PNN: From proximal algorithms to robust unfolded image restauration networks

Nelly Pustelnik
CNRS, ENS Lyon, France

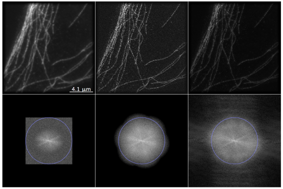
Conférence 30 ans de mathématiques pour l'imagerie optique
25 septembre 2023, Marseille

Context: Image recovery

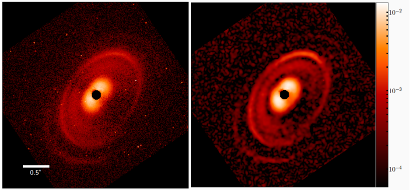
☛ **Data**: $\mathbf{z} \in \mathbb{R}^M$ degraded version of an original image $\overline{\mathbf{x}} \in \mathbb{R}^N$:

$$\boxed{\mathbf{z} = \mathbf{A}\overline{\mathbf{x}} + \mathbf{w}}$$

- $\mathbf{A} : \mathbb{R}^{M \times N}$: linear degradation (e.g. a blur)
- $\mathbf{w}$: noise (e.g. Gaussian noise)



SIM



SPHERE-IRDIS

Context: image restoration

Synthesis formulation

$\widehat{\mathbf{x}} = \mathbf{D}^* \widehat{\alpha}$ with $\mathbf{D} \in \mathbb{R}^{P \times N}$

$$\boxed{\widehat{\alpha} \in \underset{\alpha}{\operatorname{Argmin}} \frac{1}{2} \|\mathbf{A}\mathbf{D}^*\alpha - \mathbf{z}\|_2^2 + \lambda \|\alpha\|_\bullet}$$

Analysis formulation

$$\boxed{\widehat{\mathbf{x}} \in \underset{\mathbf{x}}{\operatorname{Argmin}} \frac{1}{2} \|\mathbf{A}\mathbf{x} - \mathbf{z}\|_2^2 + \lambda \|\mathbf{D}\mathbf{x}\|_\bullet}$$

$\Rightarrow$ **Equivalence for D orthonormal basis.**

[Elad, Milanfar, Ron, 2007] [Chaari, Pustelnik, Chaux, Pesquet, 2009]
[Selesnick, Figueiredo, 2009], [Carlavan, Weiss, Blanc-Féraud, 2010]
[Pustelnik, Benazza-Benhayia, Zheng, Pesquet, 2010]

Context: image restoration

Synthesis formulation

$\widehat{\mathbf{x}} = \mathbf{D}^* \widehat{\alpha}$ with $\mathbf{D} \in \mathbb{R}^{P \times N}$

$$\widehat{\alpha} \in \underset{\alpha}{\operatorname{Argmin}} \frac{1}{2} \|\mathbf{A}\mathbf{D}^*\alpha - \mathbf{z}\|_2^2 + \lambda \|\alpha\|_\bullet$$

Analysis formulation

$$\widehat{\mathbf{x}} \in \underset{\mathbf{x}}{\operatorname{Argmin}} \frac{1}{2} \|\mathbf{A}\mathbf{x} - \mathbf{z}\|_2^2 + \lambda \|\mathbf{D}\mathbf{x}\|_\bullet$$

$\Rightarrow$ **Equivalence for D orthonormal basis.**

- X-lets
- Sparse coding

- Horizontal/vertical gradients: TV
- Hessian operator
- Nonlocal total variation: weighted nonlocal gradients: NLTV
- Local dictionaries of patches

(webpage L. Duval)[Aharon, Elad, Bruckstein, 2006] [Mairal, Sapiro, Elad, 2007][Gilboa, Osher, 2008][K Bredies, K Kunisch, T Pock, 2010][Jacques, Duval, Chaux, Peyré, 2011] [S Lefkimmiatis, A Bourquard, M Unser, 2011] [Zoran, Weiss, 2011] [G Kutyniok, D Labate, 2012][Chierchia et al.,2014][Boulanger et al., 2018]...

"Semi-smooth" minimization problem

Objective

$$\text{Find } \widehat{\mathbf{x}} \in \operatorname*{Argmin}_{\mathbf{x}\in\mathcal{H}} \Big\{ \mathsf{F}(\mathbf{x}) = \mathsf{h}(\mathbf{x}) + \mathsf{g}(\mathbf{D}\mathbf{x}) \Big\}$$

- $\mathsf{h} \in \Gamma_0(\mathcal{H})$ and β-Lipschitz differentiable
- $\mathbf{D}\colon \mathcal{H} \to \mathcal{G}$ and $\mathsf{g} \in \Gamma_0(\mathcal{G})$

☛ **Remark**: Usually $\mathrm{prox}_{\mathsf{F}}$ does not have a closed form solution.

☛ **Idea**: Use splitting methods to handle h and g separately.

FB algorithm

Objective

$$\text{Find } \widehat{\mathbf{x}} \in \underset{\mathbf{x}\in\mathcal{H}}{\operatorname{Argmin}} \left\{ \mathsf{F}(\mathbf{x}) = \mathsf{h}(\mathbf{x}) + \mathsf{g}(\mathbf{D}\mathbf{x}) \right\}$$

- $\mathsf{h} \in \Gamma_0(\mathcal{H})$ and β-Lipschitz differentiable
- $\mathbf{D}\colon \mathcal{H} \to \mathcal{G}$ and $\mathsf{g} \in \Gamma_0(\mathcal{G})$

Algorithm: Let $\mathbf{x}^{[0]} \in \mathcal{H}$,
For $k = 0, 1, \ldots$

$$\mathbf{x}^{[k+1]} = \operatorname{prox}_{\gamma_k \mathsf{g}\circ\mathbf{D}}\big(\mathbf{x}^{[k]} - \gamma_k \nabla \mathsf{h}(\widetilde{\mathbf{x}}^{[k]})\big)$$

Theorem (FB): Let, for every $k \in \mathbb{N}$, $0 < \gamma_k < 2\beta^{-1}$. Then

- $(\mathbf{x}^{[k]})_{k\in\mathbb{N}}$ converges to a minimizer $\widehat{\mathbf{x}}$ of F.

[Combettes & Wajs, 2005]

Primal-dual algorithm

Objective

$$\text{Find } \widehat{\mathbf{x}} \in \underset{\mathbf{x}\in\mathbb{R}^N}{\text{Argmin}} \Big\{ \mathrm{F}(\mathbf{x}) = \mathrm{h}_1(\mathbf{x}) + \mathrm{h}_2(\mathbf{x}) + \mathrm{g}(\mathbf{D}\mathbf{x}) \Big\}$$

- $\mathrm{h}_1 \colon \mathcal{H} \to]-\infty, +\infty]$ is convex, proper and β-Lipschitz differentiable
- $\mathrm{h}_2 \in \Gamma_0(\mathcal{H})$, $\mathbf{D} \colon \mathcal{H} \to \mathcal{G}$, and $\mathrm{g} \in \Gamma_0(\mathcal{G})$

Algorithm: $\mathbf{x}^{[0]} \in \mathcal{H}$

For $k = 0, 1, \ldots$

$$\mathbf{x}^{[k+1]} = \text{prox}_{\tau \mathrm{h}_2}\Big(\mathbf{x}^{[k]} - \tau\big(\nabla \mathrm{h}_1(\mathbf{x}^{[k]}) + \mathbf{D}^*\mathbf{u}^{[k]}\big)\Big)$$

$$\mathbf{u}^{[k+1]} = \text{prox}_{\sigma \mathrm{g}^*}\Big(\mathbf{u}^{[k]} + \sigma\mathbf{D}(2\mathbf{x}^{[k+1]} - \mathbf{x}^{[k]})\Big)$$

Theorem: Choose $\tau > 0$ and $\sigma > 0$ such that $\frac{1}{\tau} - \sigma\|\mathbf{D}\|^2 > \frac{\beta}{2}$.

