Image Denoising

Redwan Mekrami

Abstract

L'acquisition d'une image numérique est le plus souvent accompagnée de l'apparition de bruit, souvent dû à une imperfection au niveau de la détection, de la transmission ou de la compression du signal, ou encore à des défauts inhérents à l'environnement comme la présence d'éclairage insuffisant ou trop prononcé. La suppression de ce bruit est un enjeu primordial dans plusieurs domaines industriels et la recherche d'algorithmes efficaces demeure un défi persistant à la croisée de plusieurs domaines scientifiques : analyse fonctionelle, probabilités, statistiques et sciences physiques. Dans cette étude, nous implémentons différents algorithmes de restitution d'une image en niveaux de gris bruitée selon un bruit gaussien.

1 Introduction

Une première modélisation consiste à représenter une image par un élément $u \in \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}^c$, où $m \times n$ correspond au nombre de pixels et c le nombre de canaux couleurs. Traditionnellement, on en utilise généralement $c = 3$ pour les images usuelles. Ces trois canaux correspondent aux canaux rouge, vert et bleu. Les valeurs que peuvent prendre les coefficients de u constituent la dynamique de l'image. Pour les images codées en 8 bits (par exemple les images JPEG, PNG), il s'agit de valeurs entières comprises entre 0 et 255. Pour une image en niveaux de gris, une valeur proche de 0 correspond à une couleur sombre et une valeur proche de 255 correspond à une couleur claire. Fondamentalement, ces valeurs doivent être rapprochées du nombre de photons mesuré par le capteur associé au pixel considéré. Une autre convention consiste alors à normaliser les valeurs de u, de sorte qu'elles appartiennent à l'intervalle $[0; 1]$. Pour une image codée en 8 bits, il suffit de diviser alors u par 255. Dans nos implémentation numérique, nous utiliserons cette seconde convention.

1.1 Le bruit dans les images numériques

Entre l'acquisition à l'aide d'un appareil photo d'une scène, et le fichier numérique contenant l'image de cette scène, il y a une succession de phénomène qui altère l'image obtenue. On parle de bruit. Ce phénomène est particulièrement visible lorsque l'on essaie de prendre une photo avec une très faible luminosité. En réalité, le bruit est toujours présent, mais dans le cas d'une scène sombre, le signal est faible, du même ordre de grandeur que le bruit, de sorte que ce dernier est plus visible.

L'opération qui cherche à supprimer le bruit d'une image, ou à restaurer la qualité initiale d'une image, est appelée débruitage. Pour la réaliser, il faut commencer par modéliser le phénomène du bruit. Un modèle simple et populaire est celui du bruit blanc gaussien : on suppose que l'image capturée g est la somme d'une image idéale u et d'un bruit blanc gaussien n :

$$g = u + n$$

Plus précisément, le bruit ne dépend pas de la couleur du point considéré, est indépendant en chaque point de l'image, et suit la même loi gaussienne.

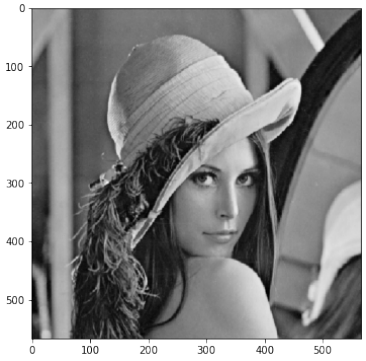

Figure 1: Image Originale

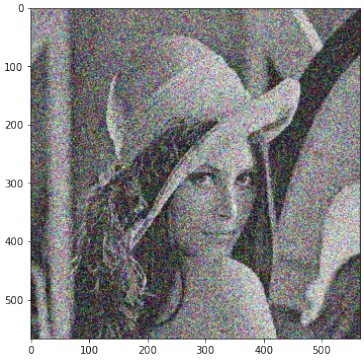

Figure 2: Image Bruitée

1.2 Gradient et divergence d'une image

Commençons tout de suite par introduire deux opérateurs qui nous seront très utiles dans la construction de nos algorithmes : l'opérateur gradient et l'opérateur divergence.

Définition 1. *L'opérateur gradient, noté ∇^h est défini à l'aide de différences finies:*

$$\nabla^h u = \left(\delta_x^h u, \delta_y^h u\right)$$

avec

$$\forall (i,j) \in [\![1;m]\!] \times [\![1;n]\!], \quad \left(\delta_x^h u\right)_{(i,j)} = \begin{cases} u_{(i+1,j)} - u_{(i,j)} & \textit{si } i \neq m \\ 0 & \textit{sinon} \end{cases}$$

et

$$\forall (i,j) \in [\![1;m]\!] \times [\![1;n]\!], \quad \left(\delta_y^h u\right)_{(i,j)} = \begin{cases} u_{(i,j+1)} - u_{(i,j)} & \textit{si } j \neq n \\ 0 & \textit{sinon} \end{cases}$$

Il s'agit des approximations de la dérivée verticale et de la dérivée horizontale, respectivement, d'une fonction $\tilde{u}$ définie sur le plan, telle que

$$\forall (i,j) \in [1;m] \times [1;n], \quad \tilde{u}(i,j) = u_{i,j}$$

Définition 2. *La divergence d'une image notée* div^h *est l'opposée de l'opérateur adjoint à* ∇^h *:*

$$\mathrm{div}^h = -\left(\nabla^h\right)^*$$

Nous pouvons énoncer sans démonstation quelques propriétés algébrique de div^h qui nous seront utiles pour la suite.

Proposition 1. *Propriété de* div^h

Pour tout $w = (w^x, w^y) \in \mathbb{R}^{m \times n} \times \mathbb{R}^{m \times n}$

$$\mathrm{div}^h w = -\left(\delta_x\right)^* w^x - \left(\delta_y\right)^* w^y$$

avec

$$\forall (i,j) \in [1;m] \times [1;n], \quad \left(\left(\delta_x\right)^* w^x\right)_{(i,j)} = \begin{cases} -w^x_{(1,j)} & \textit{si } i = 1 \\ w^x_{(i-1,j)} - w^x_{(i,j)} & \textit{si } 1 < i < m \\ w^x_{(m-1,j)} & \textit{sinon} \end{cases}$$

et

$$\forall (i,j) \in [1;m] \times \in [1;n], \quad \left(\left(\delta_y\right)^* w^y\right)_{(i,j)} = \begin{cases} -w^y_{(i,1)} & \textit{si } j = 1 \\ w^y_{(i,j-1)} - w^y_{(i,j)} & \textit{si } 1 < j < n \\ w^y_{(i,n-1)} & \textit{sinon} \end{cases}$$

De plus:

$$\left\|\left|\nabla^h\right|\right\| \leqslant 2\sqrt{2}$$

2 Approche Variationnelle

Il existe de multiples manières de débruiter une image, dans cette section, nous nous intéressons à une classe de méthodes dites variationnelles. L'idée est simple : on part d'un modèle de bruit (par exemple : bruit additif gaussien) et d'image (on y reviendra plus bas) et on définit une distance qui permet de mesurer à quel point une image v est proche de ce modèle.

Si g (l'image bruitée) est une image en niveaux de gris de taille $m \times n$, et si le bruit est un bruit additif gaussien, un exemple de telles distances est une fonction de la forme générale suivante :

$$J(v) = \frac{\lambda}{2}\|g - v\|_2^2 + R(v)$$

avec $v \in \mathbb{R}^{m\times n}, R : \mathbb{R}^{m\times n} \to \mathbb{R}^+$ et $\lambda > 0$. Le premier terme est appelé terme d'attache aux données, ou terme de fidélité. Ce terme force la reconstruction v (idéalement, l'image u) à rester proche de l'image bruitée. Il y a donc de multiples façons de le définir. Ici, on choisit la distance euclidienne au carré à l'image bruitée g (erreur quadratique moyenne). Le terme R, appelé terme de régularisation, traduit le modèle choisi pour l'image. On verra plus bas deux choix possibles, qui sont de la forme :

$$R(v) = r\left(\nabla^h v\right)$$

où $\nabla^h v = (d_x v, d_y v)$ est la discrétisation du gradient vu en première partie. L'idée sous-jacente est que le gradient d'une image "ordinaire" possède des propriétés caractéristiques. Enfin, le paramètre λ permet de pondérer l'importance relative de ces deux termes.

On remarque que, sans terme de régularité R, le minimiseur de J est donné par l'image bruitée g. Ainsi, dans cette méthode, le régularisateur R est essentiel, sans quoi il n'y a pas de débruitage possible. De surcroît, plus λ est grand, plus le terme d'attache aux données est prépondérant; autrement dit, plus λ est grand, plus le minimiseur de J sera proche des données initiales bruitées g. C'est un bon choix lorsque l'image g est peu bruitée. A contrario, lorsque l'image g est très bruitée, il faut s'en éloigner pour obtenir un résultat satisfaisant, on doit donc accentuer la régularisation. On obtient un tel résultat en choisissant une valeur faible pour λ. Le réglage de ce paramètre est donc indispensable pour obtenir le meilleur résultat possible. En théorie, sa valeur optimale est liée au niveau de bruit dans l'image.

