Nonsmooth regularizations in Machine Learning: structure of the solutions, identification, and applications

Franck Iutzeler LJK, Univ. Grenoble Alpes

Montpellier (online), Nov. 2020

Structure	Regularization
sparsity	$r = \|\cdot\|_1$
anti-sparsity	$r = \|\cdot\|_\infty$
low rank	$r = \|\cdot\|_*$
⋮	⋮

Linear inverse problems: for a chosen regularization, we seek

$$x^\star \in \arg\min_x r(x) \qquad \text{such that } Ax = b$$

Regularized **Empirical Risk Minimization** problem:

$$\text{Find} \quad x^\star \in \arg\min_{x\in\mathbb{R}^n} \quad \underbrace{\mathcal{R}\left(x; \{a_i, b_i\}_{i=1}^m\right)}_{\text{obtained from statistical modeling}} \quad + \quad \underbrace{\lambda \; r(x)}_{\text{chosen regularization}}$$

$$\text{e.g. Lasso: Find} \quad x^\star \in \arg\min_{x\in\mathbb{R}^n} \quad \sum_{i=1}^m \tfrac{1}{2}(a_i^\top x - b_i)^2 \quad + \quad \lambda \; \|x\|_1$$

Regularization can improve statistical properties (generalization, stability, ...).

- Tibshirani: Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society (1996)
- Tibshirani *et al.*: Sparsity and smoothness via the fused lasso. Journal of the Royal Statistical Society (2004)
- Vaiter, Peyré, Fadili: Model consistency of partly smooth regularizers. IEEE Trans. on Information Theory (2017)

Composite minimization

$$\text{Find} \quad x^\star \in \arg\min_{x\in\mathbb{R}^n} \quad \mathcal{R}\left(x; \{a_i, b_i\}_{i=1}^m\right) \quad + \quad \lambda\, r(x)$$

$$\text{Find} \quad x^\star \in \arg\min_{x\in\mathbb{R}^n} \quad f(x) \quad + \quad g(x)$$

smooth — non-smooth

- f: differentiable surrogate of the empirical risk $\Rightarrow$ **Gradient**
  non-linear smooth function that depends on all the data
- g: non-smooth but chosen regularization $\Rightarrow$ **Proximity operator**
  non-differentiability on some manifolds implies structure on the solutions

$$\mathbf{prox}_{\gamma g}(u) = \arg\min_{y\in\mathbb{R}^n} \left\{ g(y) + \tfrac{1}{2\gamma}\|y-u\|_2^2 \right\}$$

closed form/easy for many regularizations:

- $g(x) = \|x\|_1$
- $g(x) = TV(x)$
- $g(x) = indicator_C(x)$

Natural optimization method: **proximal gradient**

$$\begin{cases} u_{k+1} = x_k - \gamma \nabla f(x_k) \\ x_{k+1} = \mathbf{prox}_{\gamma g}(u_{k+1}) \end{cases}$$

and its stochastic variants: proximal sgd, etc.

Example: LASSO

$$\begin{array}{rlccc} \text{Find} & x^\star \in \arg\min\limits_{x\in\mathbb{R}^n} & \mathcal{R}\left(x; \{a_i, b_i\}_{i=1}^m\right) & + & \lambda\ r(x) \\ \text{Find} & x^\star \in \arg\min\limits_{x\in\mathbb{R}^n} & \frac{1}{2}\left\|Ax-b\right\|_2^2 & + & \lambda\|x\|_1 \\ & & \text{smooth} & & \text{non-smooth} \end{array}$$

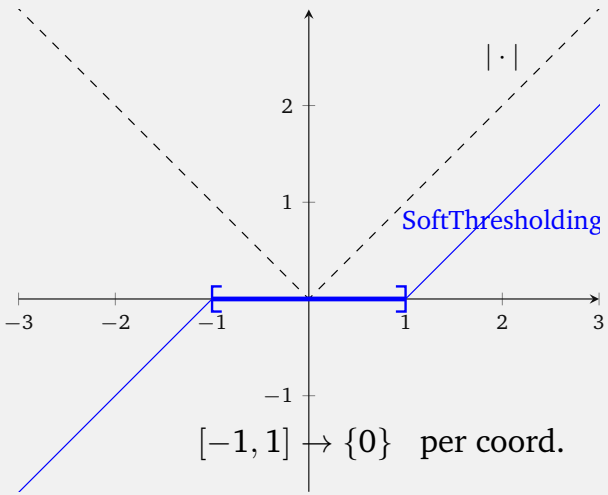
Coordinates	**Structure**	$\leftrightarrow$	**Optimality conditions**
$\forall i$	$x_i^\star = 0$	$\Leftrightarrow$	$A_i^\top (Ax^\star - b) \in [-\lambda, \lambda]$

Proximity Operator: per coordinate

$$\left[\mathbf{prox}_{\gamma\lambda\|\cdot\|_1}(u)\right]_i = \begin{cases} u_i - \lambda\gamma & \text{if } u_i > \lambda\gamma \\ 0 & \text{if } u_i \in [-\lambda\gamma; \lambda\gamma] \\ u_i + \lambda\gamma & \text{if } u_i < -\lambda\gamma \end{cases}$$

Proximal Gradient (aka ISTA):

$$\begin{cases} u_{k+1} = x_k - \gamma A^\top (Ax_k - b) \\ x_{k+1} = \mathbf{prox}_{\gamma\lambda\|\cdot\|_1}(u_{k+1}) \end{cases}$$



Example: LASSO

$$\begin{array}{rrcccc} \text{Find} & x^\star \in \arg\min\limits_{x\in\mathbb{R}^n} & \mathcal{R}\left(x; \{a_i, b_i\}_{i=1}^m\right) & + & \lambda\ r(x) \\ \text{Find} & x^\star \in \arg\min\limits_{x\in\mathbb{R}^n} & \frac{1}{2}\left\|Ax-b\right\|_2^2 & + & \lambda\|x\|_1 \\ & & \text{smooth} & & \text{non-smooth} \end{array}$$

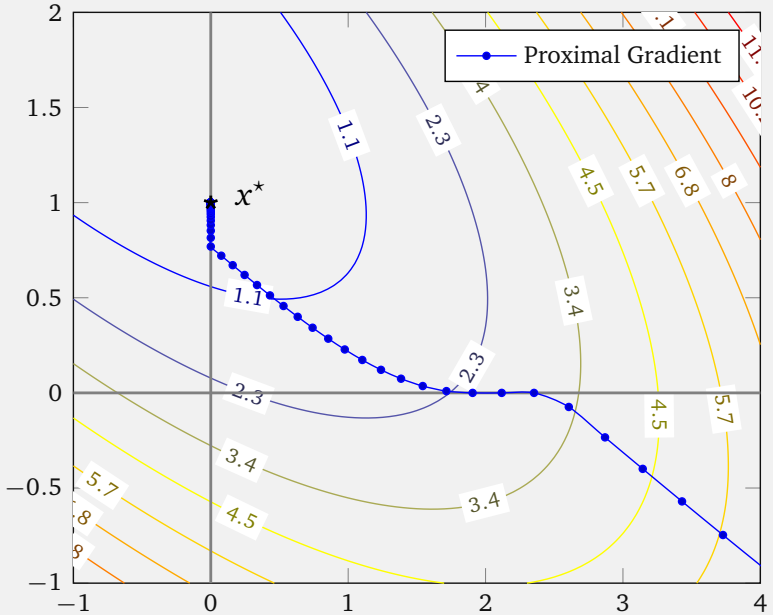
Coordinates	**Structure**	$\leftrightarrow$	**Optimality conditions**	$\leftrightarrow$	**Proximity operation**
$\forall i$	$x_i^\star = 0$	$\Leftrightarrow$	$A_i^\top (Ax^\star - b) \in [-\lambda, \lambda]$	$\Leftrightarrow$	$\left[\mathbf{prox}_{\gamma\lambda\|\cdot\|_1}(u^\star)\right]_i = 0$

$u^\star = x^\star - \gamma A^\top (Ax^\star - b)$

$$\left[\mathbf{prox}_{\gamma\lambda\|\cdot\|_1}(u)\right]_i = \begin{cases} u_i - \lambda\gamma & \text{if } u_i > \lambda\gamma \\ 0 & \text{if } u_i \in [-\lambda\gamma; \lambda\gamma] \\ u_i + \lambda\gamma & \text{if } u_i < -\lambda\gamma \end{cases}$$

Proximal Gradient (aka ISTA):

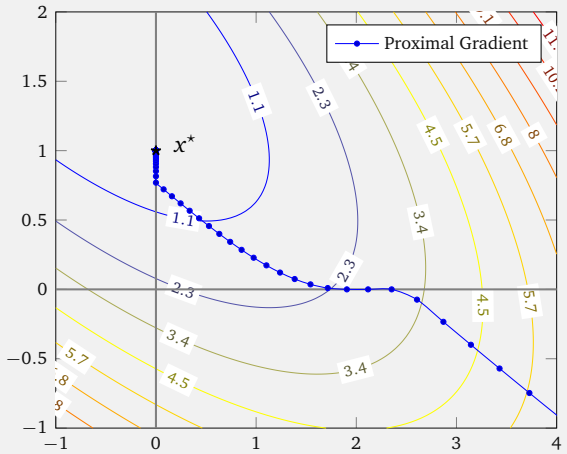
$$\begin{cases} u_{k+1} = x_k - \gamma A^\top (Ax_k - b) \\ x_{k+1} = \mathbf{prox}_{\gamma\lambda\|\cdot\|_1}(u_{k+1}) \end{cases}$$

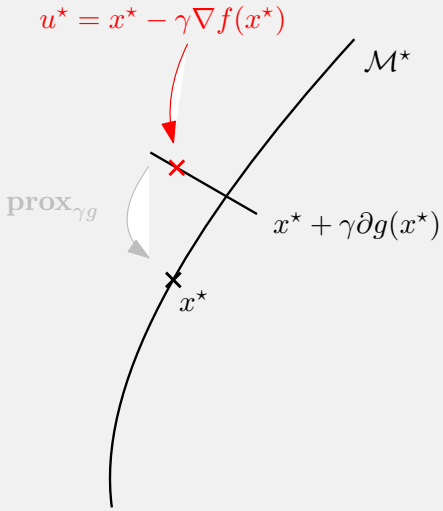


Iterates (x_k) reach the **same structure as $x^\star$** in **finite** time!

