

Introduction to Plug-and-Play methods for image restoration and to their convergence analysis

Samuel Hurault

Institut de Mathématiques de Bordeaux



Workshop on Mathematical Models for Plug-and-play Image Restoration

Human: I want to understand Plug-and-play methods for image restoration

AI: Sure thing! Plug-and-play methods for image restoration use deep learning algorithms to restore degraded images. These methods use the training data to learn the mapping between the input image and the output image, allowing them to restore images with minimal user intervention.

Human: What are the maths behind ?

AI: Plug and play methods for image restoration are based on variational-optimization methods, which are used to solve minimization problems. These methods typically use a data fidelity term and a regularization term, which correspond to the input and output images, respectively. The optimization process then minimizes the difference between the two. Additionally, the denoiser is used as an implicit prior, which helps the AI accurately restore details and color. Does this answer your question?

Human: Do I still need to do my presentation on Plug-and-play ?

AI: Yes, it is still advisable to do a presentation on Plug-and-play methods for image restoration. It will help you better understand the underlying concepts and give you a deeper insight into the topic. Furthermore, you will be able to explain the concepts to your audience in an effective and engaging way.

Human: How do I start my presentation ?

AI: It's great that you are preparing for your presentation on Plug-and-play methods for image restoration. The best way to start is by explaining the types of images that can be restored using these methods. After that, you can introduce the denoiser as an implicit prior and explain how it helps to restore details and color. Then, you can discuss the variational-optimization methods used in Plug-and-play methods and how they help to create a mapping between the input and output images. Finally, you can conclude the presentation by summarizing the key points and highlighting the advantages of using Plug-and-play methods for image restoration

Variational Image Restoration

PnP Mathematical Formulation

- 1st order optimization

- Denoising as Implicit Prior

- PnP and RED algorithms

Convergence analysis

- Fixed-point convergence

- Global convergence

Algorithm in practice

New trends in PnP image restoration

- Deep Unfolding

- Denoising diffusion models

Image Inverse Problems

Find x from observation $y = Ax + \xi$

- $y \in \mathbb{R}^m$ observation
- $x \in \mathbb{R}^n$ unknown input
- $A \in \mathbb{R}^{m \times n}$ degradation operator
- ξ random noise, generally $\xi \sim \mathcal{N}(0, \sigma^2 \text{Id}_m)$

Image Inverse Problems

Find x from observation $y = Ax + \xi$

- $y \in \mathbb{R}^m$ observation
- $x \in \mathbb{R}^n$ unknown input
- $A \in \mathbb{R}^{m \times n}$ degradation operator
- ξ random noise, generally $\xi \sim \mathcal{N}(0, \sigma^2 \text{Id}_m)$

Denoising:

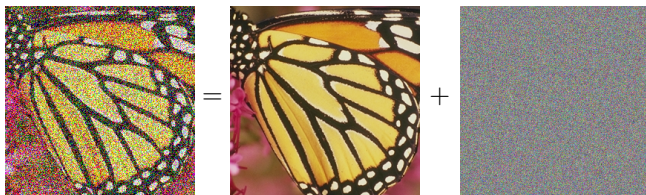


Image Inverse Problems

Find x from observation $y = Ax + \xi$

- $y \in \mathbb{R}^m$ observation
- $x \in \mathbb{R}^n$ unknown input
- $A \in \mathbb{R}^{m \times n}$ degradation operator
- ξ random noise, generally $\xi \sim \mathcal{N}(0, \sigma^2 \text{Id}_m)$

Deblurring:



Image Inverse Problems

Find x from observation $y = Ax + \xi$

- $y \in \mathbb{R}^m$ observation
- $x \in \mathbb{R}^n$ unknown input
- $A \in \mathbb{R}^{m \times n}$ degradation operator
- ξ random noise, generally $\xi \sim \mathcal{N}(0, \sigma^2 \text{Id}_m)$

Super-resolution:



Image Inverse Problems

Find x from observation $y = Ax + \xi$

- $y \in \mathbb{R}^m$ observation
- $x \in \mathbb{R}^n$ unknown input
- $A \in \mathbb{R}^{m \times n}$ degradation operator
- ξ random noise, generally $\xi \sim \mathcal{N}(0, \sigma^2 \text{Id}_m)$

Inpainting:

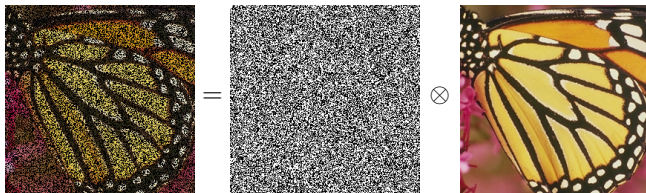


Image Inverse Problems

Find x from observation $y = Ax + \xi$

- $y \in \mathbb{R}^m$ observation
- $x \in \mathbb{R}^n$ unknown input
- $A \in \mathbb{R}^{m \times n}$ degradation operator
- ξ random noise, generally $\xi \sim \mathcal{N}(0, \sigma^2 \text{Id}_m)$

Compressed Sensing : e.g. Magnetic Resonance Imaging (MRI)

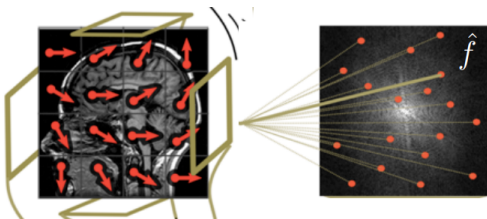


Image Inverse Problems

Find x from observation $y \sim p(y|x)$

- $y \in \mathbb{R}^m$ observation
- $x \in \mathbb{R}^n$ unknown input
- $p(y|x)$ forward model

Computed Tomography :



Maximum A-Posteriori

Find x from observation $y \sim p(y|x)$

$$\begin{aligned}x^* &\in \arg \max_{x \in \mathbb{R}^n} p(x|y) \\&= \arg \max_{x \in \mathbb{R}^n} \frac{p(y|x)p(x)}{p(y)} \\&= \arg \min_{x \in \mathbb{R}^n} -\log p(y|x) - \log p(x) \\&= \arg \min_{x \in \mathbb{R}^n} f(x) + \lambda g(x)\end{aligned}$$

Maximum A-Posteriori

Find x from observation $y \sim p(y|x)$

Maximum A-Posteriori

$$x^* \in \arg \min_{x \in \mathbb{R}^n} f(x) + \lambda g(x)$$

Maximum A-Posteriori

Find x from observation $y \sim p(y|x)$

Maximum A-Posteriori

$$x^* \in \arg \min_{x \in \mathbb{R}^n} f(x) + \lambda g(x)$$

$$\iff \arg \min_{x \in \mathbb{R}^n} \begin{array}{c} \text{data-fidelity} \\ f(x) = -\log p(y|x) \end{array}$$

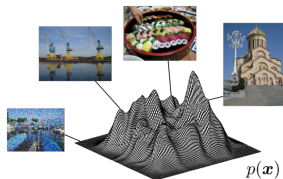
Maximum A-Posteriori

Find x from observation $y \sim p(y|x)$

Maximum A-Posteriori

$$x^* \in \arg \min_{x \in \mathbb{R}^n} f(x) + \lambda g(x)$$

$$\iff \arg \min_{x \in \mathbb{R}^n} \begin{array}{c} \text{data-fidelity} \\ f(x) = -\log p(y|x) \end{array} + \begin{array}{c} \text{regularization} \\ g \propto -\log p \end{array}$$



Maximum A-Posteriori

Find x from observation $y \sim p(y|x)$

Maximum A-Posteriori

$$x^* \in \arg \min_{x \in \mathbb{R}^n} f(x) + \lambda g(x)$$

$$\iff \arg \min_{x \in \mathbb{R}^n} \begin{array}{c} \text{data-fidelity} \\ f(x) = -\log p(y|x) \end{array} + \begin{array}{c} \text{regularization} \\ g \propto -\log p \end{array}$$



A variety of data-fidelity terms f

- Assuming Gaussian noise model $\xi \sim \mathcal{N}(0, \sigma^2 \text{Id})$,

$$f(x) = -\log p(y|x) = \frac{1}{2\sigma^2} \|Ax - y\|^2$$

\rightarrow *convex and smooth f , non-strongly convex in general*

A variety of data-fidelity terms f

- Assuming Gaussian noise model $\xi \sim \mathcal{N}(0, \sigma^2 \text{Id})$,

$$f(x) = -\log p(y|x) = \frac{1}{2\sigma^2} \|Ax - y\|^2$$

\rightarrow *convex and smooth f , non-strongly convex in general*

- With other regularities
 - Laplace / Poisson noise model \rightarrow *non-smooth f*
 - Phase retrieval \rightarrow *nonconvex f*

A variety of data-fidelity terms f

- Assuming Gaussian noise model $\xi \sim \mathcal{N}(0, \sigma^2 \text{Id})$,

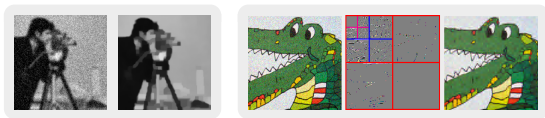
$$f(x) = -\log p(y|x) = \frac{1}{2\sigma^2} \|Ax - y\|^2$$

\rightarrow *convex and smooth f , non-strongly convex in general*

- With other regularities
 - Laplace / Poisson noise model \rightarrow *non-smooth f*
 - Phase retrieval \rightarrow *nonconvex f*
- More complex non-linear modeling of real complex physical systems (X-ray CT, electron-microscopy ...)

