Deep Unfolding of a Proximal Interior Point Method for Image Restoration

M.-C. Corbineau[1]

in collaboration with

C. Bertocchi[2], E. Chouzenoux[1], J.C. Pesquet[1], M. Prato[2]

[1]Université Paris-Saclay, CentraleSupélec, Inria, Centre de Vision Numérique, Gif-sur-Yvette, France
[2]Università di Modena e Reggio Emilia, Modena, Italy

19 November 2019

Workshop on Regularisation for Inverse Problems and Machine Learning

Jussieu, Paris

Motivation

Inverse problem in imaging

$$\boldsymbol{y} = \mathcal{D}(\boldsymbol{H}\overline{\boldsymbol{x}})$$

where $\boldsymbol{y} \in \mathbb{R}^m$ observed image, $\mathcal{D}$ degradation model, $\boldsymbol{H} \in \mathbb{R}^{m \times n}$ linear observation model, $\overline{\boldsymbol{x}} \in \mathbb{R}^n$ original image

Motivation

Inverse problem in imaging

$$\boldsymbol{y} = \mathcal{D}(\boldsymbol{H}\overline{\boldsymbol{x}})$$

where $\boldsymbol{y} \in \mathbb{R}^m$ observed image, $\mathcal{D}$ degradation model, $\boldsymbol{H} \in \mathbb{R}^{m \times n}$ linear observation model, $\overline{\boldsymbol{x}} \in \mathbb{R}^n$ original image

Variational methods

$$\underset{\boldsymbol{x} \in \mathcal{C}}{\text{minimize}} \quad f(\boldsymbol{Hx}, \boldsymbol{y}) + \lambda \mathcal{R}(\boldsymbol{x})$$

where $f : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$ data-fitting term, $\mathcal{R} : \mathbb{R}^n \to \mathbb{R}$ regularization function, $\lambda > 0$ regularization weight

- ✓ Incorporate prior knowledge about solution and enforce desirable constraints
- ✗ No closed-form solution $\to$ advanced algorithms
- ✗ Estimation of λ and tuning of algorithm parameters $\to$ time-consuming

Motivation

Inverse problem in imaging

$$\boldsymbol{y} = \mathcal{D}(\boldsymbol{H}\overline{\boldsymbol{x}})$$

where $\boldsymbol{y} \in \mathbb{R}^m$ observed image, $\mathcal{D}$ degradation model, $\boldsymbol{H} \in \mathbb{R}^{m \times n}$ linear observation model, $\overline{\boldsymbol{x}} \in \mathbb{R}^n$ original image

Variational methods

$$\underset{\boldsymbol{x} \in \mathcal{C}}{\text{minimize}} \quad f(\boldsymbol{H}\boldsymbol{x}, \boldsymbol{y}) + \lambda \mathcal{R}(\boldsymbol{x})$$

where $f : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$ data-fitting term, $\mathcal{R} : \mathbb{R}^n \to \mathbb{R}$ regularization function, $\lambda > 0$ regularization weight

- ✓ Incorporate prior knowledge about solution and enforce desirable constraints
- ✗ No closed-form solution $\rightarrow$ advanced algorithms
- ✗ Estimation of λ and tuning of algorithm parameters $\rightarrow$ time-consuming

Deep-learning methods

- ✓ Generic and very efficient architectures
- ✗ Pre-processing step : solve optimization problem $\rightarrow$ estimate regularization parameter
- ✗ Black-box, no theoretical guarantees

Motivation

Inverse problem in imaging

$$\boldsymbol{y} = \mathcal{D}(\boldsymbol{H}\overline{\boldsymbol{x}})$$

where $\boldsymbol{y} \in \mathbb{R}^m$ observed image, $\mathcal{D}$ degradation model, $\boldsymbol{H} \in \mathbb{R}^{m\times n}$ linear observation model, $\overline{\boldsymbol{x}} \in \mathbb{R}^n$ original image

Variational methods

$$\underset{\boldsymbol{x}\in\mathcal{C}}{\text{minimize}} \quad f(\boldsymbol{Hx}, \boldsymbol{y}) + \lambda\mathcal{R}(\boldsymbol{x})$$

where $f : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$ data-fitting term, $\mathcal{R} : \mathbb{R}^n \to \mathbb{R}$ regularization function, $\lambda > 0$ regularization weight

- ✓ Incorporate prior knowledge about solution and enforce desirable constraints
- ✗ No closed-form solution $\to$ advanced algorithms
- ✗ Estimation of λ and tuning of algorithm parameters $\to$ time-consuming

Deep-learning methods

- ✓ Generic and very efficient architectures
- ✗ Pre-processing step : solve optimization problem $\to$ estimate regularization parameter
- ✗ Black-box, no theoretical guarantees

$\to$ Combine benefits of both approaches : unfold proximal interior point algorithm

Deep Unfolding

- Examples
  - Sparse coding : FISTA **[Gregor and LeCun, 2010]**, ISTA **[Kamilov and Mansour, 2016]**
  - Compressive sensing : ISTA **[Zhang and Ghanem, 2018]**, ADMM **[Sun et al., 2016]**
- Principle

Iterative solver		Unfolded algorithm
for $k = 0, 1, \ldots$ $\boldsymbol{x}_{k+1} = \mathcal{A}(\boldsymbol{x}_k, \underbrace{\boldsymbol{\theta}_k}_{\text{hyperparameters}})$	$\Longrightarrow$	for $k = 0, 1, \ldots, K-1$ $\boldsymbol{x}_{k+1} = \mathcal{A}\left(\boldsymbol{x}_k, \underbrace{\mathcal{L}_k^{(\boldsymbol{\theta})}(\boldsymbol{x}_k)}_{\text{layer estimating hyperparameters}}\right)$
Estimate : $\boldsymbol{x}^* = \lim_{k\to\infty} \boldsymbol{x}_k$		Estimate : $\boldsymbol{x}^* = \boldsymbol{x}_K$

- Operators and functions included in $\mathcal{A}$ can be learned
  - ✓ Gradient backpropagation and training are simpler
  - ✗ Link to the original algorithm is weakened

Notation and Assumptions

Proximity operator

Let $\Gamma_0(\mathbb{R}^n)$ be the set of proper lsc convex functions from $\mathbb{R}^n$ to $\mathbb{R} \cup \{+\infty\}$. The **proximal operator** [http://proximity-operator.net/] of $g \in \Gamma_0(\mathbb{R}^n)$ at $\boldsymbol{x} \in \mathbb{R}^n$ is uniquely defined as

$$\mathrm{prox}_g(\boldsymbol{x}) = \underset{\boldsymbol{z} \in \mathbb{R}^n}{\mathrm{argmin}} \left(g(\boldsymbol{z}) + \frac{1}{2}\|\boldsymbol{z} - \boldsymbol{x}\|^2 \right).$$

Assumptions

$$\mathcal{P}_0 : \quad \underset{\boldsymbol{x} \in \mathcal{C}}{\text{minimize}} \quad f(\boldsymbol{Hx}, \boldsymbol{y}) + \lambda \mathcal{R}(\boldsymbol{x})$$

We assume that $f(\cdot, \boldsymbol{y})$ and $\mathcal{R}$ are twice-differentiable, $f(\boldsymbol{H}\cdot, \boldsymbol{y}) + \lambda\mathcal{R} \in \Gamma_0(\mathbb{R}^n)$ is either coercive or $\mathcal{C}$ is bounded. The feasible set is defined as

$$\mathcal{C} = \{\boldsymbol{x} \in \mathbb{R}^n \mid (\forall i \in \{1, \dots, p\}) \;\; c_i(\boldsymbol{x}) \geq 0\}$$

where $(\forall i \in \{1, \dots, p\})$, $-c_i \in \Gamma_0(\mathbb{R}^n)$. The strict interior of the feasible set is nonempty.