Remarque 1. Pourquoi minimiser l'erreur quadratique?

Lorsque le bruit est supposé indépendant en chaque pixel et suivant la même loi gaussienne de moyenne nulle, minimiser $\|g - v\|_2^2$ revient à trouver v qui maximise la vraisemblance d'observer le bruit $n = g - v$. Supposons que le bruit en chaque pixel $\varepsilon_{i,j} = g_{i,j} - v_{i,j}$ suit la loi normale

$$\varepsilon_{i,j} \sim \mathcal{N}(0, \sigma)$$

de densité de probabilité donnée par la fonction

$$p(t) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{t^2}{2\sigma^2}\right)$$

La vraisemblance de $n_{i,j}$ au vu des observations $g_{i,j}$ est la densité de probabilité associée aux erreurs $\varepsilon_{i,j}$, c'est-à-dire

$$\prod_{i=1}^{m}\prod_{j=1}^{n} p\left(g_{i,j} - v_{i,j}\right) = \prod_{i=1}^{m}\prod_{j=1}^{n}\left(\frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{\left(g_{i,j} - v_{i,j}\right)^2}{2\sigma^2}\right)\right)$$

L'idée est alors de trouver la valeur $v = (v_{i,j})_{1\leqslant i\leqslant m}$ qui maximise cette quantité :

$$\text{Maximiser } v \mapsto \prod_{i=1}^{m}\prod_{j=1}^{n} p\left(g_{i,j} - v_{i,j}\right)$$

Simplifions l'expression de la vraisemblance:

$$\prod_{i=1}^{m}\prod_{j=1}^{n} p\left(g_{i,j}-v_{i,j}\right) = \left(\frac{1}{\sigma\sqrt{2\pi}}\right)^{m\times n} \prod_{i=1}^{m}\prod_{j=1}^{n} \exp\left(-\frac{\left(g_{i,j}-v_{i,j}\right)^2}{2\sigma^2}\right)$$
$$= \left(\frac{1}{\sigma\sqrt{2\pi}}\right)^{m\times n} \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{m}\sum_{j=1}^{n}\left(g_{i,j}-v_{i,j}\right)^2\right)$$

Maximiser la vraisemblance revient donc à considérer le problème :

$$\text{Minimiser } v \mapsto \sum_{i=1}^{m}\sum_{j=1}^{n}\left(g_{i,j}-v_{i,j}\right)^2 = \|g-v\|_2^2$$

2.1 Régularisation de TIKHONOV

Nous allons commencer par un modèle simple le modèle de régularisation de TIKHONOV. Dans ce modèle, on suppose que la différence de valeurs entre deux pixels voisins est faible dans l'image que l'on souhaite restaurer (u). Autrement dit, que les variations d'intensité dans une image sont douces. Dans ce cas, en posant

$$r\left(\nabla^h v\right) = \frac{1}{2}\left\|\nabla^h v\right\|_2^2$$

on définit un terme de régularisation qui prendra une valeur d'autant plus élevée que l'image v présente des variations brutales d'intensité (des pairs de pixels voisins ayant des valeurs très différentes). Ce terme mesure donc bien une distance au modèle choisi. Dans ce modèle, les images v telles que $r\left(\nabla^h v\right)$ soit le plus faible sont les images constantes (qui ne présentent donc aucune variation d'intensité). On se retrouve donc à considérer le problème d'optimisation suivant:

$$\min_{v\in\mathbb{R}^{m\times n}} J_1(v) = \frac{\lambda}{2}\|g-v\|_2^2 + \frac{1}{2}\left\|\nabla^h v\right\|_2^2 \qquad (\mathcal{P}_1)$$

Énonçons quelques propriétés de J_1 qui nous seront utiles pour la suite.

Proposition 2. *Propriétés de J_1*

(a) J_1 est différentiable et $\nabla J_1(v) = \lambda(v-g) - \mathrm{div}^h\left(\delta_x^h v, \delta_y^h v\right)$

(b) J_1 est convexe et fortement convexe de module λ.

(c) ∇J_1 est lipschitzien, de constante de Lipschitz : $L_1 = \lambda + 8$

Le problème est donc fortement convexe, différentiable de gradient lipschitzien. On peut donc appliquer la méthode du gradient à pas fixe pour en calculer une solution.

2.1.1 Simulation numérique

On génère une image bruitée à partir d'une image que l'on estime non bruitée en ajoutant un bruit gaussien et on applique la descente de gradient à pas fixe à l'aide des résultats des propriétés de J_1.

On voit que l'image reconstruite ne semble plus bruitée (elle a perdu le grain que l'on observait dans l'image bruitée). Cependant, elle est loin du résultat espéré. En effet, cette image est floue, alors que l'image initiale est nette. Cela est du au choix de la régularisation : à cause du choix de la norme quadratique pour r, la solution du problème $(\mathcal{P}_1)$ présente un gradient de norme faible, c'est-à-dire des variations douces. Or, des variations douces se traduisent par un effet de flou.

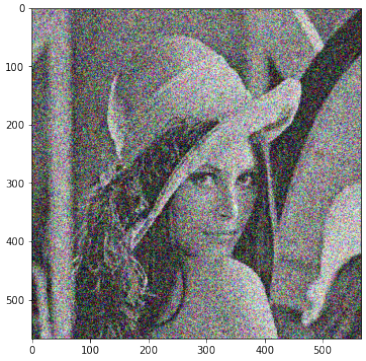

Figure 3: Image Bruitée

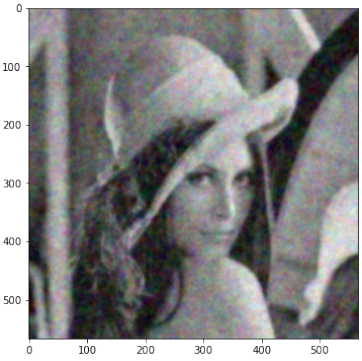

Figure 4: Débruitage par Tikhonov

2.2 Modèle de RUDIN-OSHER-FATEMI

2.2.1 Modèle : régularisation TV

Pour pallier aux défauts de la méthode précédente (à savoir des images débruitées trop floues), une meilleure régularisation consiste à remplacer la norme quadratique par la norme euclidienne :

$$r\left(\nabla^h v\right) = \|\nabla^h v\|_2$$

Ce choix est à rapprocher d'un concept plus vaste, celui de la variation totale: c'est pourquoi on parle de régularisation TV (pour Total Variation). Il est en réalité plus adapté de représenter une image, non comme une matrice u (c'est-à-dire une grille de pixels), mais une fonction $\tilde{u}$ (c'est-à-dire définie sur un domaine continu, un rectangle Ω par exemple). Or, dans ce cadre, une image présente des caractéristiques intéressantes : elle comporte des zones avec des variations très douces (voire pas de variations du tout, autrement dit des applats) délimitées par des bords francs. Or, ce genre de fonctions (on pourra penser par exemple aux fonctions constantes par morceaux) présente une variation totale: $\mathrm{TV}(\tilde{u}) = \int_\Omega |D\tilde{u}|$ faible. Lorsque $\tilde{u}$ est différentiable, sa variation totale vaut la norme de son gradient:

$$\mathrm{TV}(\tilde{u}) = \int_\Omega \|\nabla \tilde{u}(x)\|_2 \, \mathrm{d}x$$

La théorie mathématique générale sur cette quantité est celle de l'espace BV (Bounded Variations).

2.2.2 Problème d'optimisation

On est donc cette fois amené à considérer le problème d'optimisation suivant :

$$\min_{v \in \mathbb{R}^{m \times n}} J_2(v) = \frac{\lambda}{2}\|g - v\|_2^2 + \left\|\nabla^h v\right\|_2 \qquad (\mathcal{P}_2)$$

La fonction objectif J_2 n'est pas différentiable cette fois-ci, à cause du second terme. En particulier, il n'est plus possible d'utiliser la méthode du gradient. Nous allons donc modifier le problème $(\mathcal{P}_2)$ pour en proposer une variante approchée, mais différentiable, de sorte de pouvoir appliquer la méthode du gradient. Malheureusement, une telle manipulation ne préserve pas les minimiseurs, autrement dit, la solution que l'on obtiendra n'est pas assurée d'être proche de la solution du problème initial.

2.2.3 Régularisation de la fonction objectif

L'idée est de remplacer le terme non différentiable par une approximation différentiable. de gradient lipschitzien. Pour cela, on commence par s'intéresser à la fonction suivante :

$$\varphi_\alpha : \begin{cases} \mathbb{R} \to \mathbb{R}^+ \\ t \mapsto |t| - \alpha \ln\left(1 + \dfrac{|t|}{\alpha}\right) \end{cases}$$

Proposition 3. *Propriétés de la fonction φ_α*

Soit $\alpha > 0$.

