



UNIVERSITÉ DE BORDEAUX

Régularisation par débruitage

Etudiants :

Henri CHOUERI
Issa MAHAMAT
Hani Atef Abdallah MOHAMED

Professeur :

Jean-François AUJOL



Abstract

Ce rapport traite de la régularisation par débruitage (RED), présentée par Romano, Elad et Milanfar dans leur article *"The Little Engine That Could: Regularization by Denoising"* en 2016. Un développement est apporté avec le document *"Proximal Denoiser for Convergent Plug-and-Play Optimization with Nonconvex Regularization"* de Hureault, Leclaire et Papadakis (2022).

Le traitement d'image se ramène en général à la résolution d'un problème d'optimisation. La fonctionnelle à optimiser est la somme d'un terme d'attache aux données, qui mesure l'écart entre l'image observée et l'image reconstruite, et d'un terme de régularisation (ou "prior"), qui impose des contraintes sur la solution à obtenir.

Le choix de ce terme est stratégique pour obtenir une bonne reconstruction. Les auteurs proposent une approche innovante en utilisant un régularisateur basé sur un débruiteur, combinant ainsi les techniques de reconstruction d'images et de débruitage dans un même cadre. Cette approche repose sur l'idée que le débruiteur peut servir de prior dans le cadre de l'optimisation. Le choix du débruiteur découle de considérations techniques (homogénéité, invariance par changement d'échelle, neutralité). Ces propriétés conduisent à une forme pseudo-linéaire du débruiteur, ce qui justifie son utilisation dans le cadre de l'approche RED.

Ce schéma se révèle supérieur aux schémas plus classiques de *Plug-and-Play*, qui utilisent un prior fixe, ce qui limite leur capacité à s'adapter à différents types de données ou de modèles.

Il est plus versatile, mieux défini et peut s'interfacer avec n'importe quel débruiteur, y compris des réseaux de neurones.

Dans leur article de 2022, Hureault, Leclaire et Papadakis proposent de recourir à l'apprentissage profond via des réseaux de neurones pré-entraînés pour effectuer ce débruitage.

L'utilisation de réseaux de neurones capture les motifs complexes du bruit et améliore grandement les performances visuelles.

L'entraînement de ces modèles se fait à partir d'une base de paires d'images (bruitées/non bruitées), sur le critère de minimisation de l'écart quadratique moyen (MSE). Le caractère non convexe de la régularisation peut être surmonté par l'utilisation d'un opérateur proximal comme débruiteur. Combiner les techniques de débruitage avec des méthodes proximales améliore la robustesse et la flexibilité des algorithmes de type *Plug-and-Play*, et garantit que les méthodes d'optimisation convergent efficacement, même pour des débruteurs complexes comme ceux basés sur des réseaux de neurones, sous réserve de conditions de continuité de Lipschitz.

Cette approche s'avère supérieure aux approches classiques (TV, BM3D) ou non convexes (ℓ_p -normes) pour des tâches de débruitage, de défloutage et de super-résolution, comme le montrent les simulations de ces auteurs.

Contents

1	Introduction	4
2	La méthode MAP (Maximum A Posteriori approach)	4
2.1	L'approche bayésenne	4
2.2	la problématique du prior	5
3	La méthode ADMM de résolution des problèmes inverses	6
4	Le choix du débruiteur	8
4.1	Propriétés	8
4.1.1	Invariance au changement d'échelle	8
4.1.2	Forte passivité	9
4.1.3	Neutralité (non-modification d'une image non bruitée)	9
4.1.4	(pseudo-)linéarité	10
4.1.5	quasi-stabilité locale	10
4.2	Synthèse	10
4.3	Choix du débruiteur et vérifications	11
5	La régularisation par débruiteur (RED)	12
5.1	Quel débruiteur pour un prior donné ?	14
5.2	"Kerneliser" les priors	14
6	Résolution de Problèmes Inverses	14
6.1	Descente de Gradient	15
6.2	Méthode ADMM	15
6.3	Point Fixe	15
7	Comparaison des méthodes P3 et RED	16
7.1	Convexité	16
7.2	Choix d'un "prior" alternatif	16
7.3	conditions d'équivalence Plug-and-Play avec RED	17
7.4	Conclusion: Supériorité de RED sur P3	18
7.5	Application de la méthode ADMM	19
7.6	Application de la méthode Fixed Point	20
7.7	Application de la méthode Steepest Descent	21
8	Débruiteur proximal et Réseau de Neurones	21
8.1	Débruiteur proximal à descente de gradient	22
8.2	Analyse de la convergence des débruteurs proximaux dans les méthodes PnP	22
8.2.1	Convergence de PnP-PGD	23
8.2.2	Convergence de PnP-ADMM et PnP-DRS	24
8.3	Convergence de ces trois méthodes	25
8.4	Simulations avec un débruiteur basé sur un Réseau de Neurone entraîné	25
8.5	Simulations et courbes de convergence pour les réseaux de neurones entraîné	26
8.6	Observations :	27
8.7	Principe d'un Réseau de Neurones Pré entraîné	28

8.8	Simulations et courbes de convergence pour les réseaux de neurones Préen- traîné	28
8.9	Observations :	28
8.10	Simulations	29
8.11	Conclusion	29
9	Simulations menées dans ce projet	29
9.1	Comparaison des différents algorithmes de débruitage	31
10	Résultats	33
10.1	Simulation de RED avec un Filtre Médian	33
10.2	Simulation de RED entraîné par un Réseau de Neurones	35
10.3	Simulation de RED Pré entraîné par un Réseau de Neurones	37

1 Introduction

Ce rapport traite de la régularisation par débruitage (RED), présentée par Romano, Elad, Milanfar dans leur document "The Little Engine That could: regularization by Denoising" en 2016.

Le traitement d'image se ramène en général à la résolution d'un problème d'optimisation. La fonctionnelle à optimiser est la somme d'un terme d'attaches aux données, et d'un terme de régularisation (le "prior").

Le terme de régularisation stabilise les problèmes mal posés en ajoutant des contraintes pour éviter des solutions instables ou non réalistes. Ces contraintes "a priori" sont par exemple la lissité, la parcimonie ou des hypothèses spécifiques sur les données.

Le choix de ce terme est stratégique. Les auteurs proposent une approche innovante avec un régulateur laplacien basée sur un débruiteur. Cette méthode combine ainsi des techniques de reconstruction d'images et de débruitage dans le processus d'optimisation.

Nous détaillons ci-dessous la démarche suivie par les auteurs.

2 La méthode MAP (Maximum A Posteriori approach)

À partir d'une image y observée ou mesurée, et supposée dégradée par rapport à l'image originale x , on souhaite inférer une image idéale, supposée identique ou la plus proche possible de l'image originale.

2.1 L'approche bayésienne

C'est un problème d'inférence bayésienne. Il faut chercher dans l'ensemble des images possibles x , l'image x_{MAP} qui maximise la probabilité conditionnelle d'avoir x à partir de l'image y , soit $P(x|y)$.

C'est donc un problème d'optimisation :

$$\hat{x} = \underset{x}{\operatorname{Argmax}} P(x|y)$$

Ce problème de maximisation peut classiquement être transformé en le problème de minimisation suivant :

$$\hat{x} = \underset{x}{\operatorname{Argmin}} -\log(P(x|y)) = \underset{x}{\operatorname{Argmin}} -\log(P(y|x)) - \log(P(x))$$

Comme suit :

$$\begin{aligned}
 \hat{x} &= \operatorname{Argmax}_x P(x|y) \\
 &= \operatorname{Argmax}_x \frac{P(y|x)P(x)}{P(y)} \quad (\text{par la loi de Bayes}) \\
 &= \operatorname{Argmax}_x P(y|x)P(x) \quad (\text{car } P(y) \text{ n'est pas fonction de } x, \text{ et peut donc être ignoré dans le problème}) \\
 &= \operatorname{Argmax}_x \log(P(y|x)P(x)) \quad (\text{car } \log \text{ est une fonction croissante}) \\
 &= \operatorname{Argmax}_x \log(P(y|x)) + \log(P(x)) \\
 &= \operatorname{Argmin}_x -\log(P(y|x)) - \log(P(x)).
 \end{aligned}$$

Ce qui peut s'interpréter comme suit:

- Le terme $[-\log P(y|x)]$ est un terme d'attache aux données. Il condense la relation probabiliste entre l'image recherchée x et l'image observée y sous la forme d'une vraisemblance logarithmique ("log-likelihood term"). On le note :

$$\ell(y, x) = -\log P(y|x)$$

- Le terme $[-\log P(x)]$ est un terme de régularisation. Il est aussi appelé *prior*, car la probabilité de l'image x est une probabilité *a priori*. Il représente le caractère statistique du problème. Concrètement, c'est un terme qui permet de régulariser les solutions apportées par $\ell(y, x)$, en termes de stabilité et de cohérence. On le note :

$$\lambda\rho(x) = -\log P(x)$$

où λ est un paramètre traduisant la confiance en ce terme.

En synthèse, le traitement de l'image y se traduit par la résolution du problème d'optimisation :

$$\hat{x} = \operatorname{Argmin}_x \ell(y, x) + \lambda\rho(x)$$

où :

- $\ell(y, x)$ est le terme d'attache aux données,
- $\rho(x)$ est le terme de régularisation *a priori*,
- et λ est le paramètre de confiance.

2.2 la problématique du prior

- Le choix de la log-vraisemblance $\ell(y, x)$ est assez classique. Il découle en général d'une modélisation de la dégradation de l'image source, sous forme :

$$y = Hx + e$$

où l'image source x subit une dégradation traduite par l'opérateur linéaire H , suivie d'un bruitage gaussien e de variance σ^2 .

Ainsi, $y - Hx = e$ suit une loi gaussienne, et on prendra naturellement :

$$\ell(y, x) = \frac{1}{2\sigma^2} \|y - Hx\|_2^2.$$

- Le choix du terme de régularisation est beaucoup plus stratégique. Beaucoup de modèles ont été proposés. On peut citer parmi les plus utilisés:
 - Le lissage **Laplacien**,
 - La régularisation **TV (Total Variation)**,
 - Les vaguelettes **wavelets sparsity**,
 - etc.
- Cette modélisation traduit un large spectre de problèmes inverses, tels que :
 - le débruitage,
 - le défloutage,
 - la démosaïcisation,
 - la super-résolution,
 - etc.
- Le caractère aléatoire de ces problèmes découle de la contamination des mesures par un bruit aléatoire qui suit une distribution quelconque (gaussienne, Laplace, gamma, Poisson, etc.).
- La résolution de ces problèmes se fait selon l'approche P3 (**Plug-and-Play Prior**) : elle consiste à résoudre spécifiquement l'optimisation pour un prior donné (pluggé), sans se préoccuper des choix en amont de ce *prior*.

3 La méthode ADMM de résolution des problèmes inverses

La méthode **Alternating Directions Method of Multipliers (ADMM)** consiste à s'affranchir des termes de régularisation dans l'optimisation de x , en introduisant des variables intermédiaires. Les contraintes associées à ces variables sont transformées en pénalités à minimiser, sous une forme quadratique plus maniable.

Pour rappel, le problème d'optimisation est :

$$\hat{x}_{\text{MAP}} = \underset{x}{\operatorname{Argmin}} \ell(y, x) + \lambda \rho(x)$$

En introduisant une variable v (telle que $v \approx x$), le problème devient :

$$\hat{x}_{\text{MAP}}, v = \underset{x, v}{\operatorname{Argmin}} \ell(y, x) + \lambda \rho(v)$$

avec la contrainte :

$$v \approx x.$$

Cette contrainte $v \approx x$ est transformée en une pénalité à minimiser, en introduisant une nouvelle variable u traduisant l'écart entre v et x . Ainsi, $u = v - x$ à minimiser. Tout cela est équivalent à débruiter une image $v = (x+u)$ contaminée par un bruit gaussien de variance $\sigma^2 = \frac{1}{\beta}$. Le problème d'optimisation devient alors :

$$\hat{x}_{\text{MAP}}, v = \underset{x, v}{\operatorname{Argmin}} \ell(y, x) + \lambda \rho(v) + \frac{\beta}{2} \underbrace{\|x - v + u\|_2^2}_{\text{terme quadratique}}$$

Le problème est résolu de manière séquentielle :

1. **Mise à jour de x :**

$$\hat{x} = \arg \min_x \ell(y, x) + \frac{\beta}{2} \|x - v + u\|_2^2$$

2. **Mise à jour de v :**

$$\hat{v} = \arg \min_v \lambda \rho(v) + \frac{\beta}{2} \|v - x - u\|_2^2$$

3. **Mise à jour de u :**

$$\hat{u} = u + x - v$$

On a donc introduit deux paramètres :

- λ , paramètre de confiance ;
- β , pénalisant la distance entre x et v .

Pour résoudre le problème suivant:

$$\hat{v} = \underset{v}{\operatorname{Argmin}} \lambda \rho(v) + \frac{\beta}{2} \|x - v + u\|_2^2 = \underset{v}{\operatorname{Argmin}} \frac{\lambda}{\beta} \rho(v) + \frac{1}{2} \|x - v + u\|_2^2$$

L'entrée dans le débruiteur ρ est donc conditionnée par :

$$\frac{\lambda}{\beta} = \sigma^2 \quad \text{soit} \quad \sigma = \sqrt{\frac{\lambda}{\beta}}$$

Le réglage fin de ces paramètres est ardu. En pratique, on choisit un paramètre β qui augmente avec le nombre d'itérations.