- Existence of a solution to $\mathcal{P}_0$
- Twice-differentiability : training using gradient descent

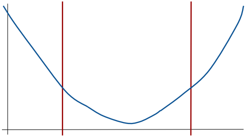
$\mathcal{B}$: logarithmic barrier

$$(\forall \boldsymbol{x} \in \mathbb{R}^n) \quad \mathcal{B}(\boldsymbol{x}) = \begin{cases} -\sum_{i=1}^{p} \ln(c_i(\boldsymbol{x})) & \text{if } \boldsymbol{x} \in \mathrm{int}\mathcal{C} \\ +\infty & \text{otherwise.} \end{cases}$$

Logarithmic barrier method

Constrained Problem

$$\mathcal{P}_0: \quad \underset{\boldsymbol{x}\in\mathcal{C}}{\text{minimize}} \quad f(\boldsymbol{Hx}, \boldsymbol{y}) + \lambda\mathcal{R}(\boldsymbol{x})$$



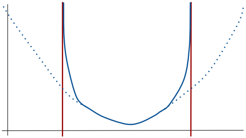
Logarithmic barrier method

Constrained Problem
$$\mathcal{P}_0 : \quad \underset{\boldsymbol{x}\in\mathcal{C}}{\text{minimize}} \quad f(\boldsymbol{Hx}, \boldsymbol{y}) + \lambda\mathcal{R}(\boldsymbol{x})$$

$$\Downarrow$$

Unconstrained Subproblem
$$\mathcal{P}_\mu : \quad \underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}} \quad f(\boldsymbol{Hx}, \boldsymbol{y}) + \lambda\mathcal{R}(\boldsymbol{x}) + \mu\boldsymbol{\mathcal{B}}(\boldsymbol{x})$$

where $\mu > 0$ is the barrier parameter.



$\mathcal{P}_0$ is replaced by a sequence of subproblems $(\mathcal{P}_{\mu_j})_{j\in\mathbb{N}}$.

- Subproblems solved approximately for a sequence $\mu_j \to 0$
- Main advantages : feasible iterates, superlinear convergence for NLP
- ✗ Inversion of an $n \times n$ matrix at each step

Proximal interior point strategy

→ Combine interior point method with proximity operator

Exact version of the proximal IPM in **[Kaplan and Tichatschke, 1998]**.

Let $\boldsymbol{x}_0 \in \mathrm{int}\mathcal{C}$, $\underline{\gamma} > 0$, $(\forall k \in \mathbb{N})$ $\underline{\gamma} \leq \gamma_k$ and $\mu_k \to 0$;
for $k = 0, 1, \ldots$ **do**
$\boldsymbol{x}_{k+1} = \mathrm{prox}_{\gamma_k(f(\boldsymbol{H}\cdot,\boldsymbol{y})+\lambda\mathcal{R}+\mu_k\mathcal{B})}(\boldsymbol{x}_k)$
end for

✗ No closed-form solution for $\mathrm{prox}_{\gamma_k(f(H\cdot,y)+\lambda\mathcal{R}+\mu_k\mathcal{B})}$

Proximal interior point strategy

→ Combine interior point method with proximity operator

Exact version of the proximal IPM in **[Kaplan and Tichatschke, 1998]**.

Let $\boldsymbol{x}_0 \in \operatorname{int}\mathcal{C}$, $\underline{\gamma} > 0$, $(\forall k \in \mathbb{N})$ $\underline{\gamma} \leq \gamma_k$ and $\mu_k \to 0$;
for $k = 0, 1, \ldots$ **do**
$\quad \boldsymbol{x}_{k+1} = \operatorname{prox}_{\gamma_k(f(\boldsymbol{H}\cdot,\boldsymbol{y})+\lambda\mathcal{R}+\mu_k\mathcal{B})}(\boldsymbol{x}_k)$
end for

✗ No closed-form solution for $\operatorname{prox}_{\gamma_k(f(H\cdot,y)+\lambda\mathcal{R}+\mu_k\mathcal{B})}$

Proposed forward–backward proximal IPM.

Let $\boldsymbol{x}_0 \in \operatorname{int}\mathcal{C}$, $\underline{\gamma} > 0$, $(\forall k \in \mathbb{N})$ $\underline{\gamma} \leq \gamma_k$ and $\mu_k \to 0$;
for $k = 0, 1, \ldots$ **do**
$\quad \boldsymbol{x}_{k+1} = \operatorname{prox}_{\gamma_k\mu_k\mathcal{B}}\left(\boldsymbol{x}_k - \gamma_k\left(\boldsymbol{H}^\top\nabla_1 f(\boldsymbol{H}\boldsymbol{x}_k, \boldsymbol{y}) + \lambda\nabla\mathcal{R}(\boldsymbol{x}_k)\right)\right)$
end for

✓ Only requires $\operatorname{prox}_{\gamma_k\mu_k\mathcal{B}}$

Proximity operator of the barrier

Let $\varphi : (\boldsymbol{x}, \alpha) \mapsto \mathrm{prox}_{\alpha\mathcal{B}}(\boldsymbol{x})$.

A neural network obtained by unfolding an iterative solver $\mathcal{A}$

- requires to compute $\mathcal{A}(\boldsymbol{x}, \boldsymbol{\theta})$.

  $\rightarrow$ expression for the proximity operator $\varphi(\boldsymbol{x}, \alpha)$?

Proximity operator of the barrier

Let $\varphi : (\boldsymbol{x}, \alpha) \mapsto \mathrm{prox}_{\alpha\mathcal{B}}(\boldsymbol{x})$.

A neural network obtained by unfolding an iterative solver $\mathcal{A}$

- requires to compute $\mathcal{A}(\boldsymbol{x}, \boldsymbol{\theta})$.

  $\rightarrow$ expression for the proximity operator $\varphi(\boldsymbol{x}, \alpha)$?
- is trained with loss function $\ell(\boldsymbol{x}_K, \overline{\boldsymbol{x}})$ by gradient descent.

$$\boldsymbol{\theta}_k = \mathcal{L}_k^{(\boldsymbol{\theta})}(\boldsymbol{x}_k)$$
$$\boldsymbol{x}_{k+1} = \mathcal{A}\left(\boldsymbol{x}_k, \boldsymbol{\theta}_k\right)$$

$\rightarrow$ first derivatives of ℓ wrt learnable parameters of hidden layers $\left(\mathcal{L}_k^{(\boldsymbol{\theta})}\right)_{0 \le k \le K-1}$?

$\rightarrow$ the chain rule requires the derivative of $\mathcal{A}$ wrt $\boldsymbol{x}$ and $\boldsymbol{\theta}$

$\rightarrow$ expressions for $J_\varphi^{(\boldsymbol{x})}(\boldsymbol{x}, \alpha)$ and $\nabla_\varphi^{(\alpha)}(\boldsymbol{x}, \alpha)$?

Proximity operator of the barrier

Let $\varphi : (\boldsymbol{x}, \alpha) \mapsto \mathrm{prox}_{\alpha\mathcal{B}}(\boldsymbol{x})$.

A neural network obtained by unfolding an iterative solver $\mathcal{A}$

- requires to compute $\mathcal{A}(\boldsymbol{x}, \boldsymbol{\theta})$.

  $\rightarrow$ expression for the proximity operator $\varphi(\boldsymbol{x}, \alpha)$?
- is trained with loss function $\ell(\boldsymbol{x}_K, \overline{\boldsymbol{x}})$ by gradient descent.

$$\begin{aligned}\boldsymbol{\theta}_k &= \mathcal{L}_k^{(\boldsymbol{\theta})}(\boldsymbol{x}_k)\\ \boldsymbol{x}_{k+1} &= \mathcal{A}\left(\boldsymbol{x}_k, \boldsymbol{\theta}_k\right)\end{aligned}$$

$\rightarrow$ first derivatives of ℓ wrt learnable parameters of hidden layers $\left(\mathcal{L}_k^{(\boldsymbol{\theta})}\right)_{0\leq k\leq K-1}$?

$\rightarrow$ the chain rule requires the derivative of $\mathcal{A}$ wrt $\boldsymbol{x}$ and $\boldsymbol{\theta}$

$\rightarrow$ expressions for $J_\varphi^{(\boldsymbol{x})}(\boldsymbol{x}, \alpha)$ and $\nabla_\varphi^{(\alpha)}(\boldsymbol{x}, \alpha)$?

