Convex Optimization and Applications
-
Alternating Direction Method of Multipliers (ADMM)

Valentin Leplat & Anh Huy Phan

Skoltech
Center for Artificial Intelligence Technology

Email: {*v.leplat,a.phan*}*@skoltech.ru*

December 05, 2023

In a nutshell

- Main Purpose: The Alternating Direction Method of Multipliers (ADMM) is an algorithm that solves convex optimization problems.
- Main Principle: **ADMM** algorithm breaks problems into smaller pieces, each of which are then easier to handle.
- Motivation: found wide application in the last decade in areas such as large-scale problems arising in statistics and machine learning.
  Reason: **ADMM** is well suited to distributed convex optimization
- Origin: developed in the 1970s, with roots in the 1950s.
  Related to many other algorithms: dual decomposition, the method of multipliers, Douglas-Rachford splitting, proximal methods, and others.
- About the roadmap of this course: we first introduce some basic assumptions. Before introducing **ADMM** itself, two methods need to be presented first, namely **Dual decomposition** and **Method of Multipliers**. **ADMM** is born from the desire to get the best of those two methods.

Roadmap

Blanket assumptions

this course: underlying space is the Euclidean space $\mathbb{R}^n$, a particular case of Hilbert space $\mathcal{X}$ of finite dimension n, that is, a Banach space equipped with:

- an inner product $\langle ., . \rangle$, here we consider the dot product $\langle x, y \rangle = \sum_i^n x_i y_i$ for $x, y \in \mathbb{R}^n$,
- induced norm $\|.\| = \sqrt{\langle ., . \rangle}$

Set of assumptions on functions

Through this course:

- focus on the minimization of a function f over a convex set by basically tackling its dual problem or a modified version of it ("Augmented").
- important to always keep in mind the set of assumptions made on the functions, in particular on f (the primal objective function).
- The most important assumptions will be highlighted in red when needed
- The derived results (mainly linked to the convergence results of the algorithms discussed in the present document) are only valid in the set of assumptions considered (= the paradigm).

Roadmap

Dual Problem

- Given $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$ and $f : \mathbb{R}^n \to \mathbb{R}$ a convex function , we are interested in solving:

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f(x) \\ \text{subject to} \quad & Ax = b, \end{aligned} \tag{1}$$

- Form the **Lagrangian** function: $L(x, y) = f(x) + \langle y, Ax - b \rangle$
  where $Ax - b$ is the residual and $y \in \mathbb{R}^m$ are referred to as Lagrangian multipliers (= dual variables).
  What does it mean ?: we allow the constraints to be violated, but there is a price to pay which is given by the term $\langle y, Ax - b \rangle$.

Dual Problem

- Build the **dual** function: $g(y) := \inf_x L(x, y)$
  Particular structure for $g(y)$: the dual function is always concave.
- Dual Problem: $\text{maximize}_y \quad g(y)$
- Let us assume that the maximization goes well. By denoting $y^\star := \underset{y \in \mathbb{R}^m}{\text{argmax}} \quad g(y)$, we may recover $x^\star := \underset{x \in \mathbb{R}^n}{\text{argmin}} \quad L(x, y^\star)$.
  Does this approach always work ? : NO. Actually this may fail even for linear problems. However, with enough and appropriate assumptions, this method works (ex: minimal curvature $\iff$ not too "flat").

Solve the Dual Problem: the Dual ascent scheme

- How to maximize the dual function ?
  $\rightarrow$ Use an **ascent** gradient method: $y^{k+1} := y^k + \alpha^k \nabla g(y^k)$
  where α^k is the step size (positive scalar), k denotes the iteration counter, and $g(y)$ is assumed to be differentiable w.r.t. y [1].
- $\nabla g(y^k) = A\tilde{x} - b$, where $\tilde{x} := \underset{x \in \mathbb{R}^n}{\operatorname{argmin}} \quad L(x, y^k)$
- Dual ascent method is an iterative scheme such that at each iteration we have two steps:

1. $x^{k+1} := \underset{x \in \mathbb{R}^n}{\operatorname{argmin}} \quad L(x, y^k)$ %x-minimization
2. $y^{k+1} := y^k + \alpha^k (Ax^{k+1} - b)$ %dual update

[1] it requires f to be closed and strictly convex, if not, a subgradient method should be used

Solve the Dual Problem: the Dual ascent scheme

Few remarks:

- Step 1: consider L, fix the dual vector and minimize over x.
- Step 2: calculate the residual
  1. If the residual is zero, optimality reached $\rightarrow$ stop
  2. Otherwise, update the dual variables by some positive number times the residual
- This method works, but again with many strong assumptions.
- *Why do we solve the dual instead of the primal ?:* The reason has to do with the notion of dual decomposition...

Additional Assumption: Separability

- Assume that function f is separable:

$$f(x) = f_1(x_1) + ... + f_N(x_N), \quad x := (x_1, ..., x_N)^T$$

where $1 \leqslant N \leqslant n$.
It means that f can be expressed as a sum of functions of individual blocks.

- then L is separable in x: $L(x, y) = \sum_i^N L_i(x_i, y) - \langle y, b \rangle$ with $L_i(x_i, y) = f_i(x_i) + \langle y, A_i x_i \rangle$ and $A_i \in \mathbb{R}^{m \times q(i)}$ where $q(i)$ denotes the size of block x_i.
- x-minimization step from dual ascent scheme splits into N independent problems:

$$x_i^{k+1} := \underset{x_i \in \mathbb{R}^{q(i)}}{\text{argmin}} \quad L_i(x_i, y^k)$$

....which can be solved in parallel !

Dual decomposition

Dual Decomposition algorithm (Everett, Dantzig, Wolfe, Benders, 1960-65) is:

1. $\forall i \in [1, N] : x_i^{k+1} := \underset{x_i \in \mathbb{R}^{q(i)}}{\text{argmin}} \quad L_i(x_i, y^k)$ %x_i-minimization in parallel
2. $y^{k+1} := y^k + \alpha^k (\sum_i^N A_i x_i^{k+1} - b)$ %dual update

- Step 1: minimization of each Lagrangian term L_i separately
- Step 2: gathering of the contributions to the equality constraints
  If the residual is zero $\rightarrow$ Stop
  Otherwise, update of the dual variables

Dual decomposition

- *What is required ?*:
  1. a scattering of the dual variables (y^k),
  2. update of x_i in parallel,
  3. a gathering ($\sum_i^N A_i x_i^{k+1}$). (provides coordination)
- $\longrightarrow$ Distributed (convex) optimization !
  $\longrightarrow$ Allow to solve large problems !
- works with a lot of assumptions; often slow.
  Reason: to have interesting properties on $g(y)$ for minimization schemes, it requires strong assumptions on f.

Roadmap

Method of Multipliers

- Goal: develop a method to make the Dual Ascent more robust (so that it would work for problems as linear problems, but not only !)
- Main idea: use an **augmented Lagrangian** (Hestenes, Powell, 1969):

$$L_\rho(x, y) = f(x) + \langle y, Ax - b\rangle + \frac{\rho}{2}\|Ax - b\|_2^2$$

where $\rho > 0$ is a constant (the penalty parameter for the constraint).

- **Method of Multipliers** (Hestenes, Powell; analysis in Bertsekas in 1982)

1. $x^{k+1} := \underset{x\in\mathbb{R}^n}{\text{argmin}} \quad L_\rho(x, y^k)$ %x-minimization
2. $y^{k+1} := y^k + \rho(Ax^{k+1} - b)$ %dual update with step length ρ

Method of Multipliers: Remarks

An interesting and alternative introduction of the concept: the method modifies the problem (1) as follows:

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f(x) + \frac{\rho}{2}\|Ax - b\|_2^2 \\ \text{subject to} \quad & Ax = b, \end{aligned} \tag{2}$$

Clearly extra term $\frac{\rho}{2}\|Ax - b\|_2^2$ does not change problem. On top of that:

Proposition

Let $h = f + g$ where f is convex function and g is a strongly convex function, then h is strongly convex function.

Hence, assuming, e.g., A has full rank, primal objective is strongly convex of parameter $\rho\sigma_{\min}^2(A)$ (f is then also strictly convex), the dual function gets good properties for the use of dual ascent method.

Method of Multipliers: Remarks

Differences with Dual Ascent method:

- Addition of a nonnegative term which is a quadratic penalty (always a cost) to the Lagrangian function.
- The gain for the dual update has a very specific step size $=$ the penalty parameter ρ.

Why ?...

Method of Multipliers: dual update step

- Let us write the first-order optimality conditions for Problem (1) (for f differentiable):
  1. $Ax^\star - b = 0$ (Primal feasibility),
  2. $\nabla_x L(x^\star, y^\star) = \nabla_x f(x^\star) + A^T y^\star = 0$ (Dual feasibility).
- Step 1 from **Method of Multipliers**: x^{k+1} is the minimizer of $L_\rho(x, y^k)$, therefore it satisfies the following condition (unconstrained optimization problem):

$$\begin{aligned} 0 &= \nabla_x L_\rho(x^{k+1}, y^k) \\ &= \nabla_x f(x^{k+1}) + A^T y^k + \rho(A^T A x^{k+1} - A^T b) \\ &= \nabla_x f(x^{k+1}) + A^T \left(y^k + \rho \left(A x^{k+1} - b \right) \right) \\ &= \nabla_x f(x^{k+1}) + A^T y^{k+1} \end{aligned} \tag{3}$$

Method of Multipliers: dual update step

- With step size ρ, after each iteration of the Method of Multipliers, algorithm generates iterates (x^{k+1}, y^{k+1}) that are *dual feasible*.
- As the algorithm progresses, we hope that the residual $Ax^{k+1} - b \to 0$ to obtain the primal feasibility and hence reach the optimality.

Method of Multipliers: take-home messages

This method is similar to dual decomposition (particular case of dual ascent for which f is separable) but with two differences that induce:

- An advantage: Adding the quadratic term [2] allows the Method of Multipliers to converge under less strong assumptions (it basically works in any cases), for instance f can be non-differentiable, take on value $+\infty$ in the case f is an indicator function for a convex set $\mathcal{C} \subseteq \mathbb{R}^n$.
- A drawback: quadratic penalty term destroys splitting for the update of x (so, cannot do decomposition).

In summary: robust but non-separable.

[2] Recently, other types of augmented terms gained interests such as the Bregman divergences or entropy.

Roadmap

ADMM

- Goal: develop a method
  1. with the good robustness of method of multipliers,
  2. which can support decomposition for allowing distributed optimization
- proposed by Gabay, Mercier, Glowinski, Marrocco in 1976 (however, we have discovered in 80s that **ADMM** could be re-derived from materials from the 50s..., probably in Moscow :)).
  The main point is: this is not new.

ADMM - (Western) Problem form

- Given f and g convex functions, we are interested in solving:

$$\begin{aligned} \min_{x,z} \quad & f(x) + g(z) \\ \text{subject to} \quad & Ax + Bz = c, \end{aligned} \tag{4}$$

where:

1. x and z are two sets of variables (previous variable x splitted into two groups),
2. the objective is required to be separable across x and z,
3. the general equality constraint links x and z.

ADMM - Algorithm

- Build the **augmented Lagrangian**:

$$L_\rho(x, z, y) = f(x) + g(z) + \langle y, Ax + Bz - c \rangle + \frac{\rho}{2}\|Ax + Bz - c\|_2^2$$

- **ADMM**: Iterative scheme, at each iteration $\rightarrow$ three steps:

1. $x^{k+1} := \underset{x}{\text{argmin}} \quad L_\rho(x, z^k, y^k)$ %x-minimization
2. $z^{k+1} := \underset{z}{\text{argmin}} \quad L_\rho(x^{k+1}, z, y^k)$ %z-minimization
3. $y^{k+1} := y^k + \rho(Ax^{k+1} + Bz^{k+1} - c)$ %dual update with step length ρ

ADMM - Remarks

1. If we minimize over x and z jointly, **ADMM** boils down to the **Method of Multipliers**.
2. We get **Decomposition** since we minimize over x with z fixed, and vice versa.