L'algorithme converge vers une solution optimale \hat{x}_{MAP} sous deux conditions :

- Le débruiteur f doit être non expansif (*contractant*).
- Il doit être indifférent au changement d'échelle de l'image.

4 Le choix du débruiteur

- Le débruiteur f est un cas particulier et simple des problèmes inverses :

$$y = x + e$$

correspond à la contamination de l'image idéale x par un bruit blanc gaussien e .

- Sa résolution est donnée par :

$$\hat{\mathbf{x}}_{\text{MAP}} = \arg \min_{\mathbf{x}} \frac{1}{2\sigma^2} \|\mathbf{y} - \mathbf{x}\|_2^2 + \lambda \rho(\mathbf{x}).$$

- Le débruiteur ("denoising engine") est toute fonction f donnant une solution à ce problème (peu importe le critère choisi, tel que MAP, MMSE, etc.). Ainsi :

$$\begin{aligned} f : [0; 255]^n &\rightarrow [0; 255]^n \\ y &\mapsto x \end{aligned}$$

- Le bruitage se révèle être un problème fondamentalement différent des autres dégradations d'image. Il est le seul susceptible, avec une forte probabilité, de transformer l'image à un degré tel qu'elle sortira du champ des images possibles, contrairement aux autres dégradations. Débruiteur est donc une opération fondamentale qui ramène une image dégradée dans le champ des "possibles". A condition de satisfaire quelques conditions:

1. Le débruitage doit être invariant à tout changement d'échelle. conséquence: Débruiteur une image redimensionnée doit être équivalent à redimensionner l'image débruitée.
2. Le débruitage ne doit pas amplifier les caractéristiques de l'image en entrée. conséquence: Le débruiteur doit être une fonction fortement "passive" (voire même contractante).
3. le débruitage ne doit pas modifier une image non bruitée !

4.1 Propriétés

4.1.1 Invariance au changement d'échelle

- Cette condition se traduit par :

$$f(cx) = cf(x),$$

où c est le facteur de redimensionnement. Cela signifie que f est une fonction homogène.

- Cela entraîne :

$$f(x) = \nabla_x f(x) \cdot x.$$

Démonstration :

$$\nabla_x f(x) \cdot x = \frac{f(x + \epsilon x) - f(x)}{\epsilon}.$$

En développant :

$$\begin{aligned}
 &= \frac{1}{\epsilon} (f((1 + \epsilon)x) - f(x)), \\
 &= \frac{(1 + \epsilon)f(x) - f(x)}{\epsilon}, \\
 &= \frac{f(x)(1 + \epsilon - 1)}{\epsilon}, \\
 &= f(x).
 \end{aligned}$$

4.1.2 Forte passivité

- Cette condition se traduit par :

$$\|f(x)\| \leq \|x\|.$$

Le débruiteur f doit être fortement passif, donc très peu réactif, donc f doit être contractante. Cela implique que le gradient est inférieur à 1. Plus précisément, le rayon spectral du gradient doit être inférieur ou égal à 1 :

$$\eta(\nabla_x f(x)) \leq 1$$

Démonstration :

$$\begin{aligned}
 \|\nabla_x f(x) \cdot x\| &\leq \|\nabla_x f(x)\| \cdot \|x\|, \\
 \|\nabla_x f(x)\| &\leq \eta(\nabla_x f(x)) \cdot \|x\| \leq \|x\|.
 \end{aligned}$$

Des centaines d'algorithmes de débruitage possèdent ces caractéristiques, comme le NLM (Non Linear Mean), le K-SVD (Kernel regression), ...

4.1.3 Neutralité (non-modification d'une image non bruitée)

- Cette condition se traduit par :

$$f(c \cdot \mathbb{I}) = c \cdot \mathbb{I},$$

où \mathbb{I} satisfait :

$$W \cdot \mathbb{I} = \mathbb{I}.$$

- Cela implique que :
 - \mathbb{I} est un vecteur propre.
 - $1 \in \text{Sp}(W)$, où $\text{Sp}(W)$ désigne le spectre de W .
- de plus :
 - Les coefficients de W sont tous positifs.
- En conséquence :
 - la matrice W est **stochastique**.

4.1.4 (pseudo-)linéarité

La phase linéaire du débruiteur peut être vue comme l'action d'un filtre ayant un opérateur pseudo-linéaire W :

$$f(y) = W_y \cdot y.$$

- Ce filtre est pseudo-linéaire, car l'opérateur associé W qui s'applique à l'image, dépend lui-même de cette image!
- Cependant, il est malgré tout linéaire car y , car :

$$f(y) = W_y \cdot y.$$

4.1.5 quasi-stabilité locale

Une conséquence de l'homogénéité (condition 1) est la quasi-stabilité de W_y au voisinage de y .

Démonstration : On écrit :

$$f(y + h) = f(y) + \nabla_y f(y) \cdot h,$$

où :

$$f(y + h) \approx W_y \cdot y + \nabla_y f(y) \cdot h.$$

Cela implique que :

$$f(y + h) \approx \underbrace{\nabla_y f(y)}_{W_y} \cdot [y + h],$$

avec W_y quasi-stable.

4.2 Synthèse

En synthèse, le cheminement que l'on a suivi, à partir des conditions que doit satisfaire un débruiteur f (homogénéité, passivité, neutralité), conduit aux propriétés suivantes :

$$\begin{aligned} f(y) &= \nabla_y f(y) \cdot y \\ f(y) &= W_y \cdot y \end{aligned}$$

où W est un opérateur (pseudo)linéaire, une matrice stochastique, avec $\eta \leq 1$.

Par identification, on a alors :

$$\mathbf{W}_y = \nabla_y \mathbf{f}(y)$$

4.3 Choix du débruiteur et vérifications

- Des centaines de débruteurs sont disponibles. L'article se focalise sur le NLM et ses variants K-SVD, BM3D, EPLL,.. ainsi que le Perceptron et les Réseaux de Neurones TRND (Trainable Non Linear Reaction Diffusion). Les auteurs ont vérifié la satisfaction des deux conditions essentielles d'homogénéité et de passivité
- l'homogénéité

L'homogénéité a été confirmée par les simulations numériques pour *K-SVD*, *BM3D*, *NLM*, *EPLL*, et *TNRD*. Les graphes $(1 + \epsilon)(f(x))$ et $f((1 + \epsilon)x)$, à la précision demandée, ressortent bien comme une droite.

- Passivité

La passivité est plus technique à démontrer, car elle nécessite le calcul du jacobien (gradient) du débruiteur f . Il faut en effet vérifier que :

$$\eta(\nabla_x f(x)) \leq 1.$$

Cependant, le rayon spectral η peut se calculer itérativement comme suit :

$$\eta(\nabla_x f(x)) = \frac{h_{k+1}^T h_k}{h_k^T h_k},$$

avec :

$$h_{k+1} = \frac{\nabla_x f(x) h_k}{\|\nabla_x f(x) h_k\|}.$$

Le vecteur h_{k+1} peut être approché à partir du développement de Taylor-Young :

$$\nabla_x f(x) \approx \frac{f(x + h_k) - f(x)}{h_k},$$

pour h_k petit.

Ainsi, h_{k+1} peut être approximé par :

$$h_{k+1} \approx \frac{f(x + h_k) - f(x)}{\|f(x + h_k) - f(x)\|_2^2}.$$

On normalise h_k à chaque itération en prenant $\|h_k\|_2 = 1$,

or : h_k est un vecteur représentant une image, donc ses pixels sont noirs, blancs ou gris.

Pour avoir cette "norme" de h_k égale à 1, il faut que :

- Les pixels (les composantes $h_k(i)$) soient de valeurs très faibles $h_k(j) \ll 1$,
- donc que les perturbations h_k soient très petites.

Cela a été vérifié sur *K-SVD*, *BM3D*, *NLM*, *EPLL*, et *TNRD*, ainsi que sur l'image "Cameraman", avec $\sigma = 5$ et un nombre de situations comparatives avec une précision de 10^{-5} .

5 La régularisation par débruiteur (RED)

Rappelons encore le problème d'optimisation :

$$\hat{x} = \arg \min_x \ell(y, x) + \lambda \rho(x),$$

Les auteurs proposent de prendre un débruiteur $f(x)$, tel que :

$$f(x) = W_x \cdot x = \nabla_x f(x).$$

La régularisation laplacienne est alors de la forme :

$$\rho_L(x) = \frac{1}{2} x^T L(x) x,$$

avec :

$$\begin{aligned} L(x) &= \nabla_x (x - f(x)) \\ &= \mathbb{I} - W_x. \end{aligned}$$

Ainsi, $\rho_L(x)$ devient :

$$\begin{aligned} \rho_L(x) &= \frac{1}{2} x^T [1 - W_x] \cdot x \\ &= \frac{1}{2} x^T [x - W_x \cdot x]. \end{aligned}$$

- Le terme $x - W_x \cdot x$ est identifié à un résidu entre l'image et son image restaurée.
- Le terme global $x^T [x - W_x \cdot x]$ est une auto-corrélation.

La condition de neutralité stipule que le débruiteur laisse inchangée une image non bruitée. Ainsi, pour une image x proche du x_{MAP} , on devrait avoir :

$$\rho_L(x) \approx 0,$$

Autrement dit :

$$x^T [x - W_x \cdot x] \approx 0.$$

cela conduit à deux possibilités:

- $x - W_x \cdot x \approx 0 \Rightarrow x \approx W_x \cdot x$, c'est-à-dire que x est un "point fixe".
- $x^T [x - W_x \cdot x] \approx 0 \Rightarrow$ les résidus sont orthogonaux à l'image.

Les résidus ne présentent plus de corrélation avec l'image de départ.

Nous avons ainsi trouvé le "prior" :

$$\lambda \rho_L(\mathbf{x}) = \frac{\lambda}{2} \mathbf{x}^T [\mathbf{x} - \mathbf{W}_x \cdot \mathbf{x}].$$

et le problème d'optimisation devient :

$$\hat{x} = \arg \min_x \ell(y, x) + \frac{\lambda}{2} x^T [x - W_x \cdot x].$$

Sa résolution nécessite de calculer le gradient de W_x . Cette difficulté peut être surmontée à partir de :

$$\mathbf{W}_x \cdot \mathbf{x} = \mathbf{f}(\mathbf{x})$$

Ainsi, en réécrivant la fonction de prior :

$$\lambda \rho_L(x) = \frac{\lambda}{2} x^T [x - f(x)],$$

on reformule le problème d'optimisation comme suit :

$$\hat{\mathbf{x}} = \arg \min_{\mathbf{x}} \ell(\mathbf{y}, \mathbf{x}) + \frac{\lambda}{2} \mathbf{x}^T [\mathbf{x} - \mathbf{f}(\mathbf{x})]$$

Avec un régularisateur laplacien qui s'écrit :

$$\rho_L(x) = \frac{1}{2} x^T [x - f(x)]$$

C'est le principe de la Régularisation par Débruitage (RED), valable pour tout filtre $f(x)$, même non pseudo-linéaire.

L'énergie totale à optimiser devient :

$$E(x) = \ell(y, x) + \frac{\lambda}{2} x^T [x - f(x)].$$

Le gradient de $E(x)$ s'écrit :

$$\nabla E(x) = \nabla_x \ell(y, x) + \lambda \nabla_x \rho_L(x).$$

En développant $\rho_L(x) = \frac{1}{2} x^T [x - f(x)]$, on obtient :

$$\begin{aligned} \nabla \rho_L(x) &= \nabla_x \left[\frac{1}{2} x^T x - \frac{1}{2} x^T f(x) \right], \\ &= x - \frac{1}{2} \nabla_x (x^T f(x)), \\ &= x - \frac{1}{2} [f(x) + \nabla_x f(x)x]. \end{aligned}$$

Ainsi :

$$\nabla E(x) = \nabla_x \ell(y, x) + \lambda \left[x - \frac{1}{2} \mathbf{f}(\mathbf{x}) - \frac{1}{2} (\nabla_x \mathbf{f}(\mathbf{x})) \mathbf{x} \right]$$

Ce qui nécessite une seule application du débruiteur f (rappelons que son gradient est approximé).

5.1 Quel débruiteur pour un prior donné ?

RED propose comme régularisateurs les familles $x^T(x - f(x))$, où f est un débruiteur quelconque.

Étant donné un prior particulier, est-il possible de déterminer le débruiteur qui le construit ? Autrement dit, étant donné $\rho(x)$, quel est le débruiteur f tel que :

$$\rho(x) = \frac{1}{2}x^T(x - f(x))$$

Un calcul à partir de la condition d'homogénéité $f(cx) = cf(x)$ conduit à (*):

$$\rho(cx) = c^2\rho(x)$$

Les priors qui satisfont cela sont nombreux (par exemple, la fonction valeur totale $\text{TV}(x) = \|\nabla x\|_1^2$).

Mieux encore, tous les régularisateurs de la forme $\rho(x) = \|Ax\|_q^2$ sont 2-homogènes et satisfaisants. On peut donc faire l'hypothèse que le prior est 2-homogène.