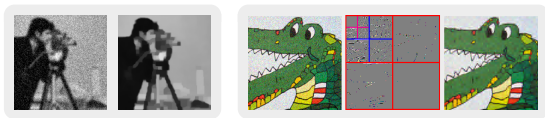
A variety of explicit image priors

Design an explicit regularization on image features: TV [Rudin et al. '92], Fourier spectrum [Ruderman '94], Wavelet sparsity [Mallat '09]

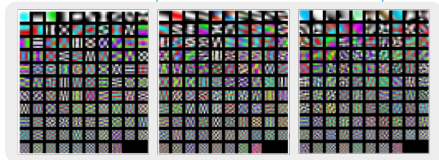


A variety of explicit image priors

Design an explicit regularization on image features: TV [Rudin et al. '92], Fourier spectrum [Ruderman '94], Wavelet sparsity [Mallat '09]

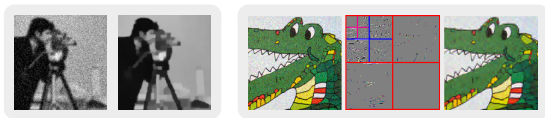


Learn an explicit prior on patches: GMM [Zoran and Weiss '11]

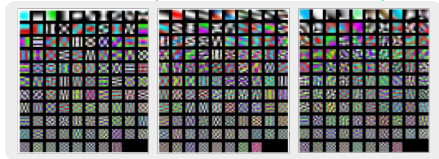


A variety of explicit image priors

Design an explicit regularization on image features: TV [Rudin et al. '92], Fourier spectrum [Ruderman '94], Wavelet sparsity [Mallat '09]



Learn an explicit prior on patches: GMM [Zoran and Weiss '11]

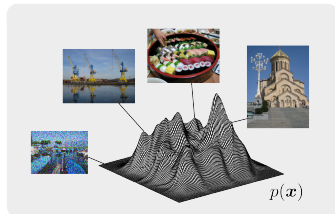


Learn an explicit deep prior
on full images (generative models):

VAE [Kingma & Welling, '13]

Normalizing flows [Dinh et al. '15]

Score-based ODE [Song et al '21]



PnP motivations

Find $x^* \in \arg \min_{x \in \mathbb{R}^n} \text{Data-fidelity}(x) + \text{Regularization}(x)$

- **Decouple** data-fidelity and regularization optimizations in iterative algorithms. [Combette & Pesquet '11] [Zoran & Weiss '11]

PnP motivations

Find $x^* \in \arg \min_{x \in \mathbb{R}^n} \text{Data-fidelity}(x) + \text{Regularization}(x)$

- **Decouple** data-fidelity and regularization optimizations in iterative algorithms. [Combette & Pesquet '11] [Zoran & Weiss '11]
- ✗ Previous explicit priors are limited.

PnP motivations

Find $x^* \in \arg \min_{x \in \mathbb{R}^n} \text{Data-fidelity}(x) + \text{Regularization}(x)$

- **Decouple** data-fidelity and regularization optimizations in iterative algorithms. [Combette & Pesquet '11] [Zoran & Weiss '11]
- ✗ Previous explicit priors are limited.
- ✓ **Image Denoising** : easier / well-understood.

PnP motivations

Find $x^* \in \arg \min_{x \in \mathbb{R}^n} \text{Data-fidelity}(x) + \text{Regularization}(x)$

- **Decouple** data-fidelity and regularization optimizations in iterative algorithms. [Combette & Pesquet '11] [Zoran & Weiss '11]

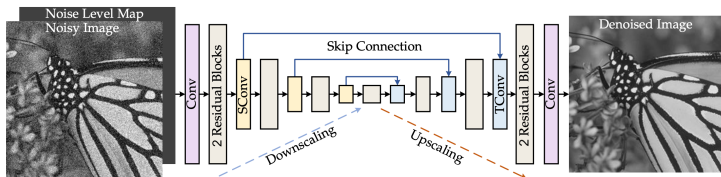
✗ Previous explicit priors are limited.

✓ **Image Denoising** : easier / well-understood.

→ State-of-the art denoisers **without explicit prior**

Filtering methods [Dabov et al. '07] [Lebrun et al. '13]

Deep denoisers [Zhang et al. '16, '17, '21] [Song et al. '19]



PnP motivations

Find $x^* \in \arg \min_{x \in \mathbb{R}^n} \text{Data-fidelity}(x) + \text{Regularization}(x)$

- **Decouple** data-fidelity and regularization optimizations in iterative algorithms. [Combette & Pesquet '11] [Zoran & Weiss '11]
- ✗ Previous explicit priors are limited.
- ✓ **Image Denoising** : easier / well-understood.
 - State-of-the art denoisers **without explicit prior**
 - Filtering methods [Dabov et al. '07] [Lebrun et al. '13]
 - Deep denoisers** [Zhang et al. '16, '17, '21] [Song et al. '19]
 - Denoiser : Noisy Image \longrightarrow Clean Image
 - Step towards the manifold of clean images : **implicit prior**

PnP motivations

Find $x^* \in \arg \min_{x \in \mathbb{R}^n} \text{Data-fidelity}(x) + \text{Regularization}(x)$

- **Decouple** data-fidelity and regularization optimizations in iterative algorithms. [Combette & Pesquet '11] [Zoran & Weiss '11]

✗ Previous explicit priors are limited.

✓ **Image Denoising** : easier / well-understood.

→ State-of-the art denoisers **without explicit prior**

Filtering methods [Dabov et al. '07] [Lebrun et al. '13]

Deep denoisers [Zhang et al. '16, '17, '21] [Song et al. '19]

Denoiser : Noisy Image \longrightarrow Clean Image

→ Step towards the manifold of clean images : **implicit prior**

From y degraded, iterate :

1. Take a denoising step
2. Enforce data-fidelity

Variational Image Restoration

PnP Mathematical Formulation

- 1st order optimization

- Denoising as Implicit Prior

- PnP and RED algorithms

Convergence analysis

- Fixed-point convergence

- Global convergence

Algorithm in practice

New trends in PnP image restoration

- Deep Unfolding

- Denoising diffusion models

1st order optimization algorithms

Find $x^* \in \arg \min_{x \in \mathbb{R}^n} F(x)$

1st order optimization algorithms

$$\text{Find } x^* \in \arg \min_{x \in \mathbb{R}^n} F(x)$$

Discretization of the Gradient Flow

$$\begin{cases} x'(t) &= -\nabla F(x(t)) \\ x(t_0) &= x_0 \end{cases}$$

1st order optimization algorithms

$$\text{Find } x^* \in \arg \min_{x \in \mathbb{R}^n} F(x)$$

Discretization of the Gradient Flow

$$\begin{cases} x'(t) &= -\nabla F(x(t)) \\ x(t_0) &= x_0 \end{cases}$$

■ Gradient Descent (*forward Euler*)

$$x_{k+1} = x_k - \tau \nabla F(x_k)$$

→ converges to $x^* \in \arg \min F(x)$ for L -smooth convex F and $\tau L < 2$.

■ Proximal Point Algorithm (*backward Euler*)

$$x_{k+1} = x_k - \tau \nabla F(x_{k+1}) = \text{Prox}_{\tau F}(x_k)$$

$$\text{where } \text{Prox}_F(y) = \arg \min_{x \in \mathbb{R}^n} \frac{1}{2} \|x - y\|^2 + F(x)$$

→ converges to $x^* \in \arg \min F(x)$ for non-smooth convex F and $\tau > 0$.

(Proximal) splitting optimization algorithms

Find $x^* \in \arg \min_{x \in \mathbb{R}^n} f(x) + g(x)$

- Gradient Descent (GD) :

$$x_{k+1} = (\text{Id} - \tau(\nabla f + \nabla g))(x_k)$$

- Proximal Gradient Descent (PGD) :

$$x_{k+1} = \text{Prox}_{\tau g} \circ (\text{Id} - \tau \nabla f)(x_k)$$

- Half Quadratic Splitting (HQS) :

$$x_{k+1} = \text{Prox}_{\tau g} \circ \text{Prox}_{\tau f}(x_k) \quad \triangle! \quad \text{does not target } f + g$$

- Douglas-Rashford Splitting (DRS) / ADMM :

$$x_{k+1} = \left(\frac{1}{2} \text{Id} + \frac{1}{2} (2 \text{Prox}_{\tau g} - \text{Id}) \circ (2 \text{Prox}_{\tau f} - \text{Id}) \right) (x_k)$$

Denoising prior

Find x from observation $y = x + \xi$

- Input distribution $p(x)$.
- Gaussian noise $\xi \sim \mathcal{N}(0, \sigma^2 \text{Id})$.
- Noisy observation $y \sim p_\sigma(y) = p * \mathcal{N}(0, \sigma^2 \text{Id})$.
→ Two **optimal** denoisers

Denoising prior

Find x from observation $y = x + \xi$

- Input distribution $p(x)$.
- Gaussian noise $\xi \sim \mathcal{N}(0, \sigma^2 \text{Id})$.
- Noisy observation $y \sim p_\sigma(y) = p * \mathcal{N}(0, \sigma^2 \text{Id})$.

