Image, Signal, Simulations

Cours de la formation
Master MApI3
Mathématiques Appliquées pour l'Ingénierie, l'Industrie et l'Innovation
4ième année – M 1

Université Paul Sabatier

Auteur : F. Malgouyres
Francois.Malgouyres@math.univ-toulouse.fr

Table des matières

Chapitre 1

La création de l'image numérique

La création d'une image numérique est faite par un appareil de mesure (scanner, appareil photo numérique, webcam, barrette CCD, ...). Malgré la diversité des appareils de mesure, elle s'écrit (à quelques approximations près) sous la forme d'une unique équation mathématique. Ce sont les éléments mathématiques utiles à l'écriture et à la compréhension de cette équation que nous allons introduire ici. Commençons par définir ce qu'est une image numérique.

1.1 Images numériques

Une image numérique est définie sur une grille à deux dimensions. Les éléments de cette grille sont appelés des *pixels*. Ainsi, une image est définie sur un ensemble

$$\{1,\dots,M\}\times\{1,\dots,N\},$$

où M et N sont des entiers strictement positifs.

De plus, à chaque pixel, l'image attribue une couleur. Il existe plusieurs façons de représenter une couleur (système RVB, HSV, niveaux de gris, ...). Le point commun entre ces méthodes est de représenter une couleur par un ou plusieurs nombres (généralement trois). Chacun d'entre eux représentant la composante de notre couleur dans la direction d'une couleur primaire de référence.

Afin de simplifier les notations dans la suite du cours, nous ne travaillerons que sur des images noir et blanc. Ceci induit une simplification notable, puisqu'une couleur, que l'on appellera maintenant *niveau de gris*, n'est plus représentée que par un seul nombre. La convention habituelle veut que la valeur 0 corresponde au **noir** et que la valeur 255 corresponde au **blanc**. Les valeurs intermédiaires donnent les différentes teintes de gris. Il nous faut par ailleurs coder ces niveaux de gris. Pour ce faire, on doit utiliser un nombre fini de niveaux de gris. On a ainsi un ensemble fini $\mathcal{C}ol \subset \mathbb{R}$ représentant nos couleurs. Par exemple, pour des niveaux de gris codés sur 8 bits, on a $\mathcal{C}ol = \{0,1,\dots,255\}$.

Ainsi, une image numérique est donnée par une suite

$$\begin{array}{rcl} \{1,\dots,M\}\times\{1,\dots,N\} & \longrightarrow & \mathcal{C}ol \\ (m,n) & \longrightarrow & u_{m,n} \end{array}$$

Dans la suite, nous utiliserons indifféremment les termes *image*, *fonction* et *suite* pour désigner une image numérique. Nous préciserons qu'une image ou une fonction n'est pas numérique en la qualifiant d'*analogique*. De plus, nous supposerons que nos images sont carrées. On a ainsi $M = N$.

1.2 La convolution

Tous les appareils de mesure commencent par faire un "moyennage", sur un voisinage d'un pixel, avant d'attribuer cette valeur au pixel. Ce moyennage ne dépend pas (on supposera en tout cas que cette dépendance, si elle existe, est négligeable) du pixel considéré. Mathématiquement, cette opération est connue sous le nom de convolution. C'est l'objet de ce chapitre.

1.2.1 De fonctions analogiques

Pour simplifier, nous ne considérerons que des fonctions, dont le module est *intégrable* [1], définies sur $\mathbb{R}^2$. Ces fonctions représentent des images analogiques. On notera $L^1(\mathbb{R}^2)$ l'ensemble des fonctions de module intégrable sur $\mathbb{R}^2$.

Pour deux fonctions h et v dans $L^1(\mathbb{R}^2)$, on note $h*v$ le *produit de convolution de h par v* et on le définit par

$$\forall (x,y) \in \mathbb{R}^2, h*v(x,y) = \int_{\mathbb{R}} \int_{\mathbb{R}} h(x-x', y-y') v(x',y') dx' dy'.$$

Il s'agit donc d'une fonction (il n'est pas très difficile de voir qu'elle est aussi dans $L^1(\mathbb{R}^2)$).