These quantities depend on $\mathcal{B}$ and on the feasible set.

$\Rightarrow$ We obtain their expressions for three types of constraints.

Proximity operator of the barrier

Affine constraints $\mathcal{C} = \left\{ \boldsymbol{x} \in \mathbb{R}^n \mid \boldsymbol{a}^\top \boldsymbol{x} \leq b \right\}$

Proposition 1

Let $\varphi : (\boldsymbol{x}, \alpha) \mapsto \mathrm{prox}_{\alpha \mathcal{B}}(\mathrm{x})$. Then, for every $(\boldsymbol{x}, \alpha) \in \mathbb{R}^n \times \mathbb{R}_+^*$,

$$\varphi(\boldsymbol{x}, \alpha) = \boldsymbol{x} + \frac{b - \boldsymbol{a}^\top \boldsymbol{x} - \sqrt{(b - \boldsymbol{a}^\top \boldsymbol{x})^2 + 4\alpha \|\boldsymbol{a}\|^2}}{2\|\boldsymbol{a}\|^2} \boldsymbol{a}.$$

In addition, the Jacobian matrix of φ wrt x and the gradient of φ wrt α are given by

$$J_\varphi^{(\boldsymbol{x})}(\boldsymbol{x}, \alpha) = \mathbb{I}_n - \frac{1}{2\|\boldsymbol{a}\|^2} \left(1 + \frac{\boldsymbol{a}^\top \boldsymbol{x} - b}{\sqrt{(b - \boldsymbol{a}^\top \boldsymbol{x})^2 + 4\alpha \|\boldsymbol{a}\|^2}} \right) \boldsymbol{a}\boldsymbol{a}^\top$$

and

$$\nabla_\varphi^{(\alpha)}(\boldsymbol{x}, \alpha) = \frac{-1}{\sqrt{(b - \boldsymbol{a}^\top \boldsymbol{x})^2 + 4\alpha \|\boldsymbol{a}\|^2}} \boldsymbol{a}$$

Proof : **[Chaux *et al.*,2007]** and **[Bauschke and Combettes,2017]**

Proximity operator of the barrier

Hyperslab constraints $\quad \mathcal{C} = \left\{ \boldsymbol{x} \in \mathbb{R}^n \mid b_m \leq \boldsymbol{a}^\top \boldsymbol{x} \leq b_M \right\}$

Proposition 2

Let $\varphi : (\boldsymbol{x}, \alpha) \mapsto \mathrm{prox}_{\alpha\mathcal{B}}(\mathrm{x})$. Then, for every $(\boldsymbol{x}, \alpha) \in \mathbb{R}^n \times \mathbb{R}_+^*$,

$$\varphi(\boldsymbol{x}, \alpha) = \boldsymbol{x} + \frac{\kappa(\boldsymbol{x}, \alpha) - \boldsymbol{a}^\top \boldsymbol{x}}{\|\boldsymbol{a}\|^2} \boldsymbol{a},$$

where $\kappa(\boldsymbol{x}, \alpha)$ is the unique solution in $]b_m, b_M[$, of the following cubic equation,

$$0 = z^3 - (b_m + b_M + \boldsymbol{a}^\top \boldsymbol{x}) z^2 + (b_m b_M + \boldsymbol{a}^\top \boldsymbol{x}(b_m + b_M) - 2\alpha\|\boldsymbol{a}\|^2) z - b_m b_M \boldsymbol{a}^\top \boldsymbol{x} + \alpha(b_m + b_M)\|\boldsymbol{a}\|^2.$$

In addition, the Jacobian matrix of φ wrt $\boldsymbol{x}$ and the gradient of φ wrt α are given by

$$J_\varphi^{(\boldsymbol{x})}(\boldsymbol{x}, \alpha) = \mathbb{I}_n - \frac{1}{\|\boldsymbol{a}\|^2} \left(\frac{(b_M - \kappa(\boldsymbol{x}, \alpha))(b_m - \kappa(\boldsymbol{x}, \alpha))}{\eta(\boldsymbol{x}, \alpha)} - 1 \right) \boldsymbol{a}\boldsymbol{a}^\top$$

and

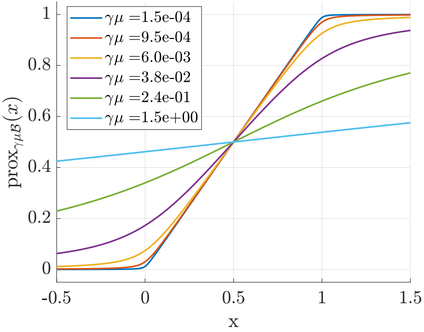
$$\nabla_\varphi^{(\alpha)}(\boldsymbol{x}, \alpha) = \frac{2\kappa(\boldsymbol{x}, \alpha) - b_m - b_M}{\eta(\boldsymbol{x}, \alpha)} \boldsymbol{a},$$

where $\eta(\boldsymbol{x}, \alpha) = (b_M - \kappa(\boldsymbol{x}, \alpha))(b_m - \kappa(\boldsymbol{x}, \alpha)) - (b_m + b_M - 2\kappa(\boldsymbol{x}, \alpha))(\kappa(\boldsymbol{x}, \alpha) - \boldsymbol{a}^\top \boldsymbol{x}) - 2\alpha\|\boldsymbol{a}\|^2$.

Proof : **[Chaux *et al.*,2007]**, **[Bauschke and Combettes,2017]** and implicit function theorem

Proximity operator of the barrier

Bound constraints $\mathcal{C} = \{x \in \mathbb{R} \mid 0 \leq x \leq 1\}$



Proximity operator of the barrier

Bounded ℓ_2-norm $\quad \mathcal{C} = \left\{ \boldsymbol{x} \in \mathbb{R}^n \mid \|\boldsymbol{x} - \boldsymbol{c}\|^2 \leq \rho \right\}$

Proposition 3

Let $\varphi : (\boldsymbol{x}, \alpha) \mapsto \mathrm{prox}_{\alpha\mathcal{B}}(\boldsymbol{x})$. Then, for every $(\boldsymbol{x}, \alpha) \in \mathbb{R}^n \times \mathbb{R}^*_+$,

$$\varphi(\boldsymbol{x}, \alpha) = c + \frac{\rho - \kappa(\boldsymbol{x}, \alpha)^2}{\rho - \kappa(\boldsymbol{x}, \alpha)^2 + 2\alpha}(\boldsymbol{x} - \boldsymbol{c}),$$

where $\kappa(\boldsymbol{x}, \alpha)$ is the unique solution in $]0, \sqrt{\rho}[$, of the following cubic equation,

$$0 = z^3 - \|\boldsymbol{x} - \boldsymbol{c}\| z^2 - (\rho + 2\alpha) z + \rho \|\boldsymbol{x} - \boldsymbol{c}\|.$$

In addition, the Jacobian matrix of φ wrt $\boldsymbol{x}$ and the gradient of φ wrt α are given by

$$J_\varphi^{(\boldsymbol{x})}(\boldsymbol{x}, \alpha) = \frac{\rho - \|\varphi(\boldsymbol{x}, \alpha) - \boldsymbol{c}\|^2}{\rho - \|\varphi(\boldsymbol{x}, \alpha) - \boldsymbol{c}\|^2 + 2\alpha} M(\boldsymbol{x}, \alpha)$$

and

$$\nabla_\varphi^{(\alpha)}(\boldsymbol{x}, \alpha) = \frac{-2}{\rho - \|\varphi(\boldsymbol{x}, \alpha) - \boldsymbol{c}\|^2 + 2\alpha} M(\boldsymbol{x}, \alpha)(\varphi(\boldsymbol{x}, \alpha) - \boldsymbol{c}),$$