→ Two **optimal** denoisers

MAP estimator

$$D_\sigma^{\text{MAP}}(y) = \arg \max_x p(x|y)$$

MMSE estimator

$$D_\sigma^{\text{MMSE}}(y) = \mathbb{E}_{x \sim p(x|y)}[x]$$

Denoising prior

Find x from observation $y = x + \xi$

- Input distribution $p(x)$.
- Gaussian noise $\xi \sim \mathcal{N}(0, \sigma^2 \text{Id})$.
- Noisy observation $y \sim p_\sigma(y) = p * \mathcal{N}(0, \sigma^2 \text{Id})$.

→ Two **optimal** denoisers

MAP estimator

$$D_\sigma^{\text{MAP}}(y) = \arg \max_x p(x|y)$$

MMSE estimator

$$D_\sigma^{\text{MMSE}}(y) = \mathbb{E}_{x \sim p(x|y)}[x]$$

$$\begin{aligned} &= \arg \max_{x \in \mathbb{R}^n} \frac{p(y|x)p(x)}{p(y)} \\ &= \arg \min_{x \in \mathbb{R}^n} -\log p(y|x) - \log p(x) \\ &= \arg \min_{x \in \mathbb{R}^n} \frac{1}{2\sigma^2} \|x - y\|^2 - \log p(x) = \text{Prox}_{-\sigma^2 \log p}(y) \end{aligned}$$

Denoising prior

Find x from observation $y = x + \xi$

- Input distribution $p(x)$.
- Gaussian noise $\xi \sim \mathcal{N}(0, \sigma^2 \text{Id})$.
- Noisy observation $y \sim p_\sigma(y) = p * \mathcal{N}(0, \sigma^2 \text{Id})$.

→ Two **optimal** denoisers

MAP estimator

$$D_\sigma^{\text{MAP}}(y) = \arg \max_x p(x|y)$$

MMSE estimator

$$D_\sigma^{\text{MMSE}}(y) = \mathbb{E}_{x \sim p(x|y)}[x]$$

Bayes :

$$D_\sigma^{\text{MAP}} = \text{Prox}_{-\sigma^2 \log p}$$

Denoising prior

Find x from observation $y = x + \xi$

- Input distribution $p(x)$.
- Gaussian noise $\xi \sim \mathcal{N}(0, \sigma^2 \text{Id})$.
- Noisy observation $y \sim p_\sigma(y) = p * \mathcal{N}(0, \sigma^2 \text{Id})$.

→ Two **optimal** denoisers

MAP estimator

$$D_\sigma^{\text{MAP}}(y) = \arg \max_x p(x|y)$$

MMSE estimator

$$D_\sigma^{\text{MMSE}}(y) = \mathbb{E}_{x \sim p(x|y)}[x]$$

Bayes :

$$D_\sigma^{\text{MAP}} = \text{Prox}_{-\sigma^2 \log p}$$

Tweedie :

$$D_\sigma^{\text{MMSE}} = \text{Id} - \nabla(-\sigma^2 \log p_\sigma)$$

Denoising prior

Find x from observation $y = x + \xi$

- Input distribution $p(x)$.
- Gaussian noise $\xi \sim \mathcal{N}(0, \sigma^2 \text{Id})$.
- Noisy observation $y \sim p_\sigma(y) = p * \mathcal{N}(0, \sigma^2 \text{Id})$.

→ Two **optimal** denoisers

MAP estimator

$$D_\sigma^{\text{MAP}}(y) = \arg \max_x p(x|y)$$

MMSE estimator

$$D_\sigma^{\text{MMSE}}(y) = \mathbb{E}_{x \sim p(x|y)}[x]$$

Bayes :

$$D_\sigma^{\text{MAP}} = \text{Prox}_{-\sigma^2 \log p}$$

Tweedie :

$$D_\sigma^{\text{MMSE}} = \text{Id} - \nabla(-\sigma^2 \log p_\sigma)$$

→ **Real** denoiser $D_\sigma \approx D_\sigma^{\text{MAP}}$ or $D_\sigma \approx D_\sigma^{\text{MMSE}}$

A denoiser provides an **implicit prior**

PnP and RED algorithms

Find x from observation $y \sim p(y|x)$

PnP and RED algorithms

Find x from observation $y \sim p(y|x)$

MAP formulation $f = -\log p(y|.)$ and $g \propto -\log p$

Find $x^* \in \arg \min f(x) + \lambda g(x)$

PnP and RED algorithms

Find x from observation $y \sim p(y|x)$

MAP formulation $f = -\log p(y|\cdot)$ and $g \propto -\log p$

Find $x^* \in \arg \min f(x) + \lambda g(x)$

$$\left\{ \begin{array}{ll} \text{GD} & : x_{k+1} = (\text{Id} - \tau(\nabla f + \lambda \nabla g))(x_k) \\ \text{HQS} & : x_{k+1} = \text{Prox}_{\tau \lambda g} \circ \text{Prox}_{\tau f}(x_k) \\ \text{PGD} & : x_{k+1} = \text{Prox}_{\tau \lambda g} \circ (\text{Id} - \tau \nabla f)(x_k) \\ \text{DRS} & : x_{k+1} = \frac{1}{2} \text{Id} + \frac{1}{2} (2 \text{Prox}_{\tau \lambda g} - \text{Id}) \circ (2 \text{Prox}_{\tau f} - \text{Id})(x_k) \end{array} \right.$$

PnP and RED algorithms

Find x from observation $y \sim p(y|x)$

MAP formulation $f = -\log p(y|\cdot)$ and $g \propto -\log p$

Find $x^* \in \arg \min f(x) + \lambda g(x)$

$$\left\{ \begin{array}{ll} \text{GD} & : x_{k+1} = (\text{Id} - \tau(\nabla f + \lambda \nabla g))(x_k) \\ \text{HQS} & : x_{k+1} = \text{Prox}_{\tau \lambda g} \circ \text{Prox}_{\tau f}(x_k) \\ \text{PGD} & : x_{k+1} = \text{Prox}_{\tau \lambda g} \circ (\text{Id} - \tau \nabla f)(x_k) \\ \text{DRS} & : x_{k+1} = \frac{1}{2} \text{Id} + \frac{1}{2} (2 \text{Prox}_{\tau \lambda g} - \text{Id}) \circ (2 \text{Prox}_{\tau f} - \text{Id})(x_k) \end{array} \right.$$

✗ Unknown prior g

PnP and RED algorithms

Find x from observation $y \sim p(y|x)$

MAP formulation $f = -\log p(y|\cdot)$ and $g \propto -\log p$

Find $x^* \in \arg \min f(x) + \lambda g(x)$

$$\left\{ \begin{array}{ll} \text{GD} & : x_{k+1} = (\text{Id} - \tau(\nabla f + \lambda \nabla g))(x_k) \\ \text{HQS} & : x_{k+1} = \text{Prox}_{\tau \lambda g} \circ \text{Prox}_{\tau f}(x_k) \\ \text{PGD} & : x_{k+1} = \text{Prox}_{\tau \lambda g} \circ (\text{Id} - \tau \nabla f)(x_k) \\ \text{DRS} & : x_{k+1} = \frac{1}{2} \text{Id} + \frac{1}{2} (2 \text{Prox}_{\tau \lambda g} - \text{Id}) \circ (2 \text{Prox}_{\tau f} - \text{Id})(x_k) \end{array} \right.$$

✗ Unknown prior g

MAP denoiser

$$D_{\sigma}(y) = \text{Prox}_{-\sigma^2 \log p}(y)$$

MMSE denoiser

$$D_{\sigma}(y) = (\text{Id} + \sigma^2 \nabla \log p_{\sigma})(y)$$

PnP and RED algorithms

Find x from observation $y \sim p(y|x)$

MAP formulation $f = -\log p(y|\cdot)$ and $g \propto -\log p$

Find $x^* \in \arg \min f(x) + \lambda g(x)$

$$\left\{ \begin{array}{ll} \text{GD} & : x_{k+1} = (\text{Id} - \tau(\nabla f + \lambda \nabla g))(x_k) \\ \text{HQS} & : x_{k+1} = \text{Prox}_{\tau \lambda g} \circ \text{Prox}_{\tau f}(x_k) \\ \text{PGD} & : x_{k+1} = \text{Prox}_{\tau \lambda g} \circ (\text{Id} - \tau \nabla f)(x_k) \\ \text{DRS} & : x_{k+1} = \frac{1}{2} \text{Id} + \frac{1}{2} (2 \text{Prox}_{\tau \lambda g} - \text{Id}) \circ (2 \text{Prox}_{\tau f} - \text{Id})(x_k) \end{array} \right.$$