By the way: we are free to minimize over x and z as many times as we want before updating the dual variables.

ADMM - Optimality conditions

- Let us write the first-order optimality conditions for Problem (4) (for f and g differentiable):
  1. $Ax^\star + Bz^\star - c = 0$ (Primal feasibility),
  2. $\nabla_x L(x^\star, z^\star, y^\star) = \nabla_x f(x^\star) + A^T y^\star = 0$ (x-Dual feasibility).
  3. $\nabla_z L(x^\star, z^\star, y^\star) = \nabla_z g(z^\star) + B^T y^\star = 0$ (z-Dual feasibility).
- Step 2 from **ADMM**: z^{k+1} is the minimizer of $L_\rho(x^{k+1}, z, y^k)$, therefore it satisfies the following condition (unconstrained optimization problem):

$$
\begin{aligned}
0 &= \nabla_z L_\rho(x^{k+1}, z^{k+1}, y^k) \\
&= \nabla_z g(z^{k+1}) + B^T y^k + \rho B^T (Ax^{k+1} + Bz^{k+1} - c) \\
&= \nabla_z g(z^{k+1}) + B^T \left(y^k + \rho \left(Ax^{k+1} + Bz^{k+1} - c \right) \right) \\
&= \nabla_x g(z^{k+1}) + B^T y^{k+1}
\end{aligned}
\tag{5}
$$

ADMM - Optimality conditions

- With step size ρ, after each iteration of **ADMM**, algorithm generates iterates $(x^{k+1}, z^{k+1}, y^{k+1})$ that are *z-dual feasible*.
- As the algorithm progresses, we hope that:
  1. the residual $Ax^{k+1} + Bz^{k+1} - c \to 0$ to obtain the primal feasibility,
  2. $(x^{k+1}, z^{k+1}, y^{k+1})$ are *x-dual feasible*.

  and hence reach the optimality.

ADMM - The scaled dual form

It is often easier to express the **ADMM** algorithm in scaled form, where we replace the dual variable y by a scaled variable $u = \frac{y}{\rho}$

- Idea: Combine linear and quadratic terms in augmented Lagrangian:

$$\begin{aligned} L_\rho(x, z, y) &= f(x) + g(z) + \langle y, Ax + Bz - c \rangle + \frac{\rho}{2}\|Ax + Bz - c\|_2^2 \\ &= f(x) + g(z) + \frac{\rho}{2}\|Ax + Bz - c + u\|_2^2 + \text{const} \end{aligned}$$

- In this form, the **ADMM** steps are:

1. $x^{k+1} := \underset{x}{\text{argmin}} \ \left(f(x) + \frac{\rho}{2}\|Ax + Bz^k - c + u^k\|_2^2\right)$ %x-minimization
2. $z^{k+1} := \underset{z}{\text{argmin}} \ \left(g(z) + \frac{\rho}{2}\|Ax^{k+1} + Bz - c + u^k\|_2^2\right)$ %z-minimization
3. $u^{k+1} := u^k + (Ax^{k+1} + Bz^{k+1} - c)$ %dual update

ADMM - The scaled dual form

Note that here the kth iterate u^{k+1} is just given by a running sum of residuals:

$$u^{k+1} = u^0 + \sum_{i=1}^{k+1} \left(Ax^i + Bz^i - c\right)$$

$$= u^0 + A\sum_{i=1}^{k+1} x^i + B\sum_{i=1}^{k+1} z^i - (k+1)c$$

ADMM - Convergence guarantees

- Under (very little !) assumptions :
  - on f and g:
    1. are convex,
    2. are closed: a function $l : \mathbb{R}^n \to \mathbb{R}$ is closed if its epigraph, denoted and defined by epi(f)=$\{(x, t)|f(x) \leqslant t$ for $x \in \mathbb{R}^n, t \in \mathbb{R}\}$, is closed.
    3. are proper: a function l is called proper if l does not take the value $-\infty$ and dom(l) is nonempty.
  - L_0 has a saddle point (roughly speaking; the primal problem needs to have a solution, otherwise there is no point)
    These assumptions do not require A and B to be full rank.
- the **ADMM** iterates satisfy, for any $\rho > 0$:
  1. *Residual* convergence: $r^k = Ax^k + Bz^k - c \to 0$ as $k \to \infty$, i.e., primal iterates approach feasibility.
  2. *Objective* convergence: $f(x^k) + g(z^k) \to f^\star + g^\star$, where $f^\star + g^\star$ is the optimal primal objective value.
  3. *Dual* convergence: $u^k \to u^\star$, where $u^\star$ is a dual solution.

ADMM - Convergence guarantees

- The convergence garantees are **very carefully** worded. Many things are not said because they are FALSE, for instance:
  1. x^k converges $\rightarrow$ FALSE
  2. z^k converges $\rightarrow$ FALSE
  3. (x^k, z^k) converge to the optimal set $\rightarrow$ FALSE
  4. (x^k, z^k) converge to $(x^\star, z^\star)$ which are the optimal primal solutions $\rightarrow$ FALSE

  In summary: we do not generically get primal convergence, but this is true under more assumptions (*Question: is it that relevant ? Because we are loosing the all point of this methods that work "no matter what" + stopping criterion ?*).
- Convergence rate: roughly, **ADMM** behaves like first-order method Theory still being developed, see, e.g., in Hong and Luo (2012), Deng and Yin (2012), Iutzeler et al. (2014), Nishihara et al. (2015)

ADMM - Practicalities and tricks

- In practice, **ADMM** obtains a relatively accurate solution in a handful of iterations, but requires many, many iterations for a highly accurate solution. Hence it behaves more like a first-order method than a second-order method.
- Choice of ρ, can greatly influence practical convergence of **ADMM**:
  - ρ is too large $\rightarrow$ not enough emphasis on minimizing $f + g$
  - ρ is too small $\rightarrow$ not enough emphasis on feasibility

  (Boyd et al, 2010) give a strategy for varying ρ that is useful in practice (but without convergence guarantees).
- Like deriving duals, transforming a problem into one that **ADMM** can handle is sometimes a bit subtle, since different forms can lead to different algorithms.

ADMM - Practicalities and tricks

We called the method **ADMM** but it turns out that **ADMM** is related to and exactly the same as many other methods, let us cite a few:

- Identical to the Douglas-Rachford operators splitting method (by choosing the right operators and split in the right way). (Douglas, Peaceman, Rachford, Lions, Mercier,...,1950s, 1979)
- proximal point algorithm (applied to the right operator to figure out) (Rockafellar 1976)
- Dykstra's alternating projections algorithm (1983),
- Spingarn's method of partial inverses (1985)
- Bregman iterative methods (2008-present) (substitute the quadratic term by Bregman divergence, more general).

Why ? How is it that different well-known algorithms are actually the "same"?: It is complicated enough to transform (reorder, change the variables, select right operator, etc) algorithm A to algorithm B and certify they are in fact the same.

Roadmap

Common patterns

Let us consider the **ADMM** scaled form,

- x-update step from **ADMM** requires minimizing $f(x) + \frac{\rho}{2}\|Ax - v\|_2^2$ with $v = -Bz^k + c - u^k$ is a constant during x-update.[3]
- similar for z-update
- several special cases come up often (particular choice for g and/or for A, B and c)
- can simplify the update by **exploiting** the structure of these special cases

[3] Note that **ADMM** can be seen as a meta-algorithm, we are dealing with more abstract objects here compared to classical optimization algorithms where you end up working with primitives such as gradients or subgradients. Except for special cases, the operators are higher levels, they require to minimize a quadratic augmented function (can be really challenging !).

Decomposition

- Assume that function f is separable:

$$f(x) = f_1(x_1) + ... + f_N(x_N), \quad x := (x_1, ..., x_N)^T$$

  where $1 \leqslant N \leqslant n$.
- Assume that $A^T A$ is block diagonal w.r.t. blocks $x := (x_1, ..., x_N)^T$, (as a simple illustration consider the case $A = I$),
- then the $f(x) + \frac{\rho}{2}\|Ax - v\|_2^2$ splits into N components,
  $\rightarrow$ update of x_i in parallel, for $i = 1, ..., N$.

Quadratic objective

- $f(x) = 1/2\langle x, Px\rangle + \langle q, x\rangle + r$
  $\rightarrow$ x-update requires to minimize a quadratic without constraints, can be done in closed form:x^{k+1} is the solution of
  $\min_x\ l(x) := 1/2\langle x, Px\rangle + \langle q, x\rangle + r + \frac{\rho}{2}\|Ax - v\|_2^2$ where
  $v = -Bz^k + c - u^k$ is a constant. Hence we are looking for x such that:

$$\begin{aligned} 0 &= \nabla l(x) \\ &= \left(P + \rho A^T A\right) x + \left(q - \rho A^T v\right) \end{aligned}$$

  Hence $x^{k+1} := \left(P + \rho A^T A\right)^{-1} \left(\rho A^T v - q\right)$
- The problem: the computation of the inverse !
  - Depending on sparsity patterns, you may use matrix inversion lemma:

$$\left(P + \rho A^T A\right)^{-1} = P^{-1} - \rho P^{-1} A^T (I + \rho A P^{-1} A^T)^{-1} A P^{-1}$$

  - Dense case: (direct method) cache factorization of $\left(P + \rho A^T A\right)$ (LU or Cholesky if symmetric pd).

Connection to proximal operators

- Consider

$$\min_x f(x) + g(x) \iff \min_{x,z} f(x) + g(z) \text{ subject to } x - z = 0$$

  a particular case of Problem (4) with $A = I_{n\times n}$, $B = -I_{n\times n}$ and $c = \bar{0}$
- Build the **augmented Lagrangian** (scaled form):

$$L_\rho(x, z, y) = f(x) + g(z) + \frac{\rho}{2}\|x - z + u\|_2^2$$

- Consider x-update with $v^k = z^k - u^k$:

$$\begin{aligned} x^{k+1} &:= \operatorname*{argmin}_x \left(f(x) + \frac{\rho}{2}\|x - v^k\|_2^2 \right) \\ &:= \mathbf{prox}_{1/\rho,f}(v^k) \end{aligned}$$

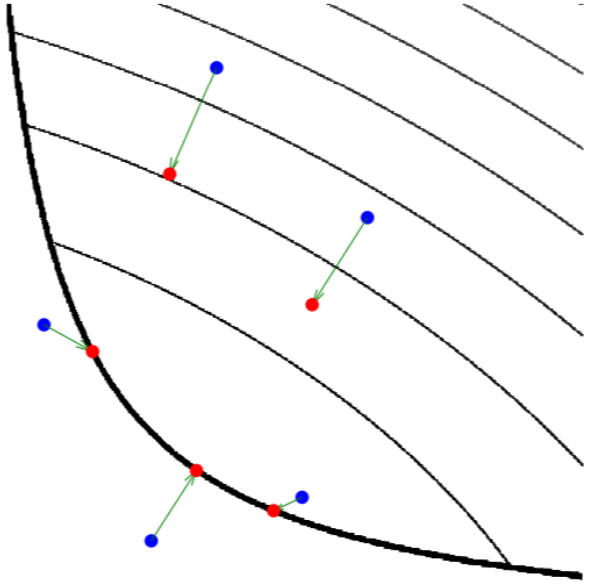
  Where $\mathbf{prox}_{1/\rho,f}(v^k)$ is referred to as the proximal operator $\mathbf{prox}_{1/\rho,f} : \mathbb{R}^n \to \mathbb{R}^n$ of f with parameter $1/\rho$.

