Proximal Neural Networks: Marrying Variational Methods and Artificial Intelligence

I – Introduction

Audrey Repetti[†] , Nelly Pustelnik[◇], Jean-Christophe Pesquet[*]

[*] CentraleSupélec, Université Paris-Saclay, Inria, Gif-sur-Yvettes, France
[◇] CNRS, ENS Lyon, Lyon, France
[†] Heriot-Watt University & Maxwell Institute for Mathematical Sciences, Edinburgh, UK

Tutorial – Eusipco 2025 – Palermo, Italy

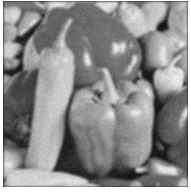
Motivation

⋆ FORWARD MODEL: $\boldsymbol{z} = \mathcal{D}(\mathbf{A}\overline{\boldsymbol{x}})$

- $\overline{\boldsymbol{x}} \in \mathbb{R}^N$: original signal
- $\mathbf{A}: \mathbb{R}^N \to \mathbb{R}^M$: linear measurement operator
- $\mathcal{D}: \mathbb{R}^M \to \mathbb{R}^M$: noise corruption process (e.g., addition of Gaussian noise)

OBJECTIVE: Find an estimate $\widehat{\boldsymbol{x}}$ of $\overline{\boldsymbol{x}}$ from $\boldsymbol{z}$

⋆ EXAMPLE: Image restoration (e.g., deblurring)



Observation

Estimate

Motivation

⋆ FORWARD MODEL: $\boldsymbol{z} = \mathcal{D}(\mathbf{A}\overline{\boldsymbol{x}})$

- $\overline{\boldsymbol{x}} \in \mathbb{R}^N$: original signal
- $\mathbf{A}: \mathbb{R}^N \to \mathbb{R}^M$: linear measurement operator
- $\mathcal{D}: \mathbb{R}^M \to \mathbb{R}^M$: noise corruption process (e.g., addition of Gaussian noise)

OBJECTIVE: Find an estimate $\widehat{\boldsymbol{x}}$ of $\overline{\boldsymbol{x}}$ from $\boldsymbol{z}$

⋆ EXAMPLE: Medical imaging (CT)

Observation

Estimate

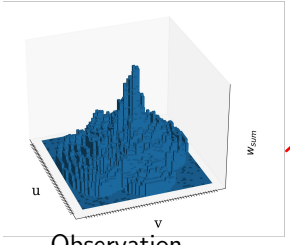
Motivation

⋆ FORWARD MODEL: $\boldsymbol{z} = \mathcal{D}(\mathbf{A}\overline{\boldsymbol{x}})$

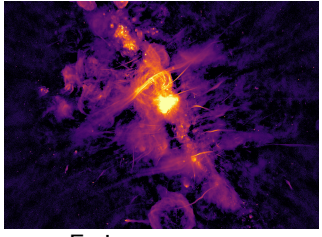
- $\overline{\boldsymbol{x}} \in \mathbb{R}^N$: original signal
- $\mathbf{A}: \mathbb{R}^N \to \mathbb{R}^M$: linear measurement operator
- $\mathcal{D}: \mathbb{R}^M \to \mathbb{R}^M$: noise corruption process (e.g., addition of Gaussian noise)

OBJECTIVE: Find an estimate $\widehat{\boldsymbol{x}}$ of $\overline{\boldsymbol{x}}$ from $\boldsymbol{z}$

⋆ EXAMPLE: Radio-interferometric imaging in astronomy



Observation



Estimate

Motivation: Case of Gaussian denoising

FORWARD MODEL: $\boldsymbol{z} = \overline{\boldsymbol{x}} + \boldsymbol{w}$ with $\boldsymbol{w} \sim \mathcal{N}(0, \sigma^2 \mathrm{Id})$

OBJECTIVE : Find an estimate $\widehat{\boldsymbol{x}}$ of $\overline{\boldsymbol{x}}$ from $\boldsymbol{z}$

IDEA : Use wavelet denoising

- Wavelets: sparse representation of most natural signals.
- Filterbank implementation of dyadic wavelet transform: $\boldsymbol{\Psi} \in \mathbb{R}^{N \times N}$.