✗ Unknown prior g

MAP denoiser

$$D_\sigma(y) = \text{Prox}_{-\sigma^2 \log p}(y)$$

MMSE denoiser

$$D_\sigma(y) = (\text{Id} + \sigma^2 \nabla \log p_\sigma)(y)$$

PnP algorithms

[Venkatakrishnan et al., '13]

$$\text{Prox}_{\tau \lambda g} \leftarrow D_\sigma$$

RED algorithms

[Romano et al., '16]

$$\nabla g \leftarrow \text{Id} - D_\sigma$$

PnP and RED algorithms

$$\text{Find } x^* \in \arg \min f(x) + \lambda g(x)$$

$$\left\{ \begin{array}{ll} \text{GD} & : x_{k+1} = (\text{Id} - \tau(\nabla f + \lambda \nabla g))(x_k) \\ \text{HQS} & : x_{k+1} = \text{Prox}_{\tau \lambda g} \circ \text{Prox}_{\tau f}(x_k) \\ \text{PGD} & : x_{k+1} = \text{Prox}_{\tau \lambda g} \circ (\text{Id} - \tau \nabla f)(x_k) \\ \text{DRS} & : x_{k+1} = \frac{1}{2} \text{Id} + \frac{1}{2} (2 \text{Prox}_{\tau \lambda g} - \text{Id}) \circ (2 \text{Prox}_{\tau f} - \text{Id})(x_k) \end{array} \right.$$

MAP denoiser

$$D_\sigma(y) = \text{Prox}_{-\sigma^2 \log p}(y)$$

MMSE denoiser

$$D_\sigma(y) = (\text{Id} + \sigma^2 \nabla \log p_\sigma)(y)$$

PnP algorithms

[Venkatakrishnan et al., '13]

$$\text{Prox}_{\tau \lambda g} \leftarrow D_\sigma$$

RED algorithms¹

[Romano et al., '16]

$$\nabla g \leftarrow \text{Id} - D_\sigma$$

¹More recently called “score-based” [Song. et al '19] for generative models

PnP and RED algorithms

$$\text{Find } x^* \in \arg \min f(x) + \lambda g(x)$$

$$\left\{ \begin{array}{ll} \text{GD} & : x_{k+1} = (\text{Id} - \tau(\nabla f + \lambda \nabla g))(x_k) \\ \text{HQS} & : x_{k+1} = \text{Prox}_{\tau \lambda g} \circ \text{Prox}_{\tau f}(x_k) \\ \text{PGD} & : x_{k+1} = \text{Prox}_{\tau \lambda g} \circ (\text{Id} - \tau \nabla f)(x_k) \\ \text{DRS} & : x_{k+1} = \frac{1}{2} \text{Id} + \frac{1}{2} (2 \text{Prox}_{\tau \lambda g} - \text{Id}) \circ (2 \text{Prox}_{\tau f} - \text{Id})(x_k) \end{array} \right.$$

MAP denoiser

$$D_\sigma(y) = \text{Prox}_{-\sigma^2 \log p}(y)$$

MMSE denoiser

$$D_\sigma(y) = (\text{Id} + \sigma^2 \nabla \log p_\sigma)(y)$$

PnP algorithms

[Venkatakrishnan et al., '13]

$$\text{Prox}_{\tau \lambda g} \leftarrow D_\sigma$$

RED algorithms

[Romano et al., '16]

$$\nabla g \leftarrow \text{Id} - D_\sigma$$

$$\left\{ \begin{array}{ll} \text{PnP-HQS} & : x_{k+1} = D_\sigma \circ \text{Prox}_{\tau f}(x_k) \\ \text{PnP-PGD} & : x_{k+1} = D_\sigma \circ (\text{Id} - \tau \nabla f)(x_k) \\ \text{PnP-DRS} & : x_{k+1} = \frac{1}{2} \text{Id} + \frac{1}{2} (2 D_\sigma - \text{Id}) \circ (2 \text{Prox}_{\tau f} - \text{Id})(x_k) \end{array} \right.$$

PnP and RED algorithms

$$\text{Find } x^* \in \arg \min f(x) + \lambda g(x)$$

$$\left\{ \begin{array}{ll} \text{GD} & : x_{k+1} = (\text{Id} - \tau(\nabla f + \lambda \nabla g))(x_k) \\ \text{HQS} & : x_{k+1} = \text{Prox}_{\tau \lambda g} \circ \text{Prox}_{\tau f}(x_k) \\ \text{PGD} & : x_{k+1} = \text{Prox}_{\tau \lambda f} \circ (\text{Id} - \tau \nabla g)(x_k) \\ \text{DRS} & : x_{k+1} = \frac{1}{2} \text{Id} + \frac{1}{2} (2 \text{Prox}_{\tau \lambda g} - \text{Id}) \circ (2 \text{Prox}_{\tau f} - \text{Id})(x_k) \end{array} \right.$$

MAP denoiser

$$D_\sigma(y) = \text{Prox}_{-\sigma^2 \log p}(y)$$

MMSE denoiser

$$D_\sigma(y) = (\text{Id} + \sigma^2 \nabla \log p_\sigma)(y)$$

PnP algorithms

[Venkatakrishnan et al., '13]

$$\text{Prox}_{\tau g} \leftarrow D_\sigma$$

RED algorithms¹

[Romano et al., '16]

$$\nabla g \leftarrow \text{Id} - D_\sigma$$

$$\left\{ \begin{array}{ll} \text{RED-GD} & : x_{k+1} = (\tau \lambda D_\sigma + (1 - \tau \lambda) \text{Id} - \tau \nabla f)(x_k) \\ \text{RED-PGD} & : x_{k+1} = \text{Prox}_{\tau f} \circ (\tau \lambda D_\sigma + (1 - \tau \lambda) \text{Id})(x_k) \end{array} \right.$$

PnP and RED algorithms

Find x from observation $y \sim p(y|x)$

Proximal Algorithms

$$x^* \in \arg \min_{x \in \mathbb{R}^n} f(x) + \lambda g(x)$$

✗ unknown prior g

PnP and RED algorithms

Find x from observation $y \sim p(y|x)$

Proximal Algorithms

$$x^* \in \arg \min_{x \in \mathbb{R}^n} f(x) + \lambda g(x)$$

✗ unknown prior g

$$\begin{array}{c} D_{\sigma}^{MAP} = \text{Prox}_{\tau g} \\ \hline D_{\sigma}^{MMSE} = \text{Id} - \nabla g \end{array} \rightarrow$$

PnP / RED algorithms

✓ implicit prior

PnP and RED algorithms

Find x from observation $y \sim p(y|x)$

Proximal Algorithms

$$x^* \in \arg \min_{x \in \mathbb{R}^n} f(x) + \lambda g(x)$$

✗ unknown prior g

$$D_\sigma \approx \text{Prox}_{\tau g}$$

$$D_\sigma \approx \text{Id} - \nabla g$$

PnP / RED algorithms

✓ implicit prior

Plugged D_σ : (Noisy image, σ) \implies Clean image

PnP and RED algorithms

Find x from observation $y \sim p(y|x)$

Proximal Algorithms

$$x^* \in \arg \min_{x \in \mathbb{R}^n} f(x) + \lambda g(x)$$

✗ unknown prior g

$$\begin{array}{l} D_\sigma \approx \text{Prox}_{\tau g} \\ D_\sigma \approx \text{Id} - \nabla g \end{array} \rightarrow$$

PnP / RED algorithms

- ✓ implicit prior
- ✓ SOTA restoration

Plugged D_σ : (Noisy image, σ) \implies Clean image

✓ CNN e.g. DRUNet [Zhang et. al '20], NCSN [Song et. al '21] ...

PnP and RED algorithms

Find x from observation $y \sim p(y|x)$

Proximal Algorithms

$$x^* \in \arg \min_{x \in \mathbb{R}^n} f(x) + \lambda g(x)$$

✗ unknown prior g

$$D_\sigma \approx \text{Prox}_{\tau g}$$

$$D_\sigma \approx \text{Id} - \nabla g$$

PnP / RED algorithms

✓ implicit prior

✓ SOTA restoration

✗ no convergence guarantees

✗ no minimization problem

Plugged D_σ : (Noisy image, σ) \implies Clean image

✓ CNN e.g. DRUNet [Zhang et. al '20], NCSN [Song et. al '21] ...



$$D_\sigma \neq \text{Prox}_{\tau g} \quad D_\sigma \neq \text{Id} - \nabla g$$

PnP and RED algorithms

Find x from observation $y \sim p(y|x)$

Proximal Algorithms

$$x^* \in \arg \min_{x \in \mathbb{R}^n} f(x) + \lambda g(x)$$

✗ unknown prior g

$$D_\sigma \approx \text{Prox}_{\tau g}$$

$$D_\sigma \approx \text{Id} - \nabla g$$

PnP / RED algorithms

✓ implicit prior

✓ SOTA restoration

✗ no convergence guarantees

✗ no minimization problem

Plugged D_σ : (Noisy image, σ) \implies Clean image

✓ CNN e.g. DRUNet [Zhang et. al '20], NCSN [Song et. al '21] ...



$$D_\sigma \neq \text{Prox}_{\tau g} \quad D_\sigma \neq \text{Id} - \nabla g$$

Objective : Find minimal conditions on D_σ to restore convergence guarantees.