(a) φ_α est de classe $\mathcal{C}^2$ et $\varphi'_\alpha : \begin{cases} \mathbb{R} \to \mathbb{R}+ \\ t \mapsto \dfrac{t}{\alpha + |t|} \end{cases}$ *et* $\varphi''_\alpha : \begin{cases} \mathbb{R} \to \mathbb{R}^+ \\ t \mapsto \dfrac{\alpha}{(\alpha + |t|)^2} \end{cases}$

(b) φ'_α est lipschitzien, de constante de Lipschitz $L \leqslant \frac{1}{\alpha}$

(c) φ_α est convexe.

La fonction φ_α approche de manière différentiable la valeur absolue, L'approximation étant d'autant meilleure que α est petit. On représente Figure 5 la courbe représentative de la valeur absolue et celle de φ_α pour $\alpha \in \{0.5, 1, 2\}$, sur $[-10; 10]$.

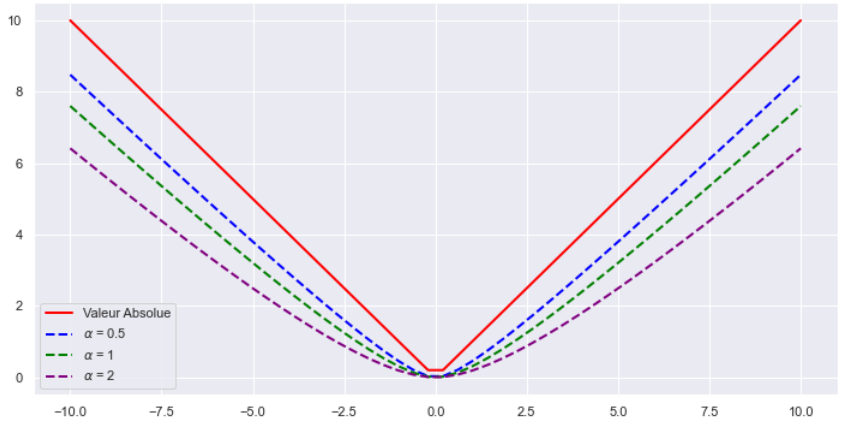

Figure 5: Approximation de la valeur absolue

Dans ce cas, on peut remplacer le second terme de J_2 par une approximation de la norme ℓ_1, et ainsi considérer le problème:

$$\min_{v \in \mathbb{R}^{m \times n}} \bar{J}_2(v) = \frac{\lambda}{2}\|g - v\|_2^2 + \sum_{i=1}^{m}\sum_{i=1}^{n} \varphi_\alpha\left(\left(\delta_x^h v\right)_{i,j}\right) + \varphi_\alpha\left(\left(\delta_y^h v\right)_{i,j}\right) \qquad (\widetilde{\mathcal{P}}_2)$$

2.3 Méthode du gradient

Nous allons utiliser la méthode du gradient pour le problème $(\widetilde{\mathcal{P}}_2)$.

Proposition 4. *Propriétés de la fonction $\bar{J}_2$*
Soit $\alpha > 0$:
(a) $\bar{J}_2$ est fortement convexe.
(b) $\bar{J}_2$ est différentiable de gradient : $\frac{\partial \bar{J}_2}{\partial v} = \lambda(v - g) - \mathrm{div}^{\mathrm{h}}\left(\varphi_{\alpha'}\left(\delta_x^h v\right), \varphi_{\alpha'}\left(\delta_y^h v\right)\right)$.
(c)$\nabla \bar{J}_2$ est lipschitzien

On implémente la méthode du gradient à pas fixe pour résoudre le problème $(\widetilde{\mathcal{P}}_2)$. On choisit $\lambda = 4$, un nombre total d'itérations de 1000 et $\alpha = 0.01$. On initialise avec $v_0 = g$. Dans l'implémentation, on voit que lorsque α est petit (donc l'approximation de la valeur absolue est meilleure), le résultat est plus satisfaisant, mais que la convergence est plus lente. Autrement dit, une meilleure approximation nécessite un temps de calcul plus long.

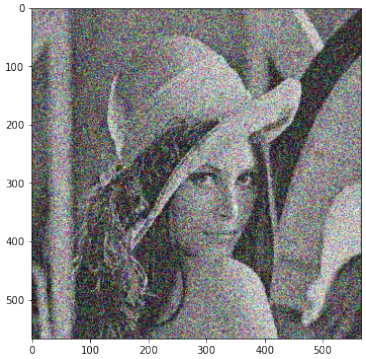

Figure 6: Image Bruitée

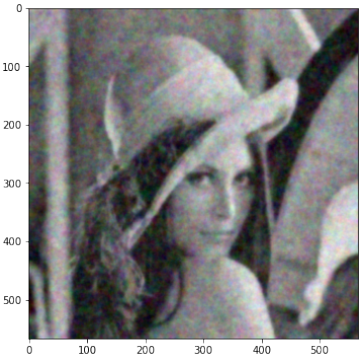

Figure 7: Débruitage par Tikhonov

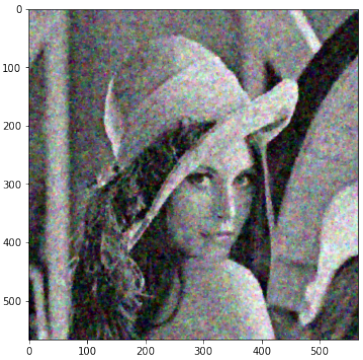

Figure 8: Débruitage par TV

Depuis l'article fondateur de Rudin, Osher et Fatemi, l'utilisation de la régularisation TV est devenue extrêmement populaire dans les problèmes liés à l'image. Parmi les raisons de ce succès, on peut citer la capacité de la TV à prendre en compte les discontinuités de l'image tout en pénalisant les oscillations et le bruit indésirables. Au cours de la dernière décennie, la conception d'algorithmes d'optimisation rapides capables de traiter la non-différentiabilité de la fonctionnelle TV a rendu ce modèle encore plus efficace et attrayant. Le principal inconvénient du modèle TV est l'effet dit d'escalier, c'est-à-dire l'apparition de zones constantes séparées par des frontières. Plusieurs variantes ont alors été proposées pour remédier à ce problème, par exemple des méthodes de lissage de la fonction TV, une autre possibilité consiste à conserver le véritable terme TV mais à changer le cadre du modèle. En exploitant cette idée, la variante TV-LSE considère un cadre bayésien, et remplace l'estimation du maximum a posteriori (MAP) (qui correspond à une interprétation classique du modèle ROF original) par l'estimation de la moyenne à posteriori qui empêche l'apparition des structures en escalier favorisées par le critère d'estimation de la MAP. Le calcul numérique de cette variante de TV-LSE nécessite l'utilisation d'un algorithme Méthode de Monte-Carlo par Chaînes de Markov (MCMC).

3 Modélisation bayésienne

Dans cette section, nous étudions le critère TV dans un cadre différent, afin d'éviter l'effet d'escalier tout en conservant l'efficacité de la mesure TV. Nous étudirons l'interprétation MAP bayésienne du débruitage ROF et introduirons un nouveau filtre de débruitage appelé TV-LSE, défini comme l'estimation de l'image permettant d'atteindre le risque bayésien des moindres carrés. Un algorithme de Metropolis est ensuite proposé pour calculer l'espérance à posteriori requise par ce nouveau filtre. Nous verrons que le débruitage TV-LSE ne souffre pas de l'effet d'escalier et produit des images plus réalistes tout en tout en gardant une bonne efficacité.

3.1 Formulation bayésienne du débruitage TV

Soit $u : \Omega \to \mathbb{R}$ est une image discrète en niveaux de gris définie sur un domaine rectangulaire $\Omega \subset \mathbb{Z}^2$. La variation totale discrète de u est définie par

$$TV(u) = \sum_{(x,y)\in\Omega} |Du(x,y)|$$

où $|Du(x,y)|$ est comme précédemment une approximation discrète de la norme du gradient de u dans (x,y). Nous utiliserons la norme euclidienne habituelle dans $\mathbb{R}^2$ et l'approximation la plus simple possible du vecteur gradient, donnée par:

$$Du(x,y) = \begin{pmatrix} u(x+1,y) - u(x,y) \\ u(x,y+1) - u(x,y) \end{pmatrix}$$

(avec la convention que les différences impliquant des pixels en dehors de Ω sont nulles). Étant donné une image bruitée g. Nous rappelons que la méthode ROF propose de calculer l'unique image u qui minimise:

$$E_\lambda(u) = \|u - g\|^2 + \lambda TV(u)$$

où $\|\cdot\|$ est la norme classique L^2 sur les images et λ est un hyperparamètre qui contrôle le niveau de débruitage. Cette formulation de minimisation d'énergie peut être traduite dans un cadre bayésien. Considérons, pour $\beta > 0$, la distribution de probabilité a priori:

$$p_\beta(u) = \frac{1}{Z_\beta} e^{-\beta TV(u)}$$

$$\text{avec} \quad Z_\beta = \int_{\mathscr{E}_\mu} e^{-\beta TV(u)} du \quad \text{et} \quad \forall \mu \in \mathbb{R}, \quad \mathscr{E}_\mu = \left\{ u \in \mathbb{R}^\Omega, \frac{1}{|\Omega|} \sum_{\mathbf{x}\in\Omega} u(\mathbf{x}) = \mu \right\}$$

La fonction p_β est une fonction de densité de probabilité sur chaque ensemble $\mathscr{E}_\mu$, qui peut être utilisée comme une distribution de probabilité a priori pour estimer la meilleure image débruitée. Si nous supposons que le bruit est un bruit blanc gaussien de moyenne nulle et de variance σ^2, alors, à partir de la formule de Bayes, nous pouvons donner la densité à posteriori:

$$p(u \mid g) = \frac{p(g \mid u)\, p_\beta(u)}{p(g)}$$

Avec :

$$p_\beta(u) = \frac{1}{Z_\beta} e^{-\beta TV(u)} \quad \text{et} \quad p(g \mid u) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-\|u-g\|^2}{2\sigma^2}}$$

Donc:

$$p(u \mid g) = \frac{1}{Z} \exp\left(-\frac{E_\lambda(u)}{2\sigma^2} \right) \tag{1}$$

où $\lambda = 2\beta\sigma^2$ et Z est un facteur de normalisation, dépendant de g et de λ, assurant que $u \mapsto p(u \mid g)$ est une fonction de densité de probabilité sur $\mathbb{R}^\Omega$.