Interpretation of proximal operators

What a prox does: $\mathbf{prox}_{\lambda,f}$(blue points) $\rightarrow$ red points

Two cases:

1. three points in the domain of the function stay in the domain and move towards the minimum of the function
2. the other two move to the boundary of the domain and towards the minimum of the function



Note that:

- thin black lines:= level curves of a convex function f
- thicker black line := the boundary of its domain
- λ := trade-off parameter between the two terms.
  What happens if λ is big/small ?

Special cases of proximal operators

When f is the indicator function:

$$I_{\mathcal{C}}(x) = \begin{cases} 0 & \text{if} \quad x \in \mathcal{C}, \\ +\infty & \text{if} \quad x \notin \mathcal{C}, \end{cases}$$

where $\mathcal{C}$ is a closed nonempty convex set. The proximal operator becomes:

$$\begin{aligned} \mathbf{prox}_{\lambda,f}(v) &:= \underset{x}{\operatorname{argmin}} \left(f(x) + \frac{1}{2\lambda}\|x - v\|_2^2 \right) \\ \iff &:= \underset{x \in \mathcal{C}}{\operatorname{argmin}} \left(\underset{=0}{f(x)} + \frac{1}{2\lambda}\|x - v\|_2^2 \right) \\ \iff &= \underset{x \in \mathcal{C}}{\operatorname{argmin}} \left(\|x - v\|_2^2 \right) = \Pi_{\mathcal{C}}(v) \end{aligned}$$

Hence the proximal operator of f reduces to Euclidean projection onto $\mathcal{C}$.
...Proximal operators can thus be viewed as generalized projections !

Special cases of proximal operators

When f is the l-1 norm of x: $f(x) = \|x\|_1 = \sum_i^n |x_i|$ (non-differentiable). The proximal operator becomes:

$$\mathbf{prox}_{\lambda,f}(v) := \underset{x}{\text{argmin}} \left(\|x\|_1 + \frac{1}{2\lambda}\|x - v\|_2^2 \right)$$

Recall $\|x\|_2^2 = \sum_i^n x_i^2$, hence the function to minimize is separable w.r.t. each x_i component. Indeed:

$$\mathbf{prox}_{\lambda,f}(v) := \underset{x}{\text{argmin}} \left(\sum_i^n \left(|x_i| + \frac{1}{2\lambda}(x_i - v_i)^2 \right) \right)$$

x-minimization splits into n independent problems:

$$\left[\mathbf{prox}_{\lambda,f}(v)\right]_i := \underset{x_i \in \mathbb{R}}{\text{argmin}} \quad |x_i| + \frac{1}{2\lambda}(x_i - v_i)^2$$

$$\iff := \underset{x_i \in \mathbb{R}}{\text{argmin}} \quad \lambda|x_i| + \frac{1}{2}(x_i - v_i)^2$$

i.e. given v_i, find the minimizer x_i of the scalar function.

Special cases of proximal operators: l-1 norm

Consider the scalar function:

$$l(x_i) = \lambda|x_i| + \frac{1}{2}(x_i - v_i)^2. \tag{6}$$

It is convex :

1. $|x_i|$ is not differentiable but convex
2. $(x_i - v_i)^2$ is differentiable and convex

The minimizer of f, denoted as $x_i^\star$, can be obtained by considering the first-order optimality condition. As $|x_i|$ is not differentiable, we use the optimality condition for non-smooth functions (no constraints).

Minimum of a non-smooth function

A point $x_i^\star$ is a minimizer of a convex function l if and only if l is subdifferentiable at $x_i^\star$ and

$$0 \in \partial l(x_i^\star). \tag{7}$$

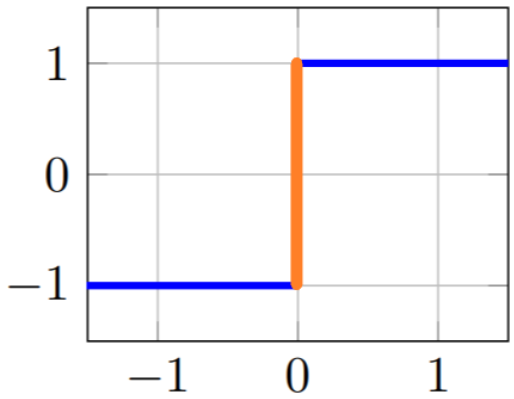
Special cases of proximal operators: l-1 norm

Few subdifferential calculus rules:

1. if $l(x) = l_1(x) + l_2(x)$, then $\partial l(x) = \partial l_1(x) + \partial l_2(x)$
2. if l_2 is differentiable, then $\partial l_2(x) = \{\nabla l_2(x)\}$

First-order optimality condition for $l(x_i)$:

1. The subdifferential of $|x_i|$ is $\text{sgn}(x_i)$ for $x_i \neq 0$ and $[-1, 1]$ for $x_i = 0$:



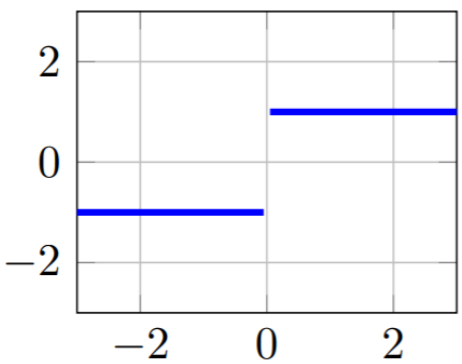
2. The gradient of $\frac{1}{2}(x_i - v_i)^2$ is $(x_i - v_i)$.

Hence (to ease the reading, we consider the case $x_i \neq 0$):

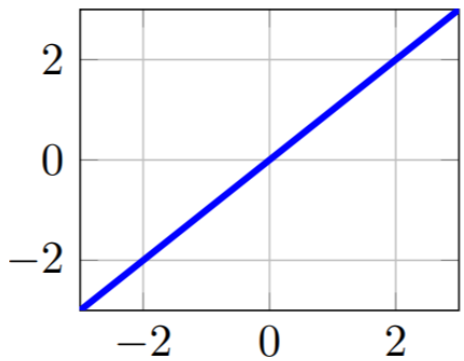
$$0 \in \partial l(x_i^\star) \iff 0 \in \lambda \text{sgn}(x_i^\star) + (x_i^\star - v_i) \iff v_i = x_i^\star + \lambda \text{sgn}(x_i^\star)$$

Special cases of proximal operators: l-1 norm

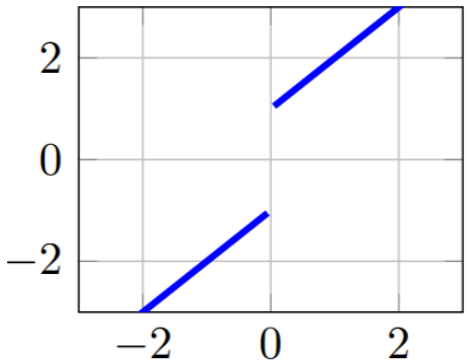
Let us know express $x_i^\star$ as a function of v_i. For ease of notation, we drop the subscript i in the **two** following slides. This can be done by swapping the xv axes of the plot of $v = x + \lambda \text{sgn}(x)$ (or more formally, computing its inverse).
As a simple illustration for the case $\lambda = 1$:



(a) $\text{sgn}(x)$



(b) x



(c) $x + \text{sgn}(x)$

Special cases of proximal operators: l-1 norm

Swapping the axes and including the case $x_i = 0$, we get the soft thresholding operator S:

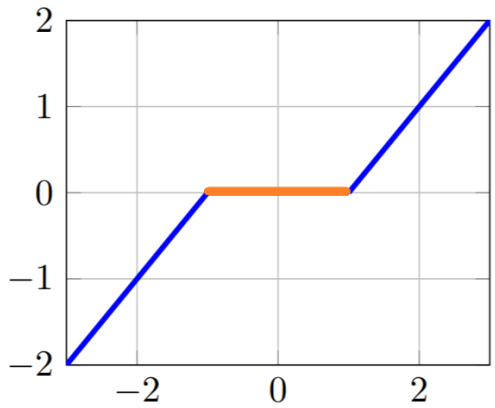


Figure: $S(v) = \text{sgn}(v)\left(|v| - 1\right)_+ = \text{sgn}(v)\text{max}\left(|v| - 1, 0\right)$

In general case with threshold λ, we have:

$$S_\lambda(v) = \text{sgn}(v)\left(|v| - \lambda\right)_+ \quad [4] \tag{8}$$

In MATLAB : sign(v).*(max(abs(v)-lambda,0));

[4] Equivalent formula is $S_\lambda(v) = (v - \lambda)_+ - (-v - \lambda)_+$

Special cases of proximal operators: l-1 norm

Finally, the proximal operator [5]:

$$\mathbf{prox}_{\lambda,f}(v) := \underset{x}{\text{argmin}} \left(\|x\|_1 + \frac{1}{2\lambda}\|x - v\|_2^2 \right)$$

can be computed by the following update rule:

$$\mathbf{prox}_{\lambda,f}(v) = S_\lambda(v)$$

where S_λ is the soft thresholding operator applied on v componentwise:

$$\left[\mathbf{prox}_{\lambda,f}(v)\right]_i = \text{sgn}(v_i)\left(|v_i| - \lambda\right)_+$$

Remark: the proximal operator has been computed in closed form !

[5] **Attention**, we are back to the original notations: $x, v \in \mathbb{R}^n$

Proximal operators and MatLab Toolbox

For further readings,

1. getting access to an open-source MatLab toolbox called Unlocbox dedicated to convex optimization (in particular proximal methods and ADMM), the reader is invited to consult:
   ▸ UNLOCBOX - documentation and ▸ UNLOCBOX - Proximal operators
2. Examples of Proximal operators that can be solved in closed form:
   ▸ Link
3. Python libraries for efficient computation of proximal operators
   ▸ THE PROXIMITY OPERATOR REPOSITORY

Smooth objective

- f is smooth (function that has continuous derivatives up to some desired order over some domain).
- for the x-update, can use standard methods for smooth minimization:
  - gradient-based methods, Newton or quasi-Newton,
  - a very good choice would be **limited-memory BFGS** since it scale to very large problems and the augmentation with a quadratic term puts this method in the best context (rely on smoothness to be efficient).

Constrained convex optimization

- Consider **ADMM** for a generic convex problem:

$$\begin{aligned} \min_{x} \quad & f(x) \\ \text{subject to} \quad & x \in \mathcal{C}, \end{aligned}$$

- **ADMM** form: take g as the indicator function of the set $\mathcal{C}$:

$$\begin{aligned} \min_{x,z} \quad & f(x) + g(z) \\ \text{subject to} \quad & x - z = 0, \end{aligned}$$

- Build the **augmented Lagrangian** (scaled form):

$$L_\rho(x, z, y) = f(x) + g(z) + \frac{\rho}{2}\|x - z + u\|_2^2$$

Constrained convex optimization

By combining the special cases of proximal operators presented previously, we get the **ADMM** steps (equivalent to Douglas-Rachford, here):

1. $x^{k+1} := \mathbf{prox}_{1/\rho,f}(z^k - u^k)$ %x-minimization
2. $z^{k+1} := \Pi_{\mathcal{C}}\left(x^{k+1} + u^k\right)$ %z-minimization
3. $u^{k+1} := u^k + (x^{k+1} - z^{k+1})$ %dual update

Alternating projections, revisited

Consider finding a point x in intersection of (simple) convex sets $\mathcal{C}$ and $\mathcal{D}$.

- Let us formulate this as an optimization problem:

$$\begin{aligned} \min_{x} \quad & 0 \\ \text{subject to} \quad & x \in \mathcal{C}, x \in \mathcal{D} \end{aligned}$$

- **ADMM** form: take f, g as the indicators function of the set $\mathcal{C}$ and $\mathcal{D}$ resp.:

$$\begin{aligned} \min_{x,z} \quad & f(x) + g(z) \\ \text{subject to} \quad & x - z = 0, \end{aligned}$$

Alternating projections, revisited

By combining the special cases of proximal operators presented previously, we get the **ADMM** steps:

1. $x^{k+1} := \Pi_{\mathcal{C}}(z^k - u^k)$ %x-minimization
2. $z^{k+1} := \Pi_{\mathcal{D}}\left(x^{k+1} + u^k\right)$ %z-minimization
3. $u^{k+1} := u^k + (x^{k+1} - z^{k+1})$ %dual update

This is like the classical alternating projections method, but now with a dual variable u (much more efficient).