The sequence $(\mathbf{x}^{[k]})_{k\in\mathbb{N}}$ converges to a minimizer $\widehat{\mathbf{x}}$ of F

[Vũ, 2013][Condat, 2013]

Context

Data: $\mathbf{z}$

Direct/acquisition model

$$\mathbf{z} = \mathcal{D}(\mathbf{A}\overline{\mathbf{x}})$$

Design a cost function

$$\widehat{\mathbf{x}}(\mathbf{z};\lambda) \in \underset{\mathrm{x}}{\mathrm{Argmin}}\ \frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{z}\|_2^2 + \lambda\|\mathbf{D}\mathbf{x}\|_\bullet$$

Minimization algorithm

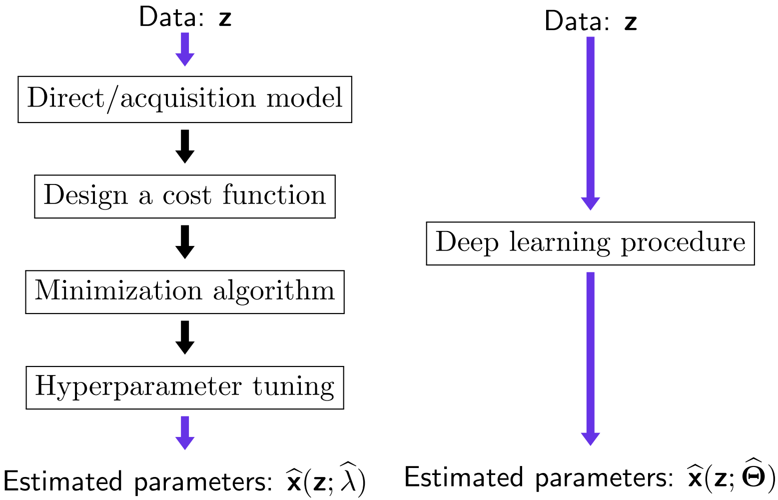
Sequence $\mathbf{x}_{k+1} = \mathsf{T}\mathbf{x}_k$

Hyperparameter tuning

$$\widehat{\lambda} \in \underset{\lambda}{\mathrm{Argmin}}\|\overline{\mathbf{x}} - \widehat{\mathbf{x}}(\mathbf{z};\lambda)\|_2^2$$

Estimated parameters: $\widehat{\mathbf{x}}(\mathbf{z};\widehat{\lambda})$

Standard learning and deep learning



Training a NN for inverse problem task

- **Database**: $\mathcal{S} = \big\{(\mathbf{z}_i, \overline{\mathbf{x}}_i) \in \mathbb{R}^M \times \mathbb{R}^N \;\big|\; i \in \{1, \ldots, \mathbb{L}\}\big\}$

Training a NN for inverse problem task

- **Database**: $\mathcal{S} = \big\{(\mathbf{z}_i, \overline{\mathbf{x}}_i) \in \mathbb{R}^M \times \mathbb{R}^N \;\big|\; i \in \{1, \ldots, \mathbb{L}\}\big\}$
  We consider two sets of images: the *training set* $(\mathbf{z}_i, \overline{\mathbf{x}}_i)_{i\in\mathbb{I}}$ of size $\sharp\,\mathbb{I}$ and the *testing set* $(\mathbf{z}_j, \overline{\mathbf{x}}_j)_{j\in\mathbb{J}}$ of size $\sharp\,\mathbb{J}$ where
  $$(\forall i \in \mathbb{I} \cup \mathbb{J}) \quad \mathbf{z}_i = \mathbf{A}\overline{\mathbf{x}}_i + \mathbf{w}_i$$

Training a NN for inverse problem task

- **Database**: $\mathcal{S} = \big\{(\mathbf{z}_i, \overline{\mathbf{x}}_i) \in \mathbb{R}^M \times \mathbb{R}^N \;\big|\; i \in \{1, \ldots, \mathbb{L}\}\big\}$
  We consider two sets of images: the *training set* $(\mathbf{z}_i, \overline{\mathbf{x}}_i)_{i\in\mathbb{I}}$ of size $\sharp\,\mathbb{I}$ and the *testing set* $(\mathbf{z}_j, \overline{\mathbf{x}}_j)_{j\in\mathbb{J}}$ of size $\sharp\,\mathbb{J}$ where
  $$(\forall i \in \mathbb{I} \cup \mathbb{J}) \quad \mathbf{z}_i = \mathbf{A}\overline{\mathbf{x}}_i + \mathbf{w}_i$$

- **Training**: The NN is trained using the *training set* to estimate:
  $$\widehat{\boldsymbol{\Theta}} \in \underset{\boldsymbol{\Theta}}{\operatorname{Argmin}} \frac{1}{\sharp\,\mathbb{I}} \sum_{i\in\mathbb{I}} \|\overline{\mathbf{x}}_i - f_{\boldsymbol{\Theta}}(\mathbf{z}_i)\|^2$$

- **Testing**: The learned NN $f_{\widehat{\boldsymbol{\Theta}}}$ is then validated on the testing set. A properly trained network should satisfy
  $$(\forall j \in \mathbb{J}) \quad \overline{\mathbf{x}}_j \approx f_{\widehat{\boldsymbol{\Theta}}}(\mathbf{z}_j).$$

Deep learning architecture

Deep learning – *Framework*

- Database : $\mathcal{S} = \big\{(\mathbf{z}_i, \overline{\mathbf{x}}_i) \in \mathbb{R}^M \times \mathbb{R}^N \;\big|\; i \in \{1, \dots, \mathbb{L}\}\big\}$
- Prediction function : $f_{\boldsymbol{\Theta}}(\mathrm{z}_i) = \eta^{[K]}\big(W^{[K]} \dots \eta^{[1]}(W^{[1]}\mathrm{z}_i + b^{[1]}) \dots + b^{[K]}\big)$

⊙ Linear operators: $W^{[1]}, \mathrm{W}^{[2]}, \dots, W^{[K]}$
⊙ Activation functions: $\eta^{[1]}, \eta^{[2]}, \dots, \eta^{[K]}$
⊙ Biais vectors: $b^{[1]}, b^{[2]}, \dots, b^{[K]}$

$$\Rightarrow \boxed{\boldsymbol{\Theta} = \{W^{[1]}, \dots, W^{[K]}, b^{[1]}, \dots, b^{[K]}\}}$$

PNN for image restoration

Pioneer work : LISTA for synthesis formulation

- **Synthesis formulation:**

$$\boxed{\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{A}\mathbf{D}^*\mathbf{x}-\mathbf{z}\|_2^2+\lambda\|\mathbf{x}\|_1}$$ where $\mathbf{H}=\mathbf{A}\mathbf{D}^* \in \mathbb{R}^{M\times\overline{N}}$

- **Forward-backward iterations:**

$$\left| \; \mathbf{x}^{[k+1]} = \mathrm{prox}_{\tau\lambda\|\cdot\|_1}\left(\mathbf{x}^{[k]}-\tau\mathbf{H}^*(\mathbf{H}\mathbf{x}^{[k]}-\mathbf{z})\right)\right.$$

- **Reformulation:**

$$\left| \; \mathbf{x}^{[k+1]} = \mathrm{prox}_{\tau\lambda\|\cdot\|_1}\left((\mathrm{Id}-\tau\mathbf{H}^*\mathbf{H})\mathbf{x}^{[k]}+\tau\mathbf{H}^*\,\mathbf{z}\right)\right.$$

- **Layer network:** [Gregor, LeCun, 2010]

$$\mathbf{x}^{[k+1]} = \underbrace{\boxed{\mathrm{prox}_{\tau\lambda\|\cdot\|_1}}}_{\eta^{[k]}}\left(\underbrace{\boxed{\mathrm{Id}-\tau\mathbf{H}^*\mathbf{H}}}_{\mathrm{W}^{[k]}}\;\mathbf{x}^{[k]}+\underbrace{\boxed{\tau\mathbf{H}^*\mathbf{z}}}_{\mathrm{b}^{[k]}}\right)$$