Par conséquent, la formulation variationnelle :

$$\arg\min_u E_\lambda(u)$$

est équivalente à la formulation bayésienne du maximum à posteriori (MAP) bayésienne:

$$\hat{u}_{MAP} = \arg\max_u p(u \mid g)$$

ce qui signifie que le filtre de débruitage ROF sélectionne l'image la plus probable u selon la distribution a posteriori $p(u \mid g)$. Toutefois, comme nous l'avons vu, l'image générée par $\hat{u}_{MAP}$ a tendance à présenter des "zones plates", c'est-à-dire des régions d'intensité uniforme, provoquant le fameux effet d'escalier.

Au lieu de la fonction de risque conduisant à l'estimation MAP, nous pouvons nous intéresser à l'erreur des moindres carrés (LSE), qui consiste à trouver la fonction $\hat{u}(g)$ qui minimise:

$$\mathbb{E}_{u,g}\left[\|u-\hat{u}(g)\|^2\right]=\int_{\mathbb{R}^\Omega}\int_{\mathscr{E}_\mu}\|u-\hat{u}(g)\|^2 p(u,g)\,dgdu$$

Ce minimum est atteint par l'espérance conditionnelle, c'est-à-dire pour :

$$\hat{u}_{LSE}:=\mathbb{E}[u\mid g]=\int_{u\in\mathbb{R}^\Omega} up(u\mid g)\,du$$

Grâce à (1), ceci peut être réécrit :

$$\hat{u}_{LSE}=\frac{\int_{\mathbb{R}^\Omega}\exp\left(-\frac{E_\lambda(u)}{2\sigma^2}\right)\cdot u du}{\int_{\mathbb{R}^\Omega}\exp\left(-\frac{E_\lambda(u)}{2\sigma^2}\right)du} \tag{2}$$

Pour calculer $\hat{u}_{LSE}$, il nous faut donc calculer ce rapport d'intégrales relativement compliqué puisqu'on intègre sur $\mathbb{R}^\Omega$. Nous allons donc utiliser les Méthodes de Monte-Carlo par chaînes de Markov et plus particulièrement l'algorithme de Metropolis-Hastings.

3.2 Algorithme MCMC pour le débruitage TV-LSE

3.2.1 Algorithme TV-LSE

Le débruitage TV-LSE nécessite d'évaluer le quotient d'intégales de l'équation (2). La dimension étant le nombre de pixel, pour éviter le fléau de la grande dimension, il nous faut utiliser les méthodes de Monte-Carlo. Nous proposons ici d'étudier une approche MCMC suivant le schéma de Metropolis afin de simuler une chaîne de Markov et utiliser le théorème ergodique pour les chaînes de Markov.

Commençons par expliciter les lois π et q de l'algorithme de Metropolis-Hastings pour le calcul de $\hat{u}_{LSE}$:

- On veut simuler la chaîne de Markov $(\Upsilon_n)_{n\in\mathbb{N}}$ qui est une séquence d'images définie sur Ω.
- La densité stationnaire π est de la forme: $\pi(u)=\frac{1}{Z}e^{-\frac{\|u-g\|^2+\lambda TV(u)}{2\sigma^2}}$
- La probabilité de transition $q(u,u')$ est de la forme: Soit $\alpha>0$,

$$q(u,u')=\frac{1}{|\Omega|}\sum_{x\in\Omega}\left(\prod_{y\neq x}\delta_{u(y)}\left(u'(y)\right)\right)\frac{1}{2\alpha}\mathbf{1}_{[u(x)-\alpha,u(x)+\alpha]}\left(u'(x)\right)$$

.

Remarque 2. Simuler selon la loi q revient à choisir un pixel $x\sim\mathcal{U}_\Omega$ uniformément sur l'ensemble des pixels et à faire une marche aléatoire uniforme $\mathcal{U}_{[u(x)-\alpha,u(x)+\alpha]}$ sur ce seul pixel. C'est un moyen simple d'avoir q à la fois symétrique et rapide à simuler.

3.3 Algorithme

Les considérations ci-dessus conduisent à l'algorithme suivant.

Algorithm 1 Continuous Hastings-Metropolis algorithm for $\hat{u}_{LSE}$

Let $\pi(u) = \exp\left[-\frac{\|u-g\|^2+\lambda TV(u)}{2\sigma^2}\right]$
Choose a maximal number of iterations n,
Choose a subsampling rate R ($= |\Omega|$ for example)
Draw Υ_0 at random from an initial distribution μ,
Set $k = 0$
Repeat

1. draw $x \sim \mathcal{U}_\Omega$
2. draw $\begin{cases} \Upsilon_{k+1/2}(x) \sim \mathcal{U}_{[\Upsilon_k(x)-\alpha, \Upsilon_k(x)+\alpha]} \\ \Upsilon_{k+1/2}(y) = \Upsilon_k(y) \quad \forall y \neq x \end{cases}$
3. $\begin{cases} \text{if } \pi\left(\Upsilon_{k+1/2}\right) > \pi\left(\Upsilon_k\right) \text{ set } \Upsilon_{k+1} = \Upsilon_{k+1/2} \\ \text{else set } \Upsilon_{k+1} = \begin{cases} \Upsilon_{k+1/2} & \text{with probability } \frac{\pi\left(\Upsilon_{k+1/2}\right)}{\pi(\Upsilon_k)} \\ \Upsilon_k & \text{with probability } 1 - \frac{\pi\left(\Upsilon_{k+1/2}\right)}{\pi(\Upsilon_k)} \end{cases} \end{cases}$
4. $k \leftarrow k+1$

Until $k = Rn+1$

Return $\frac{1}{n}\sum_{i=1}^{n} \Upsilon_{Ri}$

Comme deux images successives Υ_n et Υ_{n+1} diffèrent sur un pixel au plus, l'évolution de la chaîne $(\Upsilon_n)_{n\in\mathbb{N}}$ est très lente. Nous pouvons donc considérer la chaîne sous-échantillonnée par un facteur R, soit $(\Upsilon_{Rn})_{n\in\mathbb{N}}$. Cette chaîne sous-chaîne présente les avantages suivants :

- Lorsque R est grand, la corrélation entre les itérations est beaucoup plus faible et peut permettre une convergence plus rapide. Bien sûr, l'utilisation de la sous-chaîne nécessite le calcul de tous les états intermédiaires $(\Upsilon_{Rn+k})_{1\leqslant k\leqslant R-1}$. Par conséquent, le gain en taux de convergence pratique n'est pas très significatif.
- La moyenne empirique de la sous-chaîne $(\Upsilon_{Rn})_{n\in\mathbb{N}}$ converge vers $\hat{u}_{LSE}$ pour tout $R \in \mathbb{N}^*$ de la même manière que la moyenne empirique de la chaîne complète $(\Upsilon_n)_{n\in\mathbb{N}}$.

En pratique, la chaîne a le même taux de convergence pour une large gamme de paramètres $R > 0$. Nous allons maintenant montrer la consistance de l'algorithme:

Proposition 5. *Soit (Υ_n) la chaîne de Markov construite par l'algorithme 1. Alors pour tout $R \in \mathbb{N}^*$ sa chaîne sous-échantillonnée (Υ_{Rn}) satisfait à*

$$\frac{1}{n}\sum_{k=1}^{n} \Upsilon_{Rk} \underset{n\to\infty}{\longrightarrow} \hat{u}_{LSE} \quad p.s$$

Demonstration 1. On note $U_n = \Upsilon_{Rn}$. Le théorème ergodique sur les chaînes de Markov stipule que si la chaîne de Markov est irréductible, a une distribution stationnaire π, et $h : E \to E$ satisfait à $\sum_{u\in E} |h(u)|\pi(u) < \infty$, alors

$$\frac{1}{n}\sum_{k=1}^{n} h\left(U_k\right) \underset{n\to+\infty}{\longrightarrow} \int_E h(u)d\pi(u) \text{ p.s. et dans } L^1$$

- La chaîne (U_n) est irréductible car si u et u' sont deux images de E, alors $\mathbb{P}\left(U_n = u' \mid U_0 = u\right)$ est positif pour tout $n \geqslant \frac{\|u'-u\|_\infty}{\alpha}$, de sorte que u et u' communiquent.