Variational Image Restoration

PnP Mathematical Formulation

- 1st order optimization
- Denoising as Implicit Prior
- PnP and RED algorithms

Convergence analysis

- Fixed-point convergence
- Global convergence

Algorithm in practice

New trends in PnP image restoration

- Deep Unfolding
- Denoising diffusion models

PnP fixed-point convergence

PnP algorithms :

$$x_{k+1} = T_{PnP}(x_k)$$

$$\text{with } T_{PnP} = \begin{cases} T_{HQS} &= D_\sigma \circ \text{Prox}_{\tau f} \\ T_{PGD} &= D_\sigma \circ (\text{Id} - \tau \nabla f) \\ T_{DRS} &= \frac{1}{2} \text{Id} + \frac{1}{2} (2D_\sigma - \text{Id}) \circ (2 \text{Prox}_{\tau f} - \text{Id}) \end{cases}$$

Objective : Show that $x_k \rightarrow x^* \in \text{Fix}(T_{PnP})$.

PnP fixed-point convergence

PnP algorithms :

$$x_{k+1} = T_{PnP}(x_k)$$

$$\text{with } T_{PnP} = \begin{cases} T_{HQS} &= D_\sigma \circ \text{Prox}_{\tau f} \\ T_{PGD} &= D_\sigma \circ (\text{Id} - \tau \nabla f) \\ T_{DRS} &= \frac{1}{2} \text{Id} + \frac{1}{2} (2D_\sigma - \text{Id}) \circ (2 \text{Prox}_{\tau f} - \text{Id}) \end{cases}$$

Objective : Show that $x_k \rightarrow x^* \in \text{Fix}(T_{PnP})$.

Solutions :

- Averaged (nonexpansive) D_σ . [Sun et. al '18, '19, '21] [Hertrich et. al '20] [Terris et. al '21] [Bohra et. al '21]
- [Chan et. al '16] PnP-DRS with stepsize $\tau \rightarrow 0$ and a “bounded” denoiser.
- [Cohen et. al '20] RED-PRO reformulates as a convex minimization problem.

PnP fixed-point convergence

PnP algorithms :

$$x_{k+1} = T_{PnP}(x_k)$$

$$\text{with } T_{PnP} = \begin{cases} T_{HQS} &= D_\sigma \circ \text{Prox}_{\tau f} \\ T_{PGD} &= D_\sigma \circ (\text{Id} - \tau \nabla f) \\ T_{DRS} &= \frac{1}{2} \text{Id} + \frac{1}{2} (2D_\sigma - \text{Id}) \circ (2 \text{Prox}_{\tau f} - \text{Id}) \end{cases}$$

Objective : Show that $x_k \rightarrow x^* \in \text{Fix}(T_{PnP})$.

Solutions :

- Averaged (nonexpansive) D_σ . [Sun et. al '18, '19, '21] [Hertrich et. al '20] [Terris et. al '21] [Bohra et. al '21]
- [Chan et. al '16] PnP-DRS with stepsize $\tau \rightarrow 0$ and a “bounded” denoiser.
- [Cohen et. al '20] RED-PRO formulates a fixed-point objective as a convex minimization problem.

Definition

$T : \mathbb{R}^n \rightarrow \mathbb{R}^n$ and $\theta \in (0, 1)$. T is **θ -averaged** if there is a nonexpansive operator R such that

$$T = \theta R + (1 - \theta) \text{Id}$$

- T θ -averaged $\implies T$ nonexpansive.
- $1/2$ -averaged = “firmly nonexpansive”.

Averaged operator theory [Bauschke & Combettes '11]

Definition

$T : \mathbb{R}^n \rightarrow \mathbb{R}^n$ and $\theta \in (0, 1)$. T is **θ -averaged** if there is a nonexpansive operator R such that

$$T = \theta R + (1 - \theta) \text{Id}$$

- T θ -averaged $\implies T$ nonexpansive.
- $1/2$ -averaged = “firmly nonexpansive”.

Theorem

If T is a θ -averaged operator that admits fixed points, then the sequence $x_{k+1} = T(x_k)$ converges to a fixed point of T .

Averaged operator theory [Bauschke & Combettes '11]

Proposition

Let T_1 θ_1 -averaged and T_2 θ_2 -averaged.

- *Then $T_1 \circ T_2$ is θ -averaged with $\theta = \frac{\theta_1 + \theta_2 - 2\theta_1\theta_2}{1 - \theta_1\theta_2}$*
- *For $\alpha \in [0, 1]$, $\alpha T_1 + (1 - \alpha) \text{Id}$ is $\alpha\theta_1$ -averaged.*

Proposition

If $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is convex and L -smooth,

- *$\text{Prox}_{\tau f}$ is $\frac{\tau L}{2(1+\tau L)}$ -averaged.*
- *$\text{Id} - \tau \nabla f$ is $\frac{\tau L}{2}$ -averaged.*

Averaged operator theory for PnP convergence

PnP algorithms :

$$x_{k+1} = T_{PnP}(x_k)$$

$$\text{with } T_{PnP} = \begin{cases} T_{HQS} &= D_\sigma \circ \text{Prox}_{\tau f} \\ T_{PGD} &= D_\sigma \circ (\text{Id} - \tau \nabla f) \\ T_{DRS} &= \frac{1}{2} \text{Id} + \frac{1}{2} (2D_\sigma - \text{Id}) \circ (2 \text{Prox}_{\tau f} - \text{Id}) \end{cases}$$

Theorem

If $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is convex, L -smooth and D_σ is θ -averaged,

- If $\theta < 1$, **PnP-HQS** converges towards a fixed point of T_{HQS} .
- If $\theta < 1$ and $\tau L < 2$, **PnP-PGD** converges towards a fixed point of T_{PGD} .
- If $\theta \leq 1/2$, **PnP-DRS** converges towards a fixed point of T_{DRS} .

Averaged operator theory for PnP convergence

PnP algorithms :

$$x_{k+1} = T_{PnP}(x_k)$$

$$\text{with } T_{PnP} = \begin{cases} T_{HQS} &= D_\sigma \circ \text{Prox}_{\tau f} \\ T_{PGD} &= D_\sigma \circ (\text{Id} - \tau \nabla f) \\ T_{DRS} &= \frac{1}{2} \text{Id} + \frac{1}{2} (2D_\sigma - \text{Id}) \circ (2 \text{Prox}_{\tau f} - \text{Id}) \end{cases}$$

Theorem

If $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is convex, L -smooth and D_σ is θ -averaged,

- If $\theta < 1$, **PnP-HQS** converges towards a fixed point of T_{HQS} .
- If $\theta < 1$ and $\tau L < 2$, **PnP-PGD** converges towards a fixed point of T_{PGD} .
- If $\theta \leq 1/2$, **PnP-DRS** converges towards a fixed point of T_{DRS} .

Remark

✗ Does not extend to **nonconvex** data-fidelity term f .

- For f **strongly convex**, $\text{Prox}_{\tau f}$ and $\text{Id} - \tau \nabla f$ are contractive $\rightarrow D_\sigma$ $(1 + \epsilon)$ -Lipshitz (“almost averaged”) [Ryu et. al, '19].

How to build averaged deep denoisers ?

$$D_\sigma = \theta R_\sigma + (1 - \theta) \text{Id} \quad \text{with } R_\sigma \text{ nonexpansive.}$$

How to build averaged deep denoisers ?

$$D_\sigma = \theta R_\sigma + (1 - \theta) \text{Id} \quad \text{with } R_\sigma \text{ nonexpansive.}$$

→ How to train nonexpansive networks ?

How to build averaged deep denoisers ?

$$D_\sigma = \theta R_\sigma + (1 - \theta) \text{Id} \quad \text{with } R_\sigma \text{ nonexpansive.}$$

→ How to train nonexpansive networks ?

- Spectral normalization. [\[Miyato et. al '18\]](#) [\[Ryu et. al '19\]](#)
 - ✗ Lipschitz constant $\ll 1$ for large networks.
 - ✗ Does not allow skip connexions.

How to build averaged deep denoisers ?

$$D_\sigma = \theta R_\sigma + (1 - \theta) \text{Id} \quad \text{with } R_\sigma \text{ nonexpansive.}$$

→ How to train nonexpansive networks ?