Least Absolute Shrinkage and Selection Operator (Lasso)

- Given $y \in \mathbb{R}^n$, $X \in \mathbb{R}^{n \times p}$, we are interested in solving:

$$\min_{\beta \in \mathbb{R}^p} \quad \frac{1}{2}\|y - X\beta\|_2^2 + \lambda \|\beta\|_1 \tag{9}$$

  where $\lambda \geqslant 0$ is a given regularization parameter and $\|\beta\|_1 = \sum_i^n |x_i|_1$ is the L_1-norm of β.
- This problem is referred to as the **Lasso** problem.
- **ADMM** form:

$$\begin{aligned} \min_{\beta, \alpha \in \mathbb{R}^p} \quad & \frac{1}{2}\|y - X\beta\|_2^2 + \lambda \|\alpha\|_1 \\ \text{subject to} \quad & \beta - \alpha = 0, \end{aligned}$$

Lasso

- **ADMM** steps:

1. $\beta^{k+1} := \underset{\beta}{\text{argmin}} \ \left(\frac{1}{2}\|y - X\beta\|_2^2 + \frac{\rho}{2}\|\beta - \alpha^k + u^k\|_2^2\right)$ %β-minimization
2. $\alpha^{k+1} := \underset{\alpha}{\text{argmin}} \ \left(\lambda\|\alpha\|_1 + \frac{\rho}{2}\|\beta^{k+1} - \alpha + u^k\|_2^2\right)$ %α-minimization
3. $u^{k+1} := u^k + (\beta^{k+1} - \alpha^{k+1})$ %dual update

Lasso: Steps 1 and 2

Step 1

$f(x) = \frac{1}{2}\|y - X\beta\|_2^2$ is a quadratic, we obtain

$$\beta^{k+1} := \left(X^T X + \rho I\right)^{-1}\left(\rho v^k + X^T y\right)$$

where $v^k = \alpha^k - u^k$ and $X^T X + \rho I$ is always invertible, regardless of X. If we compute a factorization (say Cholesky) in $O(p^3)$ flops, then each β update takes $O(p^2)$ flops (complexity of forward/backward substitution).

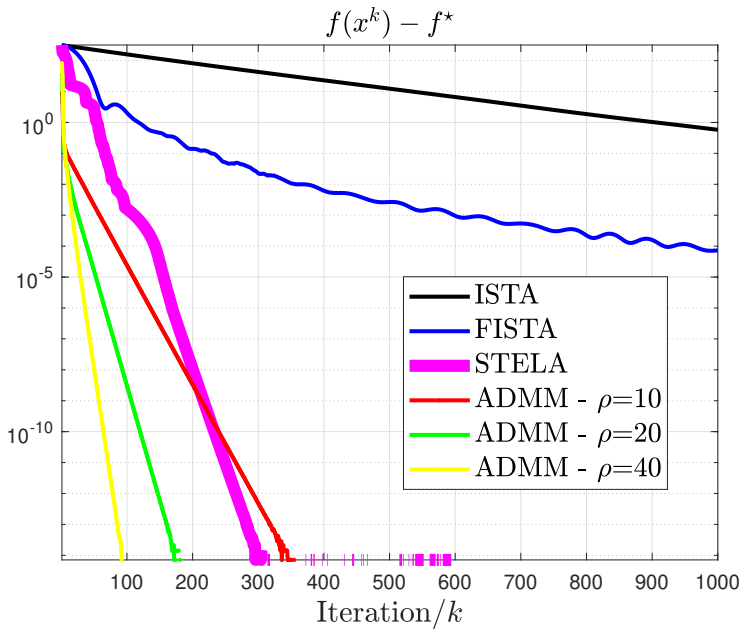
Step 2

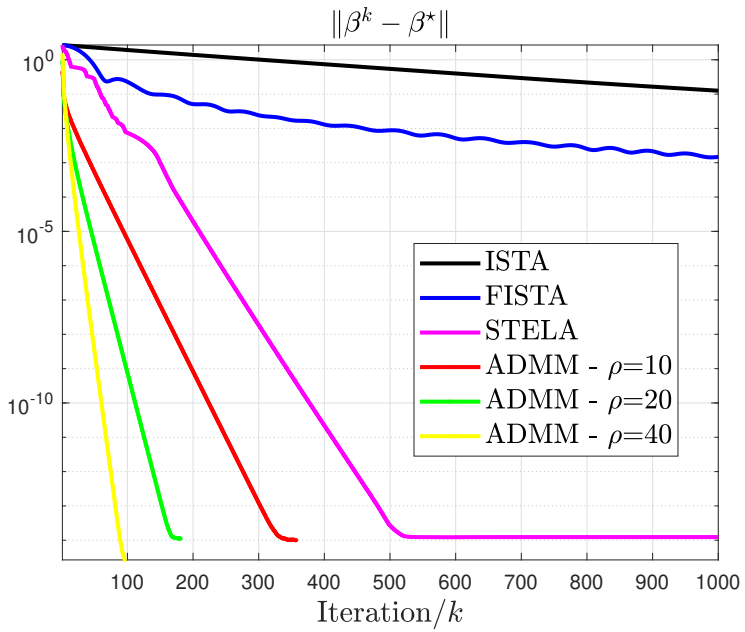
The z-update corresponds to the proximal operator associated to the l-1 norm of parameter $\frac{\lambda}{\rho}$, therefore:

$$\begin{aligned}\alpha^{k+1} &:= \mathbf{prox}_{\lambda/\rho,\|.\|_1}(\beta^{k+1} + u^k) \\ &= S_{\lambda/\rho}(\beta^{k+1} + u^k)\end{aligned}$$

Lasso: numerical experiments

Comparison of various algorithms: instances with $n = 1000$, $p = 100$

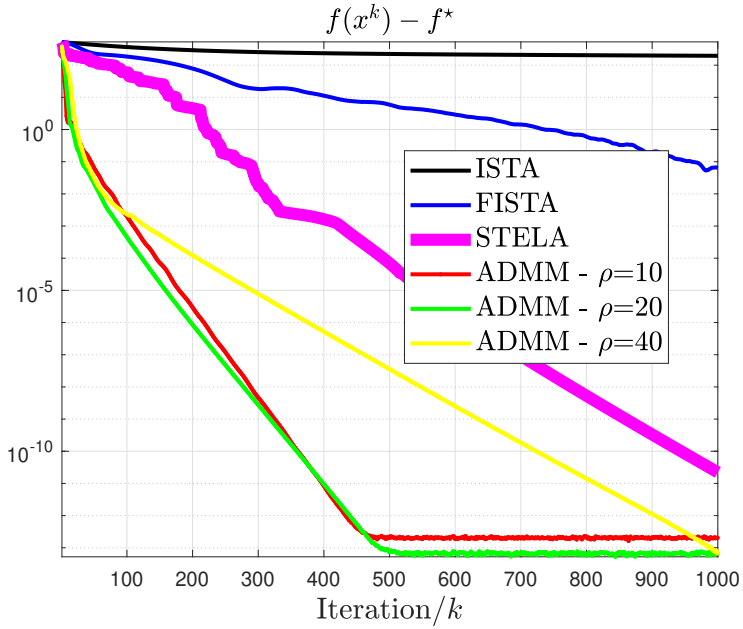


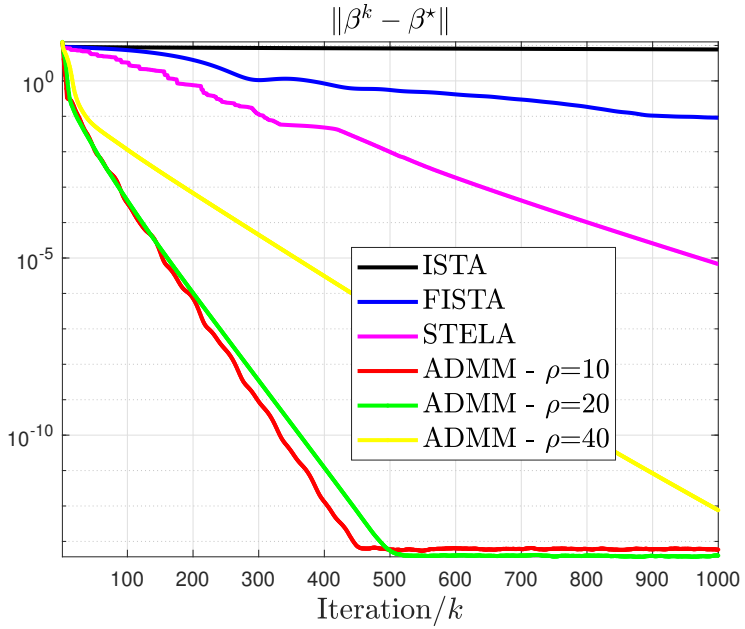


▸ MatLab Code

Lasso: numerical experiments

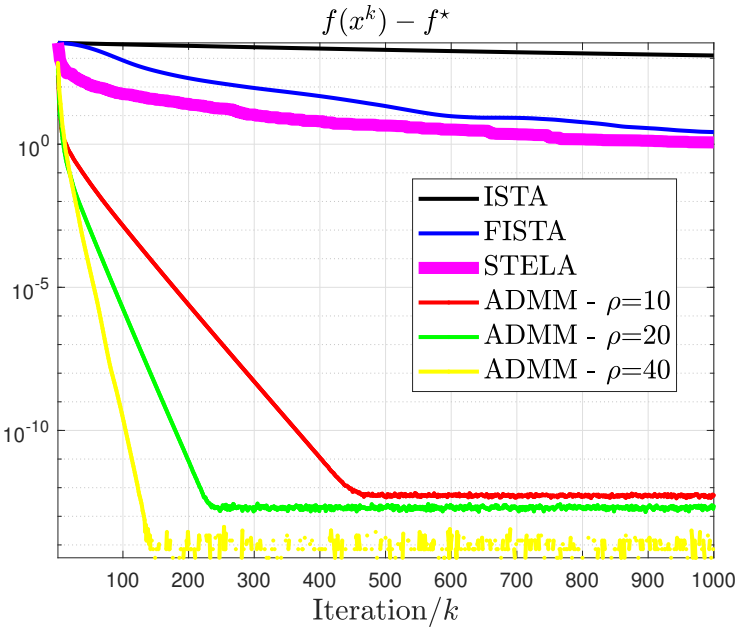
Comparison of various algorithms: instances with $n = 100$, $p = 1000$

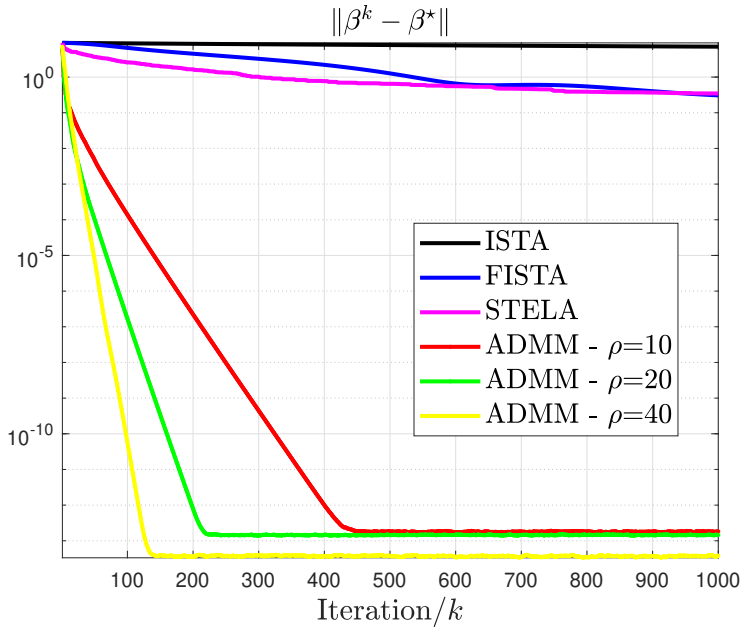




Lasso: numerical experiments

Comparison of various algorithms: instances with $n = 1000$, $p = 1000$





Remark: **STELA** algorithm (Yang and Pesavento, 2017) seems more sensible to initialization than other methods. Therefore random and various initializations should be done to compare more accurately and fairly the algorithms.