En différentiant l'équation (*) et en résolvant, nous aboutissons à :

$$f(x) = x - \nabla\rho(x)$$

Ainsi, à partir d'un prior donné, à la condition qu'il soit **2-homogène**, nous pouvons déterminer un débruiteur f qui le simule avec :

$$f(x) = x - \nabla\rho(x)$$

5.2 "Kerneliser" les priors

Étant donné un prior $\rho(x)$, peut-on avoir un débruiteur pseudo-linéaire : $f(x) = W(x)x$? Ce type de filtres est facile à implémenter, et ils révèlent le type de moyennage réalisé sur l'image.

Nous avons $f(x) = \nabla f(x)x$. Nous cherchons W tel que $f(x) = W(x)x$. Nous avons donc $(\nabla f(x) - W(x))x = 0$ pour tout x , d'où $\nabla f(x) = W(x)$.

En différentiant l'équation $f(x) = x - \nabla\rho(x)$ et en remplaçant $\nabla f(x)$ par $W(x)$, nous trouvons :

$$W(x) = I - H(\rho(x))$$

où H est la hessienne du prior ρ .

6 Résolution de Problèmes Inverses

La méthode **RED** (Regularization by Denoising) peut être utilisée pour résoudre des problèmes inverses généraux, de la forme :

$$\ell(y, x) = \frac{\lambda}{2\sigma^2}\|Hx - y\|_2^2$$

La fonction énergie à optimiser est :

$$E(x) = \ell(y, x) + \frac{\lambda}{2}x^T[x - f(x)]$$

La résolution de l'optimisation de $E(x)$ peut se faire avec plusieurs algorithmes. Les auteurs ont proposé les trois suivants :

- la descente de gradient
- La méthode **ADMM**
- La méthode du point fixe **Fixed-Point Strategy**

6.1 Descente de Gradient

Cette méthode est la plus classique. Elle suit l’itération suivante :

$$\begin{aligned}\hat{\mathbf{x}}_{k+1} &= \hat{x}_k - \mu \nabla_x E(x_k) \\ &= \hat{\mathbf{x}}_k - \mu [\mathbf{l}(\mathbf{y}, \mathbf{x}_k) + \lambda (\hat{\mathbf{x}}_k - \mathbf{f}(\hat{\mathbf{x}}_k))]\end{aligned}$$

Cette méthode est lente mais peut être accélérée par des variantes telles que le gradient conjugué, ou le SESOP (Sequential Subspace Optimization)

6.2 Méthode ADMM

Comme décrit précédemment, cette méthode repose sur l’introduction de variables auxiliaires et suit l’optimisation suivante :

$$\hat{v} = \operatorname{Argmin}_v \frac{\lambda}{2} v^T [v - f(v)] + \frac{\beta}{2} \|v - x - u\|_2^2$$

Le gradient total est :

$$\nabla_v \hat{v} = \lambda v - \frac{\lambda}{2} (f(v) + v \cdot f'(v)) + \beta(v - x - u)$$

L’annulation du gradient donne :

$$\lambda v - \frac{\lambda}{2} (f(v) + v \cdot f'(v)) + \beta(v - x - u) = 0$$

Ce qui donne :

$$v(\lambda + \beta) = \beta(x + u) + \frac{\lambda}{2} (f(v) + v \cdot f'(v))$$

Elle peut être résolu de manière itérative en utilisant la stratégie de point fixe, comme suit :

$$\lambda(v_j - f(v_{j-1})) + \beta(v_j - x - u) = 0,$$

Ce qui conduit à :

$$\mathbf{v}_j = \frac{1}{\beta + \lambda} (\lambda \mathbf{f}(\mathbf{v}_{j-1}) + \beta(\mathbf{x} + \mathbf{u}))$$

Cet algorithme est coûteux en calcul. Il peut être amélioré par une technique d’arrêt précoce ("early termination").

6.3 Point Fixe

On cherche un x qui annule le gradient, ce qui conduit à une équation implicite.

$$\nabla_x \ell(y, x) + \lambda (x - f(x)) = 0$$

le point fixe est calculé par itération:

$$\nabla_{\mathbf{x}} \ell(\mathbf{y}, \mathbf{x}_{k+1}) + \lambda (\mathbf{x}_{k+1} - \mathbf{f}(\mathbf{x}_k)) = \mathbf{0}$$

7 Comparaison des méthodes P3 et RED

7.1 Convexité

La fonction de régularisation est donnée par :

$$\rho_L(x) = \frac{1}{2}x^T(x - f(x)).$$

Le gradient de cette fonction est :

$$\begin{aligned}\nabla_x \rho_L(x) &= x - \frac{1}{2}\nabla_x(x^T f(x)) \\ &= x - \frac{1}{2}(f(x) - x\nabla_x f(x)) \\ &= x - f(x)\end{aligned}$$

(Rappel : $f(x) = \nabla_x f(x) \cdot x$)

Sa hessienne est donnée par :

$$\begin{aligned}\nabla_x^2 \rho_L(\mathbf{x}) &= I - \nabla_x f(x) \\ &\approx \mathbf{I} - \mathbf{W}_x,\end{aligned}$$

où I désigne la matrice identité.

Pour que $\nabla_x^2 \rho_L(x)$ soit semi-définie positive (et donc que la convexité soit assurée), il est nécessaire que toutes les valeurs propres soient comprises entre 0 et 1. Autrement dit, le spectre de la matrice doit être inclus dans $[0, 1]$:

$$\text{Sp}(\nabla_{\mathbf{x}} \mathbf{f}(\mathbf{x})) \subset [0, 1]$$

C'est le cas pour les filtres NLM, SC-SVD ou BM3D.

De manière plus générale, la convexité de $\rho_L(x)$ est assurée si le jacobien $\nabla f(x)$ est stable, c'est-à-dire si $\eta(x\nabla_x f(x)) \leq 1$.

Par ailleurs, si $\rho_L(x)$ est convexe, cela implique que la log-vraisemblance $\rho_L(y, x)$ est également convexe. Par conséquent, l'algorithme converge vers un optimum global.

Ainsi, la méthode RED est plus efficace que les méthodes P3 classiques, car :

- elle repose sur des hypothèses d'homogénéité et de positivité ;
- tandis que les méthodes P3 nécessitent la convexité, qui est une contrainte plus forte.

7.2 Choix d'un “prior” alternatif

Rappel : l'image optimale doit être inchangée par le débruiteur, c'est-à-dire que :

$$x \approx f(x).$$

C'est la condition qui a motivé le choix du “prior” de RED :

$$\rho_L(x) = x^T(x - f(x)).$$

Une alternative plus intuitive serait de prendre :

$$\rho_Q(\mathbf{x}) = \|\mathbf{x} - \mathbf{f}(\mathbf{x})\|_2^2$$

qui traduirait la volonté de contrôler le résidu de bruit.

En développant $\rho_Q(x)$, nous obtenons :

$$\begin{aligned}\rho_Q(x) &= \|x - Wx\|_2^2 \\ &= \langle x - Wx, x - Wx \rangle \\ &= \langle (W - I)x, (W - I)x \rangle \\ &= [(W - I)x]^T[(W - I)x] \\ &= x^T(W - I)^T(W - I)x.\end{aligned}$$

Cela peut être comparé à $x^T(I - W)x$ de RED. Nous avons ici une régularisation d'ordre 4 qui fait intervenir le carré du laplacien.

7.3 conditions d'équivalence Plug-and-Play avec RED

On va chercher les conditions qui permettent de trouver la même image cible \hat{x} avec les deux méthodes.

- Dans le cas de RED : \hat{x} est solution de :

$$\hat{x} = \arg \min_x \frac{\beta}{2} \|y - x\|_2^2 + \frac{\lambda}{2} x^T (x - f(x)).$$

En annulant le gradient de cette fonctionnelle, on trouve que \hat{x} est solution de :

$$\beta(\hat{x} - y) + \lambda(\hat{x} - f(\hat{x})) = 0.$$

- Dans le cas d'un schéma P3 : \hat{x} est simplement l'image de y par le débruiteur f :

$$\hat{x} = f(y).$$

À partir de là, on trouve :

$$f(y) - f(f(y)) = \frac{\beta}{\lambda}(y - f(y)).$$

Cela peut s'interpréter comme suit : Le résiduel du débruitage après une première activation du débruiteur f , donné par $(y - f(y))$, est le même, à une constante près, après une première itération $(f(y) - f(f(y)))$.

En considérant le caractère pseudo-linéaire du débruiteur ($f(y) = Wy$) et en supposant de plus que W est diagonale, un calcul simple donne :

$$\left(\frac{\beta}{\lambda} I - W \right) (I - W) = 0,$$

ce qui conduit à la condition nécessaire pour avoir une équivalence entre RED et P3 :

$$\mathbf{Sp}(\mathbf{W}) = \left\{ \mathbf{1}, \frac{\beta}{\lambda} \right\}$$

Cette condition est assez contraignante.

7.4 Conclusion: Supériorité de RED sur P3

les facteurs suivants démontrent la supériorité du schéma RED sur P3 :

- même complexité algorithmique Les deux schémas (RED et P3) sont globalement de même complexité algorithmique (dans le cas d'une utilisation de l'ADMM, et avec une seule itération pour le calcul de v).
- Le problème d'optimisation est mieux défini avec RED. RED propose une régularisation explicite, clairement définie à partir du débruiteur, tandis que P³ repose sur une régularisation implicite dépendant de l'algorithme utilisé.
- meilleures garanties de convergence RED offre des garanties de convergence plus robustes grâce à la convexité de son terme de régularisation.
- Flexibilité de l'optimisation P³ est étroitement lié à l'algorithme ADMM, ce qui peut limiter la flexibilité dans le choix des schémas d'optimisation. RED, en revanche, peut être utilisé avec n'importe quelle méthode d'optimisation, et pour n'importe quel problème inverse.
- Le schéma P3 est assez ardu à paramétrier.

7.5 Application de la méthode ADMM

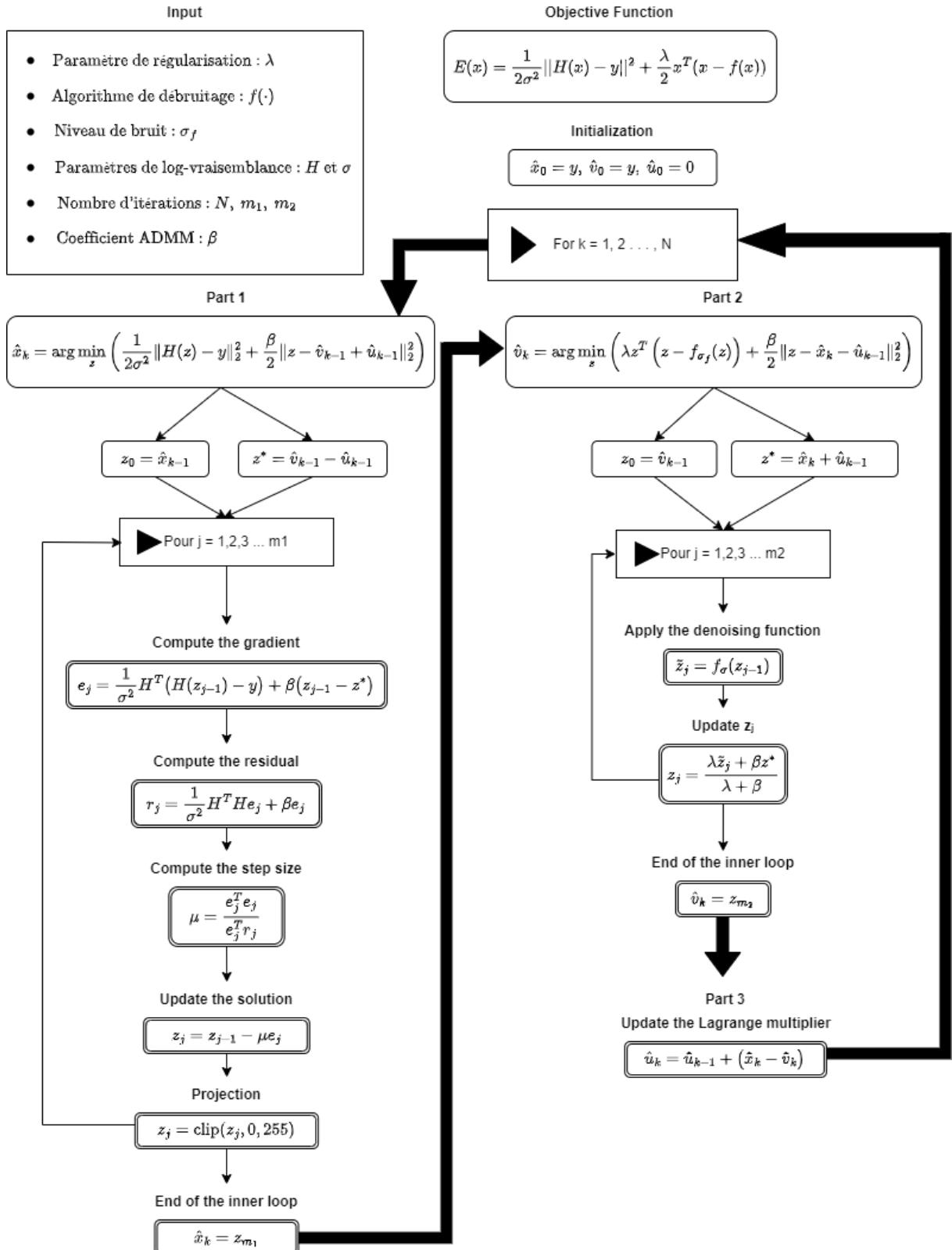


Figure 1: Diagramme décrivant l'optimisation avec la méthode ADMM dans le cadre de RED.