- Spectral normalization. [Miyato et. al '18] [Ryu et. al '19]
 - ✗ Lipschitz constant $\ll 1$ for large networks.
 - ✗ Does not allow skip connexions.
- Deep spline neural networks [Neumayer et. al, '22]
- Convolutional Proximal Neural Networks [Hertrich et. al, '20]
- Soft regularization of the training loss [Terris et. al '21].
- $D_\sigma = \text{Id} - \nabla g_\sigma$ with g_σ Input Convex Neural Network (ICNN) [Meunier et. al '21].

How to build averaged deep denoisers ?

$$D_\sigma = \theta R_\sigma + (1 - \theta) \text{Id} \quad \text{with } R_\sigma \text{ nonexpansive.}$$

→ How to train nonexpansive networks ?

- Spectral normalization. [Miyato et. al '18] [Ryu et. al '19]
 - ✗ Lipschitz constant $\ll 1$ for large networks.
 - ✗ Does not allow skip connexions.
- Deep spline neural networks [Neumayer et. al, '22]
- Convolutional Proximal Neural Networks [Hertrich et. al, '20]
- Soft regularization of the training loss [Terris et. al '21].
- $D_\sigma = \text{Id} - \nabla g_\sigma$ with g_σ Input Convex Neural Network (ICNN) [Meunier et. al '21].



Non-expansivity can harm denoising performance.

PnP and RED algorithms

Find x from observation $y \sim p(y|x)$

Proximal Algorithms

$$x^* \in \arg \min_{x \in \mathbb{R}^n} f(x) + \lambda g(x)$$

✗ unknown prior g

$$D_\sigma \approx \text{Prox}_{\tau g}$$

$$D_\sigma \approx \text{Id} - \nabla g$$

PnP / RED algorithms

✓ implicit prior

✓ SOTA restoration

✗ no convergence guarantees

✗ no minimization problem

D_σ averaged

PnP Fixed-Point convergence

$$x^* \in \{x = T_{PnP}(x)\}$$

✓ convergence guarantees

✗ no minimization problem

PnP and RED global convergence

Objective : Show that $x_k \rightarrow x^* \in \arg \min_x f(x) + g(x)$.

Solution : Make PnP/RED algorithms **real** proximal splitting algorithms **again** ... by enforcing **exactly** :

PnP algorithms

$$D_\sigma = \text{Prox}_g$$

RED algorithms

$$D_\sigma = \text{Id} - \nabla g$$

PnP and RED global convergence

Objective : Show that $x_k \rightarrow x^* \in \arg \min_x f(x) + g(x)$.

Solution : Make PnP/RED algorithms **real** proximal splitting algorithms **again** ... by enforcing **exactly** :

PnP algorithms

$$D_\sigma = \text{Prox}_g$$

RED algorithms

$$D_\sigma = \text{Id} - \nabla g$$

RED global convergence

- [Romano et. al, '16] If D_σ has **symmetric Jacobian**,

$$D_\sigma = \text{Id} - \nabla g_\sigma \quad \text{with} \quad g_\sigma(x) = \frac{1}{2} \langle x, x - D_\sigma(x) \rangle$$

✗ Not verified by common denoisers [Reehorst & Schniter, '18].

- [Cohen et. al, '22] [Hurault et. al, '22]

“Gradient Step” Denoiser :

$$D_\sigma = \text{Id} - \nabla g_\sigma(x) \quad \text{with} \quad g_\sigma \text{ neural network}$$

- ✓ SOTA denoising performance.
- ✓ $D_\sigma \approx D_\sigma^{MMSE} = \text{Id} + \sigma^2 \nabla \log p_\sigma$ (Tweedie)
 $\implies g_\sigma \approx -\sigma^2 \log p_\sigma$

PnP and RED global convergence

Objective : Show that $x_k \rightarrow x^* \in \arg \min_x f(x) + g(x)$.

Solution : Make PnP/RED algorithms **real** proximal splitting algorithms **again** ... by enforcing **exactly** :

PnP algorithms

$$D_\sigma = \text{Prox}_g$$

RED algorithms

$$D_\sigma = \text{Id} - \nabla g$$

PnP global convergence

- [Moreau, '65]

If $D_\sigma = \nabla h_\sigma$ with h_σ convex and D_σ is nonexpansive, then $\exists g_\sigma : \mathbb{R}^n \rightarrow \mathbb{R}$ convex such that

$$D_\sigma = \text{Prox}_{g_\sigma}$$

✗ Hard too enforce both conditions in the same time

- [Gribonval and Nikolova, '20]

If $D_\sigma = \nabla h_\sigma$ with h_σ convex and ~~D_σ is nonexpansive~~, then $\exists g_\sigma : \mathbb{R}^n \rightarrow \mathbb{R}$ **nonconvex** such that

$$D_\sigma = \text{Prox}_{g_\sigma}$$

→ **Prox-PnP** [Hurault et. al, '22]

$$D_\sigma = \text{Id} - \nabla \phi_\sigma = \nabla h_\sigma \quad \text{with} \quad h_\sigma(x) = \frac{\|x\|^2}{2} - \phi_\sigma(x)$$

$$h_\sigma \text{ convex} \Leftrightarrow \nabla \phi_\sigma \text{ nonexpansive} \Rightarrow \exists g_\sigma, D_\sigma = \text{Prox}_{g_\sigma}$$

✗ $D_\sigma = \text{Prox}_{g_\sigma} \neq \text{Prox}_{\tau g_\sigma}$ restrict stepsize $\tau = 1$.

PnP and RED global convergence

Objective : Show that $x_k \rightarrow x^* \in \arg \min_x f(x) + g(x)$.

Solution : Make PnP/RED algorithms **real** proximal splitting algorithms **again** ... by enforcing **exactly** :

PnP algorithms

$$D_\sigma = \text{Prox}_g$$

RED algorithms

$$D_\sigma = \text{Id} - \nabla g$$

? Convergence of proximal splitting algorithms for nonconvex g

Convergence of nonconvex proximal splitting algorithms

■ **PGD** : $x_{k+1} = \text{Prox}_{\tau f_1} \circ (\text{Id} - \tau \nabla f_2)$

■ **DRS** : $x_{k+1} = \frac{1}{2} \text{Id} + \frac{1}{2} (2 \text{Prox}_{\tau f_1} - \text{Id}) \circ (2 \text{Prox}_{\tau f_2} - \text{Id})$

Objective : Show that $x_k \rightarrow x^* \in \{0 \in \partial f_1(x) + \nabla f_2(x)\}$.

Suppose

- f_2 is L -smooth
- $f_1 + f_2$ coercive and bounded from below.
- f_1 and f_2 verify the Kurdyka-Lojasiewicz (KL) property.

Then, for $\begin{cases} (\text{PGD}) & \tau L < 1 \quad [\text{Attouch, '13}] \\ (\text{DRS}) & \tau L < 1/2 \quad [\text{Themelis \& Patrinos, '19}] \end{cases}$

(x_k) converges towards a critical point of $f_1 + f_2$.

PnP and RED algorithms

Find x from observation $y \sim p(y|x)$

Proximal Algorithms

$$x^* \in \arg \min_{x \in \mathbb{R}^n} f(x) + \lambda g(x)$$

✗ unknown prior g

$$D_\sigma \approx \text{Prox}_{\tau g}$$

$$D_\sigma \approx \text{Id} - \nabla g$$

PnP / RED algorithms

✓ implicit prior

✓ SOTA restoration

✗ no convergence guarantees

✗ no minimization problem

$$D_\sigma = \text{Prox}_{g_\sigma}$$

$$D_\sigma = \text{Id} - \nabla g_\sigma$$

D_σ averaged

PnP / RED global convergence

$$x^* \in \arg \min_{x \in \mathbb{R}^n} f(x) + \lambda g_\sigma(x)$$

✓ convergence guarantees

✓ minimization problem

PnP Fixed-Point convergence

$$x^* \in \{x = T_{PnP}(x)\}$$

✓ convergence guarantees

✗ no minimization problem

Algorithms in practice

$$\left\{ \begin{array}{ll} \text{PnP-HQS} & : x_{k+1} = D_{\sigma} \circ \text{Prox}_{\tau f}(x_k) \\ \text{PnP-PGD} & : x_{k+1} = D_{\sigma} \circ (\text{Id} - \tau \nabla f)(x_k) \\ \text{PnP-DRS} & : x_{k+1} = \frac{1}{2} \text{Id} + \frac{1}{2} (2D_{\sigma} - \text{Id}) \circ (2 \text{Prox}_{\tau f} - \text{Id})(x_k) \end{array} \right.$$

$$\left\{ \begin{array}{ll} \text{RED-GD} & : x_{k+1} = (\tau \lambda D_{\sigma} + (1 - \tau \lambda) \text{Id} - \tau \nabla f)(x_k) \\ \text{RED-PGD} & : x_{k+1} = \text{Prox}_{\tau f} \circ (\tau \lambda D_{\sigma} + (1 - \tau \lambda) \text{Id})(x_k) \end{array} \right.$$

- Manual parameter tuning of σ , λ
→ automatic via RL [Wei et. al ‘20], SURE [Ramani et. al, ‘08].
- Backtracking Line Search for τ [Hu et al. ‘22] [Hurault et al. ‘22]
- Annealing of σ [Zhang et al. ‘21] [Kadkhodaie & Simoncelli]

