Proximal Neural Networks: Wedding Variational Methods and Artificial Intelligence

VI – Conclusion and toolbox presentation

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

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

Tutorial – Eusipco 2025 – Palermo, Italy

Unified framework

Inference framework: feed-forward NN

$$(\forall \boldsymbol{x}^{[0]} \in \mathbb{R}^{N_0}) \qquad \begin{aligned} \boldsymbol{x}^{[K]} &= \mathfrak{L}_{\Theta}^{K}(\boldsymbol{x}^{[0]}) \\ &= \mathfrak{T}_{\Theta_K} \circ \ldots \circ \mathfrak{T}_{\Theta_1}(\boldsymbol{x}^{[0]}), \end{aligned}$$

Layer/iteration

$$\mathfrak{T}_{\Theta_k} \colon \mathbb{R}^{N_{k-1}} \to \mathbb{R}^{N_k} \colon \boldsymbol{x} \mapsto \mathfrak{D}_{\Lambda_k}(\mathbf{L}_k \boldsymbol{x} + \boldsymbol{b}_k),$$

- $\mathbf{L}_k \colon \mathbb{R}^{N_{k-1}} \to \mathbb{R}^{N_k}$: linear operator,
- $\boldsymbol{b}_k \in \mathbb{R}^{N_k}$: shift parameter,
- $\mathfrak{D}_{\Lambda_k} \colon \mathbb{R}^{N_k} \to \mathbb{R}^{N_k}$: nonlinear operator parametrized by Λ_k.

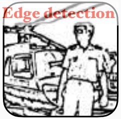
Parameters: $\Theta = \cup_{k=1}^{K} \Theta_k$ with $\Theta_k = \{\Lambda_k, \mathbf{L}_k, \boldsymbol{b}_k\}$.

Unified framework: Unfolded neural networks

$$\underbrace{\overbrace{\mathfrak{T}_{\Theta_K}}^{\mathfrak{D}_{\Lambda_K}(\mathbf{L}_K \cdot + \boldsymbol{b}_K)} \circ \ldots \circ \overbrace{\mathfrak{T}_{\Theta_1}}^{\mathfrak{D}_{\Lambda_1}(\mathbf{L}_1 \cdot + \boldsymbol{b}_1)}}$$

$$\boldsymbol{x}^{[0]} \longrightarrow \boxed{\mathfrak{L}^K_\Theta} \longrightarrow \boldsymbol{x}^{[K]}$$

Variational approaches and Proximal algorithms	
K large	$\mathfrak{D}_{\Lambda_k}$: Proximity operator
Plug-and-Play	
K large	$\mathfrak{D}_{\Lambda_k}$: NN denoiser
Unfolded Neural Network	
K small (i.e. ~ 20)	**Learned linear operator** $\mathfrak{D}_{\Lambda_k}$: Proximity operator **Learned nonlinear operator** $\mathfrak{D}_{\Lambda_k}$: NN denoiser
End-to-end	
K medium (i.e. ~ 50)	$\mathfrak{D}_{\Lambda_k}$: Activation function



Challenges for the next years

Theoretical challenges:

- Interpretation of output of (unfolded) neural networks
- Develop mathematical framework to better assess robustness of (unfolded) neural networks

Computational challenges:

- Boost expressivity of PnP and unfolded methods
- Further explore real applications

Societal challenges:

- Convince end-users that model-informed deep learning methods such as PnP and unfolded networks are reliable for decision-making processes
- Develop effective quantification measures for environmental impact of data-driven methods

  $\rightsquigarrow$ Reduce environmental impact by adoption of frugal learning strategies

Soon(ish) available...