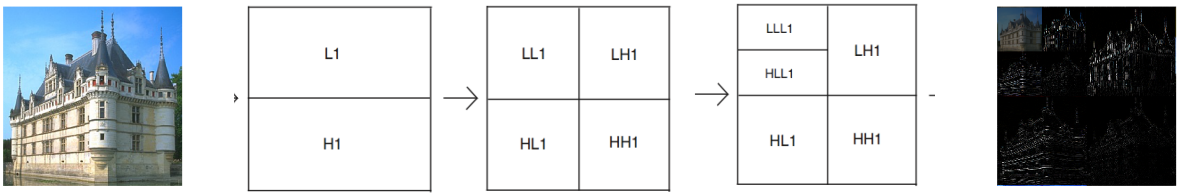
Motivation: Case of Gaussian denoising

FORWARD MODEL: $\boldsymbol{z} = \overline{\boldsymbol{x}} + \boldsymbol{w}$ with $\boldsymbol{w} \sim \mathcal{N}(0, \sigma^2 \mathrm{Id})$

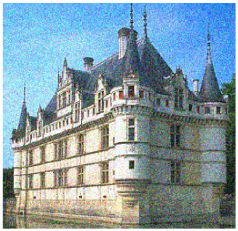
OBJECTIVE : Find an estimate $\widehat{\boldsymbol{x}}$ of $\overline{\boldsymbol{x}}$ from $\boldsymbol{z}$

IDEA : Use wavelet denoising

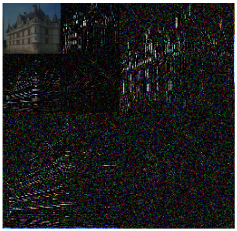
- Wavelets: sparse representation of most natural signals.
- Filterbank implementation of dyadic wavelet transform: $\boldsymbol{\Psi} \in \mathbb{R}^{N \times N}$.



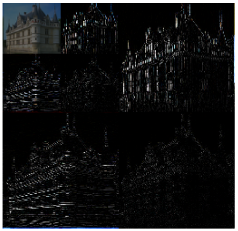
From wavelet denoising to proximity operators



$\boldsymbol{z}$



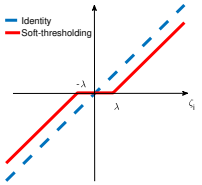
$\boldsymbol{\alpha} = \boldsymbol{\Psi} \boldsymbol{z}$



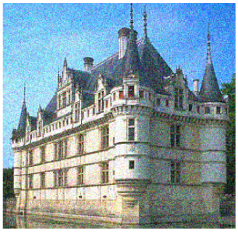
$\text{soft}_\lambda(\boldsymbol{\Psi} \boldsymbol{z})$

$\widehat{\boldsymbol{x}} = \boldsymbol{\Psi}^* \text{soft}_\lambda(\boldsymbol{\Psi} \boldsymbol{z})$

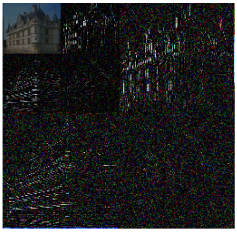
$$\begin{aligned} \text{soft}_\lambda(\boldsymbol{\alpha}) &= \left(\max\{|\alpha_i| - \lambda, 0\} \operatorname{sign}(\alpha_i) \right)_{i \in \Omega} \\ &= \underset{\boldsymbol{\nu}}{\operatorname{argmin}} \ \frac{1}{2} \|\boldsymbol{\nu} - \boldsymbol{\alpha}\|_2^2 + \lambda \|\boldsymbol{\nu}\|_1 \\ &= \text{prox}_{\lambda \|\cdot\|_1}(\boldsymbol{\alpha}) \quad \rightarrow \text{proximity operator} \end{aligned}$$



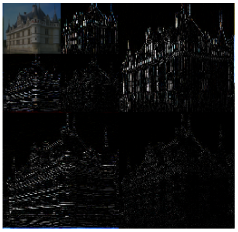
From wavelet denoising to proximity operators