Proximal Algorithms:

$$\begin{cases} u_{k+1} = x_k - \gamma \nabla f(x_k) \\ x_{k+1} = \mathbf{prox}_{\gamma g}(u_{k+1}) \end{cases}$$





> **project on manifolds**

Let $\mathcal{M}$ be a manifold and $u^\star$ such that

$$x^\star = \mathbf{prox}_{\gamma g}(u^\star) \in \mathcal{M} \quad \text{and} \quad \frac{u^\star - x^\star}{\gamma} \in \mathrm{ri}\, \partial g(x^\star)$$

If g is *partly smooth* at $x^\star$ relative to $\mathcal{M}^\star$, then

$$\mathbf{prox}_{\gamma g}(u) \in \mathcal{M}^\star$$

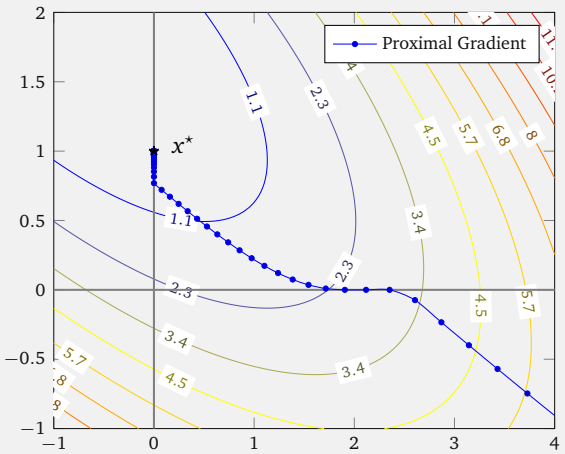
for any u close to $u^\star$.

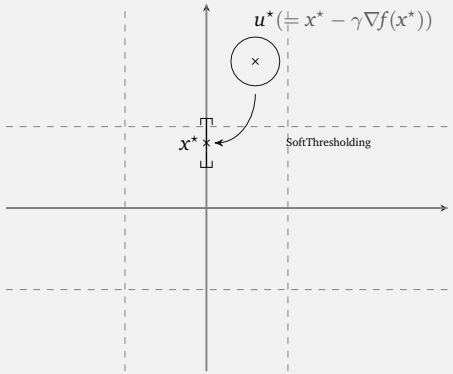
◇ Hare, Lewis: Identifying active constraints via partial smoothness and prox-regularity. Journal of Convex Analysis (2004)

◇ Daniilidis, Hare, Malick: Geometrical interpretation of the predictor-corrector type algorithms in structured optimization problems. Optimization (2006)

Proximal Algorithms:

$$\begin{cases} u_{k+1} = x_k - \gamma \nabla f(x_k) \\ x_{k+1} = \mathbf{prox}_{\gamma g}(u_{k+1}) \end{cases}$$





> **project on manifolds**

> **identify the optimal structure**

Let (x_k) and (u_k) be a pair of sequences such that

$$x_k = \mathbf{prox}_{\gamma g}(u^k) \to x^\star = \mathbf{prox}_{\gamma g}(u^\star)$$

and $\mathcal{M}$ be a manifold. If $x^\star \in \mathcal{M}$ and *"structure is stable under small perturbation of the data"*

$$\exists \varepsilon > 0 \text{ such that for all } u \in \mathcal{B}(u^\star, \varepsilon),\ \mathbf{prox}_{\gamma g}(u) \in \mathcal{M} \qquad \text{(QC)}$$

holds, then, **after some finite but unknown time,** $x_k \in \mathcal{M}$.

◇ Lewis: Active sets, nonsmoothness, and sensitivity. SIAM Journal on Optimization (2002)
◇ Fadili, Malick, Peyré: Sensitivity analysis for mirror-stratifiable convex functions. SIAM Journal on Optimization (2018)

> **Nonsmoothness** is actively studied in Numerical Optimization...
>
> Subgradients, Partial Smoothness/prox-regularity, Bregman geometry, etc.

- ◇ Hare, Lewis: Identifying active constraints via partial smoothness and prox-regularity. J. of Conv. Analysis (2004)
- ◇ Lemarechal, Oustry, Sagastizabal: The U-Lagrangian of a convex function. Trans. of the AMS (2000)
- ◇ Bolte, Daniilidis, Lewis: The Łojasiewicz inequality for nonsmooth subanalytic functions with applications to subgradient dynamical systems. SIAM J. on Optim. (2007)
- ◇ Chen, Teboulle: A proximal-based decomposition method for convex minimization problems. Math. Prog. (1994)

> **Nonsmoothness** is actively studied in Numerical Optimization...
>
> Subgradients, Partial Smoothness/prox-regularity, Bregman geometry, etc.

> ...but **often suffered** because of lack of structure/expression.
>
> Bundle methods, Gradient Sampling, Smoothing, Inexact proximal methods, etc.

- ◇ Nesterov: Smooth minimization of non-smooth functions. Mathematical Programming (2005)
- ◇ Burke, Lewis, Overton: A robust gradient sampling algorithm for nonsmooth, nonconvex optimization. SIAM J. on Optim. (2005)
- ◇ Solodov, Svaiter: A hybrid projection-proximal point algorithm. J. of Conv. Analysis (1999)
- ◇ de Oliveira, Sagastizábal: Bundle methods in the XXIst century: A bird’s-eye view. Pesquisa Operacional (2014)

> **Nonsmoothness** is actively studied in Numerical Optimization...
Subgradients, Partial Smoothness/prox-regularity, Bregman geometry, etc.

> ...but **often suffered** because of lack of structure/expression.
Bundle methods, Gradient Sampling, Smoothing, Inexact proximal methods, etc.

> For **Machine Learning objectives**, it can often be **harnessed**
Feature selection, Screening, Faster rates, etc.

◇ Bach, et al.: Optimization with sparsity-inducing penalties. FnT in Machine Learning (2012)
◇ Massias, Salmon, Gramfort: Celer: a fast solver for the lasso with dual extrapolation. ICML (2018)
◇ Liang, Fadili, Peyré: Local linear convergence of forward–backward under partial smoothness. NeurIPS (2014)
◇ O’Donoghue, Candes: Adaptive restart for accelerated gradient schemes. Foundations of Comp. Math. (2015)

> **Nonsmoothness** is actively studied in Numerical Optimization...
Subgradients, Partial Smoothness/prox-regularity, Bregman geometry, etc.

> ...but **often suffered** because of lack of structure/expression.
Bundle methods, Gradient Sampling, Smoothing, Inexact proximal methods, etc.

> For **Machine Learning objectives**, it can often be **harnessed**
Feature selection, Screening, Faster rates, etc.

> Why?

- *Explicit/“proximable”* regularizations ℓ_1, nuclear norm
- We *know* the *expressions* and *activity* of sought structures sparsity, rank
- Any converging proximal algorithm will *identify* the *optimal structure* of the problem.

▷ I. & Malick: *Nonsmoothness in Machine Learning: specific structure, proximal identification, and applications*, review/pedagogical paper to appear in Set-Valued and Variational Analysis, `https://arxiv.org/abs/2010.00848`

Thanks to the *Optimization for Machine Learning* week at CIRM in March 2020!

Let us solve a *Regularized ERM* problem with a *proximal algorithm*

$$\begin{cases} u_{k+1} & = \mathsf{Update}\left(f; \{x_\ell\}_{\ell \leq k}; \{u_\ell\}_{\ell \leq k}; \gamma\right) \\ x_{k+1} & = \mathbf{prox}_{\gamma g}(u_{k+1}) \end{cases}$$

with $x_k = \mathbf{prox}_{\gamma g}(u_k) \longrightarrow x^\star = \mathbf{prox}_{\gamma g}(u^\star)$

- The proximity operator gives a *current structure* $\mathcal{M}_k \subset \mathbb{R}^n$
  partial identif/screening
- We know that *eventually* $\mathcal{M}_k = \mathcal{M}^\star$ after some finite time
  identification

1– **Does faster minimization means faster identification ?**

2– **Can we efficiently restrict our update to $\mathcal{M}_k$?**

Example: Sparse structure and $g = \|\cdot\|_1$.