Lasso: numerical experiments

Remarks:

- Other methods exist for solving Lasso problem such as Coordinate descent-based methods that showed high effectiveness (for small size problems), see (Tibshirani, 2015) - page 11 for a numerical comparison.
- The proper choice of λ is crucial for good performance of this algorithm, but this is not an easy task. Unfortunately we are not in the place here to give you a rule of thumb what to do, since it highly depends on the application at hand. Again, consult (Beck et al 2009) for any further considerations of this matter.

Generalized lasso, revisited

- Given the usual $y \in \mathbb{R}^n$, $X \in \mathbb{R}^{n \times p}$ and an additional $D \in \mathbb{R}^{m \times p}$, the *generalized lasso* problem solves:

$$\min_{\beta \in \mathbb{R}^p} \quad \frac{1}{2}\|y - X\beta\|_2^2 + \lambda \|D\beta\|_1 \tag{10}$$

- The *generalized lasso* is computationally harder than the lasso ($D = I$).
  ADMM form:

$$\begin{aligned} \min_{\beta \in \mathbb{R}^p, \alpha \in \mathbb{R}^m} \quad & \frac{1}{2}\|y - X\beta\|_2^2 + \lambda \|\alpha\|_1 \\ \text{subject to} \quad & D\beta - \alpha = 0, \end{aligned}$$

Generalized lasso, revisited

- However, **ADMM** delivers a simple algorithm:

1. $\beta^{k+1} := \left(X^T X + \rho D^T D\right)^{-1} \left(\rho D^T v^k + X^T y\right)$ %β-minimization
2. $\alpha^{k+1} := S_{\lambda/\rho}(D\beta^{k+1} + u^k)$ %α-minimization
3. $u^{k+1} := u^k + (D\beta^{k+1} - \alpha^{k+1})$ %dual update

Intermediate steps let as an exercise !

Roadmap

1. Assumptions
2. Dual decomposition
3. Method of Multipliers
4. Alternating Direction Method of Multipliers
5. Commom patterns
6. Examples
   - Matrix Decompositions

Low rank approximation of a matrix

Disclaimer: before we present examples of LOw-RAnk Matrix APproximations (LORAMAP [6]) solved by using **ADMM**:

- we recall fundamental principles and tools from linear algebra as the notions of the rank and the Singular Value Decomposition (SVD) of a matrix $A \in \mathbb{R}^{m \times n}$,
- we recall that SVD can be used to find to best rank-r approximation of a matrix, w.r.t. any unitarily invariant norm (and without any additional constraints),
- we briefly introduce the Matrix Rank Minimization problems and the general Low Rank Regularization framework,
- we present the link between the nuclear norm and the rank of a matrix, that is nuclear norm is the convex envelope (tightest convex relaxation) of the rank (within the unit ball).

[6] Partial reproduction of the name of Nicolas Gillis ▸ ERC

LORAMAP: basics from linear algebra

- Rank and basic properties:

Rank definition

For field $\mathbb{F}$, let $A \in \mathbb{F}^{m \times n}$. Then

$$\text{rank}(A) := \dim(\text{range}(A))$$

where range(A) denotes the linear space spanned by the columns of A.
For simplicity, $\mathbb{F} = \mathbb{R}$ throughout the lecture and often $m \geqslant n$.
Lemma:

- $\text{rank}(A) = \text{rank}(A^T)$ (the dimension of the column space of the matrix is equal to the dimension of its row space)
- $\text{rank}(PAQ) = \text{rank}(A)$ for any invertible matrices $P \in \mathbb{R}^{m \times m}$ and $Q \in \mathbb{R}^{n \times n}$.
- $\text{rank}\left(\begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}\right) = \text{rank}(A_{11}) + \text{rank}(A_{22})$.

LORAMAP: basics from linear algebra

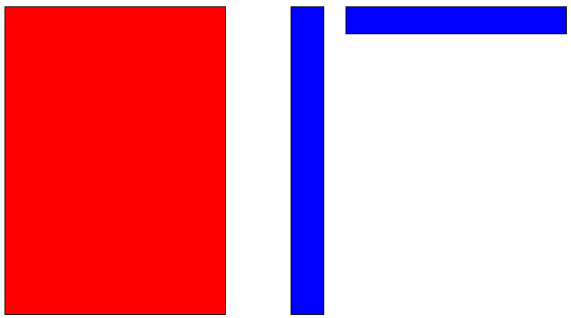
- Rank and matrix factorizations:

Lemma

A matrix $A \in \mathbb{R}^{m \times n}$ of rank r admits a factorization of the form:

$$A = BC, B \in \mathbb{R}^{m \times r}, C \in \mathbb{R}^{r \times n}.$$

We say that A has low rank if $\text{rank}(A) \ll \min(m, n)$.
Illustration of low-rank factorization:



	A	BC
#entries	mn	$mr + nr$

1. Generically (and in most applications), A has full rank.
2. Aim instead at *approximating* A by a low-rank matrix.

LORAMAP: SVD

- Fundamental tool: The Singular Value Decomposition (SVD)

Theorem

Let $A \in \mathbb{R}^{m \times n}$ with $m \geqslant n$, then there are orthogonal matrices $U \in \mathbb{R}^{m \times m}$ and $V \in \mathbb{R}^{n \times n}$ such that:

$$A = U \Sigma V^T \text{ with } \Sigma = \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \\ & 0_{(m-n) \times n} & \end{bmatrix} \in \mathbb{R}^{m \times n}$$

and $\sigma_1 \geqslant \sigma_2 \geqslant \ldots \geqslant \sigma_n \geqslant 0$.

where:

- $\sigma_1, \ldots, \sigma_n$ are called the singular values,
- $u_1, \ldots, u_n$ are called the left singular vectors,
- $v_1, \ldots, v_n$ are called the right singular vectors,
- $Av_i = \sigma_i u_i$, $A^T u_i = \sigma_i v_i$, $i = 1, \ldots, n$.

LORAMAP: SVD

Very basic properties of SVD

- $r = \text{rank}(A)$ is number of nonzero singular values of A, i.e., the rank of a matrix A is the l-0 norm of the vector composed of the singular values of A. (note that l-0 norm is actually not a norm since $\|\alpha x\| = |\alpha|\|x\|$ does not hold for all α.)
- $\text{kernel}(A) = \text{span}^7(v_{r+1}, ..., v_n)$,
- $\text{range}(A) = \text{span}(u_1, ..., u_r)$,

[7] the set of all finite linear combinations

LORAMAP: SVD

Norms: Spectral and Frobenius norm

- Given SVD $A = U\Sigma V^T$, one defines:
  - Spectral norm: $\|A\|_2 = \sigma_1$,
  - Frobenius norm:

$$\|A\|_F = \sqrt{\langle A, A \rangle}$$
$$= \sqrt{\mathrm{Tr}(A^T A)} = \sqrt{\sum_{ij} |A_{ij}|^2} = \sqrt{\sigma_1^2 + ... + \sigma_n^2}$$

- Basic properties:
  - $\|A\|_2 = \max(\|Av\|_2 | \|v\|_2 = 1)$,
  - $\|.\|_2$ and $\|.\|_F$ are both unitarily invariant, that is

$$\|QAP\|_2 = \|A\|_2, \ \|QAP\|_F = \|A\|_F \tag{11}$$

for any orthogonal matrices Q and P.

LORAMAP: best rank-r approximation

Now, let us introduce the optimal low-rank approximation problem in $\mathbb{R}^{m\times n}$, which is formulated as follows:

Problem 1

Let $A \in \mathbb{R}^{m\times n}$ and $1 \leqslant r \leqslant \min(m,n)$, find $X^* \in \mathbb{R}^{m\times n}$ with $\text{rank}(X^*) \leqslant r$ such that

$$\min_{\text{rank}(X)\leqslant r} \|A - X\| = \|A - X^*\|$$

for some given operator norm $\|.\|$.

LORAMAP: best rank-r approximation

The problem has been solved by Schmidt and was generalized by Mirsky to unitarily invariant norms:

Proposition 1

Given $A \in \mathbb{R}^{m \times n}$ and $1 \leqslant r \leqslant \min(m, n)$, then

$$\min_{\text{rank}(X) \leqslant r} \|A - X\| = \|\text{diag}(\sigma_{r+1}(A), ..., \sigma_{\min(m,n)}(A))\|$$

for any unitarily invariant norm $\|.\|$.

Hence, if an SVD of A is given by $A = \sum_i^{\min(m,n)} \sigma_i u_i v_i^T$, then an optimal solution is $X^* = \sum_i^r \sigma_i u_i v_i^T$. This solution may not be unique if the norm does not depend on all singular values or if $\sigma_r(A) = \sigma_{r+1}(A)$. Nevertheless, if the chosen norm is the Frobenius-norm and $\sigma_r(A) \neq \sigma_{r+1}(A)$, then there is a unique solution.

LORAMAP: best rank-r approximation, application

Image Processing: A is a gray-scale image, we display rank-r approximations for $r = 2, ..., 25$.

LORAMAP: best rank-r approximation, application

Remark: Rank-20 approximation already gives a good approximation of the original image. It implies that

1. The main information is " contained" within the first singular values,
2. a low-rank approximation makes sense, for compression purpose for instance.

Matrix Rank Minimization problems (RMP)

- Consider:

$$\min_{X \in \mathbb{R}^{m \times n}} \quad \text{rank}(X)$$

- As explained previously: the rank has various definition, one of them is the l-0 norm of the singular values of X: let σ be the vector holding the singular values of X, then: $\text{rank}(X) = \|\sigma\|_0$,
- Mainly due to the combinatorial nature of the l-0 norm[8]: the problem RMP is NP-hard. However, it has a trivial solution: X equal to zero matrix.
- **Important**: the Problem is purely *introductory*, indeed the optimal solution is the trivial solution (zero matrix) since there is no constraint.

[8] l-0 norm minimization problem are well known to be NP-hard. As RMP is at least as hard as l-0-norm minimization problem, so RMP is (at least) NP-hard.

General framework: Low Rank Regularization

In the last decade, several attempts have been made to unify these LORAMAP's problems within a unified framework. Let us mention here the Low Rank Regularization framework which considers the following general formulation:

- Consider:

$$\begin{aligned} \min_{X \in \mathbb{R}^{m \times n}} \quad & \operatorname{rank}(X) + \lambda L(X) \\ \text{subject to} \quad & X \in \mathcal{C}, \end{aligned} \tag{12}$$

  where:

  1. $L(X)$ represents some general regularization on X,
  2. C represents the constraints over X,
  3. Still NP-hard problem.

In the following slides, we present some tools to replace the rank function by some easier object to deal with.

Rank and nuclear norm of a matrix

- Rank function: $\text{rank}(A) : \mathbb{R}^{m \times n} \to \mathbb{N}$, can be defined as the l-0 norm of the vector of the singlar values of A.
- Nuclear norm (or less standard the trace norm):

$$\|A\|_* := \sum_{i}^{\min(m,n)} \sigma_i(A),$$

the sum of the singular values of A. Since the singular values are nonnegative by definition, the nuclear norm is equivalent to the l-1 norm of the vector of the singular values of A.