Deep learning versus proximal algorithms

☛ **Most of activation functions are proximity operator** :
ReLU, Unimodal sigmoid, Softmax ...
[Combettes, Pesquet, 2020]

Deep learning versus proximal algorithms

- **Most of activation functions are proximity operator** : ReLU, Unimodal sigmoid, Softmax . . .
  [Combettes, Pesquet, 2020]

Proposition [Le, Pustelnik, Foare, 2022]: The proximity operator of the conjugate of the ℓ_1-norm scaled by parameter $\lambda > 0$ fits the HardTanh activation function, i.e., for every $\mathbf{x} = (\mathbf{x}_i)_{1 \leqslant i \leqslant N}$:

$$\mathrm{P}_{\|\cdot\|_\infty \leqslant \lambda}(\mathbf{x}) = \mathrm{HardTanh}_\lambda(\mathbf{x}) = (\mathrm{p}_i)_{1 \leqslant i \leqslant N}$$

where

$$\mathrm{p}_i = \begin{cases} -\lambda & \text{if} \quad \mathrm{p}_i < -\lambda, \\ \lambda & \text{if} \quad \mathrm{p}_i > \lambda, \\ \mathrm{p}_i & \text{otherwise.} \end{cases}$$

Deep learning versus proximal algorithms

- **Most of activation functions are proximity operator** : ReLU, Unimodal sigmoid, Softmax ...
  [Combettes, Pesquet, 2020]
- Let $\mathrm{W}^{[k]}$ be a bounded linear operators, b_k a vector, η_k proximity operators (1/2-averaged operator), $f_{\boldsymbol{\Theta}} = \mathsf{T}_K \circ \cdots \circ \mathsf{T}_1$ with $\mathsf{T}_k = \eta_k(W_k \cdot + b_k)$ model allows to derive tight Lipschitz bounds for feedforward neural networks in order to evaluate their **robustness** i.e.

$$\|f_{\boldsymbol{\Theta}}(\mathbf{z}+\epsilon) - f_{\boldsymbol{\Theta}}(\mathbf{z})\| \leqslant \chi \|\epsilon\|$$

. [Combettes, Pesquet, 2020]

Preliminary work: DeepPDNet for Analysis formulation

☛ **Analysis formulation:**

$$\boxed{\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{A}\mathbf{x}-\mathbf{z}\|_2^2 + \|\mathbf{H}\mathbf{x}\|_1} \quad \text{where } \mathbf{H} = \lambda\mathbf{D}$$

☛ **Condat-Vũ iterations:**

$$\left|\begin{array}{ll} \mathbf{x}^{[k+1]} & = \mathbf{x}_k - \tau\mathbf{A}^*(\mathbf{A}\mathbf{x}^{[k]} - \mathbf{z}) - \tau\mathbf{H}^*\mathbf{u}^{[k]} \\ \mathbf{u}^{[k+1]} & = \operatorname{prox}_{\gamma\|\cdot\|_1^*}\left(\mathbf{u}^{[k]} + \gamma\mathbf{H}(2\mathbf{x}^{[k+1]} - \mathbf{x}^{[k]})\right) \end{array}\right.$$

☛ **Reformulation:**

$$\left|\begin{array}{l} \mathbf{x}^{[k+1]} = (\mathrm{Id} - \tau\mathbf{A}^*\mathbf{A})\mathbf{x}^{[k]} - \tau\mathbf{H}^*\mathbf{u}^{[k]} + \tau\mathbf{A}^*\mathbf{z} \\ \mathbf{u}^{[k+1]} = \operatorname{prox}_{\gamma\|\cdot\|_1^*}\left(\gamma\mathbf{H}(\mathrm{Id} - 2\tau\mathbf{A}^*\mathbf{A})\mathbf{x}^{[k]} + (\mathrm{Id} - 2\tau\gamma\mathbf{H}\mathbf{H}^*)\mathbf{u}^{[k]} + 2\tau\gamma\mathbf{H}\mathbf{A}^*\mathrm{z}\right). \end{array}\right.$$

☛ **Layer network:** [Jiu, Pustelnik, 2021]

$$\begin{bmatrix}\mathbf{x}^{[k+1]}\\ \mathbf{u}^{[k+1]}\end{bmatrix} = \underbrace{\boxed{\begin{matrix}\mathrm{Id}\\ \operatorname{prox}_{\gamma\|\cdot\|_1^*}\end{matrix}}}_{\eta^{[k]}}\left(\underbrace{\boxed{\begin{bmatrix}\mathrm{Id} - \tau\mathbf{A}^*\mathbf{A} & -\tau\mathbf{H}^*\\ \gamma\mathbf{H}(\mathrm{Id} - 2\tau\mathbf{A}^*\mathbf{A}) & \mathrm{Id} - 2\tau\gamma\mathbf{H}\mathbf{H}^*\end{bmatrix}}}_{W^{[k]}}\begin{bmatrix}\mathbf{x}^{[k]}\\ \mathbf{u}^{[k]}\end{bmatrix} + \underbrace{\boxed{\begin{bmatrix}\tau\mathbf{A}^*\mathbf{z}\\ 2\tau\gamma\mathbf{H}\mathbf{A}^*\mathbf{z}\end{bmatrix}}}_{b^{[k]}}\right)$$

Analysis formulation and the proposed DeepPDNet

$$\boxed{f_{\Theta}(\mathbf{x}) = \eta^{[K]}\left(W^{[K]} \dots \eta^{[1]}(W^{[1]}\mathbf{x} + b^{[1]}) \dots + b^{[K]}\right)}$$

☛ **Network with fixed layer: $\Theta = \{\mathbf{H}, \tau, \gamma\}$**

$$\begin{bmatrix} \mathbf{x}^{[k+1]} \\ \mathbf{u}^{[k+1]} \end{bmatrix} = \underbrace{\boxed{\begin{matrix} \mathrm{Id} \\ \mathrm{prox}_{\gamma\|\cdot\|_1^*} \end{matrix}}}_{\eta^{[k]}} \left(\underbrace{\boxed{\begin{bmatrix} \mathrm{Id} - \tau\mathbf{A}^*\mathbf{A} & -\tau\mathbf{H}^* \\ \gamma\mathbf{H}(\mathrm{Id} - 2\tau\mathbf{A}^*\mathbf{A}) & \mathrm{Id} - 2\tau\gamma\mathbf{H}\mathbf{H}^* \end{bmatrix}}}_{W^{[k]}} \begin{bmatrix} \mathbf{x}^{[k]} \\ \mathbf{u}^{[k]} \end{bmatrix} + \underbrace{\boxed{\begin{bmatrix} \tau\mathbf{A}^*\mathrm{z} \\ 2\tau\gamma\mathbf{H}\mathbf{A}^*\mathrm{z} \end{bmatrix}}}_{b^{[k]}} \right)$$

☛ **Network with variable layers: $\Theta = \{\mathbf{H}^{[k]}, \tau_k, \gamma_k, \}_{1\leqslant k\leqslant K}$**

$$\begin{bmatrix} \mathbf{x}^{[k+1]} \\ \mathbf{u}^{[k+1]} \end{bmatrix} = \underbrace{\boxed{\begin{matrix} \mathrm{Id} \\ \mathrm{prox}_{\gamma_k\|\cdot\|_1^*} \end{matrix}}}_{\eta^{[k]}} \left(\underbrace{\boxed{\begin{bmatrix} \mathrm{Id} - \tau_k\mathbf{A}^*\mathbf{A} & -\tau_k\mathbf{H}_k^* \\ \gamma_k\mathbf{H}_k(\mathrm{Id} - 2\tau_k\mathbf{A}^*\mathbf{A}) & \mathrm{Id} - 2\tau\gamma_k\mathbf{H}_k\mathbf{H}_k^* \end{bmatrix}}}_{W^{[k]}} \begin{bmatrix} \mathbf{x}^{[k]} \\ \mathbf{u}^{[k]} \end{bmatrix} + \underbrace{\boxed{\begin{bmatrix} \tau_k\mathbf{A}_k^*\mathrm{z} \\ 2\tau_k\gamma_k\mathbf{H}_k\mathbf{A}^*\mathrm{z} \end{bmatrix}}}_{b^{[k]}} \right)$$

$+$ specificities for the first and last layers.