$\boldsymbol{z}$



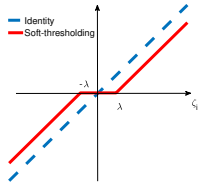
$\boldsymbol{\alpha} = \boldsymbol{\Psi} \boldsymbol{z}$



$\text{soft}_\lambda(\boldsymbol{\Psi} \boldsymbol{z})$

$\widehat{\boldsymbol{x}} = \boldsymbol{\Psi}^* \text{soft}_\lambda(\boldsymbol{\Psi} \boldsymbol{z})$

$$\begin{aligned} \text{soft}_\lambda(\boldsymbol{\alpha}) &= \big(\max\{|\alpha_i| - \lambda, 0\} \operatorname{sign}(\alpha_i)\big)_{i \in \Omega} \\ &= \underset{\boldsymbol{\nu}}{\operatorname{argmin}} \ \frac{1}{2}\|\boldsymbol{\nu} - \boldsymbol{\alpha}\|_2^2 + \lambda \|\boldsymbol{\nu}\|_1 \\ &= \text{prox}_{\lambda \|\cdot\|_1}(\boldsymbol{\alpha}) \quad \rightarrow \text{proximity operator} \end{aligned}$$



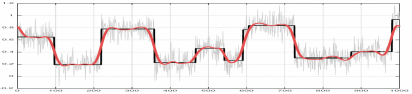
$$\boldsymbol{\Psi}^* \text{soft}_\lambda(\boldsymbol{\Psi} \boldsymbol{z}) = \underset{\boldsymbol{x}}{\operatorname{argmin}} \ \tfrac{1}{2}\|\boldsymbol{x} - \boldsymbol{z}\|_2^2 + \lambda \|\boldsymbol{\Psi} \boldsymbol{z}\|_1 = \text{prox}_{\lambda \|\boldsymbol{\Psi} \cdot\|_1}(\boldsymbol{z})$$

Piecewise constant denoising

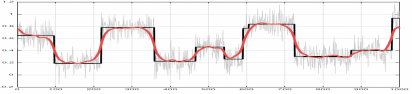
Observation model: $\boldsymbol{z} = \overline{\boldsymbol{x}} + \boldsymbol{w}$ with $\boldsymbol{w} \in \mathbb{R}^N$ realisation of $\mathcal{N}(0, \sigma^2 \mathrm{Id})$

Minimization problem: $\widehat{\boldsymbol{x}}(\boldsymbol{z}; \widehat{\lambda}) = \underset{\boldsymbol{x} \in \mathbb{R}^N}{\operatorname{argmin}} \ \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{z}\|_2^2 + \lambda \, \|\boldsymbol{\Psi}\boldsymbol{x}\|_{\bullet}^{\bullet}$ where $\begin{cases} \boldsymbol{\Psi}\boldsymbol{x} = \psi * \boldsymbol{x} \\ \lambda > 0 \end{cases}$

Linear denoising



$\psi = \begin{bmatrix} 1 & -1 \end{bmatrix}$; $\|\cdot\|_{\bullet}^{\bullet} = \|\cdot\|_2^2$; Large λ



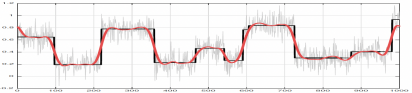
$\psi = \begin{bmatrix} 1 & -1 \end{bmatrix}$; $\|\cdot\|_{\bullet}^{\bullet} = \|\cdot\|_2^2$; Small λ

Piecewise constant denoising

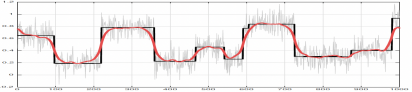
Observation model: $\boldsymbol{z} = \overline{\boldsymbol{x}} + \boldsymbol{w}$ with $\boldsymbol{w} \in \mathbb{R}^N$ realisation of $\mathcal{N}(0, \sigma^2 \mathrm{Id})$