Rank and nuclear norm of a matrix

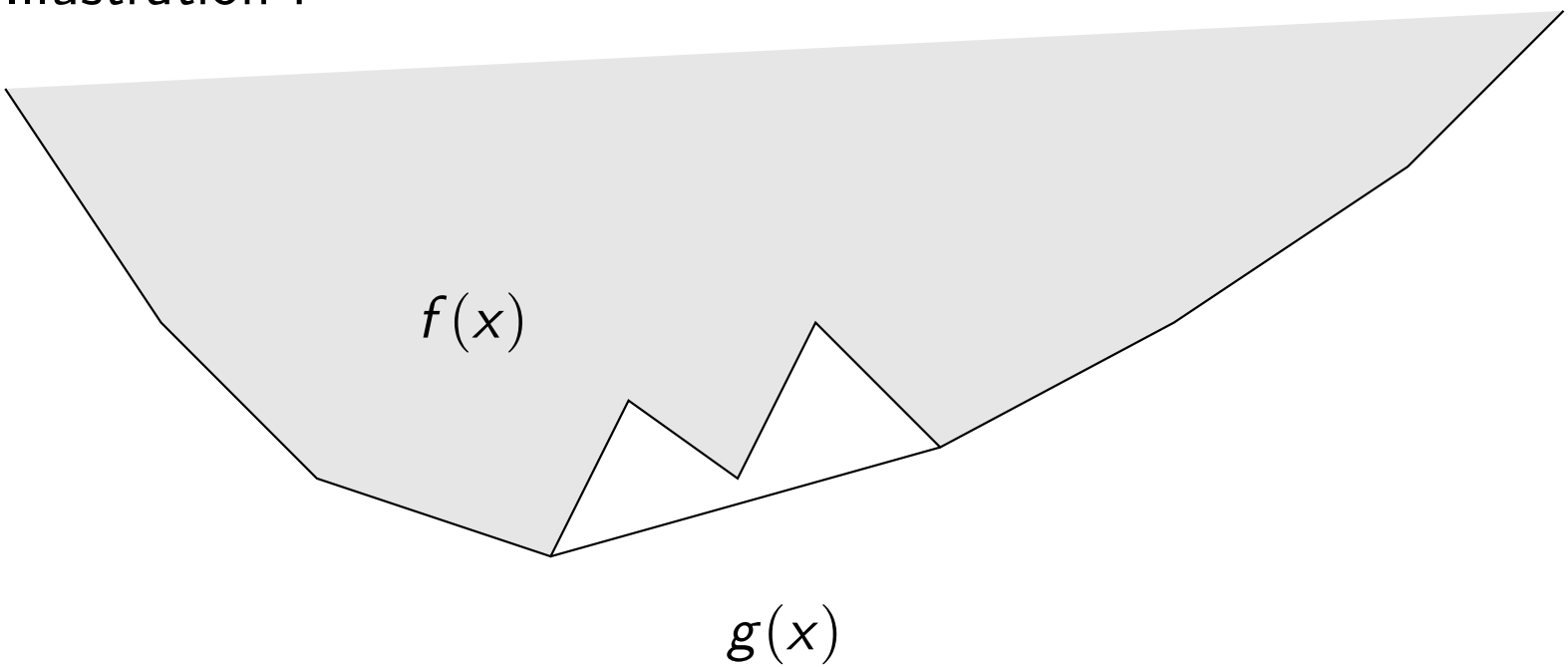
The concept of convex envelope:

Given a function $f : \mathbb{E} \to \mathbb{R}$, a function g is called the convex envelope of f if and only if g is convex and

$$g(x) \leqslant f(x), \ \forall x \in \mathbb{E}$$

i.e., g is a convex lower bound on our original function.

Illustration :



Rank and nuclear norm of a matrix

Theorem (Fazel 2002)

For $f(X) = \text{rank}(X)$ over the set

$$\mathcal{S} := \{X \in \mathbb{R}^{m \times n} | \|X\|_2 \leqslant 1\}$$

the convex envelope of f is the nuclear norm $\|X\|_*$.

- The theorem only applies to those X inside the unit ball. It tells nothing if X is outside the unit ball. In fact, if X is outside the unit ball, the output of the convex envelope will be at ∞.
- If we have $\mathcal{S}' := \{X \in \mathbb{R}^{m \times n} \|X\|_2 \leqslant M\}$, we can do a scaling to reuse the theorem : in this case the convex envelope of f on $\mathcal{S}'$ will be $1/M\|X\|_*$ for $M > 0$.
- The tool to prove the theorem is basically the convex conjugate, more particularly show that the biconjugate of the rank, that is $(\text{rank}(X))^{**}$ is the nuclear norm of X.

Rank and nuclear norm of a matrix

As a simple and intuitive illustration: consider $\mathbb{E} = \mathbb{R}^2$ and try to find the convex envelope of the following subset:

$$\mathcal{C} = \{x \in \mathbb{E} | \|x\|_0 \leqslant 1, \|x\|_2 \leqslant 1\}$$

Hint: Draw C and find the convex hull (the set of all convex combinations of every pair of points) of $\mathcal{C}$, corresponds exactly to the l-1 ball and is minimal by definition of convex hull.
Note that if we allow k non zero elements, this illustration is not valid anymore.
Interest for the following: replace the rank by the its convex relaxation (or convex envelope); the nuclear norm.

Sparse low rank decomposition: Context

- Motivation: Given a large data matrix $M \in \mathbb{R}^{m \times n}$, and know that it may be decomposed as (Candes 2011):

$$M = L_0 + S_0$$

  where L_0 has low rank and S_0 is sparse; here, both components are of arbitrary magnitude.
- We do not know:
  1. low-dimensional column and row space of L_0, neither their dimension,
  2. the locations of the nonzero entries of S_0, not even how many there are.
- Question: Can we hope to recover the low-rank and sparse components both accurately (perhaps even exactly) and efficiently? (Candes 2011)

Sparse low rank decomposition: Possible formulation

Different formulations:

1. **Classical Principal Component Analysis (PCA)**:

$$\min_{\text{rank}(L)\leqslant r} \|M - L\|$$

Advantage: can be efficiently solved via the singular value decomposition (SVD) and enjoys a number of optimality properties when the noise N_0 is small and i.i.d. Gaussian.
Drawback: brittleness with respect to grossly corrupted observations often puts its validity in jeopardy [9].
Gross errors are now ubiquitous in modern applications such as image processing, web data analysis, and bioinformatics, where some measurements may be arbitrarily corrupted

[9] a single grossly corrupted entry in M could render the estimated arbitrarily far from the true L_0.

Sparse low rank decomposition: Possible formulation

Different formulations (next):

2. (one possible) **Robust PCA**:

$$\begin{aligned} \min_{L,S \in \mathbb{R}^{m \times n}} \quad & \text{rank}(L) + \lambda \|S\|_0 \\ \text{subject to} \quad & L + S = M, \end{aligned} \tag{13}$$

Advantage: the entries in S_0 can have arbitrarily large magnitude, and their support is assumed to be sparse but unknown and motivated by a different set of applications: Video Surveillance, Face Recognition, etc.

Drawback: NP-hard and (almost) numerically intractable.

Sparse low rank decomposition: (Candes et al 2011) formulation

(Candes et al 2011) showed that the problem can be solved by tractable convex optimization:

$$\begin{aligned} \min_{L,S\in\mathbb{R}^{m\times n}} \quad & \|L\|_* + \lambda\|S\|_1 \\ \text{subject to} \quad & L + S = M, \end{aligned} \tag{14}$$

Main relaxation tool: l-0 norm replaced by l-1 norm

- Under weak assumptions[10], (14) exactly recovers the low-rank L_0 and the sparse S_0.
- It has been showed in (Chandrasekaran et al 2011) and (Wright et al 2009) that (14) and (13) are equivalent with high probability.

[10] provided that L_0 is sufficiently low-rank and S_0 is sufficiently sparse comparing to the matrix size

Sparse low rank decomposition: ADMM

Note that (14) is already in **ADMM** form.

- Build the **augmented Lagrangian** (scaled form):

$$L_\rho(L, S, U) = \|L\|_* + \lambda\|S\|_1 + \frac{\rho}{2}\|L + S - M + U\|_F^2$$

- **ADMM** steps:

1. $L^{k+1} := \underset{L}{\text{argmin}} \ \left(\|L\|_* + \frac{\rho}{2}\|L + S^k - M + U^k\|_F^2\right)$ %L-minimization
2. $S^{k+1} := \underset{z}{\text{argmin}} \ \left(\lambda\|S\|_1 + \frac{\rho}{2}\|L^{k+1} + S - M + U^k\|_F^2\right)$ %S-minimization
3. $U^{k+1} := U^k + (L^{k+1} + S^{k+1} - M)$ %dual update

Sparse low rank decomposition: ADMM

Remarks

1. Step 1: the L-update corresponds to the proximal operator associated to the nuclear norm of parameter $\frac{1}{\rho}$ that will be derived in the next slides,
2. Step 2: the S-update corresponds to the proximal operator associated to the l-1 norm of parameter $\frac{\lambda}{\rho}$ derived previously.

Sparse low rank decomposition: Singular Value Thresholding operator

In these slides we give the closed-form expression for the proximal operator associated to the nuclear norm of parameter $\frac{1}{\rho}$:

$$\begin{aligned} \mathbf{prox}_{1/\rho,\|.\|_*}(W) &:= \underset{L}{\text{argmin}} \ \left(\|L\|_* + \frac{\rho}{2}\|L - W\|_F^2 \right) \\ &:= \text{SVT}_{1/\rho}(W) = U[\Sigma - \frac{I}{\rho}]_+ V^T \end{aligned} \tag{15}$$

where I is the identity matrix of appropriate size, $W = -S + M - U$, $U\Sigma V^T = W$ (SVD of W) and $[.]_+ = \max(., 0)$.
The operator $\text{SVT}_{1/\rho}$ is called the Singular Value Thresholding operator (SVT).

Sparse low rank decomposition: SVT operator

What does SVT ?

1. Perform SVD on W and get $U\Sigma V^T$
2. Subtract all the diagonal value of Σ by $1/\rho$, denoted $\Sigma - \frac{I}{\rho}$
3. Replace negative value in $\Sigma - \frac{I}{\rho}$ by zero, denoted $[\Sigma - \frac{I}{\rho}]_+$
4. Compute $U[\Sigma - \frac{I}{\rho}]_+ V^T$

Showing that the $\mathbf{prox}_{1/\rho,\|.\|_*}$ is equal to the $\text{SVT}_{1/\rho}$ operator can be done by using sub-differential theory. Here we present a shorter proof based on the von Neumann trace inequality:

$$\text{Tr}(X^T Y) \leqslant \sum_i \sigma_i(X)\sigma_i(Y) \tag{16}$$

The equality holds X and Y share the same left and right singular vectors.

Sparse low rank decomposition: SVT operator

First we have:

$$\begin{aligned}\frac{1}{2}\|L-W\|_F^2 &= \frac{1}{2}\|L\|_F^2 - \mathrm{Tr}(L^T W) + \frac{1}{2}\|W\|_F^2 \\ &\geqslant \frac{1}{2}\left(\sum_i \sigma_i^2(L) - 2\sigma_i(L)\sigma_i(W) + \sum_i \sigma_i^2(W)\right) \\ &= \frac{1}{2}\sum_i (\sigma_i(L) - \sigma_i(W))^2\end{aligned} \tag{17}$$

Sparse low rank decomposition: SVT operator

Assume that L and W share the same left and right singular vectors, let us inject (17) in (15):

$$\begin{aligned}\mathbf{prox}_{1/\rho,\|.\|_*}(W) &:= \underset{L}{\text{argmin}} \left(\|L\|_* + \frac{\rho}{2}\|L-W\|_F^2\right) \\ &:= \underset{L}{\text{argmin}} \left(\frac{1}{\rho}\sum_i \sigma_i(L) + \frac{1}{2}\sum_i (\sigma_i(L)-\sigma_i(W))^2\right) \\ &= \underset{L}{\text{argmin}} \sum_i \left(\frac{1}{\rho}\sigma_i(L) + \frac{1}{2}(\sigma_i(L)-\sigma_i(W))^2\right)\end{aligned} \tag{18}$$

Equation (18) implies that:

$$\sigma_i\left(\mathbf{prox}_{1/\rho,\|.\|_*}(W)\right) := \underset{\sigma_i(L)\geqslant 0}{\text{argmin}} \left(\frac{1}{\rho}\sigma_i(L) + \frac{1}{2}(\sigma_i(L)-\sigma_i(W))^2\right) \tag{19}$$

which can solved in parallel and in closed form by using the soft thresholding operator.