Analysis formulation and the proposed DeepPDNet

➡ **Learn a prediction function $f_{\boldsymbol{\Theta}}$:**

$$\widehat{\boldsymbol{\Theta}} \in \underset{\boldsymbol{\Theta}}{\mathrm{Argmin}}\ \mathrm{E}(\boldsymbol{\Theta}) := \frac{1}{\sharp\,\mathbb{I}} \sum_{i \in \mathbb{I}} \|\overline{\mathbf{x}}_i - d_{\boldsymbol{\Theta}}(\mathbf{z}_i)\|^2$$

➡ **Gradient based strategy**

$$\boldsymbol{\Theta}_{k,\ell+1} = \boldsymbol{\Theta}_{k,\ell} - \gamma_{\boldsymbol{\Theta}} \frac{\partial E(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}^{[k]}}$$

Analysis formulation and the proposed DeepPDNet

➡ **Learn a prediction function $f_{\boldsymbol{\Theta}}$:**

$$\widehat{\boldsymbol{\Theta}} \in \underset{\boldsymbol{\Theta}}{\mathrm{Argmin}}\ \mathrm{E}(\boldsymbol{\Theta}) := \frac{1}{\sharp\, \mathbb{I}} \sum_{i \in \mathbb{I}} \|\overline{\mathbf{x}}_i - d_{\boldsymbol{\Theta}}(\mathbf{z}_i)\|^2$$

➡ **Gradient based strategy**

$$\boldsymbol{\Theta}_{k,\ell+1} = \boldsymbol{\Theta}_{k,\ell} - \gamma_{\boldsymbol{\Theta}} \frac{\partial E}{\partial \mathrm{u}_K} \frac{\partial \mathrm{u}_K}{\partial \mathrm{u}_{K-1}} \cdots \frac{\partial \mathrm{u}_{k+1}}{\partial \mathrm{u}_k} \frac{\partial \mathrm{u}_k}{\partial \boldsymbol{\Theta}_k}$$

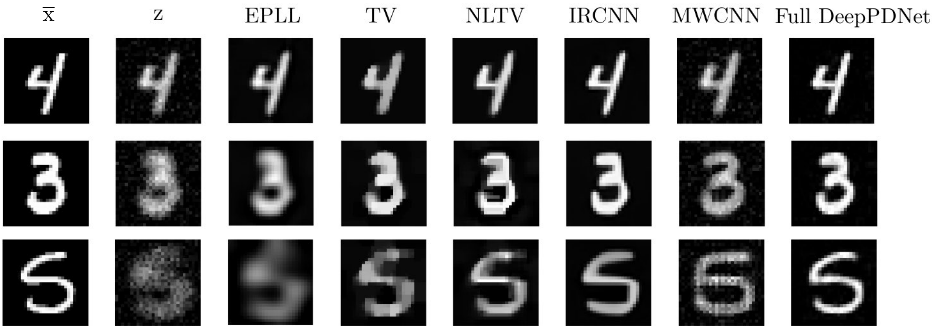
where

$$\frac{\partial \mathrm{u}_k}{\partial \mathrm{u}_{k-1}} = \frac{d\eta_k(\mathrm{v}_k)}{d\mathrm{v}_k} \mathrm{W}_k$$

$$\frac{\partial \mathrm{u}_k}{\partial \boldsymbol{\Theta}_k} = \left(\frac{\partial \eta_k(\mathrm{v}_k)}{\partial \mathrm{v}_k} \left(\frac{\partial \mathrm{W}^{[k]}}{\partial \boldsymbol{\Theta}^{[k]}} \mathrm{u}_k + \frac{\partial b_k}{\partial \boldsymbol{\Theta}_k} \right) + \frac{\partial \eta_k(\mathrm{v}_k)}{\partial \boldsymbol{\Theta}_k} \right)$$

with $\mathrm{v}_k = \mathrm{W}_k \mathrm{u}_{k-1} + b_k$ and $\mathrm{u}_k = \eta_k(\mathrm{v}_k)$

Analysis formulation and proposed DeepPDNet



[Jiu, Pustelnik, 2021]

PNN: focus on denoising task

D(i)FB algorithm

Objective

$$\text{Find } \widehat{\mathbf{x}} \in \underset{\mathbf{x}\in\mathcal{H}}{\operatorname{Argmin}} \left\{ \mathrm{F}(\mathbf{x}) = \frac{1}{2}\|\mathbf{x}-\mathbf{z}\|_2^2 + g(\mathbf{D}\mathbf{x}) + \iota_C(\mathbf{x}) \right\}$$

- $C \subset \mathcal{H}$ is a closed, convex, non-empty. $\mathbf{D}\colon \mathcal{H} \to \mathcal{G}$ and $\mathrm{g} \in \Gamma_0(\mathcal{H})$

Algorithm: Let $\mathbf{x}_0 \in \mathcal{H}$,
For $k = 0, 1, \ldots$

$$\left\lfloor \begin{array}{l} \mathbf{u}^{[k+1]} = \operatorname{prox}_{\tau_k(\nu g)^*}\Big(\mathbf{v}^{[k]} + \tau_k \mathbf{D} \mathrm{P}_C(\mathbf{z} - \mathbf{D}^\top \mathbf{v}^{[k]})\Big), \\ \mathbf{v}^{[k+1]} = (1+\rho_k)\mathbf{u}^{[k+1]} - \rho_k \mathbf{u}^{[k]}, \end{array} \right.$$

Theorem : Assume that one of the following conditions is satisfied.

1. (DFB): $\forall k \in \mathbb{N}$, $\tau_k \in (0, 2/\|\mathbf{D}\|_S^2)$, and $\rho_k = 0$.
2. (DiFB): $\forall k \in \mathbb{N}$, $\tau_k \in (0, 1/\|\mathbf{D}\|_S^2)$, and $\rho_k = \frac{t_k - 1}{t_{k+1}}$ with $t_k = \frac{k+a-1}{a}$ and $a > 2$.

Then we have

$$\widehat{\mathbf{x}} = \lim_{k\to\infty} \mathrm{P}_C(\mathbf{z} - \mathbf{D}^\top \mathbf{u}^{[k]}),$$

S(c)CP algorithm

Objective

$$\text{Find } \widehat{\mathbf{x}} \in \underset{\mathbf{x}\in\mathcal{H}}{\operatorname{Argmin}} \left\{ \mathsf{F}(\mathbf{x}) = \tfrac{1}{2}\|\mathbf{x}-\mathbf{z}\|_2^2 + g(\mathbf{D}\mathbf{x}) + \iota_C(\mathbf{x}) \right\}$$

- $C \subset \mathcal{H}$ is a closed, convex, non-empty. $\mathbf{D}\colon \mathcal{H} \to \mathcal{G}$ and $\mathrm{g} \in \Gamma_0(\mathcal{G})$

Algorithm: Let $\mathbf{x}_0 \in \operatorname{dom} \mathrm{g}$,$(\tau_k)_{k\in\mathbb{N}}$ and $(\mu_k)_{k\in\mathbb{N}}$ are positive sequences

For $k = 0, 1, \ldots$

$$\left\lfloor \begin{array}{l} \mathbf{x}^{[k+1]} = \mathrm{P}_C\left(\frac{\mu_k}{1+\mu_k}(\mathbf{z} - \mathbf{D}^\top \mathbf{u}^{[k]}) + \frac{1}{1+\mu_k}\mathbf{x}^{[k]} \right), \\ \mathbf{u}^{[k+1]} = \operatorname{prox}_{\tau_k(\nu g)^*}\left(\mathbf{u}^{[k]} + \tau_k \mathbf{D}\Big((1+\alpha_k)\mathbf{x}^{[k+1]} - \alpha_k \mathbf{x}^{[k]} \Big) \right), \end{array} \right.$$

Theorem : Assume that one of the following conditions is satisfied.