7.6 Application de la méthode Fixed Point

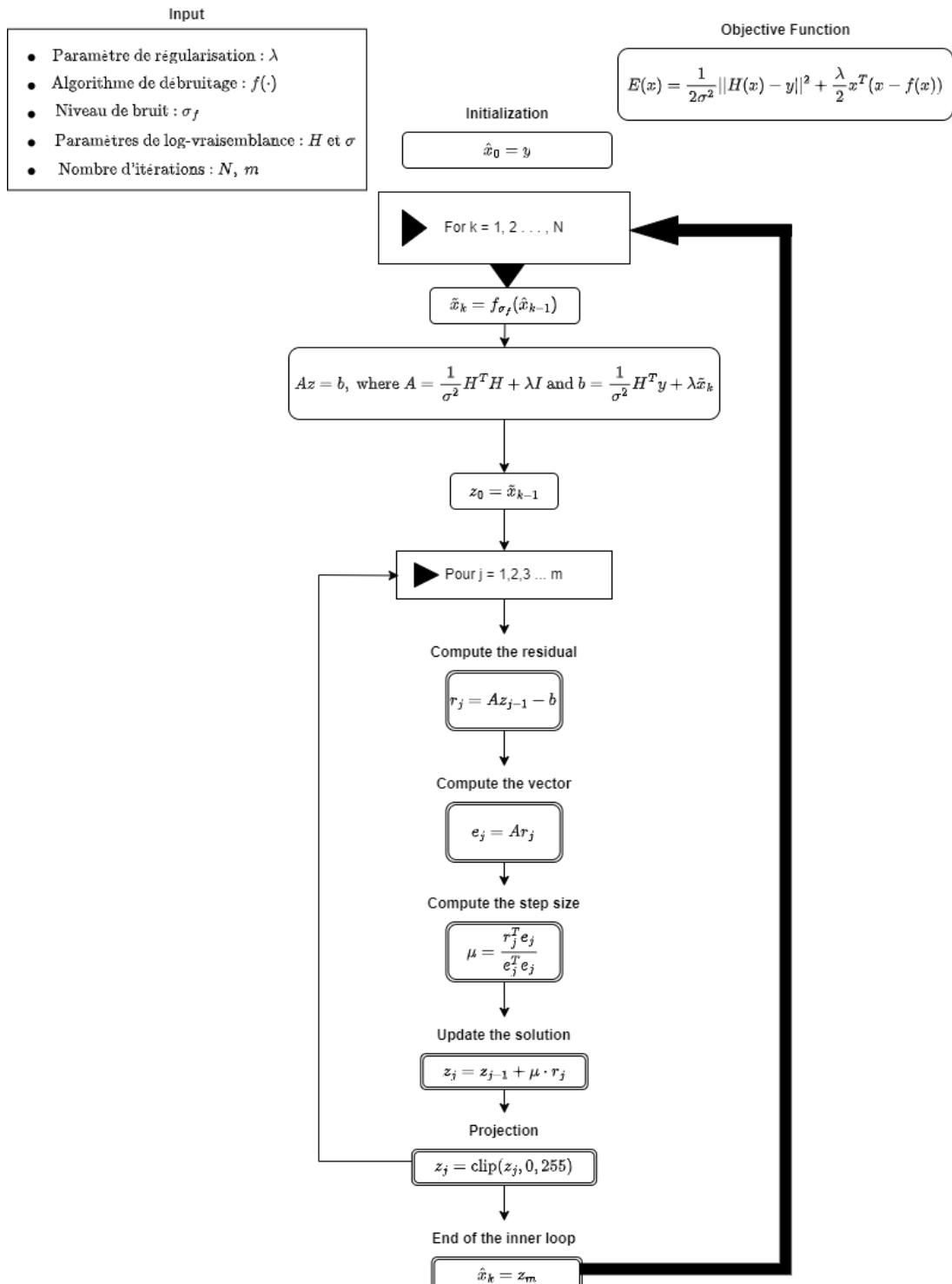


Figure 2: Diagramme décrivant l'optimisation avec la méthode Fixed Point dans le cadre de RED.

7.7 Application de la méthode Steepest Descent

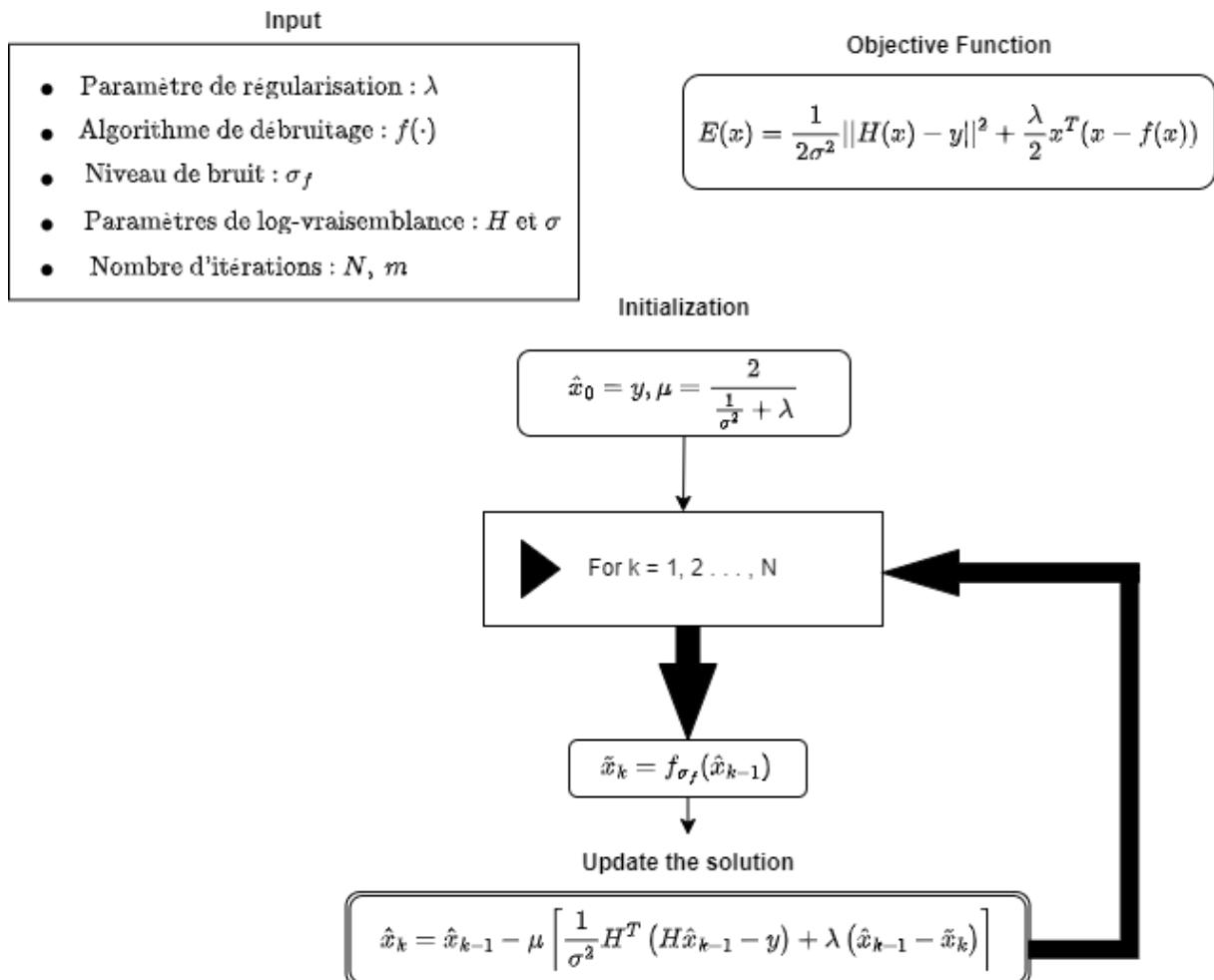


Figure 3: Diagramme décrivant l'optimisation avec la méthode Steepest Descent dans le cadre de RED.

8 Débruiteur proximal et Réseau de Neurones

Un développement a été proposé par Hureault, Leclaire et Papadakis en 2022.

(remarque: La section qui suit est une présentation de leur article et de leurs résultats mais ne s'attarde pas sur les démonstrations, techniques.)

Ces auteurs remarquent que la convergence d'un problème inverse basé sur un débruiteur n'est pas bien comprise théoriquement, et requiert des conditions peu réalistes de non expansivité, ou de forte convexité pour le terme d'attache.

L'utilisation plus récente de réseaux de neurones pour entraîner le débruiteur a donné des résultats spectaculaires, mais l'analyse théorique de la convergence demeure très incomplète.

Dans leur article, les auteurs démontrent qu'un débruiteur à descente de gradient correspond à l'opérateur proximal d'une fonction scalaire. Ils utilisent alors la théorie de convergence des algorithmes proximaux dans un cadre non convexe, pour démontrer la convergence dans le cas PnP-PGD et PnP-ADMM.

Le problème d'optimisation à résoudre est :

$$\arg \min_x \lambda f(x) + g(x)$$

où le terme d'attache aux données est de la forme

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

qui correspond au modèle linéaire $y = Ax + \zeta$. où A est un opérateur de dégradation linéaire, et un bruit blanc gaussien de variance σ^2

Dans plusieurs cas, l'opérateur A n'est pas inversible, ce qui conduit à des problèmes "mal posés", nécessitant le recours à des fonctions de régularisation $g(x)$ sur x , choisies a priori, et permettant l'utilisation d'algorithmes efficaces.

La manière d'encoder un réseau de neurones dans le prior $g(x)$ demeure une question ouverte. Une des voies possibles est d'utiliser des algorithmes proximaux de premier ordre à pas τ :

$$\text{Prox}_{\sigma,f}(x) = \arg \min_z \frac{1}{2\tau} \|x - z\|^2 + f(z).$$

Par exemple, le PGD ("Proximal Gradient Descent", ie "Forward-Backward Splitting"). Ou, dans des contextes plus généraux, le *Half Quadratic Splitting* (HQS), l'ADMM et le *Douglas-Rachford Splitting* (DRS).

8.1 Débruiteur proximal à descente de gradient

Les auteurs utilisent un débruiteur à descente de gradient (gradient-step denoiser) de la forme suivante :

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

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

et g_σ est une fonction scalaire paramétrée par un réseau de neurones.

Les auteurs montrent que ce débruiteur peut être considéré comme un opérateur proximal. Cela leur permet de généraliser le théorème de Moreau dans le cas de débruiteurs expansifs.

Dans le cas où h_σ est convexe, le débruiteur GS $D_\sigma = \nabla h_\sigma$ est lié à une certaine fonction $\phi(x)$ telle que :

$$D_\sigma(x) \in \text{Prox}_{\phi_\sigma}(x) = \arg \min_z \frac{1}{2} \|x - z\|^2 + \phi_\sigma(z)$$

De plus, si le résidu $\text{Id} - D_\sigma$ est contractant, alors il existe une fonction lisse ϕ_σ telle que :

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

et D_σ est unique.

8.2 Analyse de la convergence des débruiteurs proximaux dans les méthodes PnP

Les auteurs étudient la convergence des 3 algorithmes : PGD, ADMM et DRS dans le cas des méthodes PnP utilisant le débruiteur proximal :

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

correspondant à l'opérateur proximal d'une fonction de régularisation non convexe ϕ_σ . La fonctionnelle est alors :

$$F_{\lambda,\sigma} := \lambda f + \phi_\sigma$$

où f est le terme d'attache aux données, éventuellement non convexe, λ est un terme de régularisation, et ϕ_σ est défini dans le point précédent.

Les auteurs démontrent la convergence de ces 3 algorithmes à condition que les termes itérés soient bornés, et que $F_{\lambda,\sigma}$ vérifie les conditions de Kurdyka-Łojasiewicz (KL).

Remarque: Les conditions de Kurdyka-Łojasiewicz (KL) sont un ensemble de conditions qui permettent de garantir la convergence d'algorithmes d'optimisation dans des problèmes non convexes. Ces conditions sont particulièrement utiles dans le contexte des méthodes de type gradient, et sont souvent utilisées pour démontrer la convergence des algorithmes d'optimisation dans des espaces où la fonction objective n'est pas nécessairement convexe.