Variational Image Restoration

PnP Mathematical Formulation

- 1st order optimization
- Denoising as Implicit Prior
- PnP and RED algorithms

Convergence analysis

- Fixed-point convergence
- Global convergence

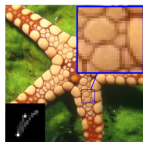
Algorithm in practice

New trends in PnP image restoration

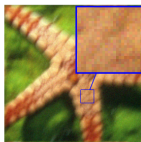
- Deep Unfolding
- Denoising diffusion models

Global convergence [Hurault et. al '22]

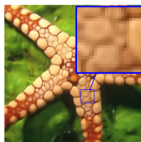
Deblurring with motion kernel and Gaussian noise std $\nu = 0.03$



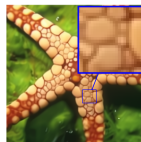
(a) Clean



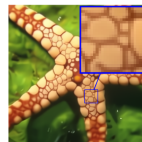
(b) Observed



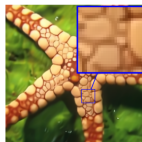
(c) IRCNN
(28.66dB)



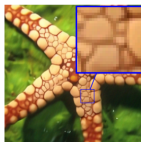
(d) DPIR
(29.76dB)



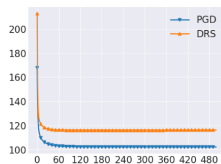
(e) GSPnP-HQS
(29.90dB)



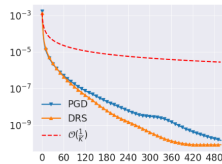
(f) Prox-PnP-PGD
(29.41dB)



(g) Prox-PnP-DRS
(29.65dB)



(h) $F_{\lambda, \sigma}(x_k)$



(i) $\min_{i \leq k} \|x_{i+1} - x_i\|^2$

Global convergence [Hurault et. al '22]

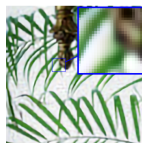
Super-resolution with scale 2, Gaussian blur kernel and Gaussian noise std $\nu = 0.01$



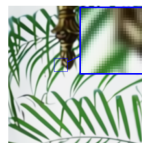
(a) Clean



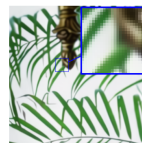
(b) Observed



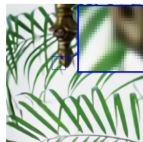
(c) IRCNN
(22.82dB)



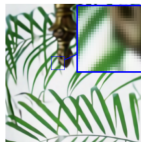
(d) DPIR
(23.97dB)



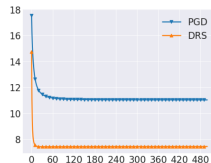
(e) GSPnP-HQS
(24.81dB)



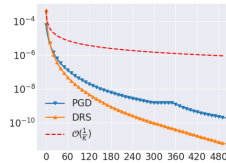
(f) Prox-PnP-PGD
(23.96dB)



(g) Prox-PnP-DRS
(24.36dB)

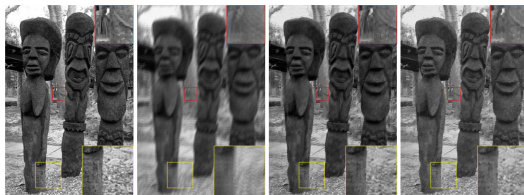


(h) $F_{\lambda, \sigma(x_k)}$



(i) $\min_{i \leq k} \|x_{i+1} - x_i\|^2$

Fixed Point convergence [Terris et. al '21]



(a) Groundtruth

(b) Observation
(20.48, 0.387)

(c) $\text{prox}_{\mu\|\Psi^{\dagger}\cdot\|_1}$
(26.13, 0.775)

(d) $\text{prox}_{\mu\|\cdot\|_{TV}}$
(26.57, 0.787)



(e) BM3D
(26.09, 0.732)



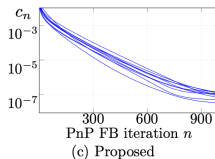
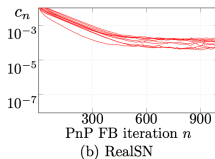
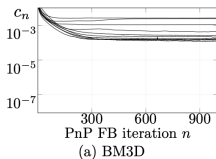
(f) RealSN
(24.68, 0.726)



(g) DnCNN
(26.12, 0.643)



(h) Proposed
(27.09, 0.789)



Variational Image Restoration

PnP Mathematical Formulation

- 1st order optimization
- Denoising as Implicit Prior
- PnP and RED algorithms

Convergence analysis

- Fixed-point convergence
- Global convergence

Algorithm in practice

New trends in PnP image restoration

- Deep Unfolding
- Denoising diffusion models

Deep Unfolding/Unrolling

- **Deep Unfolding/Unrolling (DU)** [Zhang et. al '17] [Zhang et. al '20] PnP/RED with deep denoiser D_σ^θ with trainable parameters θ .

$$\text{e.g. for } k = 1, \dots, K \quad x_{k+1} = \text{Prox}_{\tau f} \circ D_\sigma^\theta(x_k)$$

Train θ on pairs (y, x^*)

$$\mathcal{L}(\theta) = \mathbb{E} [||x_K - x^*||^2]$$

and backpropagate through the PnP/RED iterations.

- ✓ State-of-the-art results.
 - ✓ Also learns optimal regularization hyperparameters.
 - ✗ Heavy memory backpropagation
 - ✗ Specialized on single restoration tasks.
- **Deep Equilibrium (DEQ)** [Bai et. al '19, Gilton et. al '21] : DU with ∞ nb of layers.
 - ✓ Backward pass with constant memory using implicit differentiation of the fixed point equation.

Link with conditional denoising diffusion

$$\begin{cases} \text{RED-GD} & : x_{k+1} = (\tau\lambda D_\sigma + (1 - \tau\lambda) \text{Id} - \tau\nabla f)(x_k) \\ \text{RED-PGD} & : x_{k+1} = \text{Prox}_{\tau f} \circ (\tau\lambda D_\sigma + (1 - \tau\lambda) \text{Id})(x_k) \end{cases}$$

Discretization of the Gradient Flow

$$\begin{cases} \frac{d}{dt}x(t) = -\nabla f(x(t)) + \lambda(\text{Id} - D_\sigma)(x(t)) \\ x(0) = x_0 \end{cases}$$
$$\begin{cases} \frac{d}{dt}x(t) = -(\nabla \log p(y|x(t)) + \frac{\lambda}{\sigma^2} \nabla \log p_\sigma(x(t))) \\ x(0) = x_0 \end{cases}$$

[Song et. al, ‘21] one can sample $x \sim p$ discretizing the “probability flow ODE” from white noise with $\sigma(t) \rightarrow 0$.

$$\begin{cases} \frac{d}{dt}x(t) = -g(t)^2 \nabla \log p_{\sigma(t)}(x(t)) \\ x(0) \sim \mathcal{N}(0, \text{Id}) \end{cases}$$

with scaled guidance [Ho & Salimans, ‘22]

$$\begin{cases} \frac{d}{dt}x(t) = -g(t)^2 (w \nabla \log p(y|x(t)) + \nabla \log p_{\sigma(t)}(x(t))) \\ x(0) \sim \mathcal{N}(0, \text{Id}) \end{cases}$$

$\Rightarrow \text{RED} + \sigma(t) = \text{deterministic Denoising Diffusion models.}$

Key points

- Denoising provides an approximation of the prox / gradient of the true log image prior.
- A generic denoiser can be plugged in optimization algorithms to solve any IR task with known degradation model.
- State-of-the art performance using deep denoisers.
- Convergence results can be obtained with particular parametrizations on the denoiser.

Thank you !

GSPnP and ProxPnP

$D_\sigma = \text{Id} - \nabla g_\sigma$ with g_σ \mathcal{C}^1 neural network $\mathbb{R}^n \rightarrow \mathbb{R}$

$$g_\sigma(x) = \frac{1}{2} \|x - N_\sigma(x)\|^2$$

with a \mathcal{C}^1 neural network $N_\sigma : \mathbb{R}^n \rightarrow \mathbb{R}^n$ (e.g. DRUNet [Zhang et al. '21] with ELU activations)

$$D_\sigma(x) = x - \nabla g_\sigma(x) = N_\sigma(x) + J_{N_\sigma}(x)^T (x - N_\sigma(x))$$

- ✓ g_σ nonconvex, \mathcal{C}^2 , lower-bounded and KL
 \Rightarrow Function values and iterates convergence.
- ✓ **SOTA denoising** with classical denoising CNN architectures for N_σ .