$\mathcal{M}^\star$ represents the points with the same support as $x^\star$ (ie. non-selected features are put to zero).
$\mathcal{M}_k = \{x \in \mathbb{R}^n : x_i = x_{i,k}\}$ is the current structure (same support as x_k).

- Interplay between Acceleration and Identification

  Adaptive Coordinate Descent

  Quick peek 1: Distributed learning

  Quick peek 2: Predictor-Corrector methods

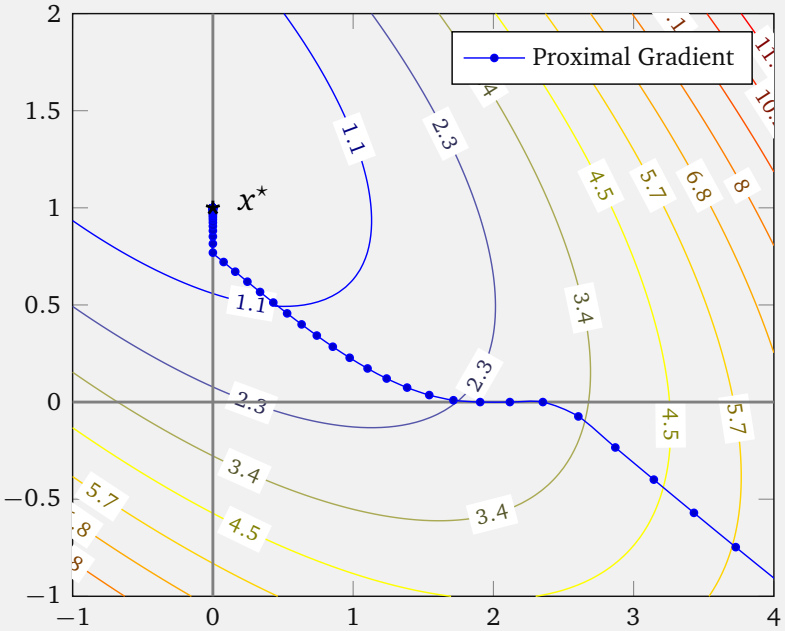
$$\begin{cases} u_{k+1} = y_k - \gamma \nabla f(y_k) \\ x_{k+1} = \mathbf{prox}_{\gamma g}(u_{k+1}) \\ y_{k+1} = x_{k+1} + \underbrace{\alpha_{k+1}(x_{k+1} - x_k)}_{\text{inertia/acceleration}} \end{cases}$$

- $\alpha_{k+1} = 0$: vanilla Proximal Gradient
- $\alpha_{k+1} = \frac{k-1}{k+3}$: **accelerated** Proximal Gradient (aka FISTA)
  Optimal rate for composite problems (coefficients may vary a little)

	PG	Accel. PG
$F(x_k) - F^\star$	$\mathcal{O}(1/k)$	$\mathcal{O}(1/k^2)$
iterates convergence	yes	yes
monotone functional decrease	yes	no
Fejér-monotone iterates	yes	no

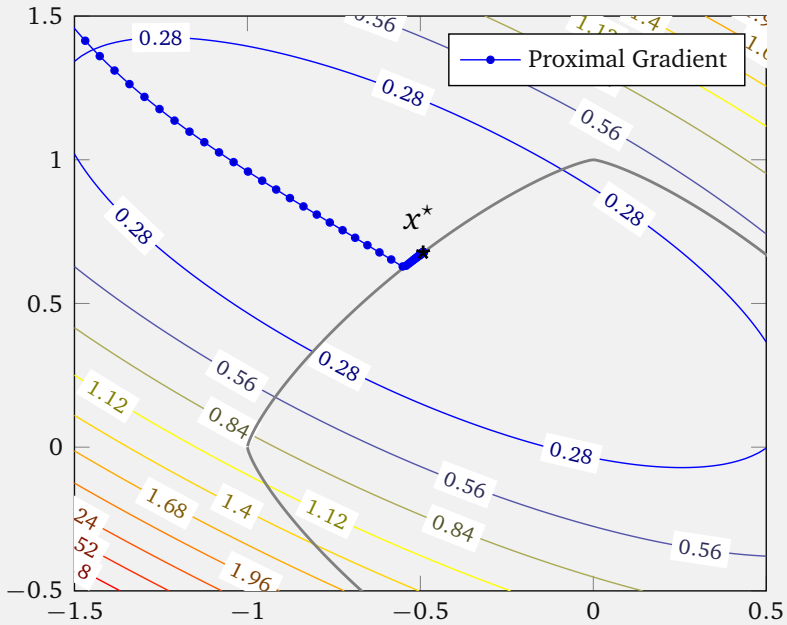
- Nesterov: A method for solving the convex programming problem with convergence rate $O(1/k^2)$. Sov. Dok. (1983)
- Beck, Teboulle: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM J. on Imag. Sci. (2009)
- Chambolle, Dossal: On the convergence of the iterates of "FISTA". J. of Optim. Theory and App. (2015)
- I., Malick: On the Proximal Gradient Algorithm with Alternated Inertia. J. of Optim. Theory and App. (2018)

$$\min_{x \in \mathbb{R}^2} \|Ax - b\|_2^2 + \lambda r(x)$$



$r(x) = \|x\|_1$

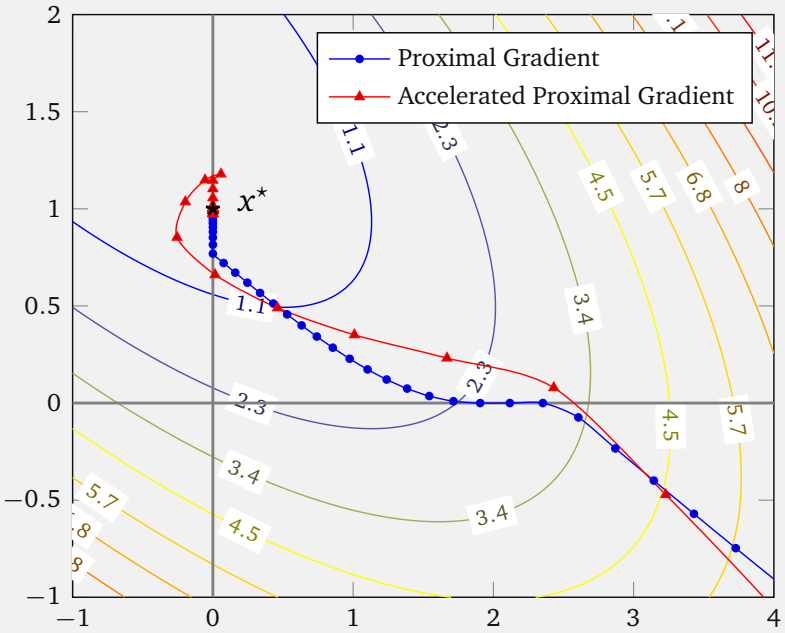
1-norm regularization



$r(x) = \max(\|x\|_{1.3} - 1; 0)$

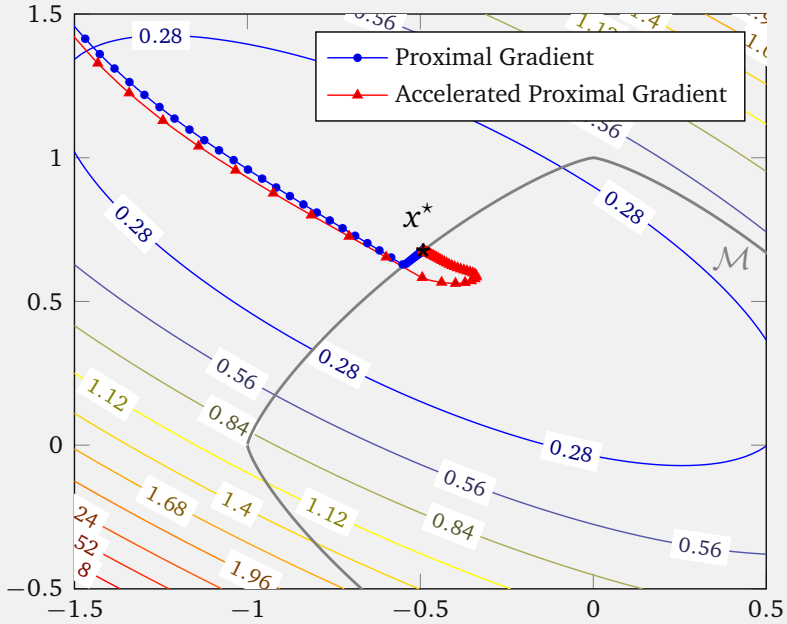
distance to 1.3-norm unit ball

$$\min_{x \in \mathbb{R}^2} \|Ax - b\|_2^2 + \lambda r(x)$$



$r(x) = \|x\|_1$

1-norm regularization



$r(x) = \max(\|x\|_{1.3} - 1; 0)$

distance to 1.3-norm unit ball

- PG identifies well;
- Accelerated PG explores well, identifies eventually, but erratically.