From Iterative Methods to Model-Informed Architectures for Data Science.
A. Repetti, N. Pustelnik, J.-C. Pesquet.
To be submitted

$\rightsquigarrow$ *Review article from proximal methods to PnP and unfolded approaches*

Python toolbox

Playing with inverse imaging problems

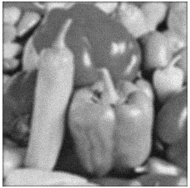
Forward model

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

- $\overline{\boldsymbol{x}} \in \mathcal{H}$ original unknown image
- $\boldsymbol{z} \in \mathcal{G}$ degraded measurements
- $\mathbf{A} \colon \mathcal{H} \to \mathcal{G}$ corresponds to the linear measurement operator
- $\mathcal{D} \colon \mathcal{G} \to \mathcal{G}$ models the degradation noise

OBJECTIVE: Find an estimate $\widehat{\boldsymbol{x}} \in \mathcal{H}$ of the original image $\overline{\boldsymbol{x}}$ from the measurements $\boldsymbol{z}$

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



Observation

Estimate

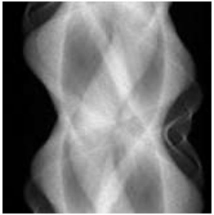
Forward model

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

- $\overline{\boldsymbol{x}} \in \mathcal{H}$ original unknown image
- $\boldsymbol{z} \in \mathcal{G}$ degraded measurements
- $\mathbf{A}\colon \mathcal{H} \to \mathcal{G}$ corresponds to the linear measurement operator
- $\mathcal{D}\colon \mathcal{G} \to \mathcal{G}$ models the degradation noise

OBJECTIVE: Find an estimate $\widehat{\boldsymbol{x}} \in \mathcal{H}$ of the original image $\overline{\boldsymbol{x}}$ from the measurements $\boldsymbol{z}$

EXAMPLE: Medical imaging (CT)



Observation

$\overset{?}{\rightsquigarrow}$

Estimate

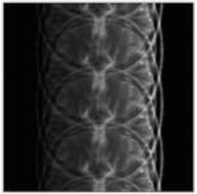
Forward model

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

- $\overline{x} \in \mathcal{H}$ original unknown image
- $z \in \mathcal{G}$ degraded measurements
- $\mathbf{A} \colon \mathcal{H} \to \mathcal{G}$ corresponds to the linear measurement operator
- $\mathcal{D} \colon \mathcal{G} \to \mathcal{G}$ models the degradation noise

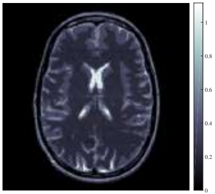
OBJECTIVE: Find an estimate $\widehat{x} \in \mathcal{H}$ of the original image $\overline{x}$ from the measurements z

EXAMPLE: Magnetic resonance imaging in medicine



Observation

$\overset{?}{\rightsquigarrow}$



Estimate

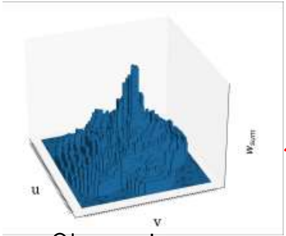
Forward model

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

- $\overline{x} \in \mathcal{H}$ original unknown image
- $z \in \mathcal{G}$ degraded measurements
- $\mathbf{A} \colon \mathcal{H} \to \mathcal{G}$ corresponds to the linear measurement operator
- $\mathcal{D} \colon \mathcal{G} \to \mathcal{G}$ models the degradation noise

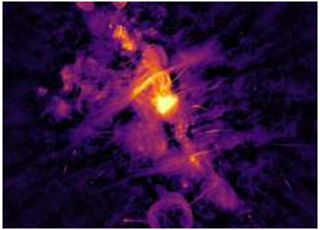
OBJECTIVE: Find an estimate $\widehat{x} \in \mathcal{H}$ of the original image $\overline{x}$ from the measurements z

EXAMPLE: Radio-interferometric imaging in astronomy



Observation

$\overset{?}{\rightsquigarrow}$



Estimate

Forward model: Examples for measurement operator $\mathbf{A}$

- **Deconvolution**
  - Most common imaging model encountered in the literature, also known as deblurring
  - $\mathbf{A}$ associated with a 2D or 3D convolution (or blur) kernel
  - For example to model motion between the scene and the camera, for defocusing of an optical imaging system, or to model atmospheric turbulence (e.g., in astronomical or satellite imaging)

Forward model: Examples for measurement operator $\mathbf{A}$

- **Deconvolution**
- **Subsampling/inpainting**
  - $\mathbf{A}$ corresponds to a *mask* operator, only selecting visible pixels
  - Used to model missing information, for example in the context of low-resolution acquisition (i.e., *super-resolution*), or from an occultation process (i.e., *inpainting*).

