}^{-1/2}} \circ \boldsymbol{D}^{1/2}(\bar{x} \mp \alpha \boldsymbol{D}^{-1}u)\rangle + \alpha\,,$$

which is strictly increasing and Lipschitz with $1+\sum_i u_i^2 d_i$.

Solving the rank-1 Proximal Mapping

Corollary: **separable case** $g(x) = \sum_{i=1}^N g_i(x_i)$

$\boldsymbol{V} = \boldsymbol{D} \pm uu^\top \in \mathbb{S}_{++}$ for $u \in \mathbb{R}^N$ and $\boldsymbol{D} = \operatorname{diag}(d_1, \dots, d_N)$. Then

$$\operatorname{prox}_g^{\boldsymbol{V}}(\bar{x}) = \Big(\operatorname{prox}_{g_i/d_i}(\bar{x}_i \mp \alpha^\star u_i/d_i)\Big)_{i=1,\dots,N}$$

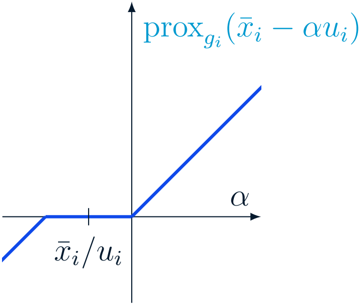
where $\alpha^\star$ is the unique root of

$$l(\alpha) = \Big\langle u, \bar{x} - \Big(\operatorname{prox}_{g_i/d_i}(\bar{x}_i \mp \alpha u_i/d_i)\Big)_{i=1,\dots,N}\Big\rangle + \alpha\,,$$

which is strictly increasing and Lipschitz with $1 + \sum_i u_i^2 d_i$.

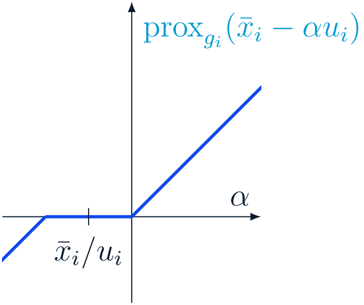
Example: $g(x) = \sum_{i=1}^{N} |x_i|$. For simplicity $D = \mathrm{id}$.

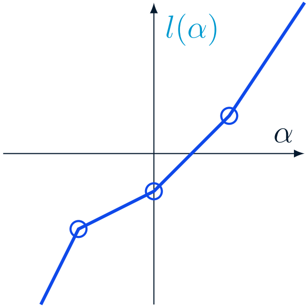
$$l(\alpha) = \langle u, \bar{x} - \Big(\mathrm{prox}_{g_i}(\bar{x}_i \mp \alpha u_i)\Big)_{i=1,\dots,N} \rangle + \alpha \,,$$



Example: $g(x) = \sum_{i=1}^{N} |x_i|$. For simplicity $D = \mathrm{id}$.

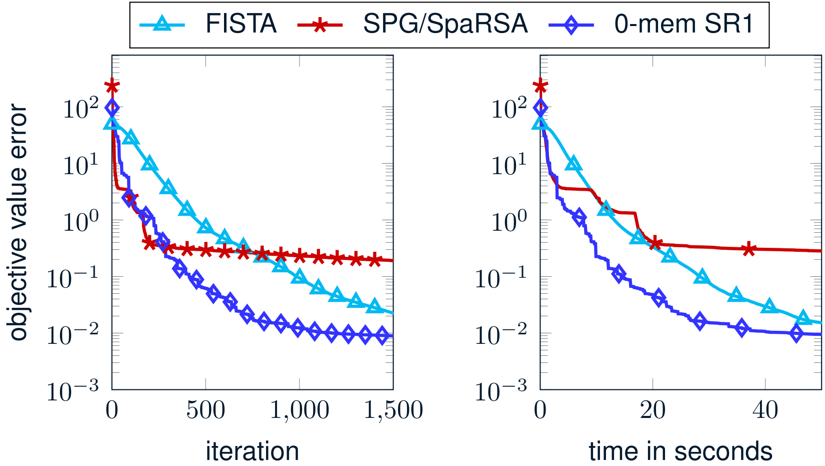
$$l(\alpha) = \langle u, \bar{x} - \Big(\mathrm{prox}_{g_i}(\bar{x}_i \mp \alpha u_i)\Big)_{i=1,\ldots,N} \rangle + \alpha \,,$$





Numerical Experiment Group Lasso

Experiment:
$$\min_{x \in \mathbb{R}^N} \frac{1}{2}\|Ax - b\|^2 + \lambda\|x\|_{\mathfrak{B},1}$$