Sparse low rank decomposition: SVT operator

Equation (19) finally becomes:

$$\begin{aligned}\sigma_i\left(\mathbf{prox}_{1/\rho,\|.\|_*}(W)\right) &:= S_{1/\rho}\left(\sigma_i\left(W\right)\right)\\ &= \mathrm{sgn}\left(\sigma_i\left(W\right)\right)\left(\left|\sigma_i\left(W\right)\right| - \frac{1}{\rho}\right)_+\\ &= \left(\sigma_i\left(W\right) - \frac{1}{\rho}\right)_+\end{aligned} \tag{20}$$

since $\sigma_i \geqslant 0$ by definition.
Finally, since L and W share the same left and right singular vectors, $\mathbf{prox}_{1/\rho,\|.\|_*}(W)$ also does, therefore:

$$\begin{aligned}\mathbf{prox}_{1/\rho,\|.\|_*}(W) &:= U[\Sigma - \frac{I}{\rho}]_+ V^T\\ &= \mathrm{SVT}_{1/\rho}(W)\end{aligned} \tag{21}$$

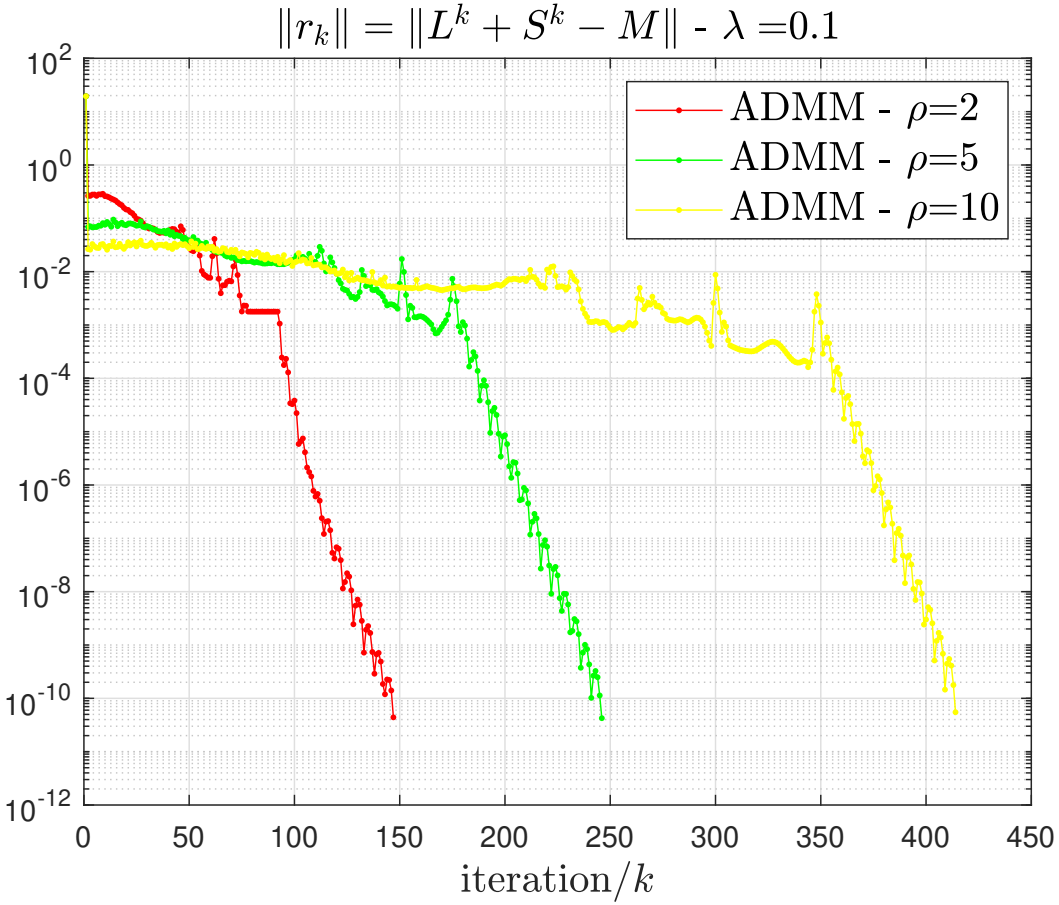
Sparse low rank decomposition: ADMM

- **ADMM** steps are:

1. $L^{k+1} := \text{SVT}_{1/\rho}(-S^k + M - U^k)$ %L-minimization
2. $S^{k+1} := S_{\lambda/\rho}(-L^{k+1} + M - U^k)$ %S-minimization
3. $U^{k+1} := U^k + (L^{k+1} + S^{k+1} - M)$ %dual update

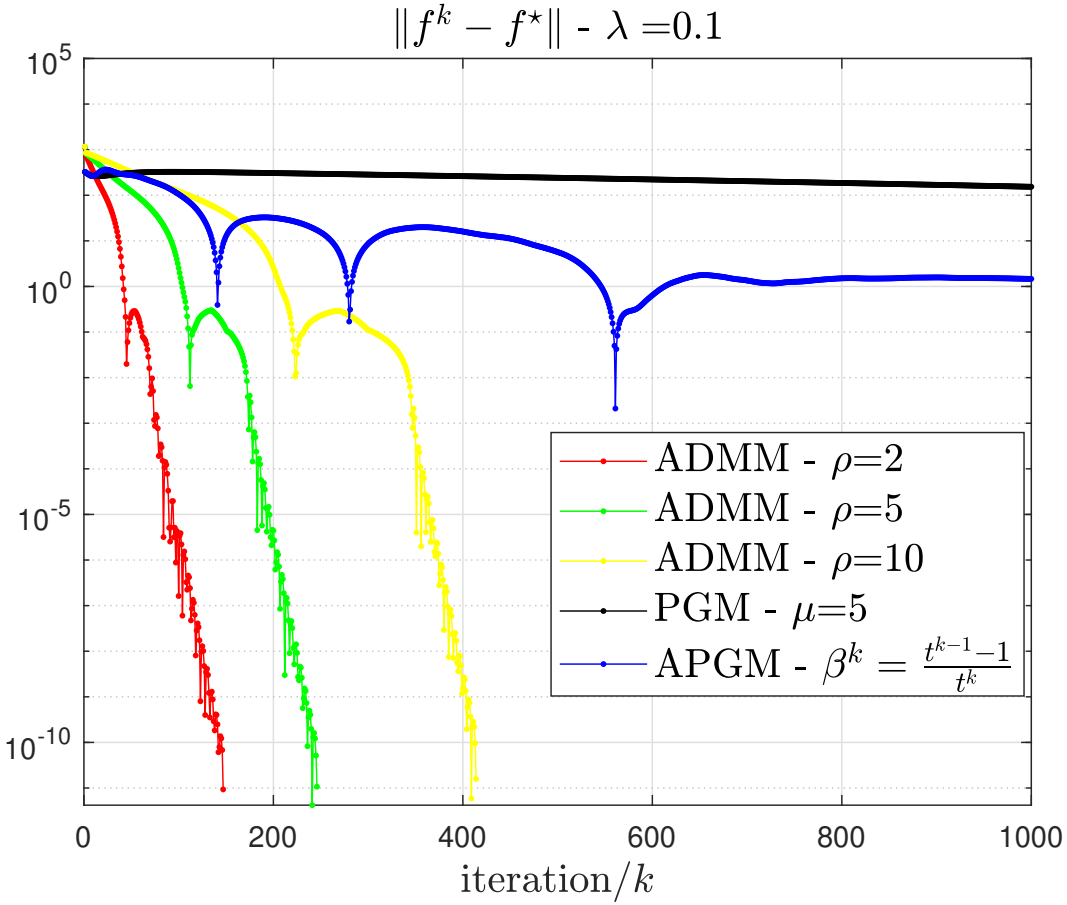
Sparse low rank decomposition: numerical experiments

Comparison of various algorithms: instance with $n = 100$, $m = 100$, $\text{rank}(L0) = 5$, $\lambda = 1/\sqrt{m}$:



▸ MatLab Code

Sparse low rank decomposition: numerical experiments



By choosing (very carefully) the **ADMM** parameter ρ and the penalty weight λ as suggested by (Candes et al 2011), **ADMM** algorithm is able to recover L_0 and S_0 with high accuracy (as the *rank*). **APGM** gives good results as well.

Sparse low rank decomposition: Example from (Candes et al 2011)

(a) Original frames (b) Low-rank $\hat{L}$ (c) Sparse $\hat{S}$

Sparse low rank decomposition: Example from (Candes et al 2011)

Remarks:

- Given a sequence of surveillance video frames, we often need to identify activities that stand out from the background.
- we stack the video frames as columns of a matrix M,
- the low-rank component L_0 naturally corresponds to the stationary background,
- the sparse component S_0 captures the moving objects in the foreground.

Nonnegative low rank approximation

Given $M \in \mathbb{R}_+^{m \times n}$ and let $1 \leqslant r \leqslant \min(m, n)$, we are interested in solving:

$$\begin{aligned} \min_{X \in \mathbb{R}^{m \times n}} \quad & \|M - X\|_F^2 \\ \text{subject to} \quad & \operatorname{rank}(X) \leqslant r, X \in \mathcal{C}, \end{aligned} \tag{22}$$

where C denotes the nonnegative orthant of dimension $m \times n$. In other words, we are looking for the optimal low-rank Frobenius-norm approximations of M (matrices with non-negative entries).
Problem (22) is non-convex. In order to provide a solution, we consider the following convex relaxation:

$$\begin{aligned} \min_{X \in \mathbb{R}^{m \times n}} \quad & \|M - X\|_F^2 + \lambda \|X\|_* \\ \text{subject to} \quad & X \in \mathcal{C}, \end{aligned} \tag{23}$$

Question: How to solve Problem (23) ?

Nonnegative low rank approximation: solution

- **ADMM** form

$$\begin{aligned} \min_{X \in \mathbb{R}^{m \times n}} \quad & \|M - X\|_F^2 + \lambda \|Z\|_* + g(X) \\ \text{subject to} \quad & X - Z = 0, \end{aligned}$$

  where $g(Z)$ is the indicator function for set C.