Forward model: Examples for measurement operator $\mathbf{A}$

- **Deconvolution**
- **Subsampling/inpainting**
- **Fourier sampling**
  - $\mathbf{A}$ can be decomposed into two linear operators: the discrete Fourier transform (i.e., 2D FFT), and the subsampling operator
    $\rightsquigarrow$ In a realistic setting, the Fourier transform should act in a continuous space (i.e., using *non-uniform FFT*)
  - Encountered for instance in medicine for magnetic resonance imaging, and in astronomy for radio-interferometric imaging

Forward model: Examples for measurement operator $\mathbf{A}$

- **Deconvolution**
- **Subsampling/inpainting**
- **Fourier sampling**
- **Radon transform**
  - $\mathbf{A}$ produces a 2D (or 3D) sinogram
  - Usually used to approximate tomography projection operators as encountered for Positron Emission Tomography (PET) or Computed Tomography (CT)
- **etc.**

Forward model: Link between degradation $\mathcal{D}$ and data fidelity h_z

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

VARIATIONAL FORMULATION: Define the estimate $\widehat{\boldsymbol{x}}$ as $\mathbf{0} \in \partial h_{\boldsymbol{z}}(\widehat{\boldsymbol{x}}) + \lambda \partial g(\widehat{\boldsymbol{x}})$

- **Additive white Gaussian noise (AWGN)**
  - Most common type of noise encountered in practice
  - Model boils down to $\boldsymbol{z} = \mathbf{A}\overline{\boldsymbol{x}} + \boldsymbol{\varepsilon}$ where $\boldsymbol{\varepsilon} \in \mathcal{G}$ is a realization of an independent identically distributed random Gaussian variable with zero mean and standard deviation $\sigma > 0$ (or diagonal covariance $\boldsymbol{\Sigma}$)
  - $h_{\boldsymbol{z}}(\boldsymbol{x}) = \frac{1}{2\sigma^2}\|\mathbf{A}\boldsymbol{x} - \boldsymbol{z}\|^2$

Forward model: Link between degradation $\mathcal{D}$ and data fidelity h_z

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

VARIATIONAL FORMULATION: Define the estimate $\widehat{\boldsymbol{x}}$ as $\mathbf{0} \in \partial h_{\boldsymbol{z}}(\widehat{\boldsymbol{x}}) + \lambda \partial g(\widehat{\boldsymbol{x}})$

- **Additive white Gaussian noise (AWGN)**
- **Coloured Gaussian noise**
  - More general version of AWGN where the the covariance Σ of the noise is not diagonal
  - $h_{\boldsymbol{z}}(\boldsymbol{x}) = \frac{1}{2}\|\mathbf{A}\boldsymbol{x} - \boldsymbol{z}\|^2_{\Sigma^{-1}}$

Forward model: Link between degradation $\mathcal{D}$ and data fidelity h_z

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

VARIATIONAL FORMULATION: Define the estimate $\widehat{\boldsymbol{x}}$ as $\mathbf{0} \in \partial h_{\boldsymbol{z}}(\widehat{\boldsymbol{x}}) + \lambda \partial g(\widehat{\boldsymbol{x}})$

- **Additive white Gaussian noise (AWGN)**
- **Coloured Gaussian noise**
- **Poisson noise**
    - Often used to model noise in low-photon-count imaging techniques
      $\rightsquigarrow$ Poisson distribution is a counting procedure that can express the number of photons received by the sensor in a given time interval
    - $h_{\boldsymbol{z}}(\boldsymbol{x}) = \sum_m \big([\mathbf{A}\boldsymbol{x}]_m - \boldsymbol{z}_m \log([\mathbf{A}\boldsymbol{x}]_m)\big)$