Solving the rank-r Proximal Mapping

Theorem: [Rank-r Proximal Mapping]
$\boldsymbol{V} = \boldsymbol{P} \pm \boldsymbol{Q}$ with $\boldsymbol{P} \in \mathbb{S}_{++}(N)$ and $\boldsymbol{Q} = \sum_{i=1}^{r} u_i u_i^\top$ with $r = \operatorname{rank}(Q) \leq N$. Denote $\boldsymbol{U} := (u_1, \ldots, u_r)$. Then

$$\operatorname{prox}_g^{\boldsymbol{V}}(\bar{x}) = \boldsymbol{P}^{-1/2} \circ \operatorname{prox}_{g \circ \boldsymbol{P}^{-1/2}} \circ \boldsymbol{P}^{1/2}(\bar{x} \mp \boldsymbol{P}^{-1}\boldsymbol{U}\alpha^\star)$$

where $\alpha^\star$ is the unique root of the mapping $\mathcal{L}\colon \mathbb{R}^r \to \mathbb{R}^r$

$$\mathcal{L}(\alpha) = \boldsymbol{U}^\top\Big(\bar{x} - \boldsymbol{P}^{-1/2} \circ \operatorname{prox}_{g \circ \boldsymbol{P}^{-1/2}} \circ \boldsymbol{P}^{1/2}(\bar{x} \mp \boldsymbol{P}^{-1}\boldsymbol{U}\alpha)\Big) + \alpha\,,$$

which is Lipschitz continuous and strongly monotone.

Discussion about Solving the Proximal Mapping

Function g	Algorithm
ℓ_1-norm	Separable: exact
Hinge	Separable: exact
ℓ_∞-ball	Separable: exact
Box constraint	Separable: exact
Positivity constraint	Separable: exact
Linear constraint	Nonseparable: exact
ℓ_1-ball	Nonseparable: Semi-smooth Newton + $\mathrm{prox}_{g \circ \boldsymbol{D}^{-1/2}}$ exact
ℓ_∞-norm	Nonseparable: Moreau identity
Simplex	Nonseparable: Semi-smooth Newton + $\mathrm{prox}_{g \circ \boldsymbol{D}^{-1/2}}$ exact

From [Becker, Fadili '12].

“Acceleration” Strategies

Acceleration Strategies:

- **Higher order Optimization?** (“non-smooth Newton method”)
  - Not tractable for large scale. (requires inverse Hessian)
- **“Accelerate” first order methods:**
  - In convex optimization, we can accelerate by *extrapolation/momentum*. (worst case $O(1/\sqrt{\varepsilon})$) (**now**)
  - In non-convex optimization, this is an *effective* heuristic.
- **Alternative acceleration strategies:**
  - Quasi-Newton schemes. (**don’t forget this**)
  - Subspace Optimization.

Accelerated Proximal Gradient Method

Algorithm: (FISTA)

- **Optimization problem**: $\min_x f(x) + g(x)$
  - $f \colon \mathbb{R}^N \to \mathbb{R}$ convex with ∇f L-Lipschitz.
  - $g \colon \mathbb{R}^N \to \overline{\mathbb{R}}$ proper, lsc, convex with simple prox.
- **Iterations** $(k \geq 0)$: Update ($x^{(0)} \in \mathbb{R}^N$) $\beta_k = \frac{k-1}{k+2}$

$$
\begin{aligned}
y^{(k)} &= x^{(k)} + \beta_k (x^{(k)} - x^{(k-1)}) \\
x^{(k+1)} &= \mathrm{prox}_{\tau g}(y^{(k)} - \tau \nabla f(y^{(k)}))
\end{aligned}
$$

Complexity estimate: $O(1/\sqrt{\varepsilon})$ $\rightsquigarrow$ **optimal method**

Accelerated Gradient Method: [Y. Nesterov: *A Method of Solving a Convex Programming Problem with Convergence Rate* $O(1/K^2)$. Soviet Mathematics Doklady 27:372–76, 1983.]