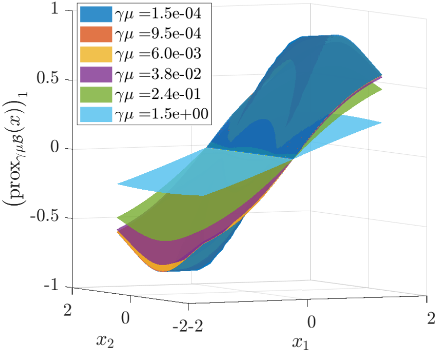
where

$$M(\boldsymbol{x}, \alpha) = \mathbb{I}_n - \frac{2(\boldsymbol{x} - \varphi(\boldsymbol{x}, \alpha))(\varphi(\boldsymbol{x}, \alpha) - \boldsymbol{c})^\top}{\rho - 3\|\varphi(\boldsymbol{x}, \alpha) - \boldsymbol{c}\|^2 + 2\alpha + 2(\varphi(\boldsymbol{x}, \alpha) - \boldsymbol{c})^\top (\boldsymbol{x} - \boldsymbol{c})}.$$

Proof : **[Bauschke and Combettes,2017]**, Sherma-Morrison lemma and implicit function theorem

Proximity operator of the barrier

Bounded ℓ_2-norm $\mathcal{C} = \left\{ \boldsymbol{x} \in \mathbb{R}^2 \mid \|\boldsymbol{x}\|^2 \leq 0.7 \right\}$



Proposed strategy

Forward–backward proximal IPM.

Let $\boldsymbol{x}_0 \in \mathrm{int}\mathcal{C}$, $\underline{\gamma} > 0$, $(\forall k \in \mathbb{N})$ $\underline{\gamma} \leq \gamma_k$ and $\mu_k \to 0$;
for $k = 0, 1, \ldots$ **do**

$$\boldsymbol{x}_{k+1} = \mathrm{prox}_{\gamma_k \mu_k \mathcal{B}} \left(\boldsymbol{x}_k - \gamma_k \left(H^\top \nabla_1 f(\boldsymbol{H}\boldsymbol{x}_k, \boldsymbol{y}) + \lambda \nabla \mathcal{R}(\boldsymbol{x}_k) \right) \right)$$

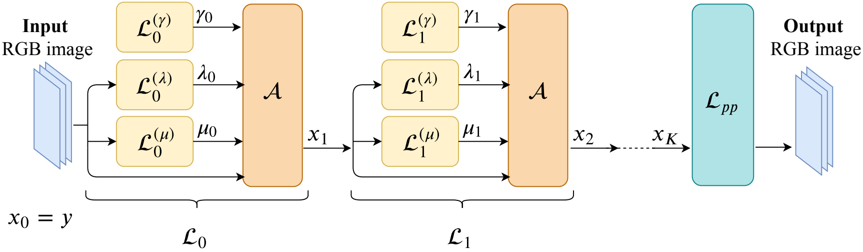
end for

- ✓ Efficient algorithm for constrained optimization
- ✗ Setting of the parameters $(\mu_k, \gamma_k)_{k \in \mathbb{N}}$?
- ✗ Finding the regularization parameter λ so as to optimize the visual quality of the solution ?

→ Unfold proximal IP algorithm over K iterations, untie γ, μ and λ across network

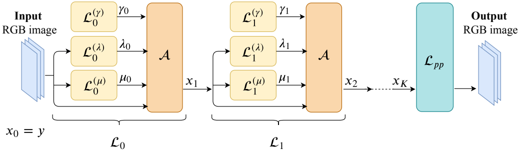
$$\mathcal{A}(\boldsymbol{x}_k, \mu_k, \gamma_k, \lambda_k) = \mathrm{prox}_{\gamma_k \mu_k \mathcal{B}} \left(\boldsymbol{x}_k - \gamma_k \left(H^\top \nabla_1 f(\boldsymbol{H}\boldsymbol{x}_k, \boldsymbol{y}) + \lambda_k \nabla \mathcal{R}(\boldsymbol{x}_k) \right) \right)$$

iRestNet architecture



Input : $x_0 = y$ blurred image

iRestNet architecture



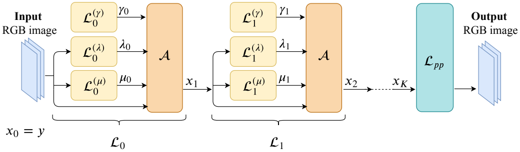
Input : $x_0 = y$ blurred image

Hidden structures

- $(\mathcal{L}_k^{(\gamma)})_{0 \leq k \leq K-1}$: estimate stepsize, positive $\rightarrow$ Softplus (smooth approx ReLU)

$$\gamma_k = \mathcal{L}_k^{(\gamma)} = \mathrm{Softplus}(a_k)$$

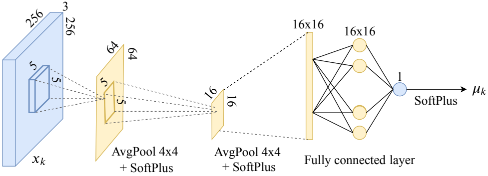
iRestNet architecture



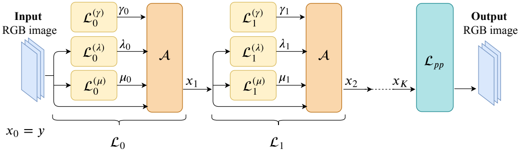
Input : $x_0 = y$ blurred image

Hidden structures

- $(\mathcal{L}_k^{(\gamma)})_{0 \leq k \leq K-1}$: estimate stepsize
- $(\mathcal{L}_k^{(\mu)})_{0 \leq k \leq K-1}$: estimate barrier parameter



iRestNet architecture

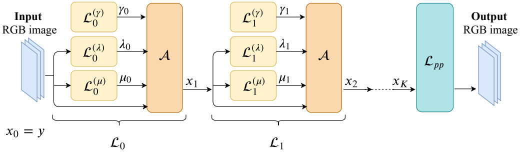


Input : $x_0 = y$ blurred image

Hidden structures

- $(\mathcal{L}_k^{(\gamma)})_{0\leq k\leq K-1}$: estimate stepsize
- $(\mathcal{L}_k^{(\mu)})_{0\leq k\leq K-1}$: estimate barrier parameter
- $(\mathcal{L}_k^{(\lambda)})_{0\leq k\leq K-1}$: estimate regularization parameter $\rightarrow$ image statistics, noise level

iRestNet architecture

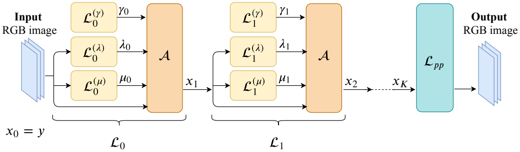


Input : $x_0 = y$ blurred image

Hidden structures

- $(\mathcal{L}_k^{(\gamma)})_{0\leq k\leq K-1}$: estimate stepsize
- $(\mathcal{L}_k^{(\mu)})_{0\leq k\leq K-1}$: estimate barrier parameter
- $(\mathcal{L}_k^{(\lambda)})_{0\leq k\leq K-1}$: estimate regularization parameter
- $\mathcal{A}(x_k, \mu_k, \gamma_k, \lambda_k) = \mathrm{prox}_{\gamma_k \mu_k \mathcal{B}}\left(x_k - \gamma_k\left(H^\top \nabla_1 f(Hx_k, y) + \lambda_k \nabla \mathcal{R}(x_k)\right)\right)$

iRestNet architecture

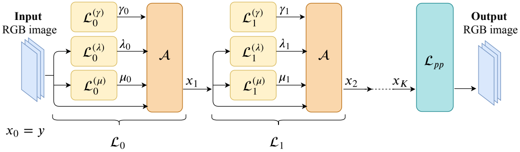


Input : $x_0 = y$ blurred image

Hidden structures

- $(\mathcal{L}_k^{(\gamma)})_{0\leq k\leq K-1}$: estimate stepsize
- $(\mathcal{L}_k^{(\mu)})_{0\leq k\leq K-1}$: estimate barrier parameter
- $(\mathcal{L}_k^{(\lambda)})_{0\leq k\leq K-1}$: estimate regularization parameter
- $\mathcal{A}(x_k, \mu_k, \gamma_k, \lambda_k) = \mathrm{prox}_{\gamma_k \mu_k \mathcal{B}}\left(x_k - \gamma_k\left(H^\top \nabla_1 f(Hx_k, y) + \lambda_k \nabla \mathcal{R}(x_k)\right)\right)$
- $\mathcal{L}_{\mathrm{pp}}$: post-processing layer $\rightarrow$ e.g. removes small artifacts