Minimization problem: $\widehat{\boldsymbol{x}}(\boldsymbol{z}; \widehat{\lambda}) = \underset{\boldsymbol{x} \in \mathbb{R}^N}{\operatorname{argmin}} \ \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{z}\|_2^2 + \lambda \, \|\boldsymbol{\Psi}\boldsymbol{x}\|_{\bullet}^{\bullet}$ where $\begin{cases} \boldsymbol{\Psi}\boldsymbol{x} = \psi * \boldsymbol{x} \\ \lambda > 0 \end{cases}$

Linear denoising

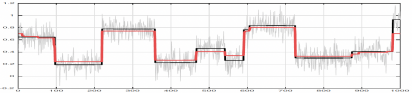


$\psi = [1 \quad -1]$; $\|\cdot\|_{\bullet}^{\bullet} = \|\cdot\|_2^2$; Large λ

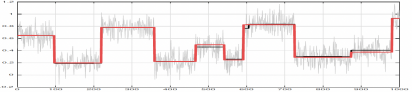


$\psi = [1 \quad -1]$; $\|\cdot\|_{\bullet}^{\bullet} = \|\cdot\|_2^2$; Small λ

Non-linear denoising



$\psi = [1 \quad -1]$ and $\|\cdot\|_{\bullet}^{\bullet} = \|\cdot\|_1$



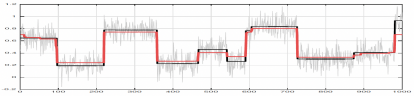
$\psi = [1 \quad -1]$ and $\|\cdot\|_{\bullet}^{\bullet} = \|\cdot\|_0$

Piecewise linear denoising

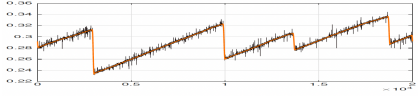
OBSERVATION MODEL: $\boldsymbol{z} = \overline{\boldsymbol{x}} + \boldsymbol{w}$ with $\boldsymbol{w} \in \mathbb{R}^N$ realisation of $\mathcal{N}(0, \sigma^2 \mathrm{Id}\,)$

MINIMIZATION PROBLEM: $\widehat{\boldsymbol{x}}(\boldsymbol{z}; \widehat{\lambda}) = \arg\min_{x \in \mathbb{R}^N} \frac{1}{2}\|x - \boldsymbol{z}\|_2^2 + \lambda \, \|\Psi x\|_{\bullet}^{\bullet}$ where $\begin{cases} \Psi x = \psi * x \\ \lambda > 0 \end{cases}$

NON-LINEAR DENOISING: PIECEWISE CONSTANT/LINEAR



$\psi = \begin{bmatrix} 1 & -1 \end{bmatrix}$ and $\|\cdot\|_{\bullet}^{\bullet} = \|\cdot\|_1$



$\psi = \begin{bmatrix} 1 & -2 & 1 \end{bmatrix}$ and $\|\cdot\|_{\bullet} = \|\cdot\|_1$

Inverse problems: Variational formulation

- ⋆ FORWARD MODEL: $\boldsymbol{z} = \mathcal{D}\big(\mathbf{A}\overline{\boldsymbol{x}}\big)$
- ⋆ VARIATIONAL APPROACH: Find $\widehat{\boldsymbol{x}} \in \underset{\boldsymbol{x}\in\mathcal{H}}{\operatorname{Argmin}}\ h_{\boldsymbol{z}}(\boldsymbol{x}) + \lambda g(\boldsymbol{x})$
  - $h_{\boldsymbol{z}}$ data fidelity term
  - $\lambda > 0$ and g regularization term (e.g., TV or ℓ_1 in a wavelet domain)
  - $\mathcal{H} = \mathbb{R}^N$ underlying signal space

EXAMPLES:

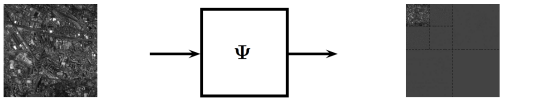
- ⋆ Gaussian noise: $h_{\boldsymbol{z}}(\boldsymbol{x}) = \frac{1}{2}\|\mathbf{A}\boldsymbol{x} - \boldsymbol{z}\|^2$
- ⋆ Poisson noise: $h_{\boldsymbol{z}}(\boldsymbol{x}) = \sum_{m=1}^{M} \left([\mathbf{A}\boldsymbol{x}]_m - z_m \log([\mathbf{A}\boldsymbol{x}]_m)\right)$
- ⋆ Energy-bounded noise: $h_{\boldsymbol{z}}(\boldsymbol{x}) = \iota_{\mathcal{B}_2(\boldsymbol{z},\epsilon)}\big(\mathbf{A}\boldsymbol{x}\big) = \begin{cases} 0 & \text{if } \mathbf{A}\boldsymbol{x} \in \mathcal{B}_2(\boldsymbol{z},\epsilon) \\ +\infty & \text{otherwise.} \end{cases}$

Inverse problems: Variational formulation

- ⋆ Forward model: $\boldsymbol{z} = \mathcal{D}(\mathbf{A}\overline{\boldsymbol{x}})$
- ⋆ Variational approach: Find $\widehat{\boldsymbol{x}} \in \underset{\boldsymbol{x}\in\mathcal{H}}{\operatorname{Argmin}} \; h_{\boldsymbol{z}}(\boldsymbol{x}) + \lambda g(\boldsymbol{x})$
  - $h_{\boldsymbol{z}}$ data fidelity term
  - $\lambda > 0$ and g regularization term (e.g., TV or ℓ_1 in a wavelet domain)
  - $\mathcal{H} = \mathbb{R}^N$ underlying signal space

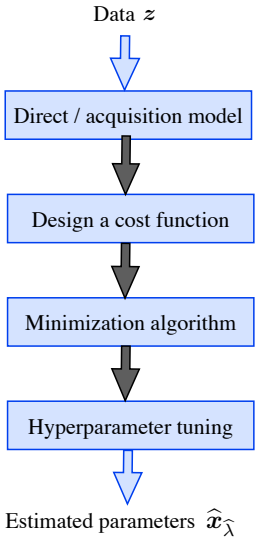
Examples of regularisation terms

- ⋆ Admissibility constraints: $g(\boldsymbol{x}) = \sum_{l=1}^{L} \iota_{C_l}(\boldsymbol{x})$
- ⋆ ℓ_1 norm (analysis approach) $g(\boldsymbol{x}) = \sum_{l=1}^{L} |[\boldsymbol{\Psi x}]_l| = \|\boldsymbol{\Psi x}\|_1$



Signal x | Frame decomposition operator | Frame coefficients

Variational pipeline



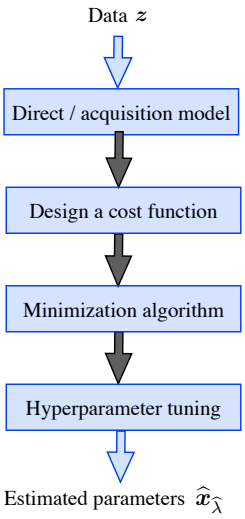
$$\boldsymbol{z} = \mathcal{D}(\mathrm{A}\boldsymbol{x})$$

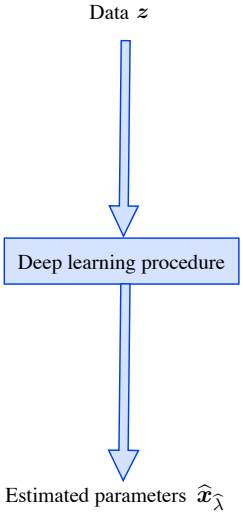
$$\text{Find } \widehat{\boldsymbol{x}}_\lambda \in \mathrm{Argmin}_{\boldsymbol{x}} h_{\boldsymbol{z}}(\boldsymbol{x}) + \lambda g(\boldsymbol{x})$$

$$\text{Sequence } \boldsymbol{x}^{[k+1]} = \mathfrak{T}_{\Theta_k}(\boldsymbol{x}^{[k]})$$

$$\widehat{\lambda} \in \mathrm{Argmin}_\lambda \|\overline{\boldsymbol{x}} - \widehat{\boldsymbol{x}}_\lambda\|_2^2$$