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

Update Step of FISTA

Update Scheme: FISTA

$$y_{\beta_k}^{(k)} = x^{(k)} + \beta_k (x^{(k)} - x^{(k-1)})$$

$$x^{(k+1)} = \operatorname*{argmin}_x g(x) + f(y_{\beta_k}^{(k)}) + \langle \nabla f(y_{\beta_k}^{(k)}), x - y_{\beta_k}^{(k)} \rangle + \frac{1}{2\tau} \|x - y_{\beta_k}^{(k)}\|^2$$

Update Step of FISTA

Update Scheme: FISTA

$$y_{\beta_k}^{(k)} = x^{(k)} + \beta_k(x^{(k)} - x^{(k-1)})$$

$$x^{(k+1)} = \operatorname*{argmin}_x g(x) + f(y_{\beta_k}^{(k)}) + \langle \nabla f(y_{\beta_k}^{(k)}), x - y_{\beta_k}^{(k)} \rangle + \frac{1}{2\tau} \|x - y_{\beta_k}^{(k)}\|^2$$

Equivalent to

$$x^{(k+1)} = \operatorname*{argmin}_{x \in \mathbb{R}^N} g(x) + \frac{1}{2\tau} \|x - \Big(y_{\beta_k}^{(k)} - \tau \nabla f(y_{\beta_k}^{(k)})\Big)\|^2$$

Update Step of FISTA

Update Scheme: **Adaptive FISTA**

$$y_{\beta_k}^{(k)} = x^{(k)} + \beta_k(x^{(k)} - x^{(k-1)})$$

$$x^{(k+1)} = \operatorname*{argmin}_x \min_{\beta_k} g(x) + f(y_{\beta_k}^{(k)}) + \langle \nabla f(y_{\beta_k}^{(k)}), x - y_{\beta_k}^{(k)} \rangle + \frac{1}{2\tau} \|x - y_{\beta_k}^{(k)}\|^2$$

Update Step of FISTA

Update Scheme: Adaptive FISTA (f **quadratic**)

$$y_{\beta_k}^{(k)} = x^{(k)} + \beta_k (x^{(k)} - x^{(k-1)})$$

$$x^{(k+1)} = \operatorname*{argmin}_x \min_{\beta_k} g(x) + f(y_{\beta_k}^{(k)}) + \langle \nabla f(y_{\beta_k}^{(k)}), x - y_{\beta_k}^{(k)} \rangle + \frac{1}{2\tau} \|x - y_{\beta_k}^{(k)}\|^2$$

... Taylor expansion around $x^{(k)}$ and optimize for $\beta_k = \beta_k(x)$ ...

$$x^{(k+1)} = \operatorname*{argmin}_x \; g(x) + \frac{1}{2} \|x - (x^{(k)} - \boldsymbol{Q}_k^{-1} \nabla f(x^{(k)}))\|_{\boldsymbol{Q}_k}^2$$

Update Step of FISTA

Update Scheme: Adaptive FISTA (f **quadratic**)

$$y^{(k)}_{\beta_k} = x^{(k)} + \beta_k (x^{(k)} - x^{(k-1)})$$

$$x^{(k+1)} = \operatorname*{argmin}_x \min_{\beta_k} g(x) + f(y^{(k)}_{\beta_k}) + \langle \nabla f(y^{(k)}_{\beta_k}), x - y^{(k)}_{\beta_k} \rangle + \frac{1}{2\tau} \| x - y^{(k)}_{\beta_k} \|^2$$

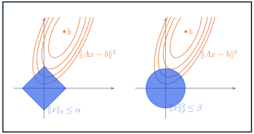
... Taylor expansion around $x^{(k)}$ and optimize for $\beta_k = \beta_k(x)$ ...

$$x^{(k+1)} = \operatorname*{argmin}_x \; g(x) + \frac{1}{2} \| x - (x^{(k)} - \boldsymbol{Q}_k^{-1} \nabla f(x^{(k)})) \|^2_{\boldsymbol{Q}_k}$$

$\boldsymbol{Q}_k$ is exactly the rank-1 modification in **SR1 quasi-Newton method.**

[O. and T. Pock: *Adaptive Fista for Non-convex Optimization*. SIAM Journal on Optimization, 29(4):2482-2503, 2019.]

Summary



Motivation non-smooth optimization

$$\min_{x\in\mathbb{R}^N} f(x) + g(x)$$

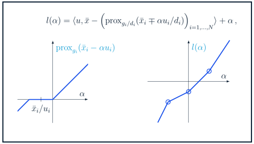
f: $\mathbb{R}^N \to \mathbb{R}$, smooth, convex, ∇f Lipschitz

g: $\mathbb{R}^N \to \overline{\mathbb{R}}$, non-smooth, convex, simple prox

Structured Optimization and Applications

$$x^{(k+1)} = \mathrm{prox}_{\tau g}(x^{(k)} - \tau\nabla f(x^{(k)}))$$

Proximal Gradient Method



Quasi-Newton Proximal Gradient Methods

$$y^{(k)} = x^{(k)} + \beta_k(x^{(k)} - x^{(k-1)})$$
$$x^{(k+1)} = \mathrm{prox}_{\tau g}(y^{(k)} - \tau\nabla f(y^{(k)}))$$

Adaptive FISTA