1. (CP): $\forall k \in \mathbb{N}$, $\tau_k\mu_k\|\mathbf{D}\|_S^2 < 1$, and $\alpha_k = 1$.
2. (ScCP): $\forall k \in \mathbb{N}$, $\alpha_k = 1/\sqrt{1+2\mu_k}$, $\mu_{k+1} = \alpha_k\mu_k$, $\tau_{k+1} = \tau_k\alpha_k^{-1}$ with $\mu_0\tau_0\|\mathbf{D}\|_S^2 \leqslant 1$.

Then we have

$$\widehat{\mathbf{x}} = \lim_{k\to\infty} \mathbf{x}^{[k]}.$$

S(c)CP to D(i)FB

Objective

$$\text{Find } \widehat{\mathbf{x}} \in \underset{\mathbf{x}\in\mathcal{H}}{\operatorname{Argmin}} \left\{ \mathsf{F}(\mathbf{x}) = \frac{1}{2}\|\mathbf{x}-\mathbf{z}\|_2^2 + g(\mathbf{D}\mathbf{x}) + \iota_C(\mathbf{x}) \right\}$$

Algorithm: For $k = 0, 1, \ldots$

$$\left\lfloor \begin{array}{l} \mathbf{x}^{[k+1]} = \mathrm{P}_C\left(\frac{\mu_k}{1+\mu_k}(\mathbf{z} - \mathbf{D}^\top \mathbf{u}^{[k]}) + \frac{1}{1+\mu_k}\mathbf{x}^{[k]}\right) \\ \mathbf{u}^{[k+1]} = \mathrm{prox}_{\tau_k(\nu g)^*}\left(\mathbf{u}^{[k]} + \tau_k \mathbf{D}\Big((1+\alpha_k)\mathbf{x}^{[k+1]} - \alpha_k \mathbf{x}_k\Big)\right) \end{array} \right.$$

☛ **S(c)CP:** Starting point.

S(c)CP to D(i)FB

Objective

Find $\widehat{\mathbf{x}} \in \underset{\mathbf{x}\in\mathcal{H}}{\operatorname{Argmin}} \left\{ \mathsf{F}(\mathbf{x}) = \frac{1}{2}\|\mathbf{x}-\mathbf{z}\|_2^2 + g(\mathbf{D}\mathbf{x}) + \iota_C(\mathbf{x}) \right\}$

Algorithm: For $k = 0, 1, \ldots$

$$\left\lfloor \begin{array}{l} \mathbf{x}^{[k+1]} = \mathrm{P}_C \left(\frac{\mu_k}{1+\mu_k}(\mathbf{z} - \mathbf{D}^\top \mathbf{u}^{[k]}) + \frac{1}{1+\mu_k}\mathbf{x}^{[k]} \right) \\ \mathbf{u}^{[k+1]} = \mathrm{prox}_{\tau_k(\nu g)^*} \left(\mathbf{u}^{[k]} + \tau_k \mathbf{D}\Big((1+\alpha_k)\mathbf{x}^{[k+1]} - \alpha_k \mathbf{x}_k \Big) \right) \end{array} \right.$$

- **S(c)CP:** Starting point.
- **Arrow-Hurwicz iterations:** $\alpha_k \equiv 0$.

S(c)CP to D(i)FB

Objective

$$\text{Find } \widehat{\mathbf{x}} \in \underset{\mathbf{x}\in\mathcal{H}}{\operatorname{Argmin}} \left\{ \mathsf{F}(\mathbf{x}) = \frac{1}{2}\|\mathbf{x}-\mathbf{z}\|_2^2 + g(\mathbf{D}\mathbf{x}) + \iota_C(\mathbf{x}) \right\}$$

Algorithm: For $k = 0, 1, \ldots$

$$\left\lfloor \begin{array}{l} \mathbf{x}^{[k+1]} = \mathrm{P}_C\left(\frac{\mu_k}{1+\mu_k}(\mathbf{z} - \mathbf{D}^\top \mathbf{u}^{[k]}) + \frac{1}{1+\mu_k}\mathbf{x}^{[k]}\right) \\ \mathbf{u}^{[k+1]} = \operatorname{prox}_{\tau_k(\nu g)^*}\left(\mathbf{u}^{[k]} + \tau_k \mathbf{D}\mathbf{x}^{[k+1]}\right) \end{array} \right.$$

- **S(c)CP:** Starting point.
- **Arrow-Hurwicz iterations:** $\alpha_k \equiv 0$.

S(c)CP to D(i)FB

Objective

$$\text{Find } \widehat{\mathbf{x}} \in \underset{\mathbf{x}\in\mathcal{H}}{\operatorname{Argmin}} \left\{ \mathsf{F}(\mathbf{x}) = \frac{1}{2}\|\mathbf{x}-\mathbf{z}\|_2^2 + g(\mathbf{D}\mathbf{x}) + \iota_C(\mathbf{x}) \right\}$$

Algorithm: For $k = 0, 1, \dots$

$$\left\lfloor \begin{array}{l} \mathbf{x}^{[k+1]} = \mathrm{P}_C\left(\frac{\mu_k}{1+\mu_k}(\mathbf{z} - \mathbf{D}^\top \mathbf{u}^{[k]}) + \frac{1}{1+\mu_k}\mathbf{x}^{[k]}\right) \\ \mathbf{u}^{[k+1]} = \mathrm{prox}_{\tau_k(\nu g)^*}\left(\mathbf{u}^{[k]} + \tau_k \mathbf{D}\mathbf{x}^{[k+1]}\right) \end{array} \right.$$

- **S(c)CP:** Starting point.
- **Arrow-Hurwicz iterations:** $\alpha_k \equiv 0$.
- **DFB:** $\mu_k \to +\infty$.

S(c)CP to D(i)FB

Objective

$$\text{Find } \widehat{\mathbf{x}} \in \underset{\mathbf{x}\in\mathcal{H}}{\operatorname{Argmin}} \left\{ \mathsf{F}(\mathbf{x}) = \frac{1}{2}\|\mathbf{x}-\mathbf{z}\|_2^2 + g(\mathbf{D}\mathbf{x}) + \iota_C(\mathbf{x}) \right\}$$

Algorithm: For $k = 0, 1, \dots$

$$\left\lfloor \begin{array}{l} \mathbf{x}^{[k+1]} = \mathrm{P}_C\left(\mathbf{z} - \mathbf{D}^\top \mathbf{u}^{[k]}\right) \\ \mathbf{u}^{[k+1]} = \operatorname{prox}_{\tau_k(\nu g)^*}\left(\mathbf{u}^{[k]} + \tau_k \mathbf{D}\mathbf{x}^{[k+1]}\right) \end{array} \right.$$

- **S(c)CP:** Starting point.
- **Arrow-Hurwicz iterations:** $\alpha_k \equiv 0$.
- **DFB:** $\mu_k \to +\infty$.

S(c)CP to D(i)FB

Objective

Find $\widehat{\mathbf{x}} \in \underset{\mathbf{x}\in\mathcal{H}}{\mathrm{Argmin}} \left\{ \mathsf{F}(\mathbf{x}) = \frac{1}{2}\|\mathbf{x}-\mathbf{z}\|_2^2 + g(\mathbf{D}\mathbf{x}) + \iota_C(\mathbf{x}) \right\}$

Algorithm: For $k = 0, 1, \ldots$

$$\left\lfloor \begin{array}{l} \mathbf{x}^{[k+1]} = \mathrm{P}_C\left(\mathbf{z} - \mathbf{D}^\top \mathbf{u}^{[k]}\right) \\ \mathbf{u}^{[k+1]} = \mathrm{prox}_{\tau_k(\nu g)^*}\left(\mathbf{u}^{[k]} + \tau_k \mathbf{D}\mathbf{x}^{[k+1]}\right) \end{array}\right.$$

- **S(c)CP:** Starting point.
- **Arrow-Hurwicz iterations:** $\alpha_k \equiv 0$.
- **DFB:** $\mu_k \to +\infty$.
- **DiFB:** Inertia step on the dual variable.