iRestNet architecture



Input : $x_0 = y$ blurred image

Hidden structures

- $(\mathcal{L}_k^{(\gamma)})_{0 \le k \le K-1}$: estimate stepsize
- $(\mathcal{L}_k^{(\mu)})_{0 \le k \le K-1}$: estimate barrier parameter
- $(\mathcal{L}_k^{(\lambda)})_{0 \le k \le K-1}$: estimate regularization parameter
- $\mathcal{A}(x_k, \mu_k, \gamma_k, \lambda_k) = \operatorname{prox}_{\gamma_k \mu_k \mathcal{B}} \left(x_k - \gamma_k \left(H^\top \nabla_1 f(Hx_k, y) + \lambda_k \nabla \mathcal{R}(x_k) \right) \right)$
- $\mathcal{L}_{\mathrm{pp}}$: post-processing layer $\rightarrow$ e.g. removes small artifacts

Training Gradient descent and backpropagation ($\nabla \mathcal{A}$ with Propositions 1-3)

Network stability

What about the network performance when the input is perturbed ?

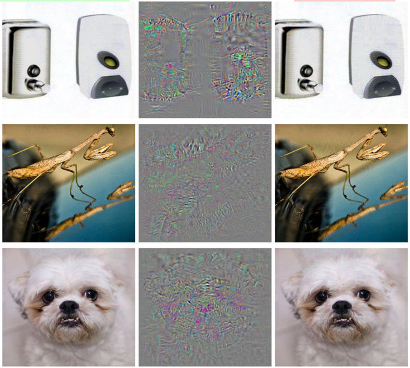
Network stability

What about the network performance when the input is perturbed ?

- Applications with high risk and legal responsibility (medical image processing, defense, etc...) $\rightarrow$ **need guarantees**
- Deep learning : lack of theoretical guarantees, e.g. AlexNet **[Szegedy *et al.*, 2013]**

Original image | Perturbation | New input

✓ Correctly classified



✗ Categorized as 'Ostrich'

FIGURE – Adversarial examples for AlexNet **[Szegedy *et al.*, 2013]**

Network stability

Formulation

- Neural network : $T(\cdot) : \mathbb{R}^n \to \mathbb{R}^n$
- Input image : $\boldsymbol{x} \in \mathbb{R}^n$
- Perturbation : $\boldsymbol{\delta x} \in \mathbb{R}^n$
- Output perturbation : $\Delta T(\boldsymbol{x}) = T(\boldsymbol{x} + \boldsymbol{\delta x}) - T(\boldsymbol{x})$
- Questions : $\|\Delta T(\boldsymbol{x})\|$? $\Delta T(\boldsymbol{x})$?

Network stability

Formulation

- Neural network : $T(\cdot) : \mathbb{R}^n \to \mathbb{R}^n$
- Input image : $\boldsymbol{x} \in \mathbb{R}^n$
- Perturbation : $\boldsymbol{\delta x} \in \mathbb{R}^n$
- Output perturbation : $\Delta T(\boldsymbol{x}) = T(\boldsymbol{x} + \boldsymbol{\delta x}) - T(\boldsymbol{x})$
- Questions : $\|\Delta T(\boldsymbol{x})\|$? $\Delta T(\boldsymbol{x})$?

Tools

1. Framework of **averaged operators**
2. iRestNet is re-written as a **generic feedforward neural network**
3. Results from the following recent work :

P. L. Combettes and J.-C. Pesquet.
Deep neural network structures solving variational inequalities
https://arxiv.org/abs/1808.07526.

Nonexpansive operators

Definition – Nonexpansiveness

Let $T : \mathbb{R}^n \to \mathbb{R}^n$. Then, T is nonexpansive if it is 1-Lipschitz continuous, i.e.,

$$(\forall \boldsymbol{x} \in \mathbb{R}^n)(\forall \boldsymbol{y} \in \mathbb{R}^n) \quad \|T(\boldsymbol{x}) - T(\boldsymbol{y})\| \leq \|\boldsymbol{x} - \boldsymbol{y}\|.$$

$\Longrightarrow$ Bound on the norm of the output variation when input is perturbed :

$$\boxed{\|\Delta T(\boldsymbol{x})\| \leq \|\boldsymbol{\delta x}\|}$$

Averaged operators

Definition – α-averaged operator

Let $T : \mathbb{R}^n \to \mathbb{R}^n$ be nonexpansive, and let $\alpha \in [0, 1]$. Then, T is α-averaged if there exists a nonexpansive operator $R : \mathbb{R}^n \to \mathbb{R}^n$ such that $T = (1 - \alpha)I_n + \alpha R$.

Averaged operators

Definition – α-averaged operator

Let $T : \mathbb{R}^n \to \mathbb{R}^n$ be nonexpansive, and let $\alpha \in [0,1]$. Then, T is α-averaged if there exists a nonexpansive operator $R : \mathbb{R}^n \to \mathbb{R}^n$ such that $T = (1-\alpha)I_n + \alpha R$.

- If T is averaged, then it is nonexpansive.
- Let $\alpha \in]0,1]$. T is α-averaged if and only if for every $\boldsymbol{x} \in \mathbb{R}^n$ and $\boldsymbol{y} \in \mathbb{R}^n$,

$$\|T(\boldsymbol{x}) - T(\boldsymbol{y})\|^2 \leq \|\boldsymbol{x} - \boldsymbol{y}\|^2 - \frac{1-\alpha}{\alpha}\|(I_n - T)(\boldsymbol{x}) - (I_n - T)(\boldsymbol{y})\|^2.$$

$\Longrightarrow$ Bound on the output variation when input is perturbed :

$$\|\Delta T(\boldsymbol{x})\|^2 \leq \|\boldsymbol{\delta x}\|^2 - \tfrac{1-\alpha}{\alpha}\|\Delta T(\boldsymbol{x}) - \boldsymbol{\delta x}\|^2$$

- In particular, as $\boldsymbol{\delta x} \to 0$, $\Delta T(\boldsymbol{x}) \to \boldsymbol{\delta x}$.
- As $\alpha \to 1$: nonexpansive.
- The smaller α is, the more stable T is.

Relation to generic deep neural networks

Feedforward architecture

$$T = R_{K-1} \circ (\boldsymbol{W}_{K-1} \cdot + \boldsymbol{b}_{K-1}) \circ \cdots \circ R_0 \circ (\boldsymbol{W}_0 \cdot + \boldsymbol{b}_0)$$

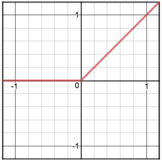
- $(R_k)_{0 \leq k \leq K-1}$ nonlinear activation functions
- $(\boldsymbol{W}_k)_{0 \leq k \leq K-1}$ linear operators (weight)
- $(\boldsymbol{b}_k)_{0 \leq k \leq K-1}$ vectors (bias parameters)

Relation to generic deep neural networks

Standard activation functions can be expressed as proximity operators

- **Rectified linear unit (ReLU)**

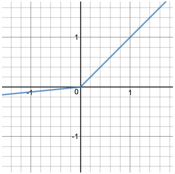
$$\varrho\colon \mathbb{R} \to \mathbb{R}\colon \xi \mapsto \begin{cases} \xi, & \text{if } \xi > 0; \\ 0, & \text{if } \xi \leq 0. \end{cases}$$



Then, $\varrho = \mathrm{proj}_{[0,+\infty[}$.