- Écrivons $\pi(u) = \frac{1}{Z}\exp\left(-\frac{E_\lambda(u)}{2\sigma^2}\right)$. Pour prouver que π est stationnaire pour la chaîne sous-échantillonnée (U_n), il suffit de prouver qu'elle est réversible (donc stationnaire) pour (Υ_n) . Le noyau de transition P de Υ_n peut être écrit :

$$P(u,u') = q\left(u,u'\right) e^{-\frac{\left(E_\lambda(u')-E_\lambda(u)\right)_+}{2\sigma^2}}$$

où $(x)_+ = \max(0,x)$ est la partie positive de x, et partie positive de x, et:

$$q\left(u,u'\right) = \frac{1}{|\Omega|}\sum_{x\in\Omega}\left(\prod_{y\neq x}\delta_{u(y)}\left(u'(y)\right)\right)\frac{1}{2\alpha}1_{[u(x)-\alpha,u(x)+\alpha]}\left(u'(x)\right)$$

est la probabilité de transition de l'algorithme de Métropolis.

Si $\pi(u) \geqslant \pi\left(u'\right)$, alors $\left(E_\lambda\left(u'\right) - E_\lambda(u)\right)_+$ est nul et de plus, on a :

$$\pi(u)P(u,u') = \pi(u)q\left(u,u'\right)$$

Mais q est symétrique et $\pi\left(u'\right) \leqslant \pi(u)$ de sorte que :

$$\pi(u)P(u,u') = \pi(u)q\left(u',u\right) = \pi\left(u'\right)P(u',u)$$

Par conséquent, π est réversible par rapport à P, donc stationnaire pour (Υ_n).

- On conclut en appliquant le théorème ergodique à la fonction $h = Id_E$, qui est π -intégrable comme requis.

3.4 Simulations numériques

On implémente l'algorithme avec $\alpha = 0.6$, $N = 100000$ et $R = 10000$

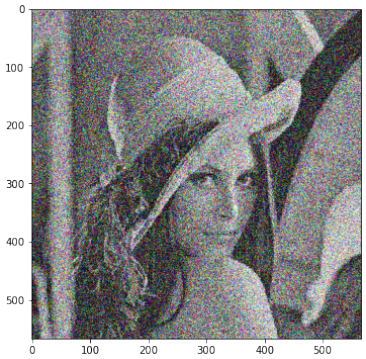

Figure 9: Image Bruitée

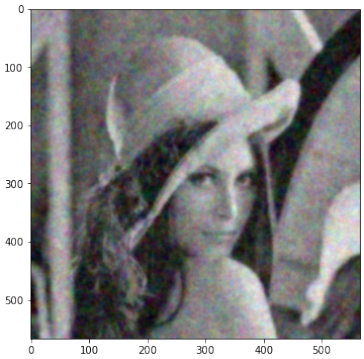

Figure 10: Débruitage Tikhonov

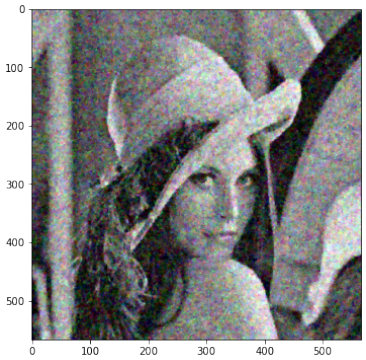

Figure 11: Débruitage par TV

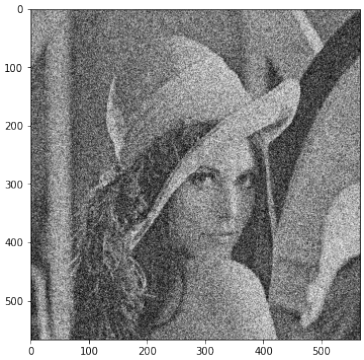

Figure 12: Débruitage par TV-LSE

On a donc procédé à n itérations en choisissant aléatoirement un pixel $x \in \Omega$ puis en faisant subir à x une expérience aléatoire en fonction de l'image à l'itération $n-1$. On a ensuite mis à jour ou non l'image selon le résultat de cette expérience aléatoire. Le problème est qu'il faut balayer l'ensemble des pixels x un grand nombre de fois pour espérer un résultat satisfaisant et que cela prend du temps (beaucoup de temps...). Nous pourrions penser à balayer l'ensemble des pixels ligne par ligne plutôt que de réaliser des tirages aléatoires, pour être sûr que tous les pixels ont été soumis à des mises à jour. L'algorithme s'arrêterait alors après un grand nombre n d'itérations. Toutefois ce processus semble long et les résultats peu prometteurs car depuis le début de notre périple nous avons omis une idée importante. Afin de construire une image interprétable, la valeur attribuée à un pixel doit dépendre des valeurs des pixels voisins. En effet, il est peu probable qu'une réalisation consituée de pixels tous très différents puissent représenter une image interprétable. Ce raisonnement simple mais puissant, qui n'est pour l'instant qu'intuitif, nous rappelle la propriété de Markov. Nous sommes donc amener à changer de cap et à construire un lien entre images et chaînes de Markov.

4 Modélisation Markovienne

Nous souhaitons construire cette fois un modèle probabiliste qui va affecter à chaque pixel une valeur selon son environnement. Ainsi, l'image dont nous disposons va être considérée comme une réalisation d'un champ aléatoire. Soit s un pixel de l'image, on peut lui associer une variable aléatoire X_s prenant ses valeurs dans $[0,1]$. Le niveau de gris x_s en s est ainsi une réalisation de la variable aléatoire X_s. On définit alors le champ aléatoire $X = (X_s)_{s\in\Omega}$ prenant ses valeurs dans $[0,1]^{|\Omega|}$.

Dans ce cadre probabiliste, l'image considérée est simplement une réalisation x du champ. La probabilité globale de x, $P(X = x)$, permet d'accéder à la vraisemblance de l'image, et les probabilités conditionnelles locales d'une valeur en un site permettent de mesurer le lien statistique entre un niveau de gris et le reste de l'image. Cette modélisation conduit à introduire la notion de champ de Markov où la valeur attribuée à un pixel dépendra des valeurs des pixels voisins. Pour ce faire, on munit Ω d'un système de voisinage V défini de la façon suivante :

$$\forall s \in \Omega, \quad \mathcal{V}_s = \{t\} \text{ tels que } \begin{cases} s \notin \mathcal{V}_s \\ t \in \mathcal{V}_s \Rightarrow s \in \mathcal{V}_t \end{cases}$$

À partir d'un système de voisinage, un système de cliques peut être déduit : une clique est soit un singleton de Ω, soit un ensemble de pixels tous voisins les uns des autres. En fonction du système de voisinage utilisé, le système de cliques sera différent et fera intervenir un nombre plus ou moins grand de pixels. On notera $\mathcal{C}$ l'ensemble des cliques relatif à un voisinage V et $\mathcal{C}_k$ l'ensemble des cliques de cardinal k.

Les interactions locales entre niveaux de gris de sites voisins peuvent alors s'exprimer comme un potentiel de clique. Soit c une clique, on lui associe le potentiel U_c dont la valeur dépend des niveaux de gris des pixels constituant la clique. En poursuivant ce raisonnement, on peut définir l'énergie globale de l'image comme la somme des potentiels de toutes les cliques:

$$U = \sum_{c\in\mathcal{C}} U_c$$

et l'énergie locale en un pixel s comme la somme des potentiels de toutes les cliques auxquelles il appartient:

$$U_s = \sum_{c\in\mathcal{C}/s\in c} U_c$$

Ainsi, le potentiel du pixel s noté U_s peut être déterminé selon les potentiels des autres pixels. Il en découle que la probabilité que le pixel s prenne une certaine valeur dépend des potentiels de ses voisins tout comme la couleur du pixel dépend des couleurs voisines. On peut assimiler la couleur du pixel à une énergie, fonction de son potentiel. De plus, ce sont les voisins du pixel s (plutôt que les pixels lointains) qui joueront un rôle important sur sa valeur. Nous allons donc considérer une image comme un champ de Markov. De façon intuitive, X est un champ de Markov si et seulement si la probabilité conditionnelle locale en un pixel n'est fonction que de la configuration du voisinage du pixel considéré. Donnons maintenant une définition formelle :

Définition 3. *Considérons x_s le niveau de gris de l'image prise au pixel s et $x^s = (x_t)_{t\neq s}$ la configuration de l'image excepté le pixel s. X est un champ de Markov ssi:*

$$P\left(X_s = x_s | x^s\right) = P\left(X_s = x_s | x_t, t \in \mathcal{V}_s\right)$$

Ainsi, le niveau de gris en un pixel ne dépend que des niveaux de gris des pixels voisins de ce pixel.