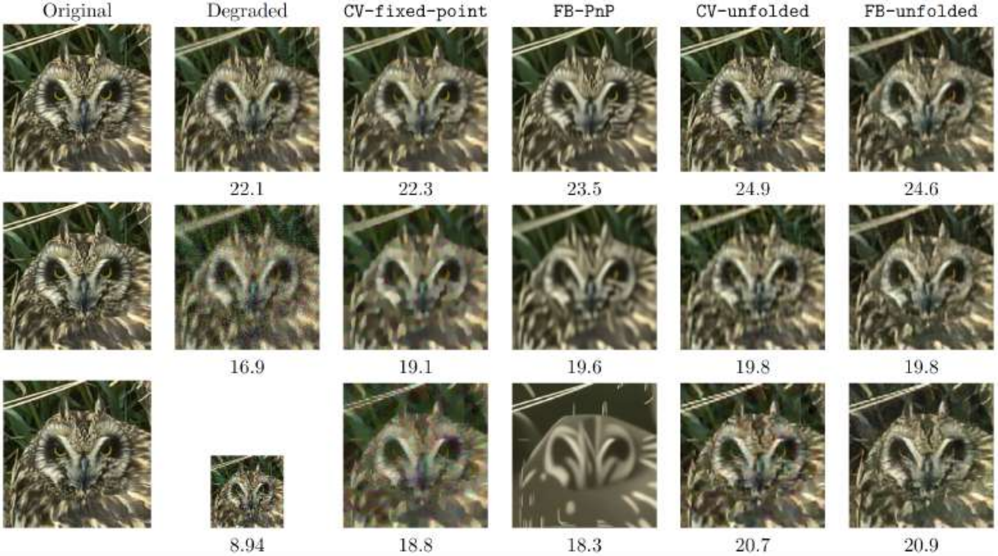
Forward model: Link between degradation $\mathcal{D}$ and data fidelity h_z

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

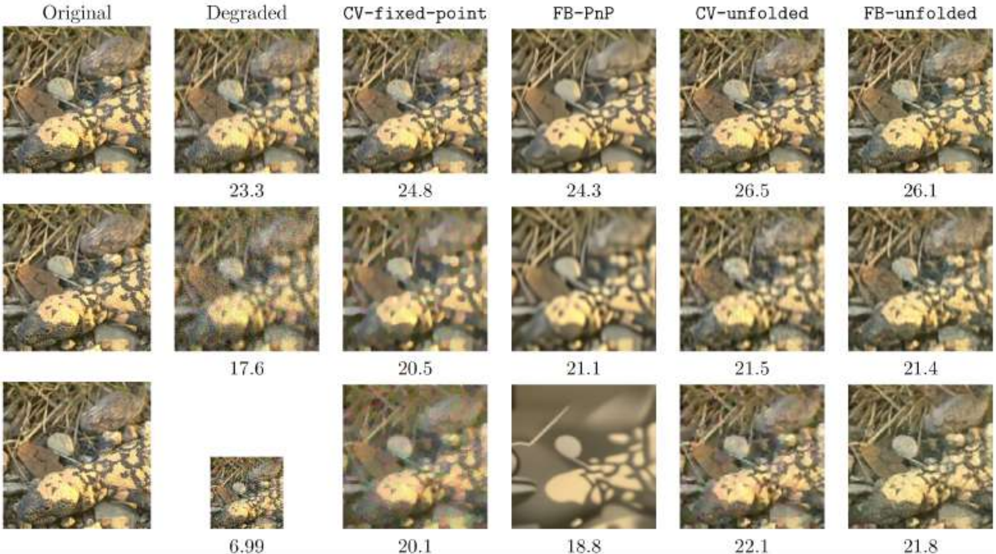
VARIATIONAL FORMULATION: Define the estimate $\widehat{\boldsymbol{x}}$ as $\mathbf{0} \in \partial h_{\boldsymbol{z}}(\widehat{\boldsymbol{x}}) + \lambda \partial g(\widehat{\boldsymbol{x}})$

- **Additive white Gaussian noise (AWGN)**
- **Coloured Gaussian noise**
- **Poisson noise**
- **Uniformly bounded noise**
  - $\mathcal{D}$ introduces a bounded noise in the sense that there exists $\varepsilon > 0$ such that $\|\mathbf{A}\overline{\boldsymbol{x}} - \boldsymbol{z}\|^2 \leq \varepsilon$
  - $h_{\boldsymbol{z}}(\boldsymbol{x}) = \iota_{\mathcal{B}_2(\boldsymbol{z},\epsilon)}(\mathbf{A}\boldsymbol{x})$ *(Morozov formulation)*
- **etc.**