- **Parametric rectified linear unit (LeakyReLU)**

$$\varrho\colon \mathbb{R} \to \mathbb{R}\colon \xi \mapsto \begin{cases} \xi, & \text{if } \xi > 0; \\ \alpha\xi, & \text{if } \xi \leq 0 \end{cases}, \qquad \alpha \in]0,1].$$



Then $\varrho = \mathrm{prox}_\phi$ where

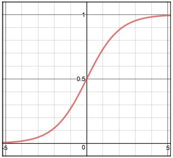
$$\phi\colon \mathbb{R} \to \mathbb{R}\colon \xi \mapsto \begin{cases} 0, & \text{if } \xi > 0; \\ (1/\alpha - 1)\xi^2/2, & \text{if } \xi \leq 0. \end{cases}$$

Relation to generic deep neural networks

Standard activation functions can be expressed as proximity operators

- **Unimodal sigmoid**

$$\varrho\colon \mathbb{R} \to \mathbb{R}\colon \xi \mapsto \frac{1}{1+e^{-\xi}} - \frac{1}{2}$$

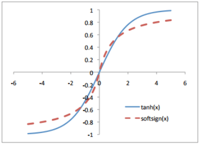


Then $\varrho = \operatorname{prox}_{\phi}$ where

$$\phi\colon \xi \mapsto \begin{cases} (\xi+1/2)\ln(\xi+1/2)+ \\ \qquad (1/2-\xi)\ln(1/2-\xi) - \dfrac{1}{2}(\xi^2+1/4) & \text{if } |\xi|<1/2; \\ -1/4, & \text{if } |\xi|=1/2; \\ +\infty, & \text{if } |\xi|>1/2. \end{cases}$$

- **Elliot activation function (SoftSign)** :

$$\varrho\colon \mathbb{R} \to \mathbb{R}\colon \xi \mapsto \frac{\xi}{1+|\xi|}.$$



We have $\varrho = \operatorname{prox}_{\phi}$, where

$$\phi\colon \mathbb{R} \to]-\infty,+\infty]\colon \xi \mapsto \begin{cases} -|\xi| - \ln(1-|\xi|) - \frac{\xi^2}{2}, & \text{if } |\xi|<1; \\ +\infty, & \text{if } |\xi| \geq 1. \end{cases}$$

Relation to generic deep neural networks

Standard activation functions can be expressed as proximity operators

- **Softmax**

$$R\colon \mathbb{R}^N \to \mathbb{R}^N\colon (\xi_k)_{1\le k\le N} \mapsto \left(\exp(\xi_k) \Bigg/ \sum_{j=1}^{N} \exp(\xi_j)\right)_{1\le k\le N} - u,$$

where $u = (1,\dots,1)/N \in \mathbb{R}^N$.
Then $R = \mathrm{prox}_\varphi$ where $\varphi = \psi(\cdot + u) + \langle \cdot \mid u\rangle$ and

$$\psi\colon \mathbb{R}^N \to]-\infty,+\infty]$$
$$(\xi_k)_{1\le k\le N} \mapsto \begin{cases} \displaystyle\sum_{k=1}^{N}\left(\xi_k \ln \xi_k - \frac{\xi_k^2}{2}\right), & \text{if } (\xi_k)_{1\le i\le N} \in [0,1]^N \text{ and } \displaystyle\sum_{k=1}^{N}\xi_k = 1;\\ +\infty, & \text{otherwise.}\end{cases}$$

Relation to generic deep neural networks

Quadratic problem $\underset{\boldsymbol{x}\in\mathcal{C}}{\text{minimize}}\ \frac{1}{2}\|\boldsymbol{H}\boldsymbol{x}-\boldsymbol{y}\|^2+\frac{\lambda}{2}\|\boldsymbol{D}\boldsymbol{x}\|^2$

Feedforward architecture $R_{K-1}\circ(\boldsymbol{W}_{K-1}\cdot+\boldsymbol{b}_{K-1})\circ\cdots\circ R_0\circ(\boldsymbol{W}_0\cdot+\boldsymbol{b}_0)$

iRestNet

$$\begin{aligned}\boldsymbol{x}_{k+1} &= \mathrm{prox}_{\gamma_k\mu_k\mathcal{B}}(\boldsymbol{x}_k-\gamma_k(\boldsymbol{H}^\top(\boldsymbol{H}\boldsymbol{x}_k-\boldsymbol{y})+\lambda_k\boldsymbol{D}^\top\boldsymbol{D}\boldsymbol{x}_k))\\ &= \mathrm{prox}_{\gamma_k\mu_k\mathcal{B}}\left([\mathbb{I}_n-\gamma_k(\boldsymbol{H}^\top\boldsymbol{H}+\lambda_k\boldsymbol{D}^\top\boldsymbol{D})]\boldsymbol{x}_k+\gamma_k\boldsymbol{H}^\top\boldsymbol{y}\right)\\ &= R_k(\boldsymbol{W}_k\boldsymbol{x}_k+\boldsymbol{b}_k)\end{aligned}$$

- $\boldsymbol{W}_k=\mathbb{I}_n-\gamma_k(\boldsymbol{H}^\top\boldsymbol{H}+\lambda\boldsymbol{D}^\top\boldsymbol{D})$ weight operator
- $\boldsymbol{b}_k=\gamma_k\boldsymbol{H}^\top\boldsymbol{y}$ bias parameter
- $R_k=\mathrm{prox}_{\gamma_k\mu_k\mathcal{B}}\to R_k$ **specific activation function**

Averageness result

Theorem 1 [Combettes and Pesquet, 2018]

Let $\alpha \in [1/2, 1]$. Let $K \geq 1$ be an integer. Let $W = W_{K-1} \circ \cdots \circ W_0$, let $\mu = \inf_{x\in\mathbb{R}^n,\, \|x\|=1} \langle Wx \mid x\rangle$, and let

$$\theta_{K-1} = \|W\| + \sum_{\ell=0}^{K-2} \sum_{0\leq j_0<\cdots<j_\ell\leq K-2} \|W_{K-1} \circ \cdots \circ W_{j_\ell+1}\|\|W_{j_\ell} \circ \cdots \circ W_{j_{\ell-1}+1}\| \cdots \|W_{j_0} \circ \cdots \circ W_0\|.$$

If one of the following conditions is satisfied :

(i) There exists $k \in \{0, \ldots, K-1\}$ such that $W_k = 0$;

(ii) $\|W - 2^K(1-\alpha)\mathbb{I}_n\| - \|W\| + 2\theta_{K-1} \leq 2^K\alpha$;

(iii) $\alpha \neq 1$, for every $k \in \{1, \ldots, K-1\}$ $W_k \neq 0$, and there exists $\eta \in [0, \alpha/((1-\alpha)\theta_{K-1})]$ such that

$$\begin{cases} \theta_{K-1} \leq 2^{K-1}\alpha \\ \alpha\theta_{K-1} + (1-\alpha)(\|\mathbb{I}_n - \eta W\| - \eta\|W\|)(\theta_{K-1} - \|W\|) \leq 2^{K-2}(2\alpha-1) + (1-\alpha)\mu, \end{cases}$$

then $T = R_{K-1} \circ (W_{K-1} \cdot + b_{K-1}) \circ \cdots \circ R_0 \circ (W_0 \cdot + b_0)$ is α-**averaged**.

Network stability result

Assumption

Consider the quadratic problem, assume that $H^\top H$ and $D^\top D$ are **diagonalizable in the same basis** $\mathcal{P}$.

Network stability result

Assumption

Consider the quadratic problem, assume that $H^\top H$ and $D^\top D$ are **diagonalizable in the same basis** $\mathcal{P}$.