Nous allons nous placer sous des hypothèses markoviennes, où l'image totale x a une énergie $U(x)$ et une probabilité de réalisation $P(X = x)$.

4.1 Étude d'un champ de Markov

Pour ce modèle, nous donnons au pixel i une couleur dépendant des couleurs de ses voisins. On peut donc écrire que la probabilité pour le pixel i d'être d'une certaine couleur, c'est-à-dire la probabilité pour la variable aléatoire X_i de prendre une valeur x_i, dépend de l'énergie de ses voisins. Le théorème de Hammersley-Clifford (1974), nous permet une expression de cette probabilité. Ce théorème met en jeu une nouvelle notion, celle de champ de Gibbs. Pour la définir, commencçons par introduire la notion de mesure de Gibbs.

Définition 4. *La mesure de Gibbs de fonction d'énergie $U : [0,1]^{|\Omega|} \to \mathbb{R}$ est la probabilité P définie sur $[0,1]^{|\Omega|}$ par:*

$$P(X = x) = \frac{1}{Z} \exp(-U(x))$$

avec:

- $U(x) = \sum_{c \in \mathcal{C}} U_c(x)$
- *$\mathcal{C}$ est le système de cliques associé à U*
- $Z = \sum_{x \in [0,1]^{|\Omega|}} \exp(-U(x))$ *est une constante de normalisation appelée fonction de partition de Gibbs.*

Nous pouvons maintenant donner la définition d'un champ de Gibbs:

Définition 5. *Un champ de Gibbs est un champ aléatoire X dont la probabilité de réalisation est donnée par la mesure de Gibbs :*

$$P(X = x) = \frac{1}{Z} \exp(-U(x)) = \frac{1}{Z} \exp\left(- \sum_{c \in \mathcal{C}} U_c(x) \right)$$

Nous allons maintenant énnoncer le théorème fondamental qui va nous permettre de relier la notion de champ de Gibbs et celle de champ de Markov afin de construire notre raisonnement:

Théorèmé 1. *Théorème de Hammersley-Clifford*
Sous les hypothèses:

- *S fini ou dénombrable.*
- *Le système de voisinage $\mathcal{V}$ est borné.*
- *L'espace des états E est discret.*

Alors:

$$\mathbf{X} \text{ est un champ de Markov relativement à } \mathcal{V} \text{ et } P(X = x) > 0 \quad \forall x \in \Omega$$

$$\Longleftrightarrow$$

$$\mathbf{X} \text{ est un champ de Gibbs de potentiel associé à } \mathcal{V}.$$

Remarque 3. Par conséquent une image modélisée comme un champ de Markov peut être vue comme un champ de Gibbs ce qui nous donne tout de suite accès à la probabilité $P(X = x)$. Toutefois pour calculer ces probabilités, il faut développer des calculs ne faisant pas intervenir ce terme Z et pour cela nous allons utiliser les probabilités conditionnelles. On note également que l'expression du champ de Gibbs nous rappelle étrangement l'expression de loi à priori que nous avions choisi dans la modélisation bayésienne.

4.2 Préparation en vue de l'algorithme

Nous allons établir des résultats sur les probabilités de réalisation des images afin de construire nos algorithmes de débruitage. Comme vu plus tôt, le calcul de la probabilité $P(X = x)$ se révèle impossible à réaliser du fait du calcul de Z. Pour s'affranchir de ce terme, nous allons utiliser les probabilités conditionnelles. En effet, une probabilité conditionnelle fait intervenir un quotient de 2 probabilités, il est donc possible de simplifier les termes Z à condition de bien choisir les probabilités en question.

Le calcul qui nous intéresse dans cette étude est celui de la probabilité pour un pixel de prendre une valeur particulière connaissant les valeurs de ses voisins. Par exemple, si tous ses voisins sont noirs, le pixel doit présenter une probabilité importante d'être également noir. On s'intéresse donc à la probabilité que le pixel X_i prenne la valeur x_i connaissant son voisinage, probabilité qu'on écrira $P\left(X_i = x_i \mid V_i\right)$. Par définition d'un champ markovien, cette probabilité est la même que la probabilité que le pixel X_i prenne la valeur x_i connaissant tout le reste de l'image, probabilité que l'on notera $P\left(X_i = x_i \mid X^i = x^i\right)$ avec x^i le vecteur contenant l'ensemble des pixels sauf le i -ème.

L'intersection des événements $(X_i = x_i)$ et $\left(X^i = x^i\right)$ permet de définir une réalisation de l'image $X = x$:

$$P\left(X_i = x_i \mid X^i = x^i\right) = \frac{P(X = x)}{P\left(X^i = x^i\right)}$$

L'événement $X^i = x^i$ peut se réécrire comme $\bigcup_{a_i \in A_i} \{X = y_i\}$ où y_i est le champ tel que le pixel i vaut a_i $\left(y_i[i] = a_i\right)$, les autres pixels étant fixés par x^i $\left(y_i[j] = x^i[j] \text{ pour } j \neq i\right)$, et où A_i est l'ensemble des valeurs pouvant être prises par le pixel i. On a ici une réunion disjointe d'événements indépendants :

$$P\left(X^i = x^i\right) = \sum_{a_i \in A_i} P\left(X = y_i\right) = \frac{\sum_{a_i \in A_i} \exp\left(-U\left(y_i\right)\right)}{Z}$$

et,

$$P(X = x) = \frac{1}{Z} \exp(-U(x))$$

donc :

$$P\left(X_i = x_i \mid V_i\right) = \frac{\exp(-U(x))}{\sum_{a_i \in A_i} \exp\left(-U\left(y_i\right)\right)}$$

Nous allons simplifier cette expression afin de pouvoir l'implémenter dans un algorithme. Définissons d'abord l'énergie locale d'un site x_j. Celle-ci correspond à la somme des énergies de chacune des cliques auxquelles x_j appartient, c'est-à-dire :

$$U_j\left(x_j \mid V_j\right) = \sum_{c \in C / j \in c} U_c\left(\left(x_{i_1}, \ldots, x_{i_l}\right), i_k \in c\right)$$

On rappelle de plus que :

$$U(x) = \sum_{c \in C} U_c\left(\left(x_{j_1}, \ldots, x_{j_l}\right), j_k \in c, x_{j_k} \in x\right)$$

Après avoir fixé un site i, on peut diviser l'ensemble des cliques en 2 sous-ensembles : celles qui contiennent le site x_i, et celles qui ne le contiennent pas :

$$U(x) = \sum_{c \in C / i \in c} U_c\left(\left(x_{j_1}, \ldots, x_{j_l}\right), x_{j_k} \in x, j_k \in c\right) + \sum_{c \in C / i \notin c} U_c\left(\left(x_{j_1}, \ldots, x_{j_l}\right), x_{j_k} \in x, j_k \in c\right)$$

On peut donc réécrire U sous la forme :

$$U(x) = \sum_{c \in C / i \notin c} U_c\left(\left(x_{j_1}, \ldots, x_{j_l}\right), x_{j_k} \in x, j_k \in c\right) + U_i\left(x_i \mid V_i, x_i = x[i]\right)$$

Ce résultat permet de simplifier l'expression de la probabilité calculée précédemment:

$$\sum_{a_i \in A_i} \exp\left(-U\left(y_i\right)\right) = \sum_{a_i \in A_i} \exp\left(-\left(\sum_{c \in C / i \notin c} U_c\left(\left(x_{j_1}, \ldots, x_{j_l}\right), x_{j_k} \in y_i, j_k \in c\right) + U_i\left(a_i \mid V_i\right)\right)\right)$$

Le terme $\sum_{c\in C/i\notin c} U_c\left((x_{j_1},\ldots,x_{j_l}),x_{j_k}\in y_i,j_k\in c\right)$ ne dépend pas de la valeur a_i choisi pour le pixel i, par définition (on s'intéresse uniquement à des x_j pris dans des cliques qui ne contiennent pas le pixel i). On a donc:

$$\sum_{a_i\in A_i}\exp\left(-U\left(y_i\right)\right)=\exp\left(-\sum_{c\in C/i\notin c}U_c\left(\left(x_{j_1},\ldots,x_{j_l}\right),x_{j_k}\in y_i,j_k\in c\right)\right)\cdot\sum_{a_i\in A_i}\exp\left(-U_i\left(a_i\mid V_i\right)\right)$$

Finalement, on obtient le résultat:

$$P\left(X_i=x_i\mid X^i=x^i\right)=\frac{\exp\left(-U_i\left(x_i\mid V_i\right)\right)}{\sum_{a_i\in A_i}\exp\left(-U_i\left(a_i\mid V_i\right)\right)}$$

Cette expression est intéressante puisque son calcul ne nécessite pas le calcul de la constante de normalisation Z. Pour pouvoir l'appliquer, il suffit de connaître l'expression de l'énergie, ainsi que l'état des voisins du pixel i (on est bien dans le cadre d'un champ markovien). Ainsi, il est possible de choisir une énergie U qui favorise certaines valeurs x_i en fonction des valeurs des pixels voisin. Le modèle de Potts semble répondre à nos attentes.