*Can we converge fast **and** identify well?*

T is a boolean function of past iterates; decides whether to accelerate or not.

$$\begin{cases} u_{k+1} = y_k - \gamma \nabla f(y_k) \\ x_{k+1} = \mathbf{prox}_{\gamma g}(u_{k+1}) \\ y_{k+1} = \begin{cases} x_{k+1} + \alpha_{k+1}(x_{k+1} - x_k) & \text{if } \mathsf{T} = 1 \\ x_{k+1} & \text{if } \mathsf{T} = 0 \end{cases} \end{cases}$$

Proposed tests:

We pre-define a collection $\mathsf{C} = \{\mathcal{M}_1, .., \mathcal{M}_p\}$ of *sought structures*

1. No Acceleration *i.e.* $\mathsf{T}^1 = 0$ when a new pattern is reached:

$$x_{k+1} \in \mathcal{M} \text{ and } x_k \notin \mathcal{M}$$

for some structure $\mathcal{M} \in \mathsf{C}$.

2. No Acceleration *i.e.* $\mathsf{T}^2 = 0$ if this means getting less structure:

$$\mathcal{T}_\gamma(x_{k+1}) \in \mathcal{M} \text{ and } \mathcal{T}_\gamma(x_{k+1} + \alpha_{k+1}(x_{k+1} - x_k)) \notin \mathcal{M}$$

for some $\mathcal{M} \in \mathsf{C}$.

where $\mathcal{T}_\gamma := \mathbf{prox}_{\gamma g}(\cdot - \gamma \nabla f(\cdot))$ is the proximal gradient operator.

Examples of sought structures: sparsity supports, rank.

Theorem

Let f, g be two convex functions such that f is L-smooth, g is lower semi-continuous, and $f + g$ is semi-algebraic with a minimizer. Take $\gamma \in (0, 1/L]$. Then, the iterates of the proposed methods with test T^1 or T^2 satisfy

$$F(x_{k+1}) - F^\star = \mathcal{O}\left(\frac{1}{k}\right)$$

for some $\mathsf{R} > 0$.
Furthermore, if the problem has a unique minimizer $x^\star$ and the qualifying constraint (QC) holds, then the iterates sequence (x_k) converges, finite-time identification happens and

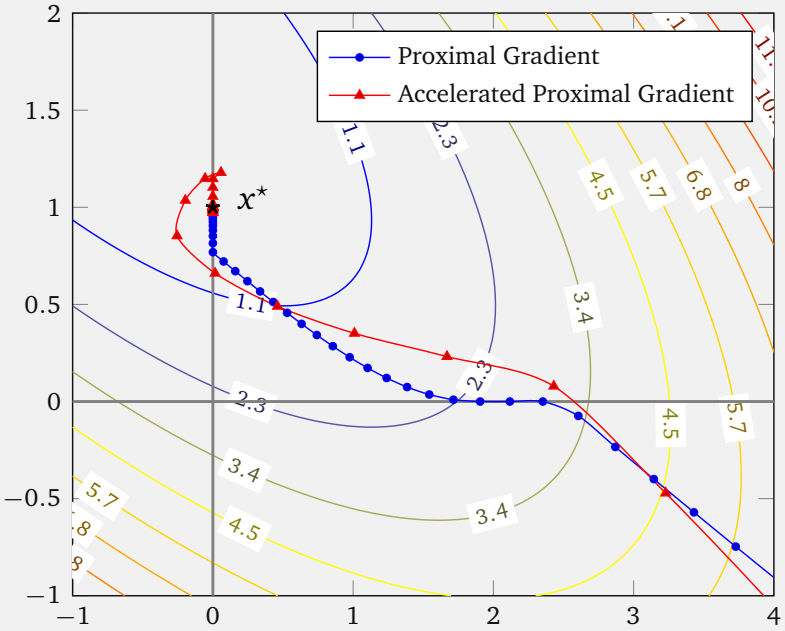
$$F(x_{k+1}) - F(x^\star) = \mathcal{O}\left(\frac{1}{k^2}\right).$$

L-smooth means that f is differentiable and ∇f is L-Lipschitz continuous.

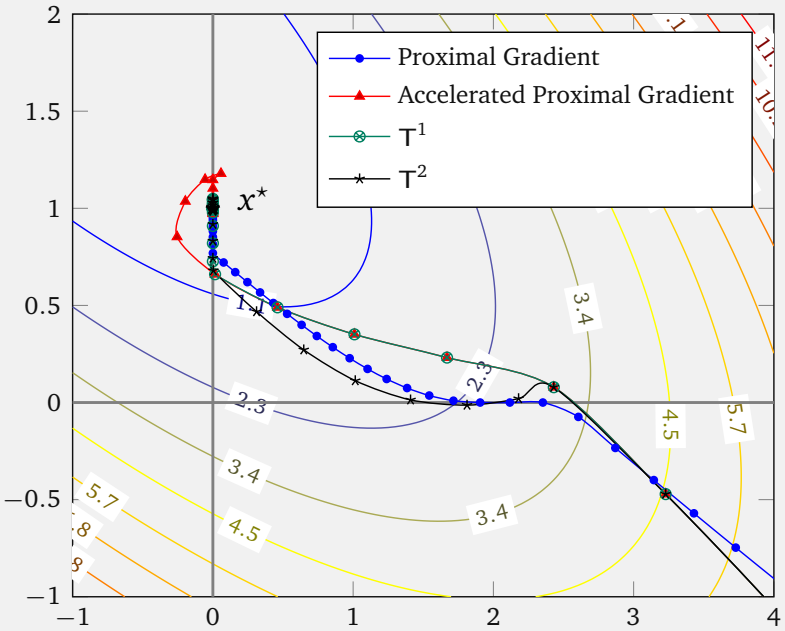
$$\exists \varepsilon > 0 \text{ such that for all } u \in \mathcal{B}(x^\star - \gamma \nabla f(x^\star), \varepsilon),\ \mathbf{prox}_{\gamma g}(u) \in \mathcal{M}^\star \qquad \text{(QC)}$$

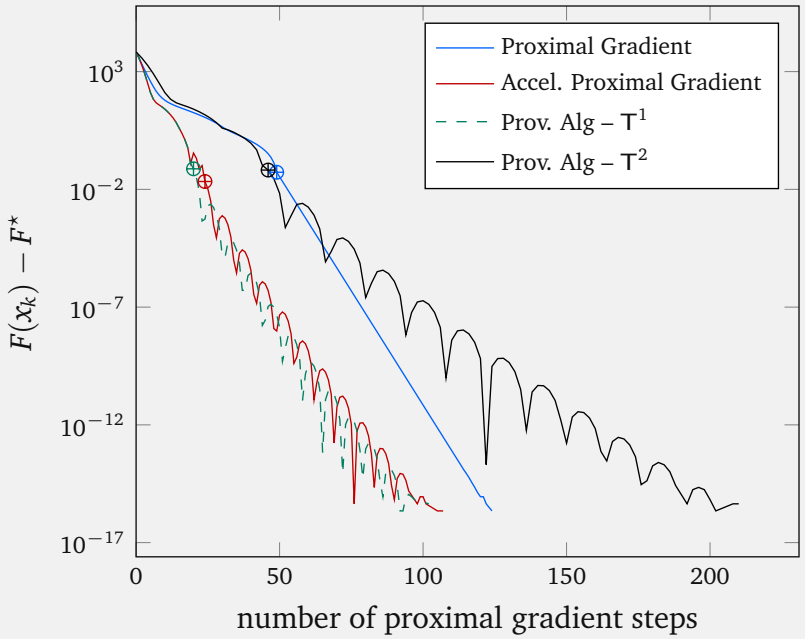
For the ℓ_1 norm, this means this means $-\nabla_i f(x^\star) \in (-\lambda; \lambda)$.

$$\min_{x \in \mathbb{R}^2} \|Ax - b\|_2^2 + \lambda \|x\|_1$$



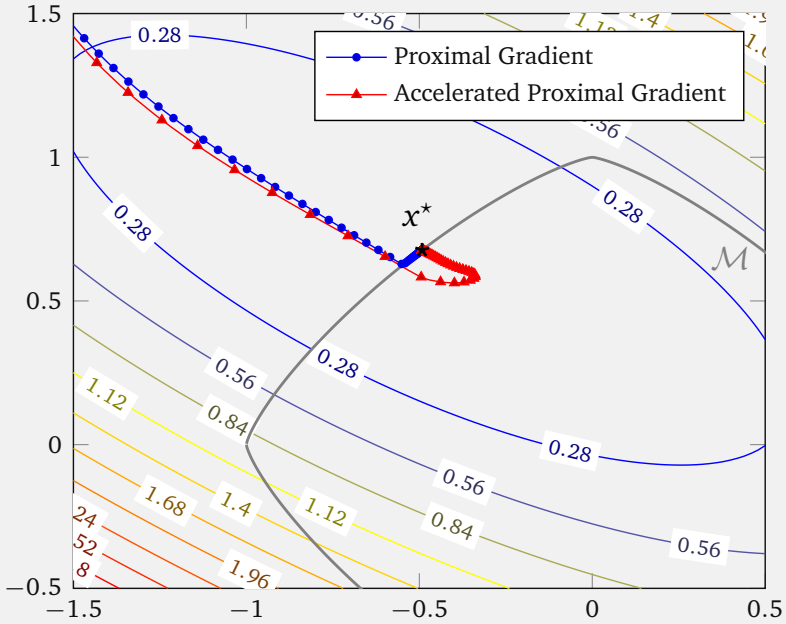
$$\min_{x \in \mathbb{R}^2} \|Ax - b\|_2^2 + \lambda \|x\|_1$$