8.2.1 Convergence de PnP-PGD

La descente de gradient classique s'écrit :

$$\text{Id} - \tau \lambda \nabla f$$

On prend un pas $\tau = 1$. L'algorithme PnP-PGD s'écrit :

$$\begin{cases} z_{k+1} = x_k - \lambda \nabla f(x_k) \\ x_{k+1} = D_\sigma(z_{k+1}) \end{cases}$$

Les auteurs prennent les hypothèses suivantes :

- f et g_σ sont bornées inférieurement.
- f est différentiable, et son gradient est L_f -Lipschitz.
- $\lambda L_f < 1$

Alors :

- $(F_{\lambda,\sigma}(x_k))$ converge.
- Le résidu $\|x_{k+1} - x_k\|$ converge vers 0.
- Les points d'accumulation de la suite (x_k) sont des points critiques (i.e., à gradient nul) de $F_{\lambda,\sigma}$.
- Si f et g_σ sont respectivement KL et semi-algébriques(*), alors la suite (x_k) est bornée et converge vers un point critique de $F_{\lambda,\sigma}$.

Remarque: une fonction f est semi-algébrique, s'il existe des relations algébriques polynomiales qui définissent cette fonction et son domaine

8.2.2 Convergence de PnP-ADMM et PnP-DRS

La version classique de PnP-ADMM pour un pas τ s'écrit :

$$\begin{cases} y_{k+1} = \text{Prox}_{\tau \lambda f}(x_k), \\ z_{k+1} = D_\sigma(y_{k+1} + x_k), \\ x_{k+1} = x_k + (y_{k+1} - z_{k+1}). \end{cases}$$

Eckstein et Fukushima ont montré que PnP-ADMM est équivalent à PnP-DRS :

$$\begin{cases} y_{k+1} = \text{Prox}_{\tau \lambda f}(x_k), \\ z_{k+1} = D_\sigma(2y_{k+1} - x_k), \\ x_{k+1} = x_k + \beta(z_{k+1} - y_{k+1}). \end{cases}$$

Les auteurs se contentent donc de prouver la convergence pour PnP-DRS. Il s'agit de minimiser la somme de deux fonctions non convexes, dont l'une est différentiable à gradient de Lipschitz.

Ils distinguent deux cas selon la différentiabilité ou non de f :

Cas 1 : f est différentiable et à gradient de Lipschitz

La fonction f est supposée différentiable à gradient de Lipschitz. Les auteurs prennent un pas $\tau = 1$ et un paramètre $\beta = 1$. L'algorithme PnP-DRS devient :

$$\begin{cases} y_{k+1} = \text{Prox}_{\lambda f}(x_k), \\ z_{k+1} = D_\sigma(2y_{k+1} - x_k) = \text{Prox}_{\phi_\sigma}(2y_{k+1} - x_k), \\ x_{k+1} = x_k + (z_{k+1} - y_{k+1}). \end{cases}$$

Sous les hypothèses suivantes :

- g_σ est de classe C^2 et son gradient est L -Lipschitz avec $L > 1$.
- f est convexe, différentiable et son gradient est L_f -Lipschitz.
- g_σ est borné inférieurement.
- $\lambda L_f > 1$.

Alors, la convergence est assurée avec les résultats suivants :

- La suite $(F_{\lambda, \sigma}(x_k))$ converge.
- Le résidu $\|y_k - z_k\|$ converge vers zéro.
- Les suites (y_k) et (z_k) ont les mêmes points d'accumulation, qui sont tous critiques pour $F_{\lambda, \sigma}$.
- De plus, si f et g_σ sont respectivement de type KL et semi-algébriques, alors la suite (y_k, z_k, x_k) est bornée et convergente. En outre, y_k et z_k convergent vers le même point critique de $F_{\lambda, \sigma}$.

Cas 2 : f n'est pas différentiable

Les auteurs permutent les étapes proximales et de débruitage pour se débarrasser des contraintes sur le paramètre λ .

Ils prennent les hypothèses suivantes :

- g_σ est de classe C^2 et son gradient est L -Lipschitz avec $L > \frac{1}{2}$.
- $\text{Im}(D_\sigma)$ est convexe.
- f est propre et semi-continue inférieurement.
- f et g_σ sont bornés inférieurement.

Ils montrent que la convergence de l'algorithme est alors assurée, avec les résultats suivants :

- La suite $F_{\lambda,\sigma}(x_k)$ converge.
- Le résidu $\|y_k - z_k\|$ converge vers 0.
- Tout point d'accumulation de la suite (y_k, z_k, x_k) est un point critique de $F_{\lambda,\sigma}$.
- De plus, si f et g_σ sont respectivement de type KL et semi-algébriques, alors la suite (y_k, z_k, x_k) est bornée et converge. En outre, y_k et z_k convergent vers un point stationnaire de $F_{\lambda,\sigma}$.

8.3 Convergence de ces trois méthodes

la convergence est analysée sur les courbes de l'énergie au fil des itérations pour les trois méthodes : ADMM, point fixe et descente de gradient.

8.4 Simulations avec un débruiseur basé sur un Réseau de Neurone entraîné

Le Réseau de Neurone utilise une architecture simple basée sur des couches convolutionnelles avec un encodeur et un décodeur.

L'encodeur (avec 2 couches, de 32 filtres de taille 3x3 pour la 1ere, et de 16 filtres pour la 2eme) réduit progressivement la dimensionnalité de l'image d'entrée avec des couches ReLU et des normalisations par lot. Le décodeur (2 couches aussi, de 32 filtres de taille 3x3 pour la première et un seul filtre pour la 2eme) applique ensuite des convolutions pour reconstruire l'image débruitée, avec une activation Sigmoïde.

Nous avons conçu deux versions du modèle : une avec l'option de biais activée et l'autre sans biais.

8.5 Simulations et courbes de convergence pour les réseaux de neurones entraîné

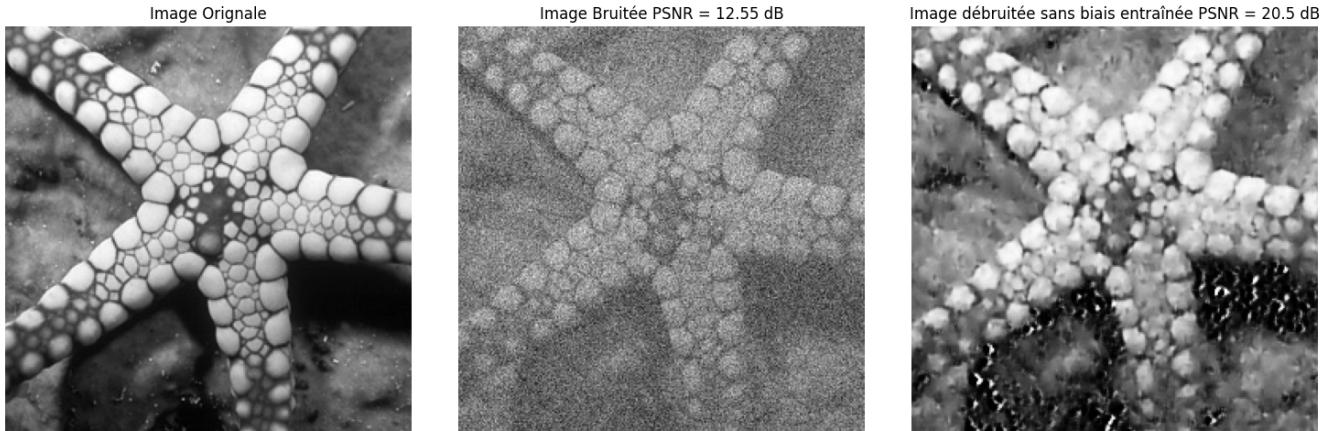


Figure 4: Comparaison entre l'image originale, bruitée et débruitée par le Réseau de Neurones entraîné sans biais.

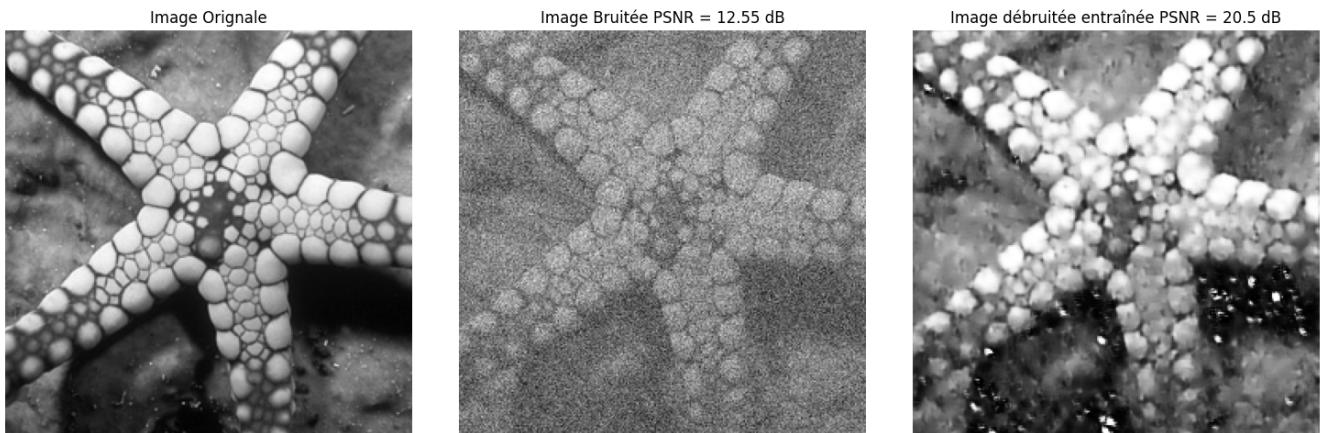


Figure 5: Comparaison entre l'image originale, bruitée et débruitée par le Réseau de Neurones entraîné avec biais.

Observations : Ce réseau de neurones a été entraîné sur un ensemble de 3000 images de dimensions 256×256 pixels pendant 10 époques, dans le but d'effectuer le débruitage des images. Les observations visuelles montrent que l'image débruitée (avec ou sans biais) présente une amélioration significative en termes de netteté et de PSNR par rapport à l'image bruitée.

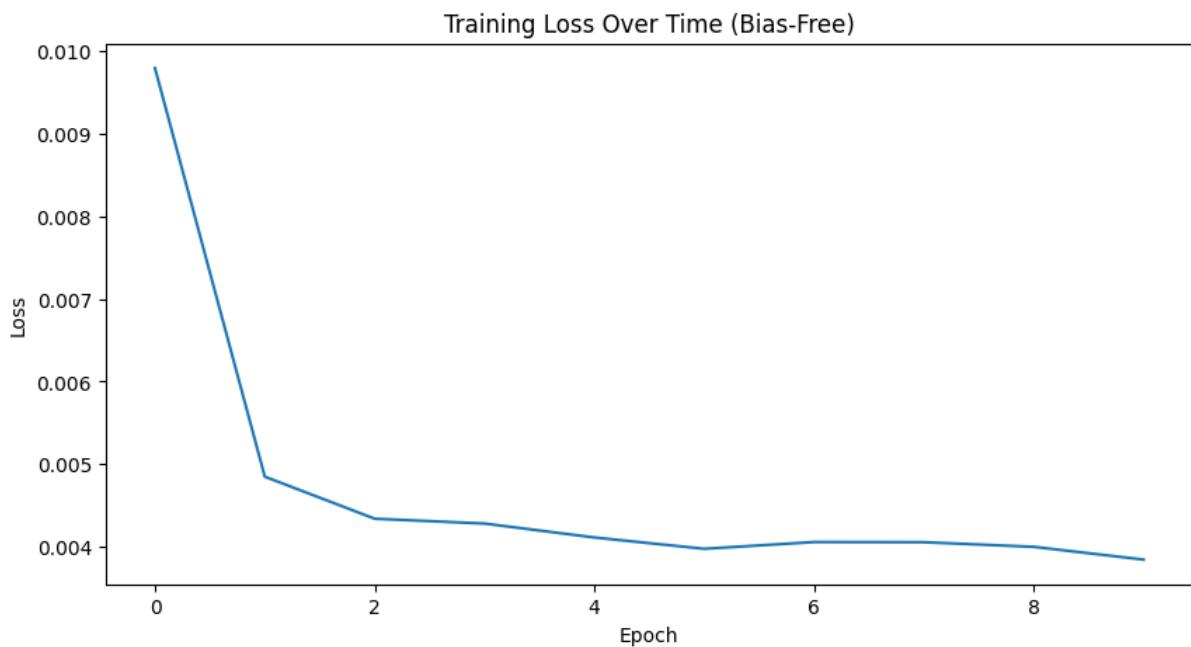


Figure 6: Courbes de convergence comparatives pour le réseau de neurones entraîné sans biais.