4.3 Le modèle de Potts

Il s'agit de la généralisation du modèle d'Ising, adaptée à un ensemble E de cardinal N, comme $E=[\![0,255]\!]/255$. La différence principale avec le modèle d'Ising est que seuls les potentiels liés aux cliques d'ordre 2 sont définis. Il n'y a pas de terme d'énergie lié aux cliques d'ordre 1, correspondant à un champ magnétique externe. L'énergie de ce modèle est:

$$U(x)=\beta\sum_{c=(s,t)\in C}\left(\mathbf{1}_{\{x_s\neq x_t\}}-\mathbf{1}_{\{x_s=x_t\}}\right)$$

Un tel modèle tend à créer des zones homogènes de taille d'autant plus grande que β est grand. Après le choix d'un potentiel, il faut procéder à la minimisation de son énergie totale. On procède par tirage de configurations. L'idée générale est de tirer pour chaque pixel une valeur aléatoire et de lui attribuer cette valeur si elle permet la diminution de l'énergie totale. Les algorithmes les plus utilisés pour réaliser ces tirages, l'échantillonneur de Gibbs et l'algorithme de Metropolis, fonctionnent de façon similaire. Nous allons présenter ici la méthode de l'échantillonnage de Gibbs.

4.4 L'échantillonneur de Gibbs

Pour chacune des n itérations de l'algorithme, on balaye l'ensemble des pixels.

Pour chaque pixel s :

1. Calcul de la probabilité locale, connaissant la configuration des voisins V_s pour l'image à l'itération $n-1$:
$$\mathbb{P}\left(X_s=x_s\mid V_s\right)=\frac{\exp\left(-U_s\left(x_s\mid V_s\right)\right)}{\sum_{a_s\in A_s}\exp\left(-U_s\left(a_s\mid V_s\right)\right)}$$
2. Mise à jour du pixels par tirage aléatoire selon la loi $\mathbb{P}\left(X_s=x_s\mid V_s\right)$.

Après avoir implémenté un algorithme de tirage de configuration, il faut implémenter un algorithme convergeant vers une image solution du problème de minimisation de l'énergie totale pour cela, nous allons utiliser l'algorithme du recuit simulé.

4.5 Recuit simulé

Le recuit simulé est une méthode classique de minimisation d'énergie fréquemment utilisée en physique. En traitement d'image, l'algorithme consiste en n itérations au cours desquelles on réalise des tirages de configuration (ici selon l'échantillonneur de Gibbs). Cependant, ces tirages ne dépendent plus que de l'énergie de configuration, mais aussi d'une quantité $T^{(n)}$ qui mesure le degré d'aléatoire introduit dans ces tirages, qu'on nomme température et qui décroît à chaque itération. Partant d'une température $T^{(0)}$ assez grande et de l'image à traiter, l'algorithme est le suivant:

Pour chaque itération n,

1. Tirage d'une configuration en remplaçant les énergies $U(x)$ par les quantités $U(x)/T^{(n)}$ pour les tirages de mise à jour des pixels.
2. Diminution de la température selon une décroissance logarithmique. La décroissance logarithmique est nécessaire pour obtenir la convergence en probabilité de l'algorithme vers l'image qui minimise l'énergie. En pratique, cette décroissance est trop lente et on préfère utiliser une décroissance linéaire ou quadratique, ce qui peut provoquer une convergence vers un minimum seulement local de l'énergie. Cependant, dans notre étude avoir un minimum local et un coup de calcul faible est plus interessant qu'un minimum global et un coup de calcul élevé.

4.6 Simulations numériques

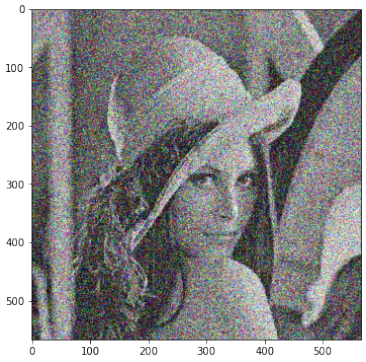

Figure 13: Image Bruitée

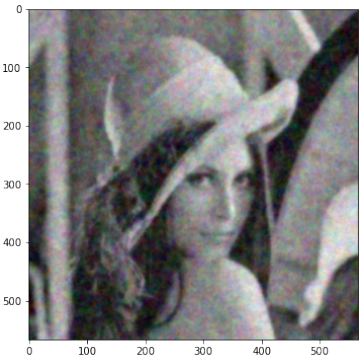

Figure 14: Débruitage Tikhonov

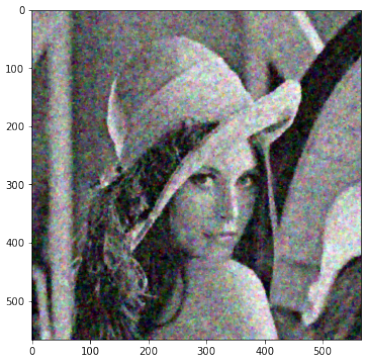

Figure 15: Débruitage par TV

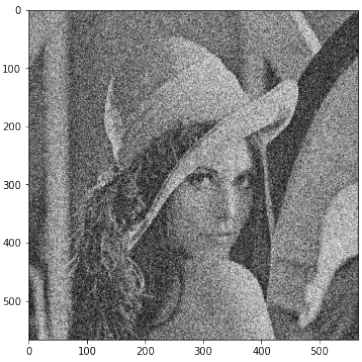

Figure 16: Débruitage par TV-LSE

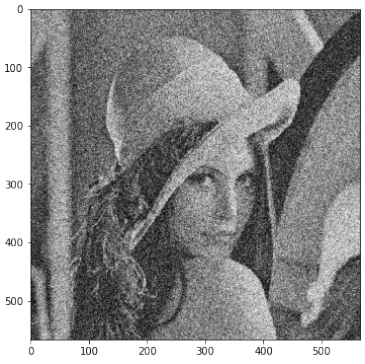

Figure 17: Débruitage Markovien

On teste nos algorithmes avec une image qui en contient que deux nuance de gris:

Figure 18: Image Bruitée

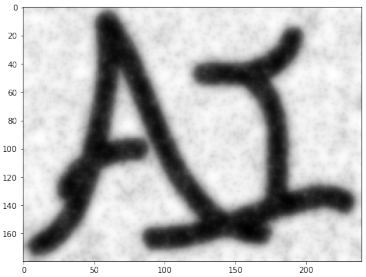

Figure 19: Débruitage Tikhonov

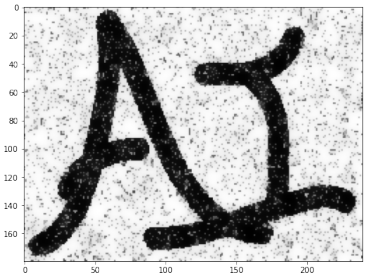

Figure 20: Débruitage par TV

Figure 21: Débruitage par TV-LSE

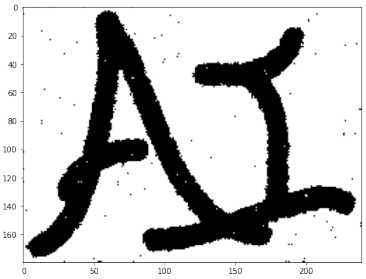

Figure 22: Débruitage Recuit

Pour une image avec plusieurs niveaux de gris, l'algorithme nous donne des résultats mauvais et est très lent puisqu'il faut calculer les probabilités des 255 nuances de gris possibles. Toutefois on voit que pour une image avec deux nuances de gris, il est très performant mais qu'il reste lent. Une idée naturelle qui nous vient à l'esprit et de se demander si, parmi plusieurs nuances de gris, il est possible, plutôt de qu'échantilloner avec l'échantillonneur de Gibbs, de simplement choisir la nuance qui à la plus grande probabilité selon ses voisins. C'est l'idée derrière l'Iterated Conditional Mode (ICM).

4.7 Iterated Conditional Mode (ICM)

La méthode Iterated Conditional Mode (ICM) consiste à tester toutes les nuances de gris pour chaque pixel et à mettre à jour avec la configuration qui permet la diminution d'énergie la plus importante. Même si l'on teste à chaque itération toutes les nuances de gris, l'absence du caractère probabiliste permet à l'ICM d'être beaucoup plus rapide. En revanche, on rien ne nous dit que l'on converge vers le minimum global, ce qui encore une fois n'est pas un problème tant que l'image de fin est de bonne qualité.

L'algorithme ICM est le suivant :
On balaye l'ensemble des pixels et pour chaque pixel s :

1. Calcul des probabilités conditionnelles locales, pour toutes les valeurs possibles de λ dans E du site :

$$\mathbb{P}\left(X_s = \lambda \mid V_s\right)$$

2. Mise à jour de la valeur par le λ maximisant la probabilité conditionnelle locale :

$$x_s(k+1) = \text{Argmax}_\lambda \, P\left(X_s = \lambda \mid V_s\right)$$

Remarque 4. On remarque l'algorithme ICM fonctionne exactement comme l'échantillonneur de Gibbs avec le recuit simulé mais sans le tirage aléatoire. On fixe la température à 1 et pour chaque pixel, on choisit directement le niveau de gris qui maximise la probabilité.