Deep learning pipeline





From optimisation to deep learning: Construction, advantages and challenges

Reconstruction methods: From variational to deep learning

IDEA: Define $\boldsymbol{x}^{[K]} = \mathfrak{T}_{\Theta_K}\Big(\mathfrak{T}_{\Theta_{K-1}}\Big(\ldots \mathfrak{T}_{\Theta_1}(\boldsymbol{x}^{[0]})\Big)\Big)$ such that $\boldsymbol{x}^{[K]} \approx \widehat{\boldsymbol{x}}$

- $K \to +\infty$ Variational-based methods (e.g., proximal algorithms)
  EXAMPLES: *Forward-backward, Douglas-Rachford, ADMM, Primal-dual Condat-Vũ, etc.*
- $K \ll +\infty$ Deep learning-based methods (e.g., end-to-end network)
  EXAMPLES: *MLP, DnCNN, Unet, Auto-Encoder, GANs, etc.*

Reconstruction methods: From variational to deep learning

IDEA: Define $\boldsymbol{x}^{[K]} = \mathfrak{T}_{\Theta_K}\left(\mathfrak{T}_{\Theta_{K-1}}\left(\ldots \mathfrak{T}_{\Theta_1}(\boldsymbol{x}^{[0]})\right)\right)$ such that $\boldsymbol{x}^{[K]} \approx \widehat{\boldsymbol{x}}$

- $K \to +\infty$ Variational-based methods (e.g., proximal algorithms)
  and hybrid methods (plug-and-play algorithms)
- $K \ll +\infty$ Deep learning-based methods (e.g., end-to-end network)
  and hybrid methods (unfolded networks)

Reconstruction methods: From variational to deep learning

IDEA: Define $\boldsymbol{x}^{[K]} = \mathfrak{T}_{\Theta_K}\left(\mathfrak{T}_{\Theta_{K-1}}\left(\ldots \mathfrak{T}_{\Theta_1}(\boldsymbol{x}^{[0]})\right)\right)$ such that $\boldsymbol{x}^{[K]} \approx \widehat{\boldsymbol{x}}$

- $K \to +\infty$ Variational-based methods (e.g., proximal algorithms)
  and hybrid methods (plug-and-play algorithms)
- $K \ll +\infty$ Deep learning-based methods (e.g., end-to-end network)
  and hybrid methods (unfolded networks)

KEY FACT: $\mathfrak{T}_{\Theta_k}$ has same form for all cases: $\mathfrak{T}_{\Theta_k} : \mathbb{R}^{N_{k-1}} \to \mathbb{R}^{N_k} : \boldsymbol{x} \mapsto \mathfrak{D}_{\Lambda_k}(\mathbf{L}_k \boldsymbol{x} + \boldsymbol{b}_k)$

- $\mathbf{L}_k : \mathbb{R}^{N_{k-1}} \to \mathbb{R}^{N_k}$ is a linear operator (e.g., sparsity or measurement operators)
- $\boldsymbol{b}_k \in \mathbb{R}^{N_k}$ is a bias (e.g., link to data/observations $\boldsymbol{z}$)
- $\mathfrak{D}_{\Lambda_k} : \mathbb{R}^{N_k} \to \mathbb{R}^{N_k}$ is a nonlinear operator (e.g., thresholding, ReLU/positivity projection)

From variational methods...

VARIATIONAL APPROACH: Find $\widehat{\boldsymbol{x}} \in \underset{\boldsymbol{x}\in\mathbb{R}^N}{\operatorname{Argmin}} \left\{ f(\boldsymbol{x}) = h(\boldsymbol{x}) + g(\boldsymbol{x}) \right\}$

VARIATIONAL FORMULATION: Find $\widehat{\boldsymbol{x}} \in \mathbb{R}^N$ such that $0 \in \partial f(\widehat{\boldsymbol{x}}) = \partial h(\widehat{\boldsymbol{x}}) + \partial g(\widehat{\boldsymbol{x}})$

IDEA: Choose your *favourite* (fixed-point) iterative (proximal) algorithm:

$$(\forall k \in \mathbb{N}) \quad \boldsymbol{x}^{[k+1]} = \mathfrak{T}(\boldsymbol{x}^{[k]})$$

From variational methods...