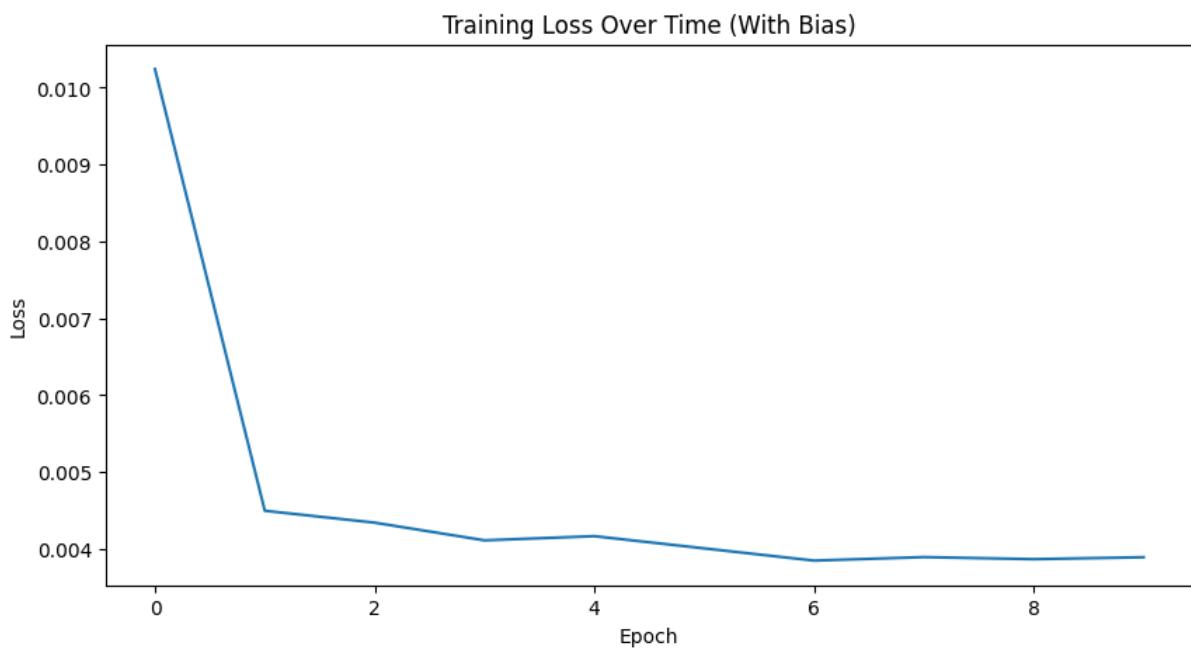


Figure 7: Courbes de convergence comparatives pour le réseau de neurones entraîné avec biais.

8.6 Observations :

Les graphes montrent une stabilisation progressive de la fonction de perte dès les premières époques. Pour les modèles avec et sans biais, une diminution rapide des pertes confirme la convergence de l'apprentissage.

8.7 Principe d'un Réseau de Neurones Pré entraîné

Le modèle de débruitage préentraîné utilise un réseau de neurones convolutionnel (CNN) d'où l'on a retiré tous les biais. Cette absence de biais garantit une invariance à l'échelle, si on multiplie tous les pixels par un même facteur, le résultat sera lui aussi simplement multiplié par ce facteur. Grâce à cet ajustement architectural, le réseau parvient à généraliser beaucoup mieux, il filtre naturellement le bruit, tout en préservant les détails importants de l'image.

Nous analyserons et comparerons les performances des versions avec et sans biais, afin de démontrer les avantages de cette architecture.

8.8 Simulations et courbes de convergence pour les réseaux de neurones Préentraîné

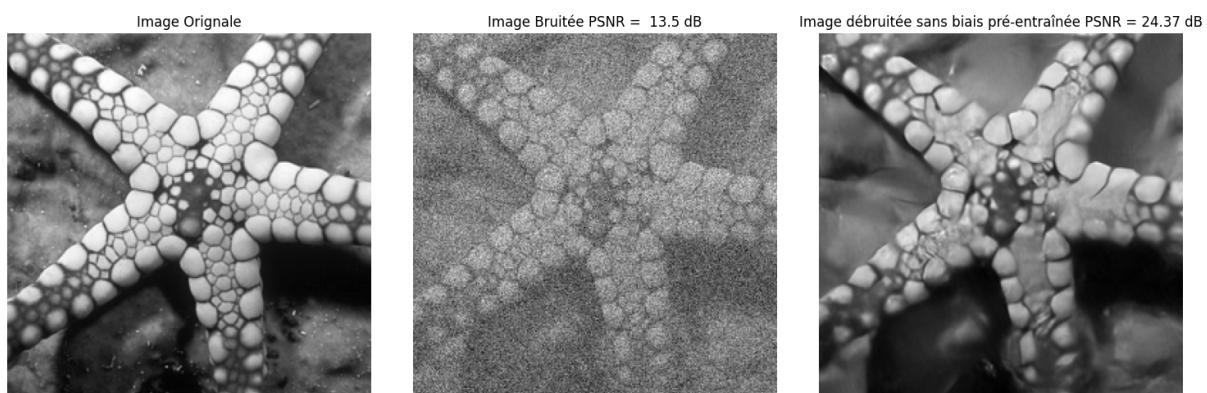


Figure 8: Comparaison entre l'image originale, bruitée et débruitée par le Réseau de Neurones Préentraîné sans biais.

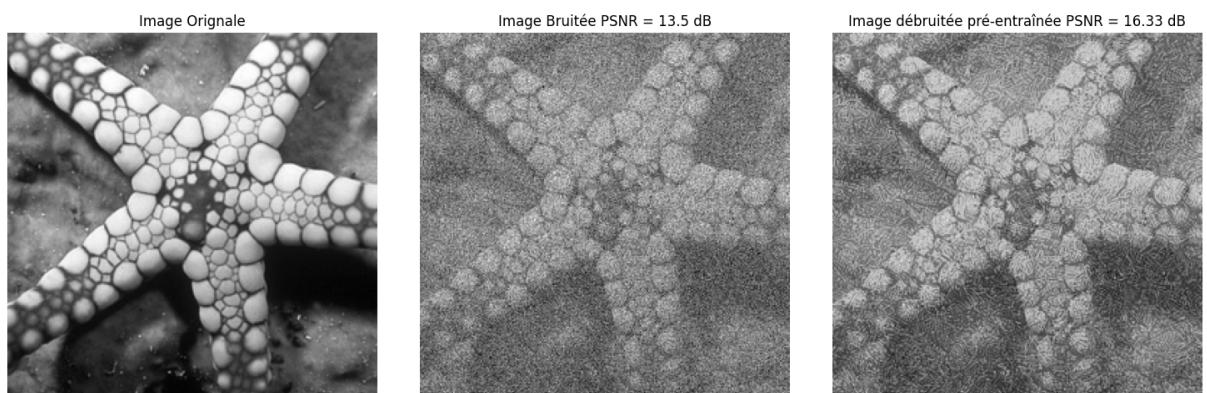


Figure 9: Comparaison entre l'image originale, bruitée et débruitée par le Réseau de Neurones Préentraîné avec biais.

8.9 Observations :

L'algorithme pré-entraîné sans biais élimine beaucoup plus de bruit, obtenant un PSNR nettement plus élevé tout en préservant davantage les détails ; à l'inverse, la ver-

sion avec biais, bien que déjà meilleure qu'une image bruitée, ne parvient pas à un niveau similaire de netteté ou de clarté.

8.10 Simulations

Les auteurs ont entraîné un réseau de neurones à descente de gradient selon les conditions pré-citées, ce qui le rend assimilable à un opérateur proximal :

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

Ils ont construit une fonction de régularisation :

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

avec N_σ un réseau de neurones paramétré avec l'architecture DRUNet. Ils ont remplacé la fonction d'activation ReLU par Softplus (qui est C^∞).

Trois types de simulations ont été menées :

- **Débruitage** : Il s'avère que le paramétrage de Lipschitz dégrade la performance, mais pour une valeur de $\mu = 10^{-2}$, le Prox-DRUNet a des performances comparables aux classiques FFDNet et DnCNN.
- **Défloutage** : L'image est dégradée par un opérateur A et un bruit gaussien d'écart-type ν . Les auteurs démontrent la supériorité de leur algorithme Prox-PnP-DRS sur les algorithmes IRCNN et DPIR, qui n'offrent pas de garantie de convergence.
- **Super-résolution** : Leur algorithme Prox-PnP-DRS réalise la meilleure performance sur tout le spectre d'échelles et de bruit. Cependant, la performance est dégradée dans le cas de l'utilisation d'un débruiteur non expansif plutôt qu'un résiduel non expansif.

8.11 Conclusion

Hureault Leclaire et Papadakis entraînent un débruiteur comme opérateur proximal d'une fonction de régularisation non convexe. Même quand ils relaxent la condition de non-expansivité, leur algorithme est du niveau des débruteurs les plus avancés.

Ils montrent que les algorithmes PnP-PGD, PnP-ADMM et PnP-DRS, quand ils sont utilisés avec le débruiteur qu'ils ont construit, sont sûrs de converger vers des points stationnaires d'une fonctionnelle explicite.

PnP-DRS est l'algorithme qui montre les meilleures performances, sous la condition que $\text{Im}(D_\sigma)$ soit convexe.

9 Simulations menées dans ce projet

Nous avons simulé les méthodes ADMM, Point Fixe, et Gradient Descent dans plusieurs configurations : PnP classiques, RED, RED avec Réseau de neurones (avec et sans biais), etc.

Les critères de comparaison utilisés étaient le PSNR, la convergence de la fonctionnelle,

le nombre d’itérations pour atteindre 28 dB, les temps, ainsi que la comparaison visuelle, plus subjective.

Pour rappel : **Le PSNR (Peak Signal to Noise Ratio)** évalue la qualité des images reconstruites ou débruitées par rapport à une image de référence (originale). Il mesure le rapport entre le signal maximal (l’image originale) et le bruit (l’erreur de reconstruction ou d’approximation) introduit par un algorithme. Plus le PSNR est élevé, meilleure est la qualité perçue de l’image reconstruite ou débruitée par rapport à l’originale.

$$\text{PSNR} = 10 \times \log_{10} \left(\frac{R^2}{\text{MSE}} \right)$$

Où :

- R : la valeur maximale possible des pixels de l’image. Pour une image 8 bits par pixel, $R = 255$. Si l’image est en format à virgule flottante entre 0 et 1, $R = 1.0$.
- MSE (Mean Squared Error) : l’erreur quadratique moyenne entre l’image originale X et l’image reconstruite Y , définie par :

$$\text{MSE} = \frac{1}{m \times n} \sum_{i=1}^m \sum_{j=1}^n (X(i, j) - Y(i, j))^2$$

Où m et n sont les dimensions de l’image (hauteur et largeur), et $X(i, j)$ et $Y(i, j)$ sont les valeurs des pixels à la position (i, j) dans les images originale et reconstruite, respectivement.

Le PSNR permet de quantifier l’efficacité du débruitage. Un PSNR autour de 30 dB signifie généralement que la qualité perçue est encore acceptable pour des applications générales.

Nous nous sommes fixés un objectif de 28 dB. (il n’a pu être atteint dans le cas de la méthode Fixed-Point, et nous nous sommes limités à 26dB)

9.1 Comparaison des différents algorithmes de débruitage

Nous avons effectué les simulations suivantes avec un débruiteur classique (Filtre Médián 3*3 et celles d'un Réseau de Neurones Entraîné et Pré entraîné):

schéma RED : Gradient Descent, Accelerated Gradient Descent, Fixed-Point, ADMM

Méthode	Input (dB)	Output (dB)	Iterations	Temps (s)
ADMM				
$\beta = 0.05, \sigma_f = 0.001,$ $\sigma = 0.01, \lambda = 0.1$	21.66	28.71	$N = 70,$ $m_1 = 5, m_2 = 1$	29
Gradient Descent				
$\sigma_f = 0.001, \sigma = 0.01, \lambda = 0.1$	21.66	28.68	$N = 1000$	31
Accelerated GD				
$\sigma_f = 0.001, \sigma = 0.01, \lambda = 0.1$	21.66	28.64	$N = 110$	4
Fixed Point				
$\sigma_f = 0.001, \sigma = 0.01,$ $\lambda = 0.1$	21.66	26.21	$N = 100,$ $m = 5$	45

Table 1: Résultats des différentes méthodes pour RED avec un filtre médian.

Méthode	Biais	Input (dB)	Output (dB)	Iterations	Temps (s)
ADMM					
$\beta = 0.05, \sigma_f = 0.001,$ $\sigma = 0.01, \lambda = 0.1$	Avec biais	21.66	28.86	$N = 70,$ $m_1 = 5, m_2 = 1$	42
$\beta = 0.05, \sigma_f = 0.001,$ $\sigma = 0.01, \lambda = 0.1$	Sans biais	21.66	28.87	$N = 70,$ $m_1 = 5, m_2 = 1$	42
Gradient Descent					
$\sigma_f = 0.001, \sigma = 0.01, \lambda = 0.1$	Avec biais	21.66	28.84	$N = 1000$	121
$\sigma_f = 0.001, \sigma = 0.01, \lambda = 0.1$	Sans biais	21.66	28.81	$N = 1000$	109
Accelerated GD					
$\sigma_f = 0.001, \sigma = 0.01, \lambda = 0.1$	Avec biais	21.66	28.83	$N = 110$	12
$\sigma_f = 0.001, \sigma = 0.01, \lambda = 0.1$	Sans biais	21.66	28.80	$N = 110$	9
Fixed Point					
$\sigma_f = 0.001, \sigma = 0.01,$ $\lambda = 0.1$	Avec biais	21.66	26.66	$N = 100,$ $m = 5$	121
$\sigma_f = 0.001, \sigma = 0.01,$ $\lambda = 0.1$	Sans biais	21.66	25.68	$N = 100,$ $m = 5$	109

Table 2: Résultats des différentes méthodes pour RED entraîné avec un Réseau de Neurones, en distinguant les cas avec biais et sans biais.