Notation

For every $p \in \{1, \dots, n\}$ let $\beta_H^{(p)}$ and $\beta_D^{(p)}$ denote the p^{th} eigenvalue of $H^\top H$ and $D^\top D$ in $\mathcal{P}$, resp. Let β_- and β_+ be defined by

$$\beta_- = \min_{1 \le p \le n} \prod_{k=0}^{K-1} \left(1 - \gamma_k \left(\beta_H^{(p)} + \lambda_k \beta_D^{(p)}\right)\right) \text{ and } \beta_+ = \max_{1 \le p \le n} \prod_{k=0}^{K-1} \left(1 - \gamma_k \left(\beta_H^{(p)} + \lambda_k \beta_D^{(p)}\right)\right).$$

Let $\theta_{-1} = 1$ and, for every $k \in \{0, \dots, K-1\}$,

$$\theta_k = \sum_{l=0}^{k} \theta_{l-1} \max_{1 \le q_l \le n} \left| \left(1 - \gamma_k \left(\beta_H^{(q_l)} + \lambda_k \beta_D^{(q_l)}\right)\right) \dots \left(1 - \gamma_l \left(\beta_H^{(q_l)} + \lambda_l \beta_D^{(q_l)}\right)\right) \right|.$$

Network stability result

Assumption

Consider the quadratic problem, assume that $H^\top H$ and $D^\top D$ are **diagonalizable in the same basis** $\mathcal{P}$.

Notation

For every $p \in \{1, \dots, n\}$ let $\beta_H^{(p)}$ and $\beta_D^{(p)}$ denote the p^{th} eigenvalue of $H^\top H$ and $D^\top D$ in $\mathcal{P}$, resp. Let β_- and β_+ be defined by

$$\beta_- = \min_{1 \le p \le n} \prod_{k=0}^{K-1} \left(1 - \gamma_k \left(\beta_H^{(p)} + \lambda_k \beta_D^{(p)}\right)\right) \text{ and } \beta_+ = \max_{1 \le p \le n} \prod_{k=0}^{K-1} \left(1 - \gamma_k \left(\beta_H^{(p)} + \lambda_k \beta_D^{(p)}\right)\right).$$

Let $\theta_{-1} = 1$ and, for every $k \in \{0, \dots, K-1\}$,

$$\theta_k = \sum_{l=0}^{k} \theta_{l-1} \max_{1 \le q_l \le n} \left| \left(1 - \gamma_k \left(\beta_H^{(q_l)} + \lambda_k \beta_D^{(q_l)}\right)\right) \dots \left(1 - \gamma_l \left(\beta_H^{(q_l)} + \lambda_l \beta_D^{(q_l)}\right)\right) \right|.$$

Theorem 2

Let $\alpha \in [1/2, 1]$. If one of the following conditions is satisfied :

(i) $\beta_+ + \beta_- \le 0$ and $\theta_{K-1} \le 2^{K-1}(2\alpha - 1)$;

(ii) $0 \le \beta_+ + \beta_- \le 2^{K+1}(1-\alpha)$ and $2\theta_{K-1} \le \beta_+ + \beta_- + 2^K(2\alpha - 1)$;

(iii) $2^{K+1}(1-\alpha) \le \beta_+ + \beta_-$ and $\theta_{K-1} \le 2^{K-1}$,

then the operator $R_{K-1} \circ (W_{K-1} \cdot + b_{K-1}) \circ \cdots \circ R_0 \circ (W_0 \cdot + b_0)$ is α-**averaged**.

Numerical experiments

Image deblurring

$$y = H\overline{x} + \omega$$

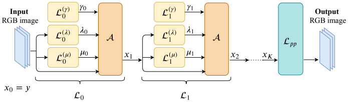
- $H \in \mathbb{R}^n \times \mathbb{R}^n$: circular convolution with known blur
- $\omega \in \mathbb{R}^n$: additive white Gaussian noise with standard deviation σ
- $y \in \mathbb{R}^n$, $\overline{x} \in \mathbb{R}^n$: RGB images

Variational formulation

$$\underset{x \in \mathcal{C}}{\text{minimize}} \quad \frac{1}{2}\|Hx - y\|^2 + \lambda \sum_{i=1}^{n} \sqrt{\frac{(D_\mathrm{h}x)_i^2 + (D_\mathrm{v}x)_i^2}{\delta^2} + 1}$$

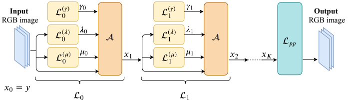
- $\mathcal{C} = \{x \in \mathbb{R}^n \mid (\forall i \in \{1, \ldots, n\}) \;\; x_{\min} \leq x_i \leq x_{\max}\}$
- δ : smoothing parameter, $\delta = 0.01$ for iRestNet
- $D_\mathrm{h} \in \mathbb{R}^{n \times n}$, $D_\mathrm{v} \in \mathbb{R}^{n \times n}$: horizontal and vertical spatial gradient operators

Network characteristics



- Number of layers : $K = 40$

Network characteristics



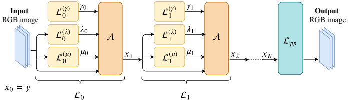
- Number of layers : $K = 40$
- **Estimation of regularization parameter**

$$\lambda_k = \mathcal{L}_k^{(\lambda)}(x_k) = \frac{\widehat{\sigma}(y) \times \mathrm{Softplus}(\mathrm{b_k})}{\eta(x_k) + \mathrm{Softplus}(\mathrm{c_k})}$$

- $\eta(x_k)$: standard deviation of $[(D_\mathrm{h} x_k)^\top (D_\mathrm{v} x_k)^\top]^\top$
- Estimation of noise level **[Ramadhan *et al.*,2017]**, $\widehat{\sigma}(y) = \mathrm{median}(|\mathrm{W_H y}|)/0.6745$
- $|W_\mathrm{H} y|$: vector gathering the absolute value of the diagonal coefficients of the first level Haar wavelet decomposition of the blurred image

$\rightarrow$ iRestNet does not require knowledge of noise level

Network characteristics



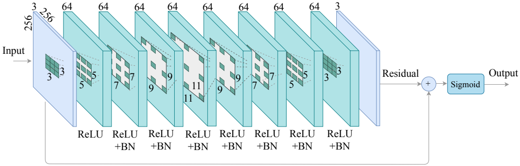
- Number of layers : $K = 40$
- **Estimation of regularization parameter**

$$\lambda_k = \mathcal{L}_k^{(\lambda)}(x_k) = \frac{\widehat{\sigma}(y) \times \mathrm{Softplus}(\mathrm{b_k})}{\eta(x_k) + \mathrm{Softplus}(\mathrm{c_k})}$$

  - $\eta(x_k)$: standard deviation of $[(D_\mathrm{h} x_k)^\top (D_\mathrm{v} x_k)^\top]^\top$
  - Estimation of noise level **[Ramadhan *et al.*,2017]**, $\widehat{\sigma}(y) = \mathrm{median}(|\mathrm{W_H y}|)/0.6745$
  - $|W_\mathrm{H} y|$: vector gathering the absolute value of the diagonal coefficients of the first level Haar wavelet decomposition of the blurred image

  $\rightarrow$ iRestNet does not require knowledge of noise level

- Post-processing $\mathcal{L}_\mathrm{pp}$ **[Zhang *et al.*,2017]**



Numerical experiments

Dataset

- Training set : 200 RGB images from BSD500 + 1000 images from COCO
- Validation set : 100 validation images from BSD500
- Test sets : 200 test images from BSD500, Flickr30 test set (30 images)

Numerical experiments

Dataset

- Training set : 200 RGB images from BSD500 + 1000 images from COCO
- Validation set : 100 validation images from BSD500
- Test sets : 200 test images from BSD500, Flickr30 test set (30 images)

Test configurations

- GaussianA : Gaussian kernel with std=1.6, $\sigma = 0.008$
- GaussianB : Gaussian kernel with std=1.6, $\sigma \in [0.01, 0.05]$
- GaussianC : Gaussian kernel with std=3, $\sigma = 0.04$
- Motion : motion kernel from **[Levin *et al.*,2009]** $\sigma = 0.01$
- Square : 7×7 square kernel, $\sigma = 0.01$

Numerical experiments

Dataset

- Training set : 200 RGB images from BSD500 + 1000 images from COCO
- Validation set : 100 validation images from BSD500
- Test sets : 200 test images from BSD500, Flickr30 test set (30 images)

Test configurations

- GaussianA : Gaussian kernel with std=1.6, $\sigma = 0.008$
- GaussianB : Gaussian kernel with std=1.6, $\sigma \in [0.01, 0.05]$
- GaussianC : Gaussian kernel with std=3, $\sigma = 0.04$
- Motion : motion kernel from **[Levin *et al.*,2009]** $\sigma = 0.01$
- Square : 7×7 square kernel, $\sigma = 0.01$

Training

- Loss : Structural SImilarity Measure (SSIM) **[Wang *et al.*, 2004]**, ADAM optimizer
- $\mathcal{L}_0, \ldots, \mathcal{L}_{29}$ trained individually, $\mathcal{L}_{\mathrm{pp}} \circ \mathcal{L}_{39} \circ \cdots \circ \mathcal{L}_{30}$ trained end-to-end $\rightarrow$ low memory
- Implemented with Pytorch using a GPU, $\sim$3-4 days per training

Numerical experiments

Competitors

Variational approach

- VAR : solution to $\mathcal{P}_0$ with projected gradient algorithm, (λ, δ) leading to best SSIM.