S(c)CP to D(i)FB

Objective

$$\text{Find } \widehat{\mathbf{x}} \in \underset{\mathbf{x}\in\mathcal{H}}{\text{Argmin}} \left\{ \mathsf{F}(\mathbf{x}) = \frac{1}{2}\|\mathbf{x}-\mathbf{z}\|_2^2 + g(\mathbf{D}\mathbf{x}) + \iota_C(\mathbf{x}) \right\}$$

Algorithm: For $k = 0, 1, \dots$

$$\left\lfloor \begin{array}{l} \mathbf{x}^{[k+1]} = \mathrm{P}_C\left(\mathbf{z} - \mathbf{D}^\top \mathbf{v}^{[k]}\right) \\ \mathbf{u}^{[k+1]} = \mathrm{prox}_{\tau_k(\nu g)^*}\left(\mathbf{u}^{[k]} + \tau_k \mathbf{D}\mathbf{x}^{[k+1]}\right) \\ \mathbf{v}^{[k+1]} = (1+\rho_k)\mathbf{u}^{[k+1]} - \rho_k \mathbf{u}^{[k]} \end{array} \right.$$

- **S(c)CP:** Starting point.
- **Arrow-Hurwicz iterations:** $\alpha_k \equiv 0$.
- **DFB:** $\mu_k \to +\infty$.
- **DiFB:** Inertia step on the dual variable.

Arrow- Hurwicz algorithm

The reformulation of Arrow-Hurwicz can be written as:

$$\begin{aligned} \mathrm{L}_{\mathbf{z},\nu,\Theta_k} : \quad & \mathcal{H} \times \mathcal{G} \to \mathcal{H} \\ & (\mathbf{x}^{[k]}, \mathbf{u}^{[k]}) \mapsto \mathrm{L}_{\mathbf{z},\Theta_{k,\mathcal{P}},\mathcal{P}}(\mathbf{x}, \mathrm{L}_{\Theta_{k,\mathcal{D}},\mathcal{D}}(\mathbf{x}^{[k]}, \mathbf{u}^{[k]})), \end{aligned}$$

with

$$\mathrm{L}_{\nu,\Theta_{k,\mathcal{D}},\mathcal{D}}(\mathbf{x},\mathbf{u}) = \mathrm{prox}_{\tau_k(\nu g)^*}\left(\tau_k \mathbf{D}\mathbf{x} + \mathbf{u}\right),$$

$$\mathrm{L}_{\mathbf{z},\Theta_{k,\mathcal{P}},\mathcal{P}}(\mathbf{x},\mathbf{u}) = \mathrm{P}_C\left(\frac{1}{1+\mu_k}\mathbf{x} - \frac{\mu_k}{1+\mu_k}\mathbf{D}^\top \mathbf{u} + \frac{\mu_k}{1+\mu_k}\mathbf{z}\right),$$

[Le, Repetti, Pustelnik, 2023]

Deep Arrow-Hurwicz building block

The reformulation of Arrow-Hurwicz can be written as:

$$\begin{aligned} \mathrm{L}_{\mathbf{z},\nu,\Theta_k}: \quad & \mathcal{H}\times\mathcal{G} \to \mathcal{H} \\ & (\mathbf{x}^{[k]},\mathbf{u}^{[k]}) \mapsto \mathrm{L}_{\mathbf{z},\Theta_{k,\mathcal{P}},\mathcal{P}}(\mathbf{x}^{[k]},\mathrm{L}_{\Theta_{k,\mathcal{D}},\mathcal{D}}(\mathbf{x}^{[k]},\mathbf{u}^{[k]})), \end{aligned}$$

with

$$\mathrm{L}_{\nu,\Theta_{k,\mathcal{D}},\mathcal{D}}(\mathbf{x},\mathbf{u}) = \mathrm{prox}_{\tau_k(\nu g)^*}\left(\tau_k \mathbf{D}_{k,\mathcal{D}}\mathbf{x} + \mathrm{Id}\mathbf{u}\right),$$

$$\mathrm{L}_{\mathbf{z},\Theta_{k,\mathcal{P}},\mathcal{P}}(\mathbf{x},\mathbf{u}) = \mathrm{P}_C\left(\frac{1}{1+\mu_k}\mathbf{x} - \frac{\mu_k}{1+\mu_k}\mathbf{D}_{k,\mathcal{P}}^\top\mathbf{u} + \frac{\mu_k}{1+\mu_k}\mathbf{z}\right),$$

[Le, Repetti, Pustelnik, 2023]

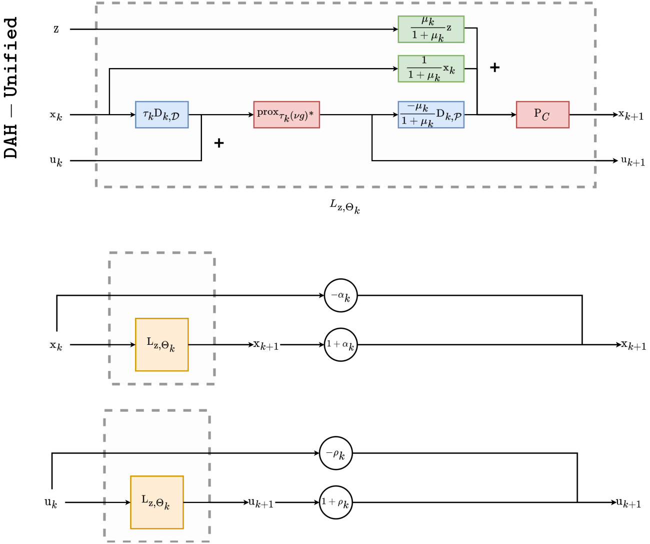


Figure: Architecture of the proposed `DAH-Unified` block for the k-th layer. `Inertial step` for ScCP (top) and DiFB (bottom).

	Θ_k	Comments
DDFB-LFO	$\mathbf{D}_{k,\mathcal{P}}$, $\mathbf{D}_{k,\mathcal{D}}$	absorb τ_k in $\mathbf{D}_{k,\mathcal{D}}$
DDiFB-LFO	$\mathbf{D}_{k,\mathcal{P}}$, $\mathbf{D}_{k,\mathcal{D}}$, α_k	fix α_k, and absorb τ_k in $\mathbf{D}_{k,\mathcal{D}}$
DDFB-LNO	$\mathbf{D}_{k,\mathcal{P}} = \mathbf{D}_{k,\mathcal{D}}^\top$	define $\tau_k = 1.99\\|\mathbf{D}_k\\|^{-2}$
DDiFB-LNO	$\mathbf{D}_{k,\mathcal{P}} = \mathbf{D}_{k,\mathcal{D}}^\top$	fix $\alpha_k = \frac{t_k-1}{t_{k+1}}$, $t_{k+1} = \frac{k+a-1}{a}$, $a > 2$, and $\tau_k = 0.99\\|\mathbf{D}_k\\|^{-2}$
DCP-LFO	$\mathbf{D}_{k,\mathcal{P}}$, $\mathbf{D}_{k,\mathcal{D}}$, μ	learn $\mu = \mu_0 = \cdots = \mu_K$, and absorb τ_k in $\mathbf{D}_{k,\mathcal{D}}$
DScCP-LFO	$\mathbf{D}_{k,\mathcal{P}}$, $\mathbf{D}_{k,\mathcal{D}}$, μ_0	learn μ_0, absorb τ_k in $\mathbf{D}_{k,\mathcal{D}}$, and fix $\alpha_k = (1+2\mu_k)^{-1/2}$, and $\mu_{k+1} = \alpha_k\mu_k$
DCP-LNO	$\mathbf{D}_{k,\mathcal{P}} = \mathbf{D}_{k,\mathcal{D}}^\top$, μ	learn $\mu = \mu_0 = \cdots = \mu_K$, and fix $\tau_k = 0.99\mu^{-1}\\|\mathbf{D}_k\\|^{-2}$
DScCP-LNO	$\mathbf{D}_{k,\mathcal{P}} = \mathbf{D}_{k,\mathcal{D}}^\top$, μ_k	learn μ_k, and fix $\alpha_k = (1+2\mu_k)^{-1/2}$, and $\tau_k = 0.99\mu_k^{-1}\\|\mathbf{D}_k\\|^{-2}$