Méthode	Biais	Input (dB)	Output (dB)	Iterations	Temps (s)
ADMM					
$\beta = 0.05, \sigma_f = 0.001, \sigma = 0.01, \lambda = 0.1$	Avec biais	21.66	28.71	$N = 70, m_1 = 5, m_2 = 1$	198
$\beta = 0.05, \sigma_f = 0.001, \sigma = 0.01, \lambda = 0.1$	Sans biais	21.66	28.74	$N = 70, m_1 = 5, m_2 = 1$	261
Gradient Descent					
$\sigma_f = 0.001, \sigma = 0.01, \lambda = 0.1$	Avec biais	21.66	1263	$N = 1000$	31
$\sigma_f = 0.001, \sigma = 0.01, \lambda = 0.1$	Sans biais	21.66	2036	$N = 1000$	30
Accelerated GD					
$\sigma_f = 0.001, \sigma = 0.01, \lambda = 0.1$	Avec biais	21.66	28.64	$N = 110$	150
$\sigma_f = 0.001, \sigma = 0.01, \lambda = 0.1$	Sans biais	21.66	28.65	$N = 110$	266
Fixed Point					
$\sigma_f = 0.001, \sigma = 0.01, \lambda = 0.1$	Avec biais	21.66	29.09	$N = 100, m = 5$	93
$\sigma_f = 0.001, \sigma = 0.01, \lambda = 0.1$	Sans biais	21.66	29.18	$N = 100, m = 5$	205

Table 3: Résultats des différentes méthodes pour RED Pré entraînée avec un Réseau de Neurones, en distinguant les cas avec biais et sans biais.

10 Résultats

10.1 Simulation de RED avec un Filtre Médian

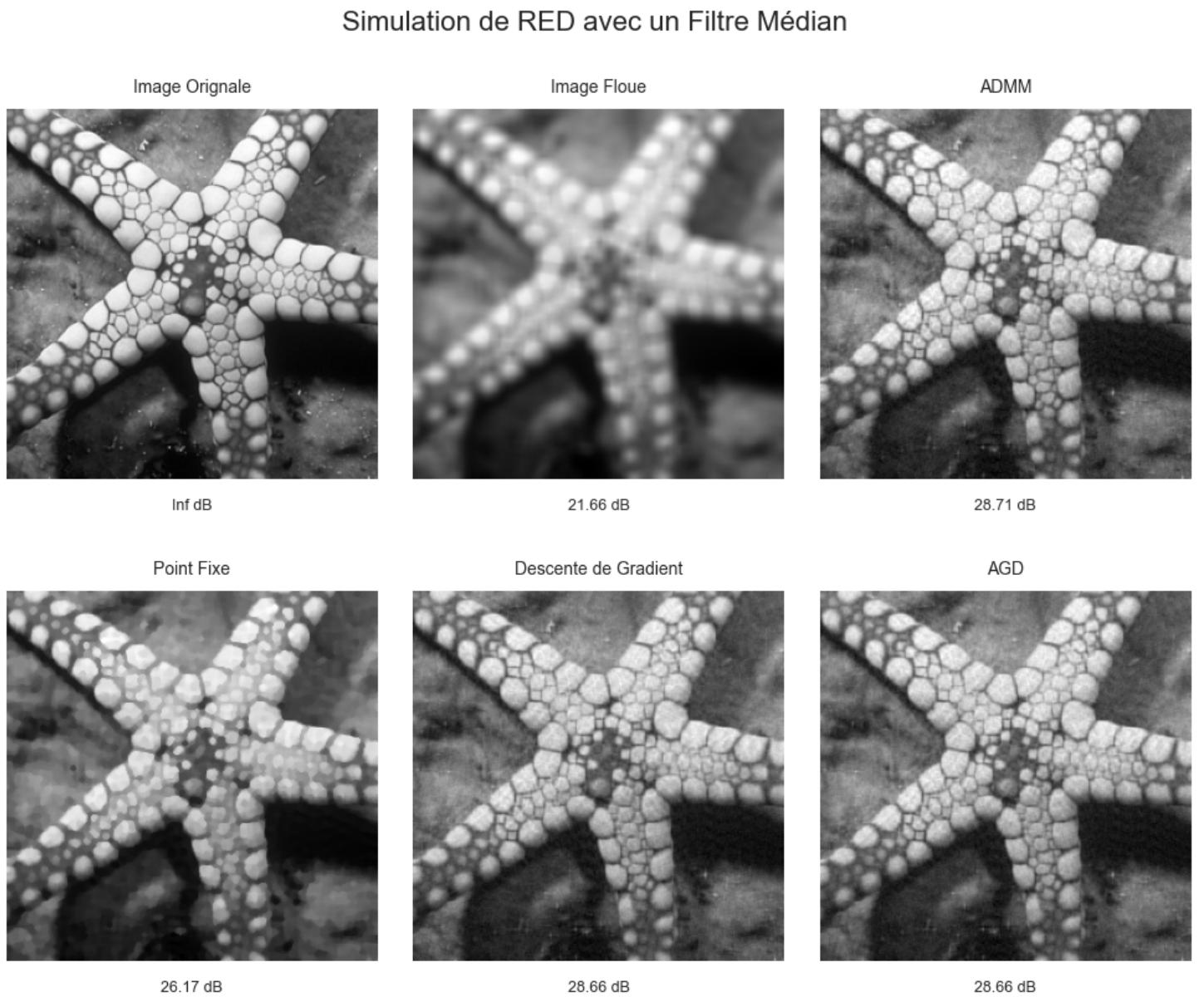


Figure 10: Comparaison des résultats de RED avec ceux d'un Filtre Médian.

Les résultats montrent qu'ADMM offre la meilleure netteté (28.71 dB), suivi de près par AGD et la Descente de Gradient (28.66 dB). Point Fixe présente une amélioration plus modérée (26.17 dB). Ces valeurs de PSNR montrent que tous les algorithmes réussissent à déflouter les images, tout en préservant la texture et le contraste, et en limitant les artefacts.

Courbe de Convergence RED avec un Filtre Médian

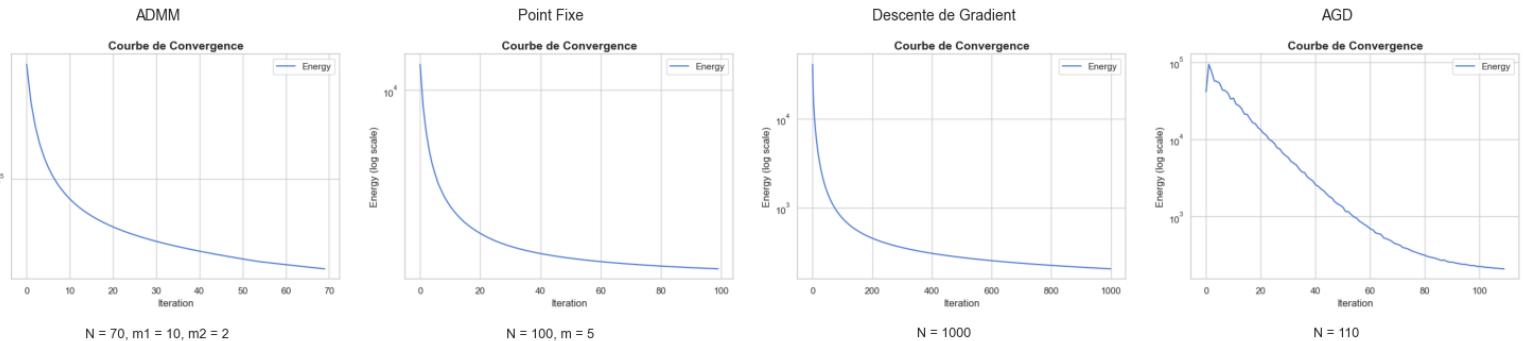


Figure 11: Courbes de convergences de RED avec celles d'un Filtre Médian.

Les courbes d'énergie révèlent qu'ADMM et Point Fixe baissent rapidement l'énergie, ADMM étant légèrement plus performant. La Descente de Gradient converge régulièrement, mais exige un plus grand nombre d'itérations, tandis qu'AGD diminue aussi de façon continue. Pour atteindre 28 dB, AGD est le plus rapide (4 s), suivi d'ADMM (29 s) et de la Descente de Gradient (31 s). En revanche, Point Fixe ne parvient pas à atteindre ce niveau, et reste en dessous de 28 dB.

10.2 Simulation de RED entraîné par un Réseau de Neurones

Simulation de RED entraînée par un Réseau de Neurones

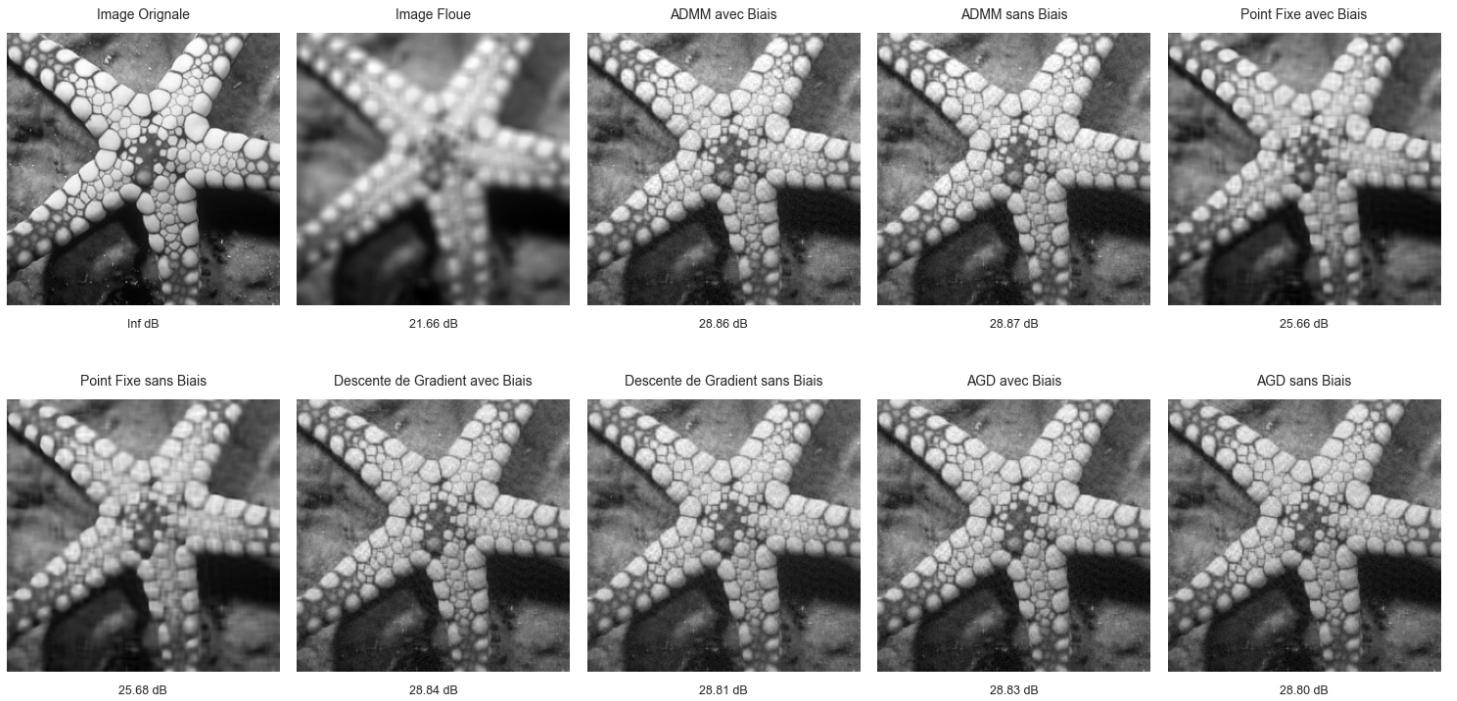


Figure 12: Comparaison des résultats de RED avec ceux d'un Réseau de Neurones entraîné.

Les résultats montrent qu'ADMM fournit la meilleure netteté (28.86 dB avec Biais, 28.87 dB sans Biais), suivi par la Descente de Gradient (28.84 dB avec Biais, 28.81 dB sans Biais) et AGD (28.83 dB avec Biais, 28.80 dB sans Biais). Point Fixe, avec un PSNR proche de 25.66 dB (avec Biais) et 25.68 dB (sans Biais), est moins performant. Dans l'ensemble, ces valeurs de PSNR indiquent que l'utilisation d'un réseau de neurones comme débruiteur (denoiser) permet d'obtenir des images de bonne qualité, avec une texture et un contraste préservés, et peu d'artéfacts.