VARIATIONAL APPROACH: Find $\widehat{\boldsymbol{x}} \in \underset{\boldsymbol{x}\in\mathbb{R}^N}{\operatorname{Argmin}} \left\{ f(\boldsymbol{x}) = h(\boldsymbol{x}) + g(\boldsymbol{x}) \right\}$

VARIATIONAL FORMULATION: Find $\widehat{\boldsymbol{x}} \in \mathbb{R}^N$ such that $0 \in \partial f(\widehat{\boldsymbol{x}}) = \partial h(\widehat{\boldsymbol{x}}) + \partial g(\widehat{\boldsymbol{x}})$

IDEA: Choose your *favourite* (fixed-point) iterative (proximal) algorithm:

$$(\forall k \in \mathbb{N}) \quad \boldsymbol{x}^{[k+1]} = \mathfrak{T}(\boldsymbol{x}^{[k]})$$

EXAMPLE: Proximal gradient algorithm

$$\mathfrak{T} = \operatorname{prox}_{\gamma h}(\mathrm{Id} - \gamma \nabla g)$$

PROXIMITY OPERATOR: $\boldsymbol{x}^* = \operatorname{prox}_{\gamma g}(\boldsymbol{u}) = \underset{\boldsymbol{x}}{\operatorname{argmin}} \ \gamma g(\boldsymbol{x}) + \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{u}\|^2$

$\rightsquigarrow$ $\boldsymbol{x}^*$ is a MAP estimator for the Gaussian denoising problem $\boldsymbol{u} = \boldsymbol{x} + \boldsymbol{w}$

... to plug-and-play methods (and more powerfull regularizers)

PnP methods: Replace prox by **powerful regularizer/denoiser**, hand-crafted (e.g., BM3D) or learned (e.g., neural network)

Example:
- FB: $(\forall k \in \mathbb{N})\ \boldsymbol{x}^{[k+1]} = \mathrm{prox}_{\gamma g}\left(\boldsymbol{x}^{[k]} - \gamma \nabla h(\boldsymbol{x}^{[k]})\right)$
- PnP-FB: $(\forall k \in \mathbb{N})\ \boldsymbol{x}^{[k+1]} = \mathfrak{D}_{\Lambda}\left(\boldsymbol{x}^{[k]} - \gamma \nabla h(\boldsymbol{x}^{[k]})\right)$ where $\mathfrak{D}_{\Lambda}$ is a **regularizer/denoiser**

... to plug-and-play methods (and more powerfull regularizers)

PnP methods: Replace prox by **powerful regularizer/denoiser**, hand-crafted (e.g., BM3D) or learned (e.g., neural network)

Example: PnP-FB: $(\forall k \in \mathbb{N})\ \boldsymbol{x}^{[k+1]} = \mathfrak{D}_\Lambda\left(\boldsymbol{x}^{[k]} - \gamma \nabla h(\boldsymbol{x}^{[k]})\right)$ where $\mathfrak{D}_\Lambda$ is a **regularizer/denoiser**

Remarks:

How to build reliable PnP methods?

PnP iterations: Can we use any scheme?

NN architectures: Can we use any denoising NN?

Theoretical understanding (for reliable decision making processes)?

Asymptotic convergence: Does $(\boldsymbol{x}^{[k]})_{k\in\mathbb{N}}$ still converge?

Characterisation of the limit point: If $(\boldsymbol{x}^{[k]})_{k\in\mathbb{N}}$ converges to $\widehat{\boldsymbol{x}}$, what is $\widehat{\boldsymbol{x}}$?

See, e.g., [Hasannasab *et al.*, 2020], [Terris *et al.*, 2020], [Cohen *et al.*, 2021], [Pesquet *et al.*, 2021], [Hurault *et al.*, 2022], [Laumont *et al.*, 2022], ...