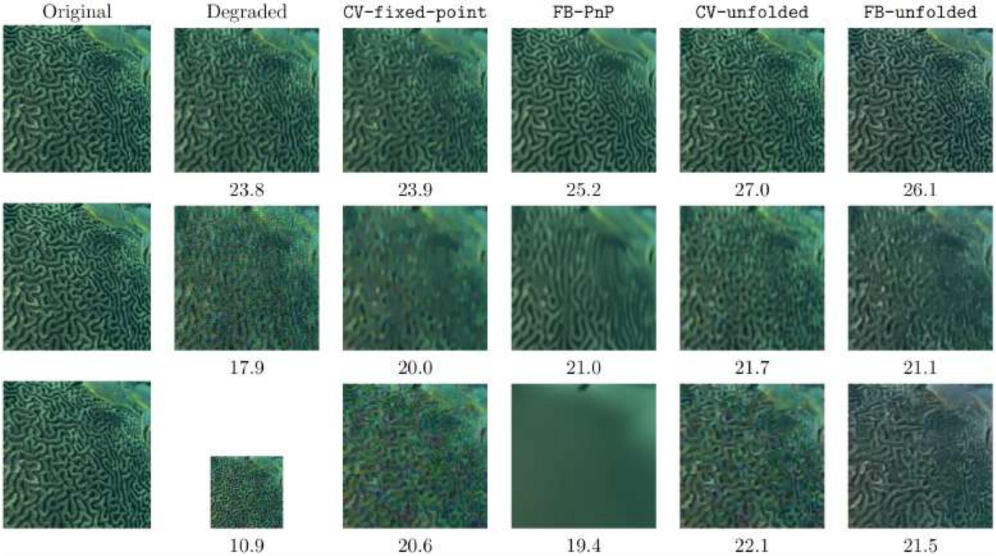
Problem solvers: Results



Problem solvers: Results

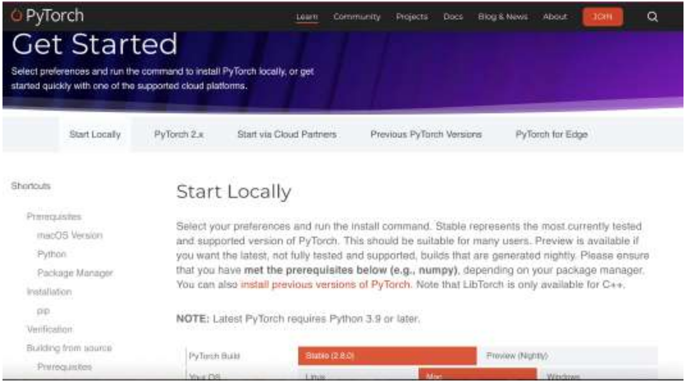


Problem solvers: Results



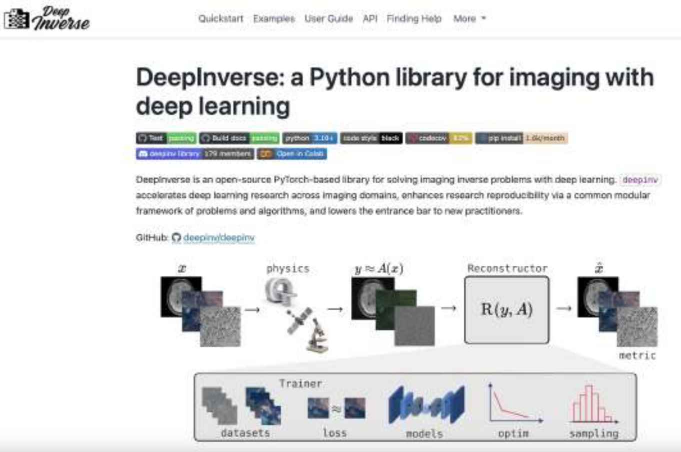
Python Toolbox: Based on DeepInverse and Pytorch

Goal: Find $\widehat{\Theta} \in \operatorname*{Argmin}_{\Theta} \; \frac{1}{|\mathbb{I}|} \sum_{j\in\mathbb{I}} \ell\big(\overline{\boldsymbol{x}}_j, \mathfrak{L}^K_\Theta(\boldsymbol{z}_j)\big).$