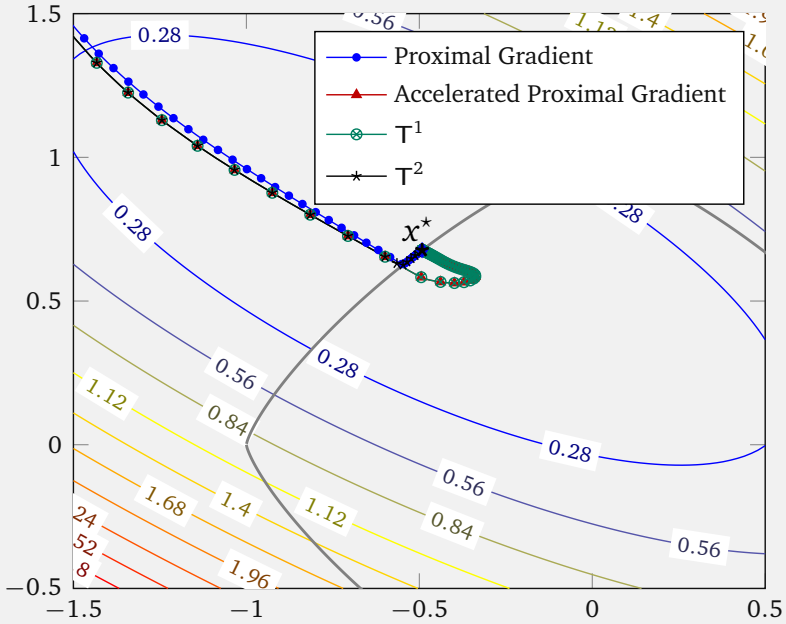


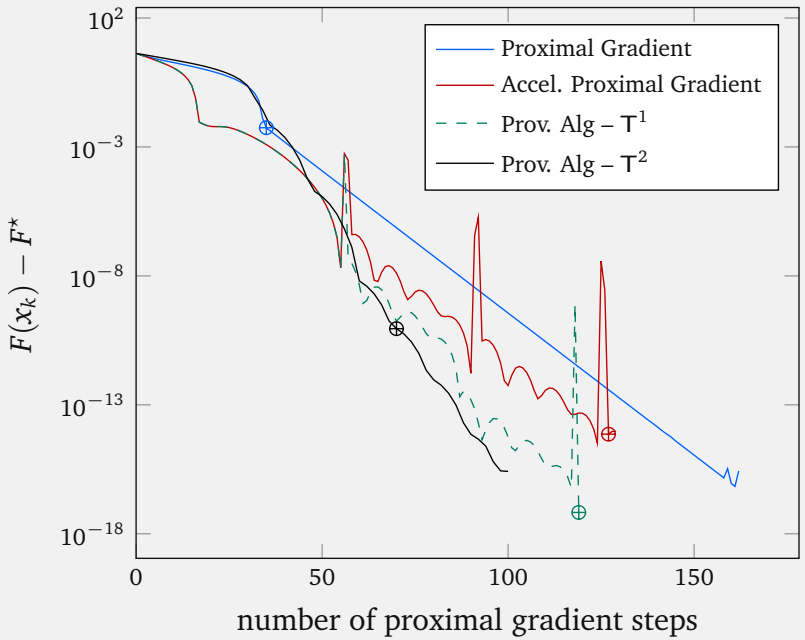
$\oplus$ marks identification time

$$\min_{x \in \mathbb{R}^2} \|Ax - b\|_2^2 + \lambda \max(|x\|_{1.3} - 1; 0)$$



$$\min_{x \in \mathbb{R}^2} \|Ax - b\|_2^2 + \lambda \max(|x\|_{1.3} - 1; 0)$$

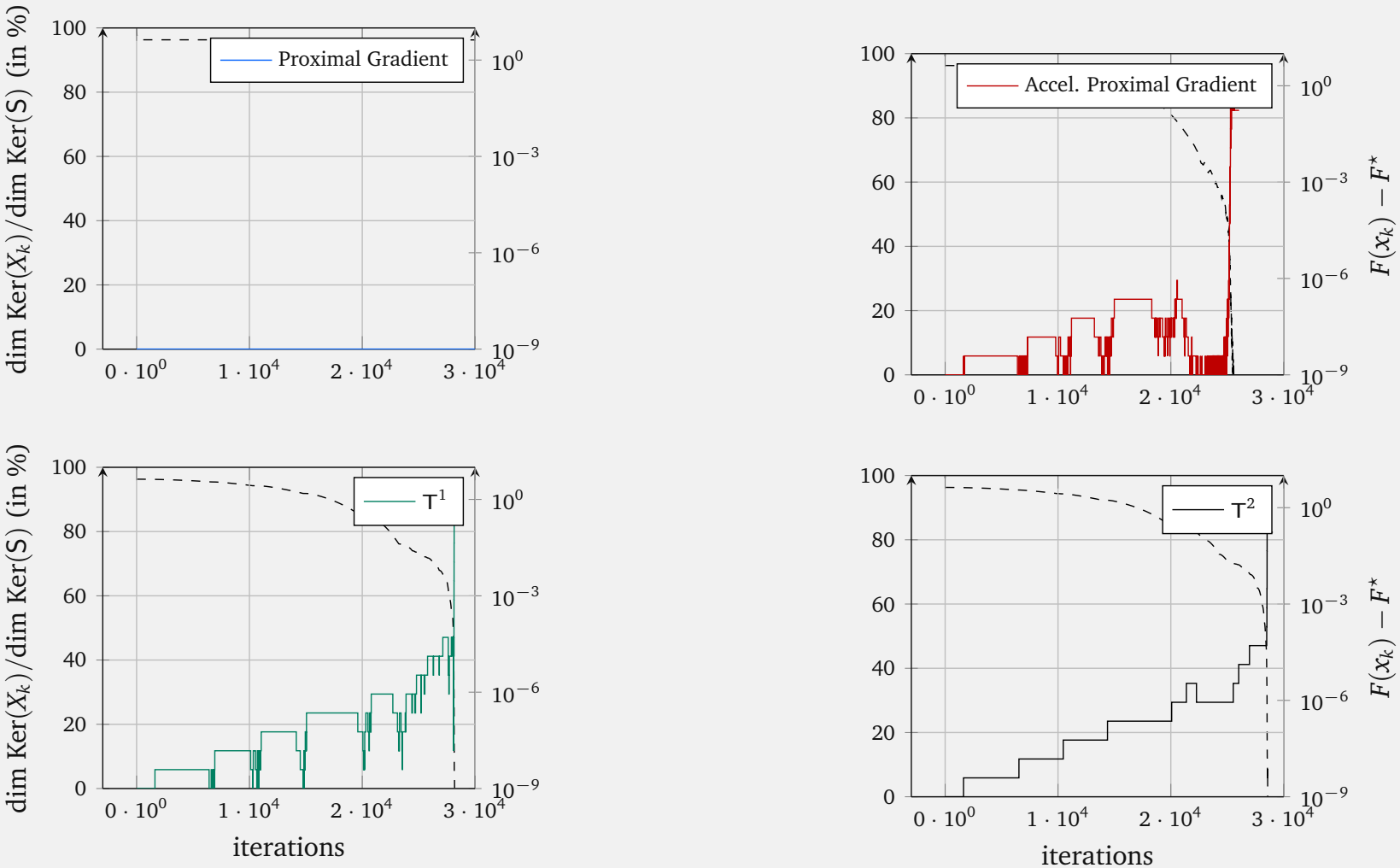




⊕ marks identification time

$$\min_{X \in \mathbb{R}^{20\times 20}} \|AX - B\|_F^2 + \lambda \|X\|_*$$

- $S \in R^{20\times 20}$ is a **rank 3** matrix;
- $A \in \mathbb{R}^{(16\times 16)\times(20\times 20)}$ is drawn from the normal distribution;
- $B = AS + E$ with E drawn from the normal distribution with variance .01



- acceleration can hurt identification for the proximal gradient algorithm;

  $\Rightarrow$ Faster convergence does not means faster structure identification

  $\Rightarrow$ Accuracy vs. Structure tradeoff for the learning problem
- we proposed a method with stable identification behavior, maintaining an accelerated convergence rate.

▷ Bareilles & I.: *On the Interplay between Acceleration and Identification for the Proximal Gradient algorithm*, Computational Optimization and Applications, 2020, `https://arxiv.org/abs/1909.08944`. Try it in Julia on `https://github.com/GillesBareilles/Acceleration-Identification`

Interplay between Acceleration and Identification

- **Adaptive Coordinate Descent**

Quick peek 1: Distributed learning

Quick peek 2: Predictor-Corrector methods

$$\text{Find} \quad x^\star \in \arg\min_{x \in \mathbb{R}^n} \overbrace{\frac{1}{2}\|Ax - b\|_2^2}^{f(x)} + \overbrace{\lambda \|x\|_1}^{g(x)}$$

Using a **Quadratic Program (QP)** solver in low dimension!