4.8 Simulations numériques

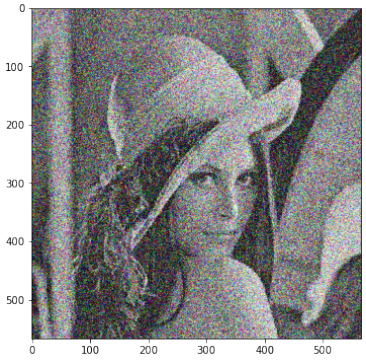

Figure 23: Image Bruitée

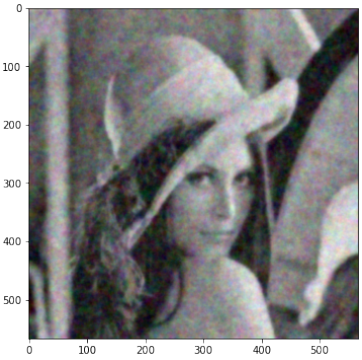

Figure 24: Débruitage Tikhonov

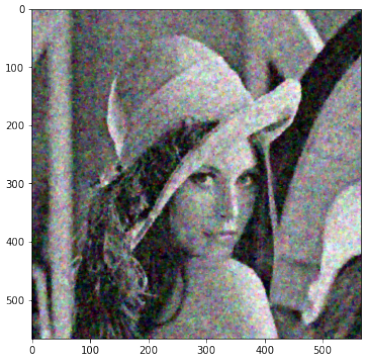

Figure 25: Débruitage par TV

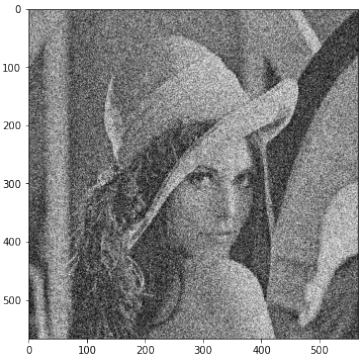

Figure 26: Débruitage par TV-LSE

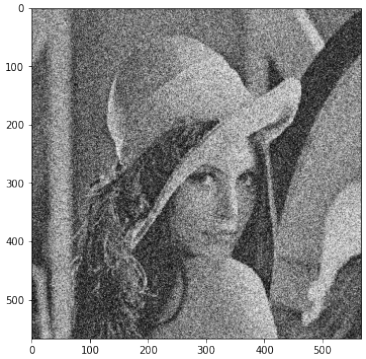

Figure 27: Débruitage Markovien

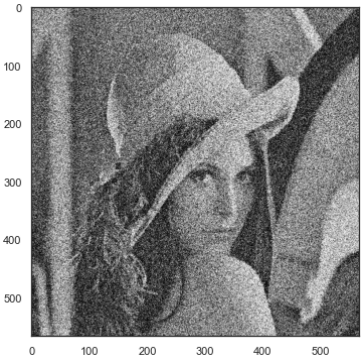

Figure 28: Débruitage ICM

Figure 29: Image Bruitée

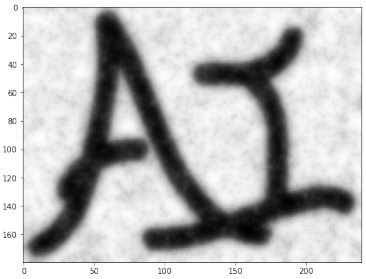

Figure 30: Débruitage Tikhonov

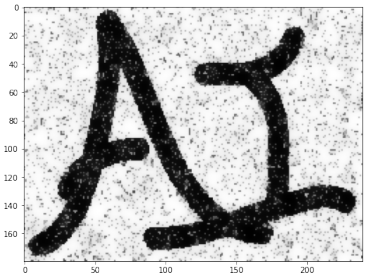

Figure 31: Débruitage par TV

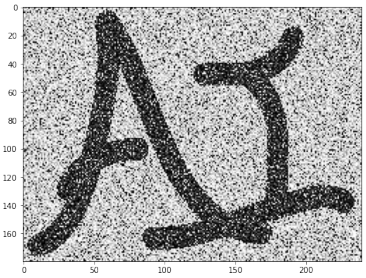

Figure 32: Débruitage par TV-LSE

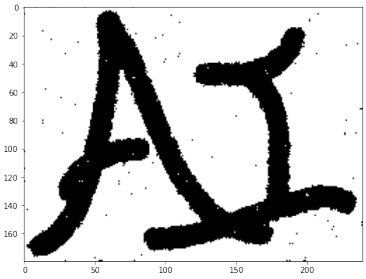

Figure 33: Débruitage Recuit

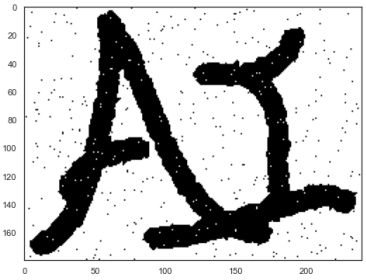

Figure 34: Débruitage ICM

La méthode ICM donne de très bons résultats avec une image en deux niveaux de gris et est bien plus rapide que le recuit simulé. Par contre avec des images en plusieurs niveau de gris l'algorithme n'arrive pas à correctement débruiter. Cela est dû à la structure trop complexe de l'image.

5 Conclusion

Pour donner un indice de qualité à nos résultats, on peut utiliser le PSNR : Peak Signal-to-Noise Ratio. Le PSNR est défini par l'erreur quadratique moyenne (MSE). Étant donné une image sans bruit I de taille $m \times n$ et notre resultat K, la MSE est définie comme suit :

$$MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - K(i,j)]^2$$

Le PSNR est alors défini comme suit pour des images codées en 8 bits :

$$PSNR = 10 \cdot \log_{10} \left(\frac{255^2}{MSE} \right) = 20 \cdot \log_{10} (255) - 10 \cdot \log_{10}(MSE)$$

Plus le PSNR est grand et plus notre image est de bonne qualité.

Algorithme	Paramètres	PNSR Lena	PNSR AI
Image bruitée	$\sigma = 1$	5.87	11.76
Tikonov	$\tau = 0.1$, $\epsilon = 0.001$, $N = 1000$	13.37	15.99
TV	$\tau = 0.003$, $\epsilon = 0.001$, $N = 1000$, $\alpha = 0.01$, $\lambda = 4$	13.99	18.8
TV-LSE	$N = 10000$, $\alpha = 0.5$, $R = 1000$, $\lambda = 8$	5.84	11.60
Gibbs + Recuit simulé	$T = 1/log(n)$, $T(n) \geqslant 9.5$	5.88	16.30
ICM	$N = 100000$	5.88	17.24

Débruiter une image est une étape importante dans plusieurs domaines avancés comme la médecine ou l'imagerie satellite. Nous avons présenté plusieurs méthodes de débruitage d'images en noir et blanc reposant sur l'optimisation, les statistiques bayésiennes et la théorie des champs de Markov. Notre étude nous a permis d'interpréter le débruitage d'une image comme un problème d'optimisation classique puis de voir les choses sous une autre perspective en formulant le même problème sous la forme d'un problème bayésien de maximisation de la probabilité à postériori. À partir de là nous avons pu nous intéresser un autre critère : celui de la minimisation de l'erreur des moindres carrés dont nous avons approché la solution avec l'algorithme de Métropolis-Hasting. Enfin l'idée que dans une image cohérente, un pixel ressemblait à ses voisins nous a permis de nous a tourner vers les champs de Markov et d'utiliser l'échantillonnage de Gibbs et le recuit simulé pour fabriquer de nouveaux algorithmes. Même si parfois les résultats n'ont pas été formidables (TV-LSE), nous avons pu utiliser des notions très différentes pour résoudre le même problème, ce qui à mon sens, est plus intéressant que les résultats en eux-mêmes.

References

E. Angelini, I. Bloch, S. Ladjal, M. Sigelle, and F. Tupin. Cours ima 203 polycopié. 2015.

J. Delon. Modèles stochastiques pour l'analyse d'image. 2020.

B. Delyon. Simulation et modélisation. 2020.

A. Guyader. Optimisation stochastique. 2021.

P. Héas. Méthodes de simulation de monte-carlo en analyse d'images. 2015.

C. Louchet. Modèles variationnels et bayésiens pour le débruitage d'images : de la variation totale vers les moyennes non-locales. 2008.

C. Louchet and L. Moisan. Total variation denoising using posterior expectation. In *European Signal Processing Conference Lausanne, Switzerland*, 2008.

P. Tan. Méthodes du premier ordre pour l'optimisation non convexe et non lisse. 2020.

G. Winkler. Image analysis, random fields, and dynamic monte carlo methods: A mathematical introduction. 1995.