... and to unfolded (end-to-end) networks

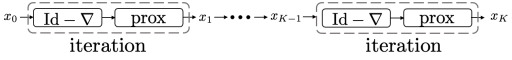
UNFOLDED METHODS: Unroll your favourite iterative (proximal) algorithm over a **fixed** number of iterations, and learn parameters inside

... and to unfolded (end-to-end) networks

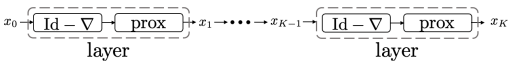
UNFOLDED METHODS: Unroll your favourite iterative (proximal) algorithm over a **fixed** number of iterations, and learn parameters inside

EXAMPLE OF LASSO [Gregor & Lecun, 2010]: Find $\widehat{\boldsymbol{x}} \in \underset{\boldsymbol{x}\in\mathbb{R}^N}{\operatorname{Argmin}} \ \frac{1}{2}\|\mathbf{W}\boldsymbol{x}-\boldsymbol{z}\|^2 + \eta\|\boldsymbol{x}\|_1$

- FB iterations: $\widehat{\boldsymbol{x}} = \lim_{k\to\infty} \boldsymbol{x}^{[k]}$ with $(\forall k \in \mathbb{N})\ \boldsymbol{x}^{[k+1]} = \operatorname{prox}_{\gamma\eta\|\cdot\|_1}\Big(\boldsymbol{x}^{[k]} - \gamma\mathbf{W}^*(\mathbf{W}\boldsymbol{x}^{[k]} - \boldsymbol{z})\Big)$



- Unfolded FB-NN: $\widehat{\boldsymbol{x}} \approx \mathfrak{T}_{\Theta_K} \circ \cdots \circ \mathfrak{T}_{\Theta_1}(\boldsymbol{x}^{[0]})$ with $\mathfrak{T}_{\Theta_k}(\boldsymbol{x}) = \underbrace{\operatorname{prox}_{\gamma\eta\|\cdot\|_1}}_{\text{Non-linearity (learn.)}}\Big(\underbrace{(\mathrm{Id} - \gamma\mathbf{W}^*\mathbf{W})}_{\text{Linearity (learn.)}}\boldsymbol{x} + \underbrace{\gamma\mathbf{W}^*\boldsymbol{z}}_{\text{Bias (learn.)}}\Big)$



REMARK: General formulation $\mathfrak{T}_{\Theta_k}(\boldsymbol{x}) = \mathfrak{D}_{\Lambda_k}(\mathbf{L}_k\boldsymbol{x} + \boldsymbol{b}_k)$

... and to unfolded (end-to-end) networks

UNFOLDED METHODS: Unroll your favourite iterative (proximal) algorithm over a **fixed** number of iterations, and learn parameters inside

REMARKS: General formulation $\widehat{\boldsymbol{x}} \approx \mathfrak{T}_{\Theta_K} \circ \cdots \circ \mathfrak{T}_{\Theta_1}(\boldsymbol{x}^{[0]})$ with $\mathfrak{T}_{\Theta_k}(\boldsymbol{x}) = \mathfrak{D}_{\Lambda_k}(\mathbf{L}_k \boldsymbol{x} + \boldsymbol{b}_k)$

- $\mathfrak{D}_{\Lambda_k}$ can be a denoising NNs [Adler & Öktem, 2018] (K very small), or can be an activation functions (K can be large) [Jiu & Pustelnik, 2021]

  $\rightsquigarrow$ Many activation functions are **proximity** operators of *convex* functions, and **derivatives** of smooth *convex* functions [Combettes & Pesquet, 2020a, 2020b] (e.g., ReLU, Unimodal sigmoid, ...)

- Unfolded networks are end-to-end (i.e., $K < +\infty$), so convergence results have limited interest.

- Instead, one can look at their stability:

  sensivity of $\widehat{\boldsymbol{x}}$ with respect to perturbations on $\boldsymbol{x}^{[0]}$ and $\boldsymbol{z}$

  See, e.g., [Bertochi *et al.*, 2020], [Chouzenoux *et al.*, 2025],...

Outline of the tutorial

① Introduction

② Proximal algorithms

③ Feed-forward neural networks

④ Plug-and-play

⑤ Unfolded schemes

⑥ Hands in code

⑦ Conclusions