- Build the **augmented Lagrangian** (scaled form):

$$L_\rho(X, Z, U) = \|M - X\|_F^2 + g(X) + \lambda \|Z\|_* + \frac{\rho}{2} \|X - Z + U\|_F^2$$

Nonnegative low rank approximation: solution

- **ADMM** steps:

1. $X^{k+1} := \underset{X}{\text{argmin}} \ \left(g(X) + \|M - X\|_F^2 + \frac{\rho}{2}\|X - Z^k + U^k\|_F^2\right)$
   %X-minimization
2. $Z^{k+1} := \underset{Z}{\text{argmin}} \ \left(\lambda\|Z\|_* + \frac{\rho}{2}\|X^{k+1} - Z + U^k\|_F^2\right)$ %Z-minimization
3. $U^{k+1} := U^k + (X^{k+1} - Z^{k+1})$ %dual update

- Remark on Step 2: We have seen that the update of Z corresponds to the computation of the Singular Value Threshold operator of parameter λ/ρ in this context. Hence $Z^{k+1} := \text{SVT}_{\lambda/\rho}(X^{k+1} + U^k)$

Nonnegative low rank approximation: solution

Step 1 computation:

- We have:

$$
\begin{aligned}
X^{k+1} &:= \underset{X}{\operatorname{argmin}} \left(g(X) + \|M - X\|_F^2 + \frac{\rho}{2}\|X - Z^k + U^k\|_F^2 \right) \\
\iff &:= \underset{X \in \mathcal{C}}{\operatorname{argmin}} \left(\underset{=0}{g(X)} + \|M - X\|_F^2 + \frac{\rho}{2}\|X - Z^k + U^k\|_F^2 \right) \\
\iff &= \underset{X \in \mathcal{C}}{\operatorname{argmin}} \left(\|X - \frac{1}{1 + \frac{\rho}{2}} \left(M + \frac{\rho}{2} \left(Z^k - U^k \right) \right) \|_F^2 \right) \\
&= \Pi_{\mathcal{C}} \left(\frac{1}{1 + \frac{\rho}{2}} \left(M + \frac{\rho}{2} \left(Z^k - U^k \right) \right) \right)
\end{aligned}
$$

The third equivalence is let as an exercise.

Nonnegative low rank approximation: solution

- **ADMM** steps:

1. $X^{k+1} := [\left(\frac{1}{1+\frac{\rho}{2}}\left(M + \frac{\rho}{2}\left(Z^k - U^k\right)\right)\right)]_+$ %X-minimization
2. $Z^{k+1} := \mathrm{SVT}_{\lambda/\rho}(X^{k+1} + U^k)$ %Z-minimization
3. $U^{k+1} := U^k + (X^{k+1} - Z^{k+1})$ %dual update

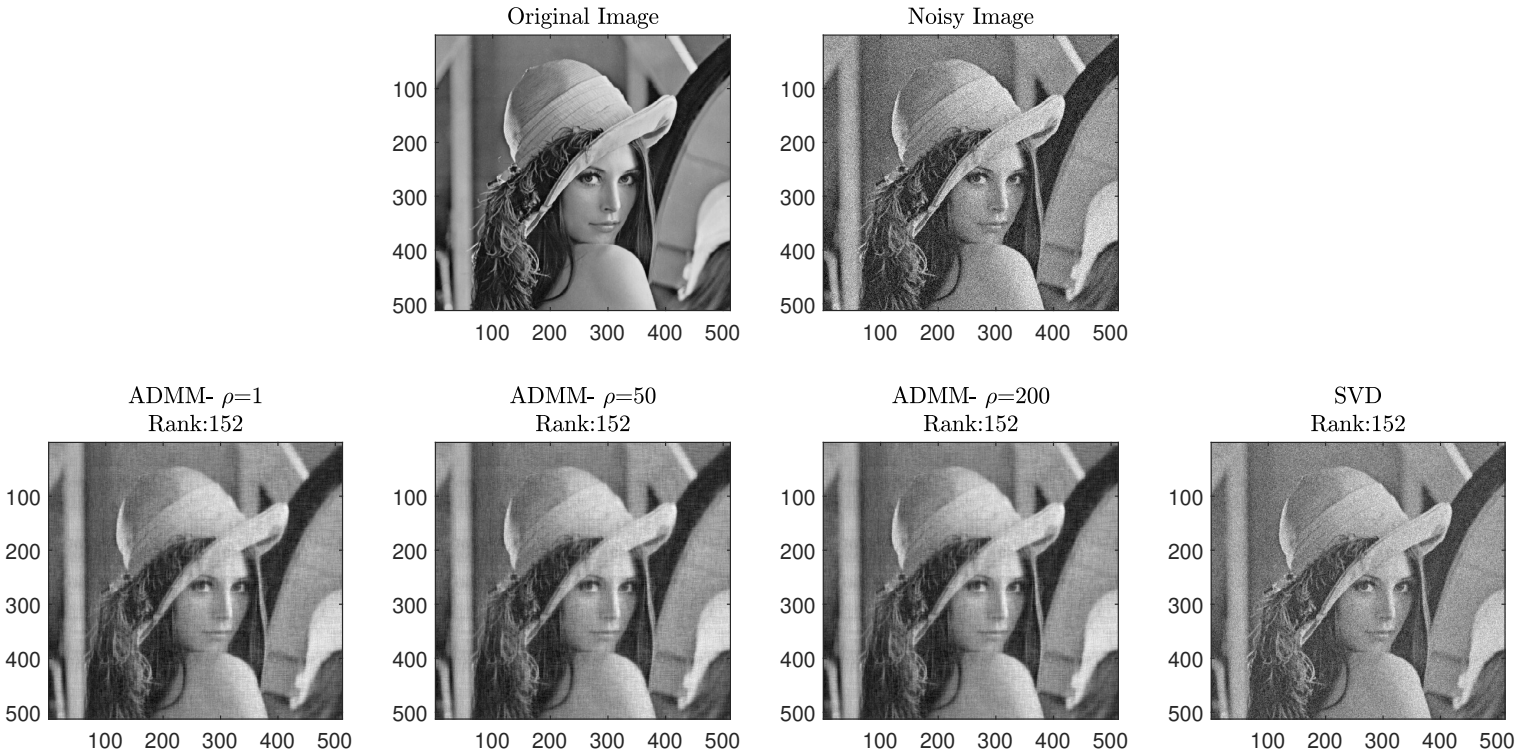
Nonnegative low rank approximation: solution

Summary and Remarks:

- We showed that **ADMM** can be used to solve Problem (23),
- Problem (23) is a convex relaxation of Problem (22).
- others methods have been proposed to tackle Problem (22), see (Grussler and Rantzer 2015) for more insights.
  In a nutshell, non-convex approach's such as Alternating Non-negative Least-Squares seems to outperform, in average, the convex relaxation-based algorithms.
- Potential application: Denoising. Here the input matrix $\tilde{M}$ is a gray-scale image corrupted with additive Gaussian noise ϵ, that is $\tilde{M} = M + \epsilon$ where M is the original image we want to recover.

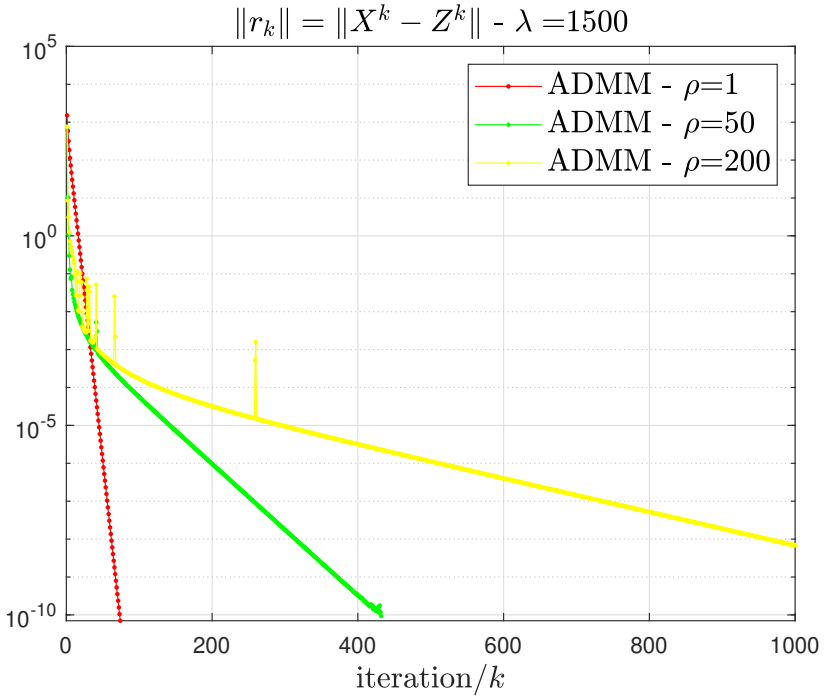
Next slides show the results obtained with our **ADMM** algorithm to tackle the denoising problem, for which the image has been corrupted with White Gaussian noise, that is Gaussian noise with zero mean and variance of 0.01.

Nonnegative low rank approximation: Denoising



Comment: the results are not impressive, more powerful unsupervised approaches should be followed such as the Total variation approach (discussed later).

Nonnegative low rank approximation: Denoising



▸ MatLab Code

Comment: the results are not impressive, more powerful unsupervised approaches should be followed such as the Total variation approach (discussed later).

Nonnegative low rank approximation - Comments

- Some research directions focus on developing efficient methods for computing the best rank-r nonnnegative low-rank approximation as is, let us cite (Song and Ng, 2020)
- You may find here some demos where we compare (Song and Ng, 2020) with ADMM methods.

  ▸ Colab demo

  Trying to theoretically understand why ADMM gives more "fuzzy" results than the ones obtained with (Song and Ng, 2020) could be a good lead :).

Total variation Approach

- Given a vectorized gray-scale image $y = \text{vec}(Y)$ potentially corrupted by noise, the problem is given by:

$$\min_x \|y - x\|_2^2 + \lambda \text{TV}(x) \tag{24}$$

where:

  - $\text{TV}(x) = \|Dx\|_1$ (= ▸ anisotropic TV)
  - D is the column stacked derivative operator (▸ more details),

- **ADMM** form

$$\begin{aligned} \min_x \quad & \|y - x\|_F^2 + \lambda \|z\|_1 \\ \text{subject to} \quad & Dx - z = 0, \end{aligned}$$

- **Demo:** use this ▸ MatLab code
  Requirements: you should install ▸ cvx first

References

Boyd, Parikh, Chu, Peleato and Eckstein (2010)

Distributed Optimization and Statistical Learning via the Alternating Method of Multipliers

Machine Learning Vol. 3, No. 1 (2010) 1–122

Y.Nesterov. (2004)

Introductory Lectures on Convex Optimization: A Basic Course.

Kluwer Academic Publishers..

X.Zhou. (2018)

On the Fenchel Duality between Strong Convexity and Lipschitz Continuous Gradient.

arXiv 1803.06573.

N.Parikh and S.Boyd. (2013)

Proximal Algorithms.

*Foundations and Trends in Optimization.*Vol. 1, No. 3 (2013) 123–231

References

R.Tibshirani (2015)

Alternating Direction Method of Multipliers

Stanford lectures/seminars Convex Optimization 10-725

A.Beck and M.Teboulle. (2013)

A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems

SIAM J. IMAGING SCIENCES Vol. 2, No. 1, pp. 183–202

Y. Yang and M. Pesavento. (2017)

A Unified Successive Pseudoconvex Approximation Framework

IEEE Transactions on Signal Processing vol. 65, no. 13, pp. 3313-3327, Dec 2017

Andersen Ang. (2021)

Personal Website

https://angms.science/index.html Teaching - Notes section

References

M. Fazel (2002)

Matrix rank minimization with application

PhD thesis Stanford University

E.J. Candes, X. Li, Y. Ma and J. Wright. (2011)

Robust Principal Component Analysis?

Journal of the ACM vol. 58, iss. 3, May 2011

V. Chandrasekaran, S. Sanghavi, P. A. Parrilo, and A. S. Willsky. (2011)

Rank-sparsity incoherence for matrix decomposition

SIAM Journal on Optimization vol. 21, no. 2, pp. 572–596, 2011.

J. Wright, A. Ganesh, S. Rao, Y. Peng, and Y. Ma. (2009)

Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization

Advances in Neural Information Processing Systems 22

References

D. Kressner (2018)

Low Rank Approximation Lecture 1

Lectures EPFL

C. Grussler and A. Rantzer (2015)

On optimal low-rank approximation of non-negative matrices

2015 IEEE 54th Annual Conference on Decision and Control, December 15-18, 2015. Osaka, Japan

. G.-J. Song M. Ng (2020)

Nonnegative low rank matrix approximation for nonnegative matrices

Applied Mathematics Letters, vol. 105, 2020.


That's all Folks!