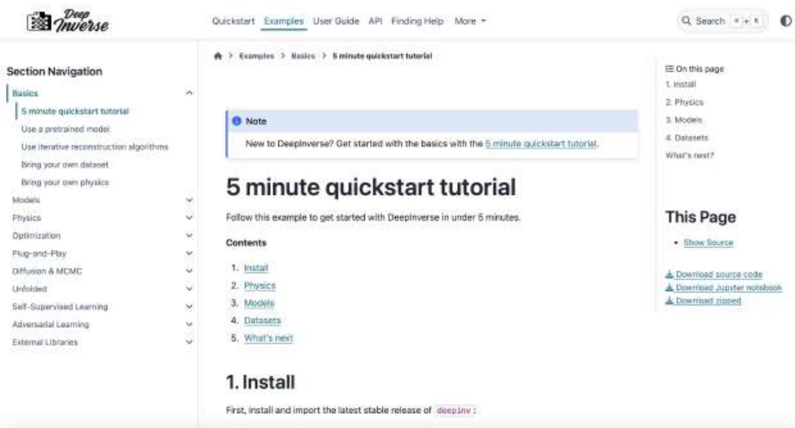
Python Toolbox: Based on DeepInverse and Pytorch

Goal: Design $\mathfrak{L}_{\Theta}^{K}$



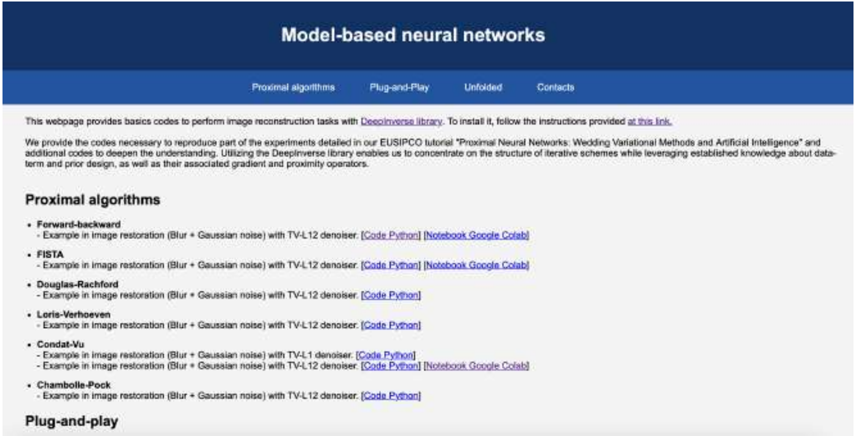
Python Toolbox: Based on DeepInverse and Pytorch

Goal: Design $\mathfrak{L}_{\Theta}^{K}$



Python Toolbox

https://perso.ens-lyon.fr/nelly.pustelnik/PNN/



Model-based neural networks

Proximal algorithms Plug-and-Play Unfolded Contacts

This webpage provides basics codes to perform image reconstruction tasks with Deepinverse library. To install it, follow the instructions provided at this link.

We provide the codes necessary to reproduce part of the experiments detailed in our EUSIPCO tutorial "Proximal Neural Networks: Wedding Variational Methods and Artificial Intelligence" and additional codes to deepen the understanding. Utilizing the DeepInverse library enables us to concentrate on the structure of iterative schemes while leveraging established knowledge about data-term and prior design, as well as their associated gradient and proximity operators.

Proximal algorithms

- **Forward-backward**
  - Example in image restoration (Blur + Gaussian noise) with TV-L12 denoiser. [Code Python] [Notebook Google Colab]
- **FISTA**
  - Example in image restoration (Blur + Gaussian noise) with TV-L12 denoiser. [Code Python] [Notebook Google Colab]
- **Douglas-Rachford**
  - Example in image restoration (Blur + Gaussian noise) with TV-L12 denoiser. [Code Python]
- **Loris-Verhoeven**
  - Example in image restoration (Blur + Gaussian noise) with TV-L12 denoiser. [Code Python]
- **Condat-Vu**
  - Example in image restoration (Blur + Gaussian noise) with TV-L1 denoiser. [Code Python]
  - Example in image restoration (Blur + Gaussian noise) with TV-L12 denoiser. [Code Python] [Notebook Google Colab]
- **Chambolle-Pock**
  - Example in image restoration (Blur + Gaussian noise) with TV-L12 denoiser. [Code Python]

Plug-and-play