Training loss : GS-Denoiser - Prox-Denoiser

$$\mathbb{E}_{x, \xi_\sigma} \left[\|D_\sigma(x + \xi_\sigma) - x\|^2 + \mu \max(\|J_{\nabla g_\sigma}(x + \xi_\sigma)\|_S, 1 - \epsilon) \right]$$

$\sigma(./255)$	5	15	25
DRUNet	40.19	33.89	31.25
GS-Denoiser	40.26	33.90	31.26
Prox-Denoiser	40.12	33.60	30.82

Table: Denoising PSNR on the CBSD68 dataset, for various σ .

GSPnP and ProxPnP

$D_\sigma = \text{Id} - \nabla g_\sigma$ with g_σ \mathcal{C}^1 neural network $\mathbb{R}^n \rightarrow \mathbb{R}$

$$g_\sigma(x) = \frac{1}{2} \|x - N_\sigma(x)\|^2$$

with a \mathcal{C}^1 neural network $N_\sigma : \mathbb{R}^n \rightarrow \mathbb{R}^n$ (e.g. DRUNet [Zhang et al. '21] with ELU activations)

$$D_\sigma(x) = x - \nabla g_\sigma(x) = N_\sigma(x) + J_{N_\sigma}(x)^T (x - N_\sigma(x))$$

- ✓ g_σ nonconvex, \mathcal{C}^2 , lower-bounded and KL
 \Rightarrow Function values and iterates convergence.
- ✓ **SOTA denoising** with classical denoising CNN architectures for N_σ .

Training loss : GS-Denoiser - Prox-Denoiser

$$\mathbb{E}_{x, \xi_\sigma} \left[\|D_\sigma(x + \xi_\sigma) - x\|^2 + \mu \max(\|J_{\nabla g_\sigma}(x + \xi_\sigma)\|_S, 1 - \epsilon) \right]$$

$\sigma(./255)$	5	15	25
GS-DRUNet	1.26	1.96	3.27
Prox-DRUNet	0.92	0.99	0.96

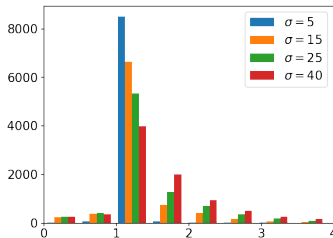
Table: $\max_x \|J_{\nabla g_\sigma}(x)\|_S$ on the CBSD68 dataset, for various σ .

More on g_σ (GSPnP)

$$g_\sigma(x) = \frac{1}{2} \|x - N_\sigma(x)\|^2$$

with a neural network $N_\sigma : \mathbb{R}^n \rightarrow \mathbb{R}^n$

- g_σ is bounded from below, \mathcal{C}^1 and with $L > 1$ Lipschitz gradient



- **Coercivity** to bound iterates x_k :

$$\hat{g}_\sigma(x) = g_\sigma(x) + \frac{1}{2} \|x - \text{Proj}_C(x)\|^2$$

for a large convex compact set C (never activated in practice)

More on ϕ_σ (ProxPnP)

Let $g_\sigma : \mathbb{R}^n \rightarrow \mathbb{R}$ a \mathcal{C}^{k+1} function with $k \geq 1$ and ∇g_σ L -Lipschitz with $L < 1$. Then, for $h_\sigma : x \rightarrow \frac{1}{2}\|x\|^2 - g_\sigma(x)$ and $D_\sigma := \nabla h_\sigma = \text{Id} - \nabla g_\sigma$,

- (i) h_σ is $(1 - L)$ -strongly convex and $\forall x \in \mathbb{R}^n$, $J_{D_\sigma}(x)$ is positive definite
- (ii) D_σ is injective, $\text{Im}(D_\sigma)$ is open and, $\forall x \in \mathbb{R}^n$,
 $D_\sigma(x) = \text{Prox}_{\phi_\sigma}(x)$, with $\phi_\sigma : \mathcal{X} \rightarrow \mathbb{R} \cup \{+\infty\}$ defined by

$$\phi_\sigma(x) \propto \begin{cases} g_\sigma(D_\sigma^{-1}(x)) - \frac{1}{2}\|D_\sigma^{-1}(x) - x\|^2 & \text{if } x \in \text{Im}(D_\sigma), \\ +\infty & \text{otherwise,} \end{cases} \quad (1)$$

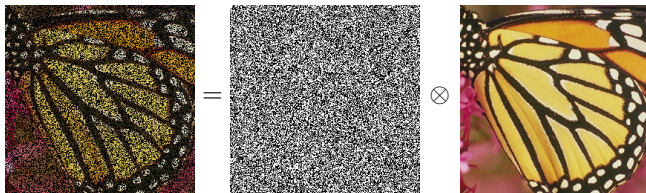
- (iii) ϕ_σ is $\frac{L}{L+1}$ weakly convex.

Inpainting

Inpainting ($\nu = 0$): A diagonal, with values in $\{0, 1\}$,

$$f(x) = \iota_{\{x \mid Ax=y\}}$$

$$\text{Prox}_{\tau f}(x) = \Pi_{\{x \mid Ax=y\}}(x) = Ay - Ax + x$$



Inpainting



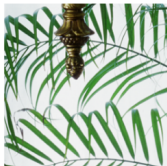
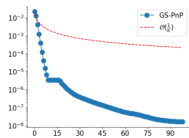
Clean



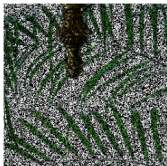
Observed



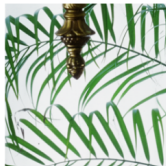
GS-PnP (31.65dB)



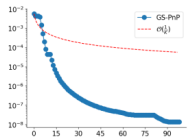
Clean



Observed



GS-PnP (33.65dB)



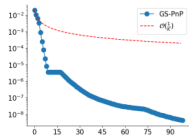
Clean



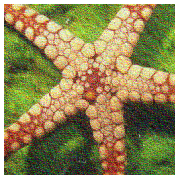
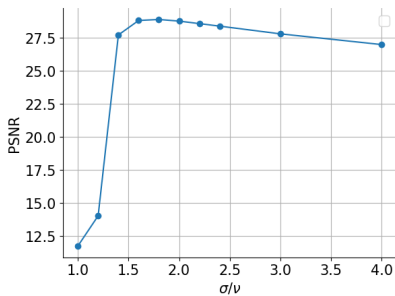
Observed



GS-PnP (33.71dB)



Influence of the parameter σ



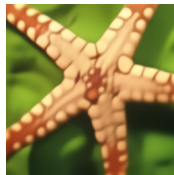
$\sigma/\nu = 1$



$\sigma/\nu = 1.8$

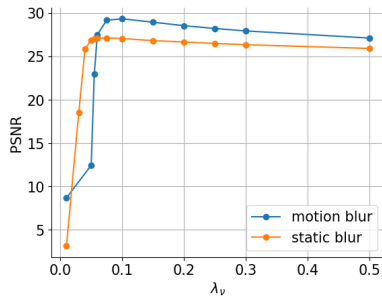


$\sigma/\nu = 4$



$\sigma/\nu = 10$

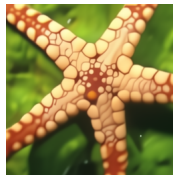
Influence of the parameter λ



$$\lambda_\nu^2 = 1$$



$$\lambda_\nu^2 = 1.8$$



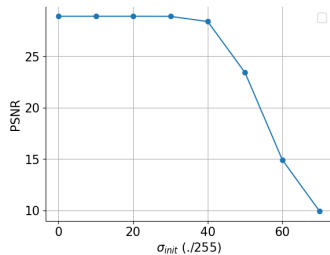
$$\lambda_\nu^2 = 4$$



$$\lambda_\nu^2 = 10$$

Influence of the initialization

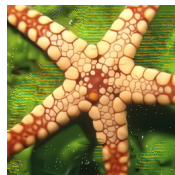
$$x_0 = y + n \text{ with } n \sim \mathcal{N}(0, \sigma_{init} \text{Id})$$



$$\sigma_{init} = 20/255$$



$$\sigma_{init} = 40/255$$



$$\sigma_{init} = 60/255$$

The first proximal steps on the data-fidelity term (with a large τ) prevent the algorithm to be stuck in a poor local minimum.

Fixed-points of the GS-Denoiser

$$x_{k+1} = \mathcal{P}_{[0,1]^n}(x_k - \tau \nabla g_\sigma(x_k))$$



(a) init



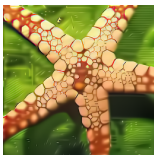
(b) $\sigma = 0$



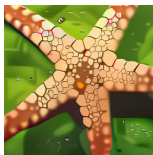
(c) $\sigma = 0.1$



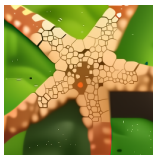
(d) $\sigma = 0.5$



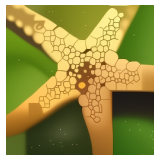
(e) $\sigma = 1$



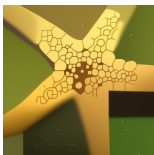
(f) $\sigma = 2.5$



(g) $\sigma = 5$



(h) $\sigma = 10$



(i) $\sigma = 15$



(j) $\sigma = 20$



(k) $\sigma = 30$



(l) $\sigma = 40$