Simulation de RED entraînée par un Réseau de Neurones

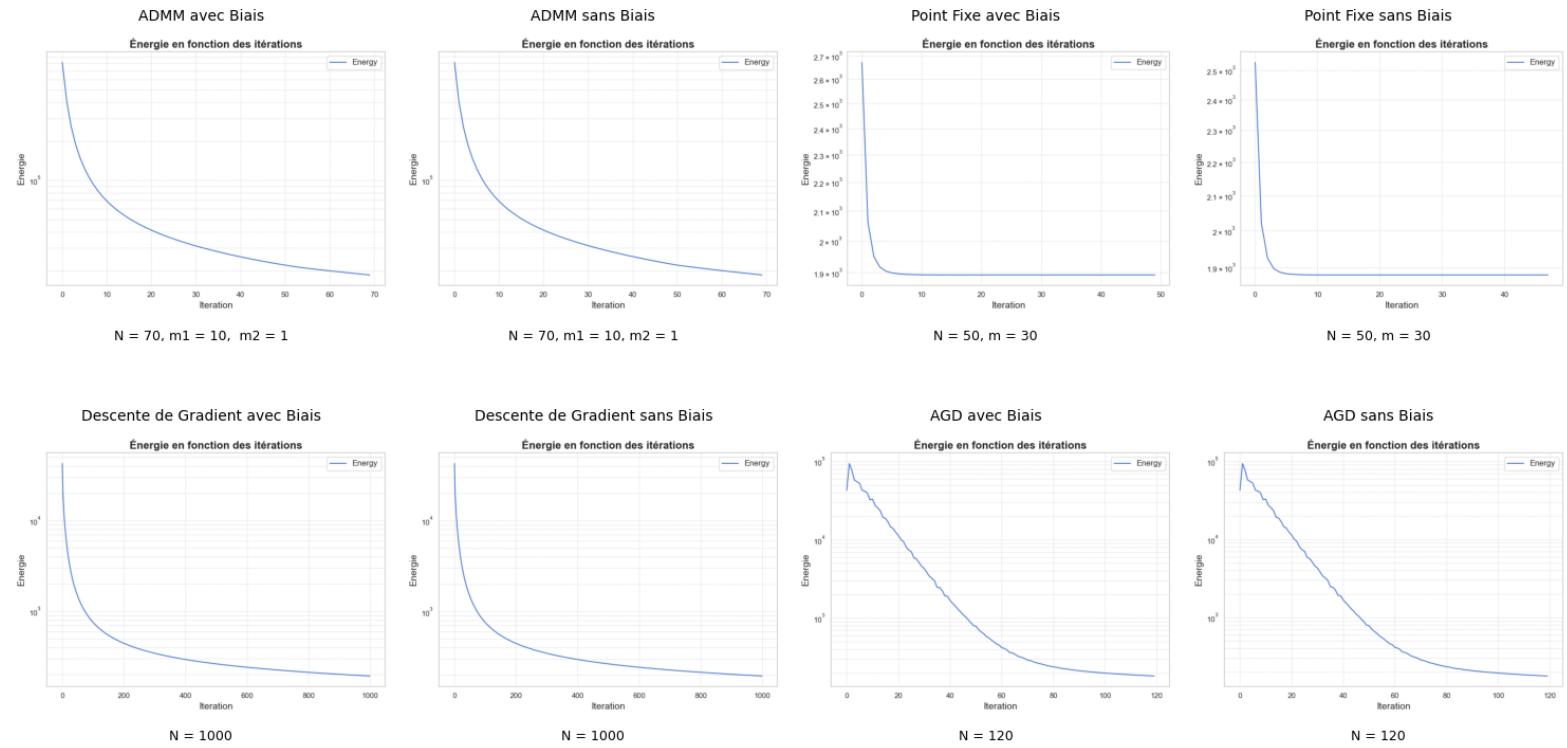


Figure 13: Courbes de convergence de RED comparées avec celles d'un Réseau de Neurones entraîné.

Du point de vue du temps de calcul, AGD est le plus rapide (12 s avec Biais, 9 s sans Biais). ADMM suit (42 s avec Biais, 41 s sans Biais), puis la Descente de Gradient (89 s avec Biais, 79 s sans Biais). Point Fixe est le plus lent (121 s avec Biais, 109 s sans Biais). En conclusion, chaque méthode converge correctement, mais le choix de l'algorithme et la présence ou l'absence de biais influencent fortement la durée nécessaire pour obtenir une image défloutée de bonne qualité.

Remarque: Les temps rapportés correspondent à la durée nécessaire pour que chaque algorithme atteigne un PSNR de 28 dB.

10.3 Simulation de RED Pré entraîné par un Réseau de Neurones

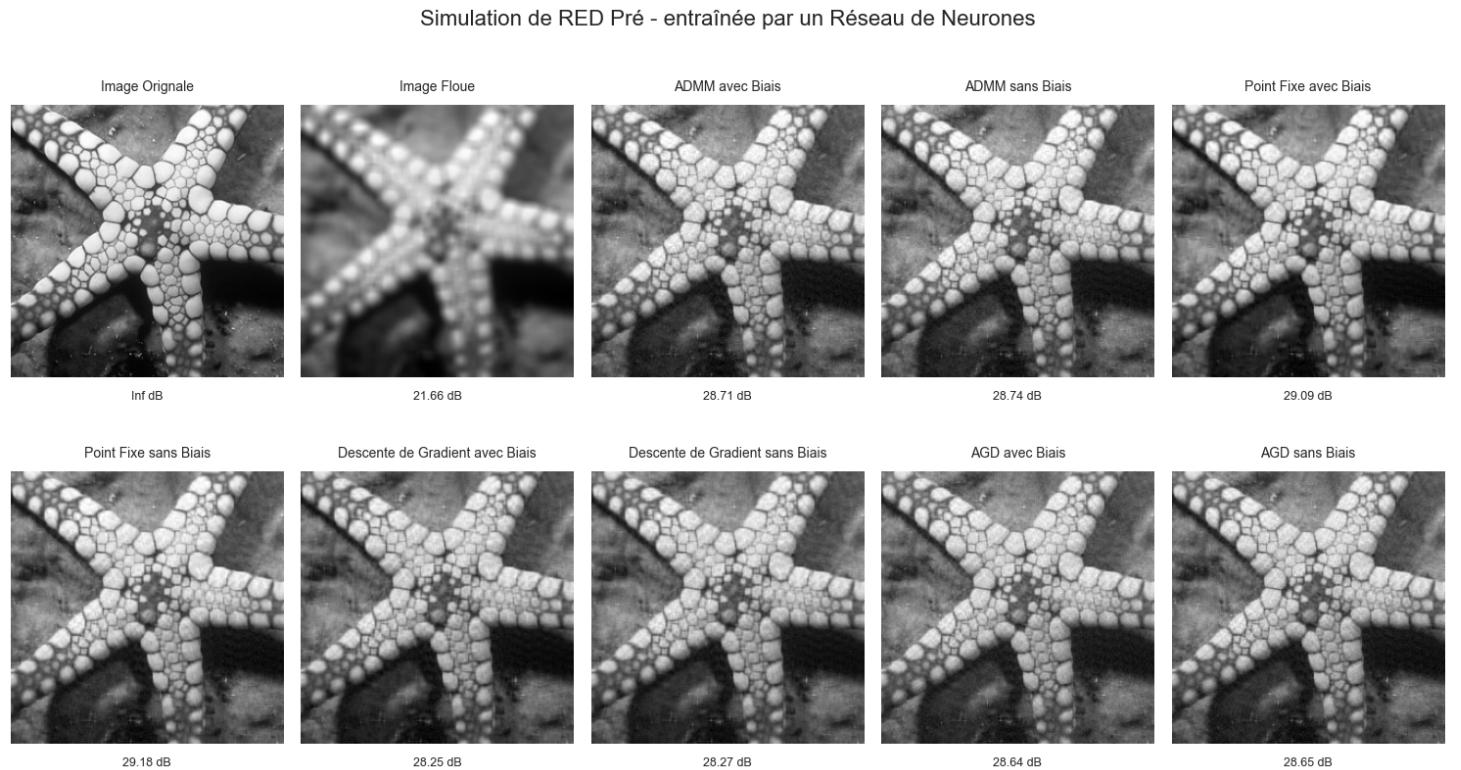


Figure 14: Comparaison des résultats de RED avec ceux d'un Réseau de Neurones Préentraîné.

Les résultats montrent que Point Fixe sans Biais offre la meilleure netteté (29.18 dB), suivi de près par Point Fixe avec Biais (29.09 dB). ADMM (28.71 dB avec Biais, 28.74 dB sans Biais) et AGD (28.64 dB avec Biais, 28.65 dB sans Biais) affichent également une bonne qualité de restauration, tandis que la Descente de Gradient (28.25 dB avec Biais, 28.27 dB sans Biais) est légèrement moins performante. Dans l'ensemble, ces valeurs de PSNR montrent que tous les algorithmes parviennent à déflouter les images en préservant texture, contraste et en limitant les artefacts.

Courbe de Convergence Pré - entraînée par un Réseau de Neurones

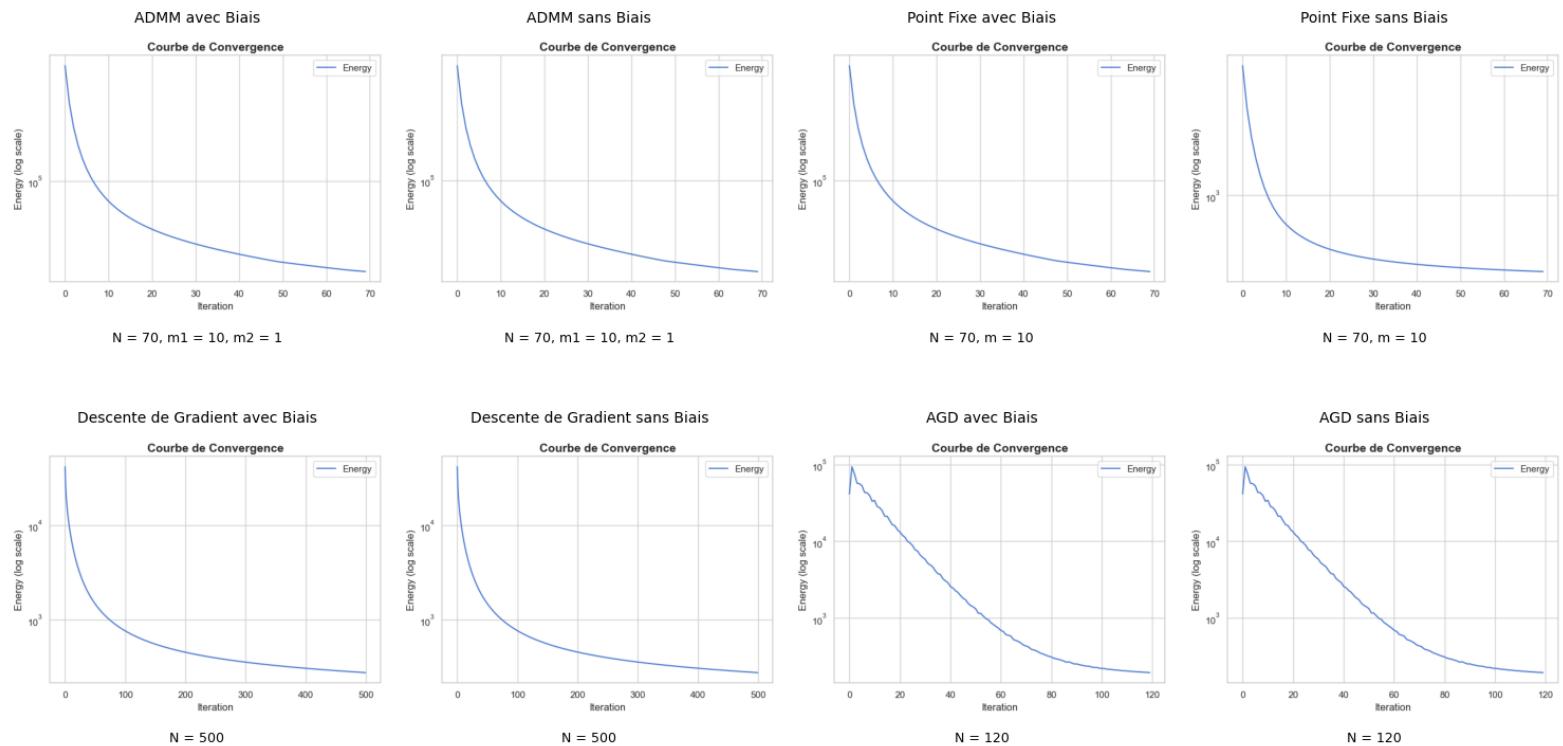


Figure 15: Courbes de convergence de RED avec ceux d'un Réseau de Neurones Préentraîné.

Les courbes d'énergie révèlent aussi que certains algorithmes obtiennent des résultats satisfaisants rapidement. Avec Biais, Point Fixe est le plus rapide (93 s), devant AGD (150 s) et ADMM (198 s), alors que la Descente de Gradient requiert 1263 s. Sans Biais, Point Fixe conserve sa rapidité (205 s), suivi d'ADMM (261 s), d'AGD (266 s) et, loin derrière, de la Descente de Gradient (2036 s). Au final, toutes les méthodes convergent correctement, mais elles diffèrent nettement au niveau du temps de calcul.

Remarque : Ici, le débruitage est assuré par un modèle de réseau de neurones pré-entraîné, ce qui influence le temps de traitement. Selon la taille et la complexité de ce réseau, l'inférence peut être plus ou moins rapide. De plus, les temps rapportés correspondent à la durée nécessaire pour que chaque algorithme atteigne un PSNR de 28 dB.