Works for - other regularizations eg. elastic net

- Friedman, Hastie, Tibshirani: glmnet R package (2009)
- — : Regularization paths for generalized linear models via coordinate descent. J. of Stat. Softw. (2010)
- Ndiaye, Fercoq, Gramfort, Salmon: Gap-safe screening rules for sparsity enforcing penalties. JMLR (2017)
- Massias, Gramfort, Salmon: Celer: a Fast Solver for the Lasso with Dual Extrapolation. ICML (2018)

$$\text{Find} \quad x^\star \in \arg\min_{x \in \mathbb{R}^n} \overbrace{\frac{1}{2}\|Ax-b\|_2^2}^{f(x)} + \overbrace{\lambda\|x\|_1}^{g(x)}$$

- Grad. of f: per coordinate

$$\nabla_i f(x) = A_i^\top (Ax - b)$$

- Prox. of g: per coordinate

$$\left[\mathbf{prox}_{\gamma\lambda\|\cdot\|_1}(u)\right]_i = \mathrm{ST}_{\gamma\lambda}(u_i) = \begin{cases} u_i - \lambda\gamma & \text{if } u_i > \lambda\gamma \\ 0 & \text{if } u_i \in [-\lambda\gamma; \lambda\gamma] \\ u_i + \lambda\gamma & \text{if } u_i < -\lambda\gamma \end{cases}$$

Proximal Gradient (aka ISTA): for *all* coordinates i

$$\begin{cases} u_{i,k+1} = x_{i,k} - \gamma A_i^\top (Ax_k - b) \\ x_{i,k+1} = \mathrm{ST}_{\gamma\lambda}(u_{i,k+1}) \end{cases}$$

Works for
- most Generalized Linear models eg. logistic
- other regularizations eg. elastic net

◇ Friedman, Hastie, Tibshirani: glmnet R package (2009)
◇ — : Regularization paths for generalized linear models via coordinate descent. J. of Stat. Softw. (2010)
◇ Ndiaye, Fercoq, Gramfort, Salmon: Gap-safe screening rules for sparsity enforcing penalties. JMLR (2017)
◇ Massias, Gramfort, Salmon: Celer: a Fast Solver for the Lasso with Dual Extrapolation. ICML (2018)

$$\text{Find} \quad x^\star \in \arg\min_{x \in \mathbb{R}^n} \overbrace{\frac{1}{2}\|Ax-b\|_2^2}^{f(x)} + \overbrace{\lambda\|x\|_1}^{g(x)}$$

- Grad. of f: per coordinate

$$\nabla_i f(x) = A_i^\top (Ax - b)$$

- Prox. of g: per coordinate

$$\left[\mathbf{prox}_{\gamma\lambda\|\cdot\|_1}(u)\right]_i = \mathrm{ST}_{\gamma\lambda}(u_i) = \begin{cases} u_i - \lambda\gamma & \text{if } u_i > \lambda\gamma \\ 0 & \text{if } u_i \in [-\lambda\gamma; \lambda\gamma] \\ u_i + \lambda\gamma & \text{if } u_i < -\lambda\gamma \end{cases}$$

Coordinate Descent:
for ***one*** coordinate i
chosen at random, or by importance, or by screening, etc.

$$\begin{cases} u_{i,k+1} = x_{i,k} - \gamma A_i^\top (Ax_k - b) \\ x_{i,k+1} = \mathrm{ST}_{\gamma\lambda}(u_{i,k+1}) \end{cases}$$

Works for
- most Generalized Linear models eg. logistic *but screening rules are looser*
- other regularizations eg. elastic net *as long as the prox is separable*

◇ Friedman, Hastie, Tibshirani: glmnet R package (2009)
◇ — : Regularization paths for generalized linear models via coordinate descent. J. of Stat. Softw. (2010)
◇ Ndiaye, Fercoq, Gramfort, Salmon: Gap-safe screening rules for sparsity enforcing penalties. JMLR (2017)
◇ Massias, Gramfort, Salmon: Celer: a Fast Solver for the Lasso with Dual Extrapolation. ICML (2018)

Disclaimer: We assume that the identified structure is linear
eg: ℓ_1/ℓ_2-group lasso, 1D TV-fused lasso but not separability of g.

$$\begin{cases} u_k &= x_k - \gamma \nabla f(x_k) \\ z_k &= u_k \\ x_{k+1} &= \mathbf{prox}_{\gamma g}(z_k) \end{cases}$$

> Vanilla Proximal gradient identifies but does not use it
full gradient computed at each iteration

We again pre-define a collection $\mathsf{C} = \{\mathcal{M}_1, .., \mathcal{M}_p\}$ of *sought structures* (eg. sparsity patterns $\mathcal{M}_i = \{x \in \mathbb{R}^n : x_i = 0\}$.

Disclaimer: We assume that the identified structure is linear
eg: ℓ_1/ℓ_2-group lasso, 1D TV-fused lasso but not separability of g.

$$\begin{cases} & \text{Observe } \mathcal{M}_k = \mathbb{R}^n \bigcap_{i:x_k \in \mathcal{M}_i} \mathcal{M}_i \\ u_k &= x_k - \gamma \nabla f(x_k) \\ z_k &= \text{proj}_{\mathcal{M}_k}(u_k) + \text{proj}^{\perp}_{\mathcal{M}_k}(z_{k-1}) \\ x_{k+1} &= \textbf{prox}_{\gamma g}(z_k) \end{cases}$$

> Direct Use of Identification may not converge
eg: starting with 0

We again pre-define a collection $\mathsf{C} = \{\mathcal{M}_1, .., \mathcal{M}_p\}$ of *sought structures* (eg. sparsity patterns $\mathcal{M}_i = \{x \in \mathbb{R}^n : x_i = 0\}$.
If we *knew* that $\mathcal{M}^\star \in \mathcal{M}_i$ (eg. looking at the suboptimality gap), we could *drop* the i-th coordinate update, ie. do *screening*.

Disclaimer: We assume that the identified structure is linear
eg: ℓ_1/ℓ_2-group lasso, 1D TV-fused lasso but not separability of g.

$$\begin{cases} & \text{Observe } \mathcal{M}_k = \mathbb{R}^n \bigcap_{i:x_k \in \mathcal{M}_i} (\xi_{k,i}\mathcal{M}_i + (1-\xi_{k,i})\mathbb{R}^n) \text{ for } \xi_{k,i} \sim \mathcal{B}(p) \\ u_k &= x_k - \gamma \nabla f(x_k) \\ z_k &= \text{proj}_{\mathcal{M}_k}(u_k) + \text{proj}^{\perp}_{\mathcal{M}_k}(z_{k-1}) \\ x_{k+1} &= \mathbf{prox}_{\gamma g}(z_k) \end{cases}$$

> Mixing Identification and Randomized "coordinate" descent biases
convergence issues

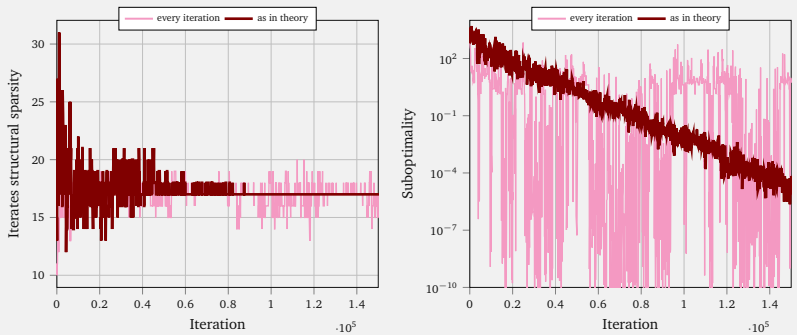
We again pre-define a collection $\mathsf{C} = \{\mathcal{M}_1, .., \mathcal{M}_p\}$ of *sought structures* (eg. sparsity patterns $\mathcal{M}_i = \{x \in \mathbb{R}^n : x_i = 0\}$.

Disclaimer: We assume that the identified structure is linear
eg: ℓ_1/ℓ_2-group lasso, 1D TV-fused lasso but not separability of g.

$$\begin{cases} & \text{Observe } \mathcal{M}_k = \mathbb{R}^n \bigcap_{i:x_k \in \mathcal{M}_i} (\xi_{k,i}\mathcal{M}_i + (1-\xi_{k,i})\mathbb{R}^n) \text{ for } \xi_{k,i} \sim \mathcal{B}(p) \\ & \text{and compute } \mathbf{P}_k = \mathbb{E}\text{proj}_{\mathcal{M}_k} \text{ and } Q_k = (\mathbf{P}_k)^{-1/2} \\ u_k & = Q_k\left(x_k - \gamma \nabla f(x_k)\right) \\ z_k & = \text{proj}_{\mathcal{M}_k}(u_k) + \text{proj}^{\perp}_{\mathcal{M}_k}(z_{k-1}) \\ x_{k+1} & = \mathbf{prox}_{\gamma g}(Q_k^{-1} z_k) \end{cases}$$

> Unbiasing with Q_k works *after identification*
but before... no, which prevents identification...