Machine learning approaches

- EPLL **[Zoran and Weiss, 2011]** : Bayesian approach, Gaussian mixture model with learned parameters, deblurred image = MAP estimate
- MLP **[Schuler *et al.*,2013]** : Multi-Layer Perceptron network fed with a pre-deconvolved image produced by a Wiener deconvolution filter.

Machine learning approaches based on deep unfolding

- IRCNN **[Zhang *et al.*,2017]** (require noise level) : empirical algorithm derived from an augmented Lagrangian formulation and unfolded over 30 iterations, CNN is used as a denoiser to update the splitting variable.
- PDHG **[Meinhardt *et al.*, 2017]** : maximum of 30 iterations of a primal dual hybrid gradient algorithm, proximity operator of the second regularization function replaced by a NN.
- FCNN **[J. Zhang *et al.*, 2017]** : unfolded algorithm, regularization function learned by a NN.

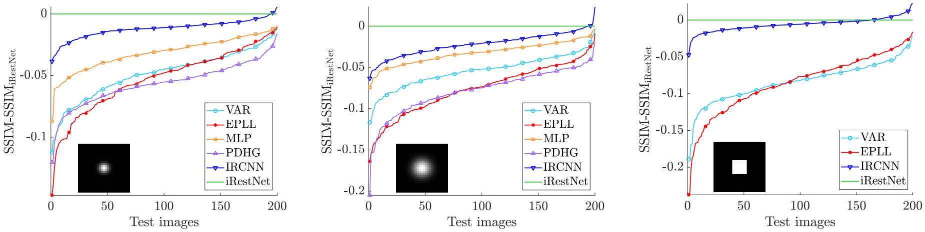
$\rightarrow$ MLP, EPLL and IRCNN require knowledge of noise level $\rightarrow$ for GaussianB use noise standard deviation estimation given by **[Mallat, 1999, Section 11.3.1]**.

Results

- ✓ Higher average SSIM than competitors
- ✓ Higher SSIM on almost all images

	GaussianA	GaussianB	GaussianC	Motion	Square
Blurred	0.676	0.526	0.326	0.549	0.544
VAR	0.804	0.723	0.587	0.829	0.756
EPLL **[Zoran and Weiss, 2011]**	0.800	0.708	0.565	0.839	0.755
MLP **[Schuler *et al.*, 2016]**	0.821	0.734	0.608	n/a	n/a
PDHG **[Meinhardt *et al.*, 2017]**	0.796	0.716	0.563	n/a	n/a
IRCNN **[K. Zhang *et al.*, 2017]**	0.841	0.768	0.619	0.907	0.834
FCNN **[J. Zhang *et al.*, 2017]**	n/a	n/a	n/a	0.847	n/a
iRestNet	**0.853**	**0.787**	**0.641**	**0.910**	**0.840**

TABLE – SSIM results on the BSD500 test set.



✓ Short execution time : $\sim$ 1.4 sec per image

✓ Similar performance on a different test set

	GaussianA	GaussianB	GaussianC	Motion	Square
Blurred	0.723	0.545	0.355	0.590	0.579
VAR	0.857	0.776	0.639	0.869	0.818
EPLL **[Zoran and Weiss, 2011]**	0.860	0.770	0.616	0.887	0.827
MLP **[Schuler *et al.*, 2016]**	0.874	0.798	0.668	n/a	n/a
PDHG **[Meinhardt *et al.*, 2017]**	0.853	0.781	0.623	n/a	n/a
IRCNN **[K. Zhang *et al.*, 2017]**	0.885	0.819	0.676	**0.930**	**0.886**
FCNN **[J. Zhang *et al.*, 2017]**	n/a	n/a	n/a	0.890	n/a
iRestNet	**0.892**	**0.833**	**0.696**	**0.930**	**0.886**

TABLE – SSIM results on the Flickr30 test set.

Visual results

✓ Better contrast and more details

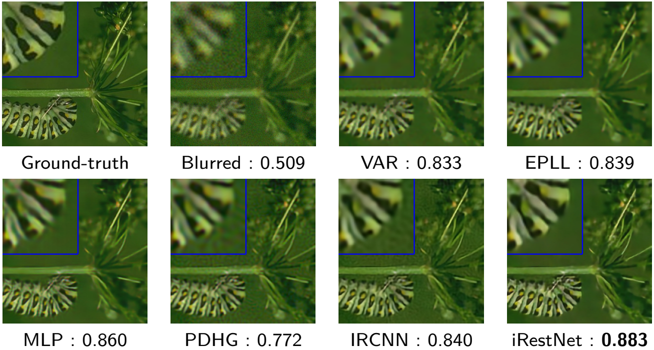
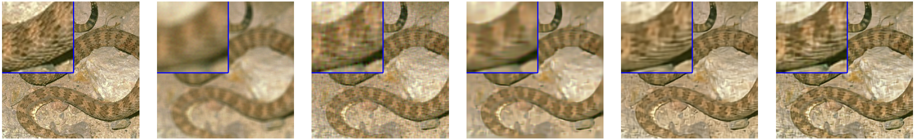


FIGURE – Visual results and SSIM obtained on one image from the BSD500 test set degraded with GaussianB.

Visual results



Ground-truth Blurred : 0.344 VAR : 0.622 EPLL : 0.553 IRCNN : 0.685 iRestNet : **0.713**

FIGURE – Visual results and SSIM obtained on one image from the BSD500 test set degraded with Square.

Ground-truth Blurred : 0.576 VAR : 0.844 EPLL : 0.849

IRCNN : 0.906 FCNN : 0.856 iRestNet : **0.909**

FIGURE – Visual results and SSIM obtained on one image from the Flickr30 test set degraded with Motion.

Conclusion

- Novel architecture based on an unfolded proximal interior point algorithm
- Allows to apply hard constraints on the image
- Expression and gradient of the proximity operator of the barrier

$\rightarrow$ Different application (classification, ...)

$\rightarrow$ When degradation is unkown : blind or semi-blind deconvolution

Codes

`https://mccorbineau.github.io`
`https://github.com/mccorbineau/iRestNet`

Related publications

iRestNet

C. Bertocchi, E. Chouzenoux, M.-C. Corbineau, J.-C. Pesquet, M. Prato

Deep unfolding of a proximal interior point method for image restoration

To appear in Inverse Problems, 2019.

Network stability

P. L. Combettes and J.-C. Pesquet.

Deep neural network structures solving variational inequalities

https://arxiv.org/abs/1808.07526.

Proximal interior point methods

M.-C. Corbineau, E. Chouzenoux and J.-C. Pesquet.

PIPA : a new proximal interior point algorithm for large-scale convex optimization.

Proceedings of the 20th IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 2018.

M.-C. Corbineau, E. Chouzenoux and J.-C. Pesquet.

Geometry-texture decomposition/reconstruction using a proximal interior point algorithm

Proceedings of the 10th IEEE Sensor Array and Multichannel Signal Processing Workshop (SAM), 2018.

E. Chouzenoux, M.-C. Corbineau and J.-C. Pesquet.

A proximal interior point algorithm with applications to image processing

To appear in Journal of Mathematical Imaging and Vision, 2019.

Thank you !