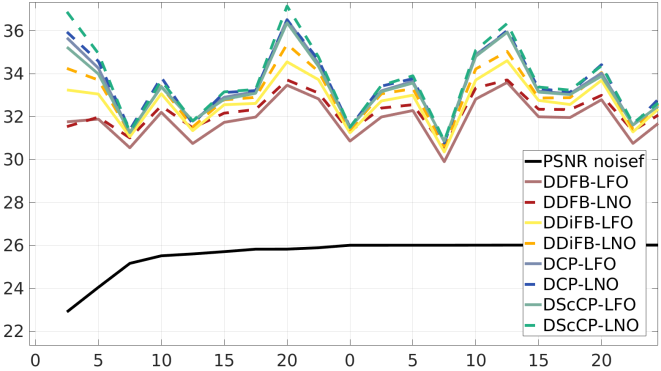
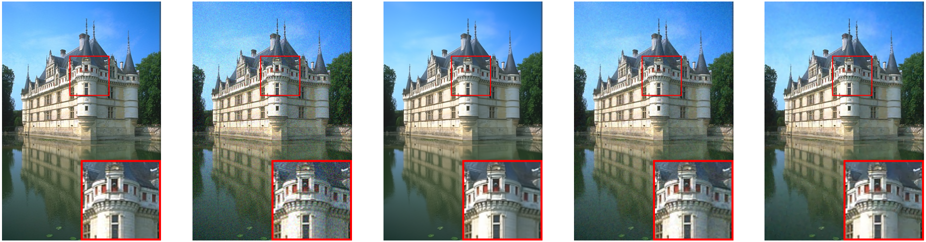


Figure: **Training Setting 2: Denoising performance.** PSNR values obtained with the proposed s (with $(K, J) = (20, 64)$), for 20 images of BSDS500 validation set, degraded with noise level $\delta = 0.05$.



Noisy	**DRUnet**	DDFB-LNO	DScCP-LNO
26.03dB	**35.81dB**	32.81dB	34.74dB

Denoising performance on Gaussian noise. Example of denoised images (and PSNR values) for Gaussian noise $\delta = 0.05$ obtained with DRUnet and the proposed DDFB-LNO and DScCP-LNO, with $(K, J) = (20, 64)$.

NN robustness

- Given an input $\mathbf{z}$ and a perturbation ϵ, the error on the output can be upper bounded :

$$\|f_\Theta(\mathbf{z}+\epsilon) - f_\Theta(\mathbf{z})\| \leqslant \chi \|\epsilon\|.$$

where χ certificated of the robustness.

- [Combettes, Pesquet, 2020]: χ can be upper bounded by:

$$\chi \leqslant \prod_{k=1}^{K} \Big(\|W_{k,\mathcal{P}}\|_S \times \|W_{k,\mathcal{D}}\|_S \Big).$$

- [Pesquet, Repetti, Terris, Wiaux, 2021, 2020]: tighter bound by Lipschitz continuity:

$$\chi \approx \max_{(\mathbf{z}_s)_{s\in\mathbb{I}}} \| \mathrm{J}\, f_\Theta(\mathbf{z}_s)\|_S.$$

where J denotes the Jacobian operator.

NN robustness

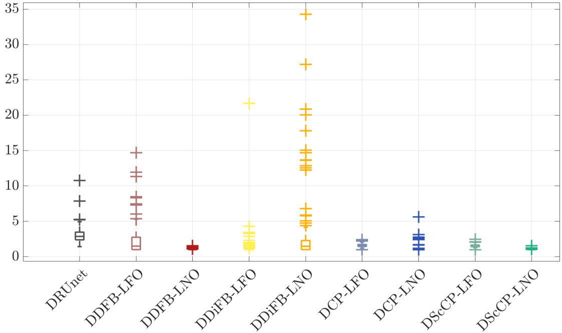


Figure: **Training Setting 2: Robustness.** Distribution of $(\| \mathrm{J}\, f_\Theta(\mathbf{z}_s) \|_S)_{s \in \mathbb{J}}$ for 100 images extracted from BSDS500 validation dataset $\mathbb{J}$, for the proposed PNNs and DRUnet.

PnP based on PNN

- **Variational approach** $(k \to \infty)$:

$$\mathrm{x}^{[k+1]} = \boxed{\mathrm{prox}_{\gamma g \circ \mathbf{D}}}(\mathrm{x}^{[k]} - \gamma \mathbf{A}^*(\mathbf{A}\mathrm{x}^{[k]} - \mathrm{z}))$$

- **PnP** $(k \to \infty)$:

$$\mathrm{x}^{[k+1]} = \boxed{f_{\widehat{\Theta}}}(\mathrm{x}^{[k]} - \gamma \mathbf{A}^*(\mathbf{A}\mathrm{x}^{[k]} - \mathrm{z}))$$

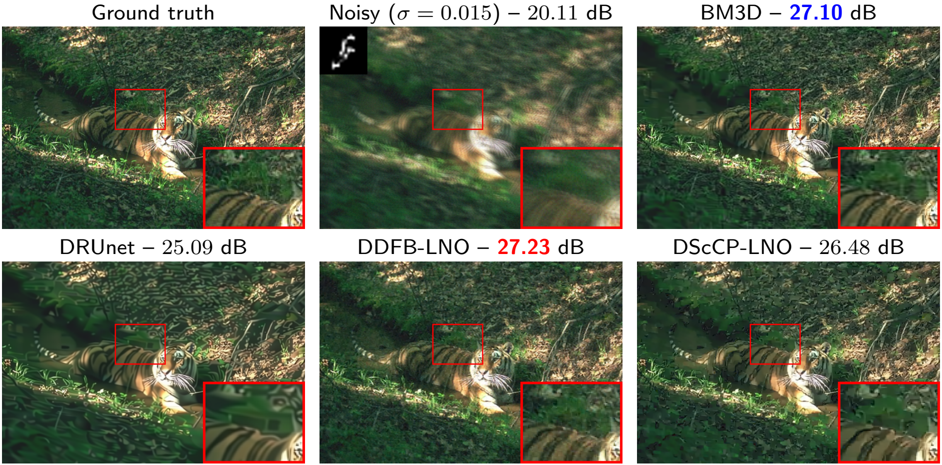


Figure: **Deblurring (Training Setting 2): Restoration performance.** Restoration example for $\sigma = 0.015$, with parameters $\gamma = 1.99$ and β chosen optimally for each scheme.