TV-regularized logistic regression:

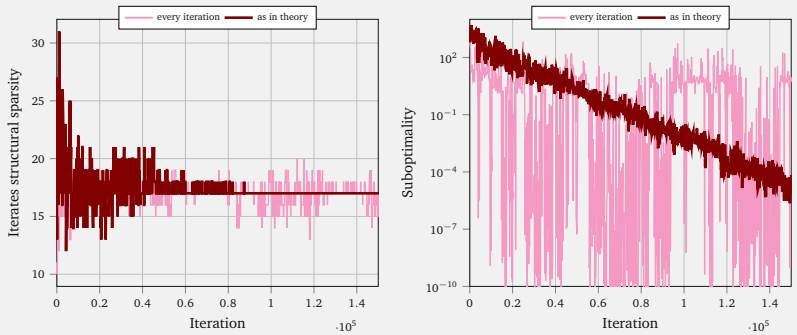


Disclaimer: We assume that the identified structure is linear
eg: ℓ_1/ℓ_2-group lasso, 1D TV-fused lasso but not separability of g.

$$\begin{cases} & \text{Observe } \mathcal{M}_k = \mathbb{R}^n \bigcap_{i:x_\ell \in \mathcal{M}_i} (\xi_{k,i}\mathcal{M}_i + (1-\xi_{k,i})\mathbb{R}^n) \text{ for } \xi_{k,i} \sim \mathcal{B}(p) \\ & \text{and compute } \mathbf{P}_k = \mathbb{E}\text{proj}_{\mathcal{M}_k} \text{ and } Q_k = (\mathbf{P}_k)^{-1/2} \textit{sometimes}, \text{ else keep prev. dist.} \\ u_k & = Q_k\left(x_k - \gamma \nabla f(x_k)\right) \\ z_k & = \text{proj}_{\mathcal{M}_k}(u_k) + \text{proj}^{\perp}_{\mathcal{M}_k}(z_{k-1}) \\ x_{k+1} & = \mathbf{prox}_{\gamma g}(Q_k^{-1} z_k) \end{cases}$$

> Structure adaptation can be performed at *some* iterations
depends on the *amount of change* $\|Q_{k-1}Q_k^{-1}\|$ and *harshness* of the sparsification $\lambda_{\min}(Q_k)$

TV-regularized logistic regression:



Theorem (informal)

Let f, g be two convex functions such that f is L-smooth, μ-strongly convex, g is lower semi-continuous. Take $\gamma \in (0, 2/(\mu + L)]$. Then, one can devise a adaptation strategy such that the iterates of the previous method satisfy

$$\mathbb{E}\|x_k - x^\star\| = \mathcal{O}\left(\left(1 - \lambda \frac{\gamma\mu L}{\mu + L}\right)^{a_k}\right)$$

where a_k is the number of adaptations performed before k and $\lambda = \inf_k \lambda_{\min}(\mathbb{E}\text{proj}_{\mathcal{M}_k})$.

Furthermore, if the problem has a unique minimizer $x^\star$ and the qualifying constraint (QC) holds, then the iterates sequence (x_k) converges, finite-time identification happens and

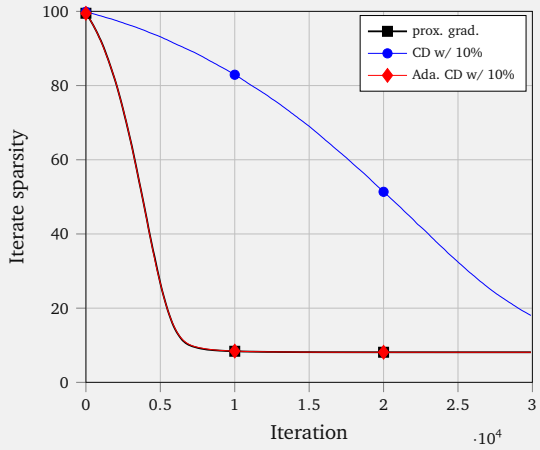
$$\|x_k - x^\star\| = \mathcal{O}_{\mathbb{P}}\left(\left(1 - 2\lambda \frac{\gamma\mu L}{\mu + L}\right)^{k}\right).$$

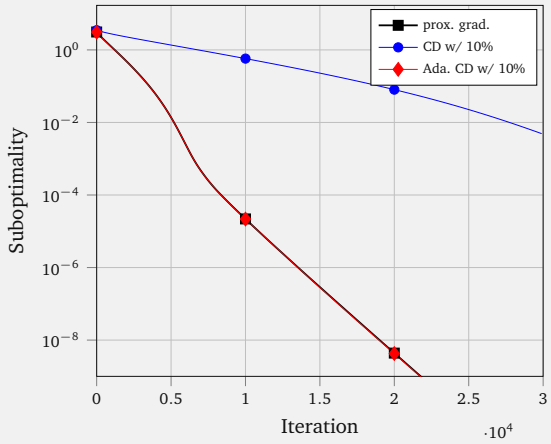
$$\exists \varepsilon > 0 \text{ such that for all } u \in \mathcal{B}(x^\star - \gamma\nabla f(x^\star), \varepsilon),\ \mathbf{prox}_{\gamma g}(u) \in \mathcal{M}^\star \tag{QC}$$

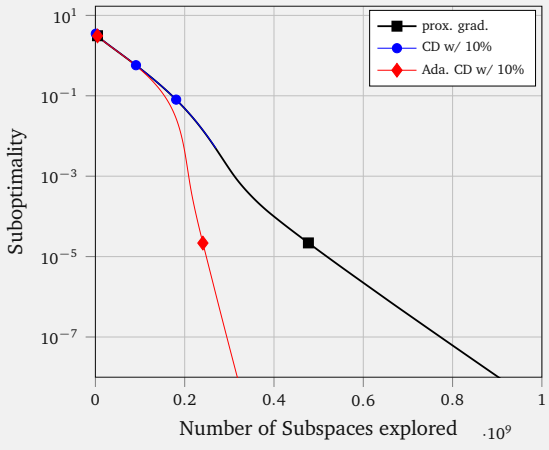
For the ℓ_1 norm, this means this means $-\nabla_i f(x^\star) \in (-\lambda; \lambda)$.

Logistic regression on a1a (1605×143),

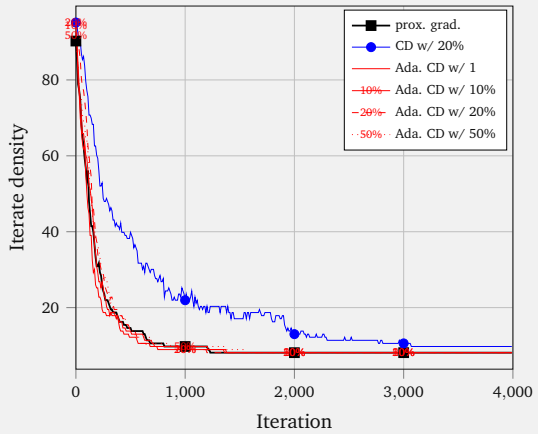
with ℓ_1-reg. 90% final coordinate sparsity

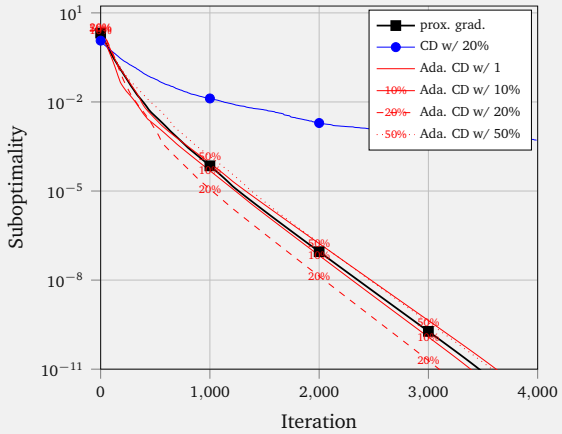


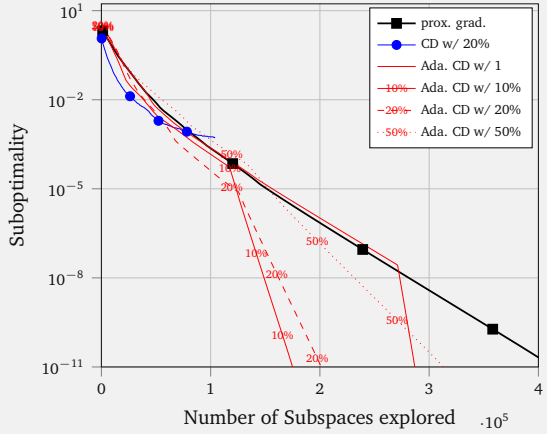




with TV-reg. 90% final *jump* sparsity







- Iterate structure enforced by nonsmooth regularizers can be used to adapt the selection probabilities of coordinate descent/sketching;
- Before identification, adaptation *has to be moderate*;
- *Qualified* minimizers once again grant better results in theory.