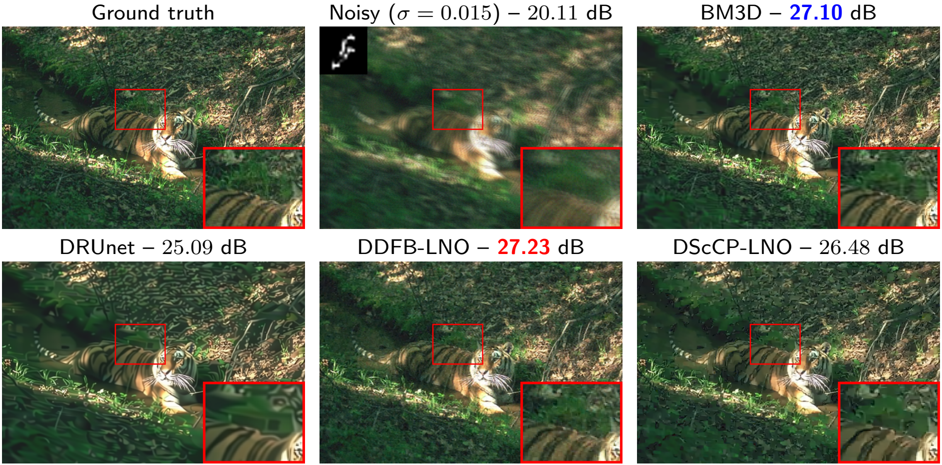


Figure: **Deblurring (Training Setting 2): Restoration performance.** Restoration example for $\sigma = 0.015$, with parameters $\gamma = 1.99$ and β chosen optimally for each scheme.

PnP based on PNN

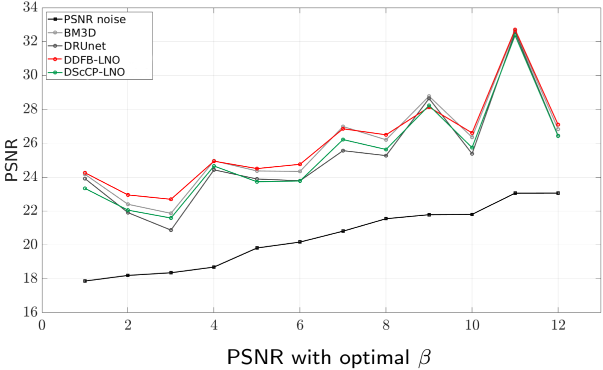


Figure: **Deblurring (Training Setting 2): Restoration performance.** Best PSNR values obtained with DDFB-LNO, DScCP-LNO, DRUnet and BM3D, on 12 images from BSDS500 validation set degraded, with $\sigma = 0.03$.

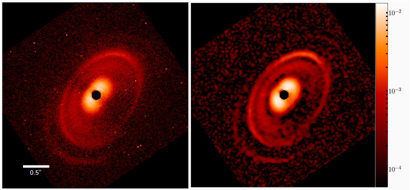
The study case of circumstellar environnement reconstruction

Direct model

➡ **Data**: $\mathbf{z} \in \mathbb{R}^M$ degraded version of an original image $\overline{\mathbf{x}} \in \mathbb{R}^N$:

$$\boxed{\mathbf{z} = \mathbf{A}\overline{\mathbf{x}} + \mathbf{w}}$$

- $\mathbf{A} : \mathbb{R}^{M \times N}$: linear degradation (e.g. a blur)
- $\mathbf{w}$: noise (e.g. Gaussian noise)



SPHERE-IRDIS

FB formulations

- **Variational approach** ($k \to \infty$):

$$\mathrm{x}^{[k+1]} = \boxed{\mathrm{prox}_{\gamma g \circ \mathbf{D}}} \left(\mathrm{x}^{[k]} - \gamma \mathbf{A}^* (\mathbf{A}\mathrm{x}^{[k]} - \mathrm{z})\right)$$

- **PnP** ($k \to \infty$):

$$\mathrm{x}^{[k+1]} = \boxed{f_{\widehat{\Theta}}} \left(\mathrm{x}^{[k]} - \gamma \mathbf{A}^* (\mathbf{A}\mathrm{x}^{[k]} - \mathrm{z})\right)$$

- **Unfolded** (k fixed):

$$\mathbf{x}^{[k+1]} = \boxed{f_{\Theta}} \left(\mathbf{x}^{[k]} - \gamma \mathbf{A}^* (\mathbf{A}\mathrm{x}^{[k]} - \mathrm{z})\right)$$

Primal-Dual formulations

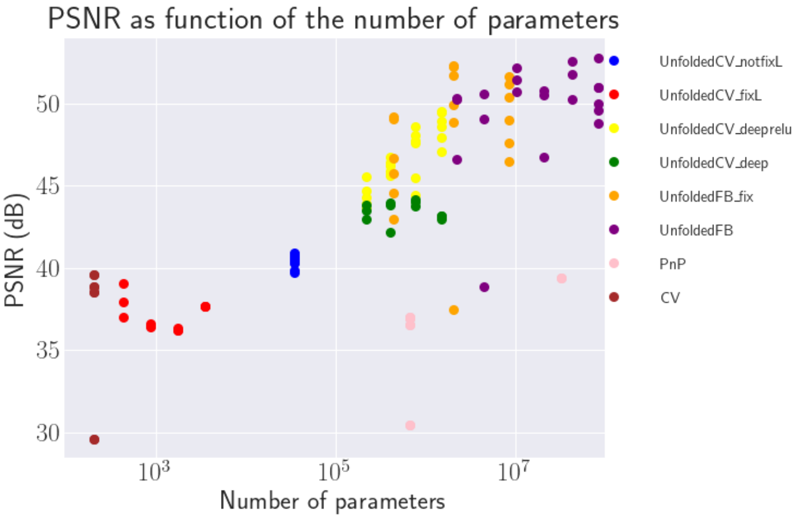
- **Variational approach** ($k \to \infty$):

$$\mathbf{x}^{[k+1]} = \mathbf{x}^{[k]} - \tau \mathbf{A}^*(\mathbf{A}\mathbf{x}^{[k]} - \mathbf{z}) - \tau \mathbf{D}^* \mathbf{u}^{[k]}$$
$$\mathbf{u}^{[k+1]} = \mathrm{prox}_{\gamma \|\cdot\|_1^*}\big(\mathbf{u}^{[k]} + \gamma \mathbf{D}(2\mathbf{x}^{[k+1]} - \mathbf{x}^{[k]})\big)$$

- **Unfolded** (k fixed):

$$\mathbf{x}^{[k+1]} = \mathbf{x}^{[k]} - \tau_k \mathbf{A}^*(\mathbf{A}\mathbf{x}^{[k]} - \mathbf{z}) - \tau_k \mathbf{D}_k^* \mathbf{u}^{[k]}$$
$$\mathbf{u}^{[k+1]} = \mathrm{prox}_{\gamma_k \|\cdot\|_1^*}\big(\mathbf{u}^{[k]} + \gamma_k \mathbf{D}_k(2\mathbf{x}^{[k+1]} - \mathbf{x}^{[k]})\big)$$

Summary


PSNR as function of the number of parameters
PSNR (dB)
Number of parameters
50
45
40
35
30
10^3
10^5
10^7
UnfoldedCV_notfixL
UnfoldedCV_fixL
UnfoldedCV_deeprelu
UnfoldedCV_deep
UnfoldedFB_fix
UnfoldedFB
PnP
CV

Conclusions and open questions

- PNN gives an intuition to build a neural network.
- PNN leads to better results than PnP. Faster solution even weaker “convergence” guarantees.
- Stuck on standard variational formulation not sufficient.
- More complex penalization (non-linearities).

Refs

- M. Jiu and N. Pustelnik, A deep primal-dual proximal network for image restoration, IEEE JSTSP Special Issue on Deep Learning for Image/Video Restoration and Compression, vol. 15, no. 2,pp. 190–203, Feb. 2021.
- M. Jiu, N. Pustelnik Alternative Design of DeepPDNet in the Context of Image Restoration, IEEE Signal Processing Letters, vol. 29, pp. 932 - 936, 2022.
- H.T.V. Le, N. Pustelnik, M Foare, The faster proximal algorithm, the better unfolded deep learning architecture? The study case of image denoising, EUSIPCO, Belgrade, Serbie, 29 Aug - 2 Sept. 202.
- H.T.V. Le, A. Repetti, N. Pustelnik, PNN: From proximal algorithms to robust unfolded image denoising networks and Plug-and-Play methods, arXiv:2308.03139, 2023.