▷ Grishchenko, I., & Malick: *Proximal Gradient Methods with Adaptive Subspace Sampling*, Mathematics of Operation Research, 2020, `https://arxiv.org/abs/2004.13356`

Interplay between Acceleration and Identification

Adaptive Coordinate Descent

- Quick peek 1: Distributed learning

Quick peek 2: Predictor-Corrector methods

Algorithm

Worker i updates w/ local data (f_i)

$$x_i^{k+1/2} = x^k - \gamma \nabla f_i(x^k)$$

for all $i = 1, .., M$

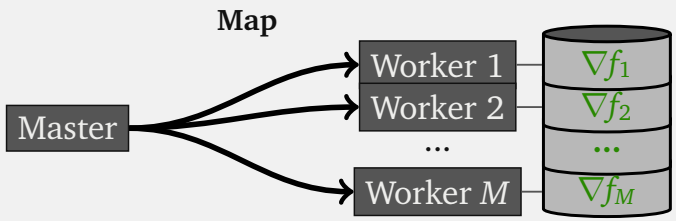
Master gathers the local variables

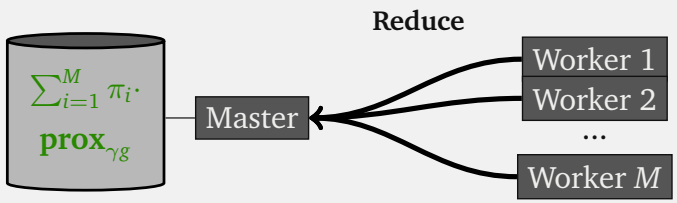
$$\bar{x}^{k+1} = \sum_{i=1}^M \pi_i x_i^{k+1/2}$$

Master performs a proximity operation

$$x_1^{k+1} = .. = x_M^{k+1} = \mathbf{prox}_{\gamma g}\left(\bar{x}^{k+1}\right)$$

Implementation





Distributed Proximal Gradient

Master:

Initialize $\bar{x} = \bar{x}^0$,
while *not converged* **do**
  when *all* workers have finished:
  Receive (x_i) from each of them
  $\bar{x} \leftarrow \sum_{i=1}^M \pi_i x_i$
  $x \leftarrow \mathbf{prox}_{\gamma g}(\bar{x})$
  Broadcast x to all agents
  $k \leftarrow k + 1$
Interrupt all slaves
Output x

Worker i:

Initialize $x = x_i = \bar{x}$,
while *not interrupted by master* **do**
  Receive the most recent x
  $x_i \leftarrow x - \gamma \nabla f_i(x)$
  Send x_i to the master

$$f_i(x) = \frac{1}{|\mathcal{S}_i|} \sum_{j \in \mathcal{S}_i} \ell_j(x)$$

Local risk at worker i

Communications may soon become the bottleneck in distributed learning, hence the rise of asynchronous methods.

DAve-PG

Master:

Initialize $\bar{x}$
while *not converged* **do**
  when a worker finishes:
  Receive adjustment Δ from it
  $\bar{x} \leftarrow \bar{x} + \Delta$
  $x \leftarrow \mathbf{prox}_{\gamma g}(\bar{x})$
  Send x to the agent in return
  $k \leftarrow k + 1$
Interrupt all slaves
Output x

Worker i:

Initialize $x = x_i = \bar{x}$,
while *not interrupted by master* **do**
  Receive the most recent x
  $x_i \leftarrow x - \gamma \nabla f_i(x)$
  $\Delta \leftarrow \pi_i \left(x_i - x_i^{prev} \right) \quad x_i^{prev} \leftarrow x_i$
  Send adjustment Δ to master

$$f_i(x) = \frac{1}{|\mathcal{S}_i|} \sum_{j \in \mathcal{S}_i} \ell_j(x)$$

Local risk at worker i

- With *sparsity inducing* regularizers (eg. ℓ_1 norm), *master-to-worker* communications will eventually become sparse $\Rightarrow$ **Use it!**
  identification of proximal methods
- Unfortunately, *worker-to-master* communications stay dense...
  Idea: sparsify adaptively!

- Whenever structure appears, it can often be used numerically
  storage, communications
- Importance of qualified solutions and adaptation frequency
  to achieve best theoretical performance

▷ Mishchenko, I., Malick, Amini: *A Delay-tolerant Proximal-Gradient Algorithm for Distributed Learning*, ICML 2018
`http://proceedings.mlr.press/v80/mishchenko18a.html`

▷ Grishchenko, I., Malick, Amini: *Distributed Learning with Sparse Communications by Identification*, 2020 `https://arxiv.org/abs/1812.03871`

Interplay between Acceleration and Identification

Adaptive Coordinate Descent

Quick peek 1: Distributed learning

- Quick peek 2: Predictor-Corrector methods

Recall that when solving a *Regularized ERM* problem with proximal gradient

$$\begin{cases} u_{k+1} &= x_k - \gamma \nabla f(x_k) \\ x_{k+1} &= \mathbf{prox}_{\gamma g}(u_{k+1}) \end{cases}$$

the proximity operator outputs a *current structure* $\mathcal{M}_k \subset \mathbb{R}^n$ ($x_k \in \mathcal{M}_k$) and *eventually* $\mathcal{M}_k = \mathcal{M}^\star$.

Recall that when solving a *Regularized ERM* problem with proximal gradient

$$\begin{cases} \text{Observe } \mathcal{M}_k, \text{ then } y_{k+1} & = \text{RiemannianStep}_{f+g}(x_k, \mathcal{M}_k) \\ u_{k+1} & = y_k - \gamma \nabla f(y_k) \\ x_{k+1} & = \mathbf{prox}_{\gamma g}(u_{k+1}) \end{cases}$$

the proximity operator outputs a *current structure* $\mathcal{M}_k \subset \mathbb{R}^n$ ($x_k \in \mathcal{M}_k$) and *eventually* $\mathcal{M}_k = \mathcal{M}^\star$.

Predictor-Corrector methods: perform a Riemannian step on $\mathcal{M}_k$, then a proximal step to correct the structure, and so on.

- ◇ Lemaréchal, Oustry, Sagastizábal: The U-Lagrangian of a convex function. Trans. of the AMS (2000)
- ◇ Daniilidis, Hare, Malick: Geometrical interpretation of the predictor-corrector type algorithms in structured optimization problems. Optimization (2006)

Recall that when solving a *Regularized ERM* problem with proximal gradient

$$\begin{cases} \text{Observe } \mathcal{M}_k, \text{ then } y_{k+1} & = \text{RiemannianStep}_{f+g}(x_k, \mathcal{M}_k) \\ u_{k+1} & = y_k - \gamma \nabla f(y_k) \\ x_{k+1} & = \mathbf{prox}_{\gamma g}(u_{k+1}) \end{cases}$$

the proximity operator outputs a *current structure* $\mathcal{M}_k \subset \mathbb{R}^n$ ($x_k \in \mathcal{M}_k$) and *eventually* $\mathcal{M}_k = \mathcal{M}^\star$.

Predictor-Corrector methods: perform a Riemannian step on $\mathcal{M}_k$, then a proximal step to correct the structure, and so on.

Numerical boost: Riemannian (truncated) Newton methods can be orders of magnitude faster than proximal gradient.

eg. cv. in ≈ 1 iteration for the lasso if a (true) small enough support is detected; reduction to smaller rank for matrix regression.

- ◇ Lemaréchal, Oustry, Sagastizábal: The U-Lagrangian of a convex function. Trans. of the AMS (2000)
- ◇ Daniilidis, Hare, Malick: Geometrical interpretation of the predictor-corrector type algorithms in structured optimization problems. Optimization (2006)

- Regularized ERM problems often have a *particular* proximal structure
  current structure knowledge is often deemed impossible in nonsmooth optimization since eg. numerically checking the rank of a matrix is hard; however, after a proximal step that thresholds singular values, the numerical/theoretical rank is known!
- Non-convex regularizations can work
  you may use ℓ_0 semi norm, rank for a matrix

▷ Bareilles, I., Malick: *???*, to appear soonish!

- Machine Learning problems often have a *noticeable structure*;
  sparsity, low rank
- This structure is identified progressively by *proximal methods*;
  + CD, Var. Red., Distributed methods, etc.
- For most problem, we *do not know* if the identified structure is optimal;
  adaptivity is key
- Nevertheless, it can be used to boost numerical performance;
  low complexity model
- *Structure vs. Optimality* tradeoff in Optimization for ML.
  structure is better than overfitting

▷ I., Malick: *Nonsmoothness in Machine Learning: specific structure, proximal identification, and applications*, review/pedagogical paper, SVVA, 2020, https://arxiv.org/abs/2010.00848

▷ Bareilles, I.: *On the Interplay between Acceleration and Identification for the Proximal Gradient algorithm*, COAP, 2020, https://arxiv.org/abs/1909.08944.

▷ Grishchenko, I., Malick: *Proximal Gradient Methods with Adaptive Subspace Sampling*, MOR, 2020, https://arxiv.org/abs/2004.13356

▷ Grishchenko, I., Malick, Amini: *Distributed Learning with Sparse Communications by Identif.*,2020 https://arxiv.org/abs/1812.03871

Thanks to ANR JCJC *STROLL* & IDEX UGA IRS *DOLL* & PGMO

Thank you! – Franck IUTZELER http://www.iutzeler.org