L'intuition qu'il faut avoir de $h*v(x,y)$ est bien celle d'un "moyennage" de v au voisinage de (x,y). Par exemple, si l'on prend $h(x,y) = \mathbf{1}_{]-\frac{1}{2},\frac{1}{2}]^2}(x,y)$, on a, pour tout $(x,y) \in \mathbb{R}^2$,

$$\begin{aligned} h*v(x,y) &= \int_{\mathbb{R}} \int_{\mathbb{R}} h(x-x', y-y') v(x',y') dx' dy'. \\ &= \int_{y-\frac{1}{2}}^{y+\frac{1}{2}} \int_{x-\frac{1}{2}}^{x+\frac{1}{2}} v(x',y') dx' dy'. \end{aligned} \tag{1.1}$$

qui est bien la moyenne de v sur un carré de côté 1, centré en (x,y). Modifier le support de h revient à changer le support sur lequel on fait la moyenne et modifier les valeurs de h revient à ajouter une pondération (certains points du voisinage de (x,y) comptant plus que d'autres). Il est important de noter que le moyennage ne dépend que de h et reste le même quel que soit $(x,y) \in \mathbb{R}^2$. On dit que l'opérateur de convolution est *invariant par translation*. On appelle h le *noyau de convolution*. En général, pour que la convolution soit vraiment un moyennage, on prend h tel que

$$\int_{\mathbb{R}^2} h(x,y) dx dy = 1$$

La convolution est la première dégradation subie par l'image analogique. Ainsi, à un appareil de mesure correspond un noyau de convolution h (celui-ci peut être du à l'optique, un compteur de photons, ...).

1.2.2 De suites finies

Nous profitons de l'introduction de la convolution pour les fonctions analogiques pour la définir pour des suites finies. Soit $(h_{m,n})_{1 \le m \le N, 1 \le n \le N}$ et $(v_{m,n})_{1 \le m \le N, 1 \le n \le N}$, deux suites finies. On note $h*v$, le *produit de convolution de h par v*, la suite

$$(h*v)_{m,n} = \sum_{m'=1}^{N} \sum_{n'=1}^{N} h_{m-m', n-n'} v_{m',n'}.$$

1. Par intégrable, on veut dire que l'intégrale, sur son domaine de définition, du module de la fonction existe.

Ici, on suppose que h est périodisé en dehors de $\{1,\dots,N\}^2$. Ceci veut dire que l'on définit

$$\forall (m,n) \in \{1,\dots,N\}^2, \forall (k,l) \in \mathbb{Z}, h_{m+kN,n+lN} = h_{m,n}.$$

Il est à noter que, même si h et v sont des images numériques, $h * v$ n'est pas forcément une image numérique car elle prend, à priori, des valeurs hors de $\mathcal{C}ol$. Nous verrons par la suite comment remédier à ce problème.

L'intuition est la même pour le produit de convolution entre des fonctions numériques et des fonctions analogiques. On a un effet de moyennage de v autour des points (m,n). Vous pouvez à titre d'exercice retrouver une formule de type de (1.1) pour $h_{m,n} = \frac{1}{9}\mathbf{1}_{|\{-1,0,1\}^2}(m,n)$ et $h_{m,n} = \frac{1}{25}\mathbf{1}_{|\{-2,-1,0,1,2\}^2}(m,n)$.

1.3 Le fenêtrage

Dans le chapitre précédent, pour la convolution entre fonctions analogiques, le domaine est $\mathbb{R}^2$. Or, en pratique, une image numérique ne représente qu'une partie finie de l'ensemble de la scène observable. Il y a donc un *fenêtrage* de cette image analogique avant la numérisation. Le choix de la fenêtre correspond à ce que les photographes appellent le cadrage.

Ainsi, après la convolution, on a une image analogique $h * v$ dont on ne gardera que la partie intérieure à une fenêtre $[0,N]^2$. Ceci revient mathématiquement à la multiplier par $\mathbf{1}_{|[0,N]^2}$.

Cette perte d'informations joue un rôle important près des bords de l'image. En effet, lors de la convolution (voir Figure 1.1), si (x,y) est tel qu'il existe (x',y') hors de $[0,N]^2$ tel que $h(x-x',y-y') \neq 0$, $h * v(x,y)$ dépend de v en un endroit où on la connaît peu (en pratique pas).

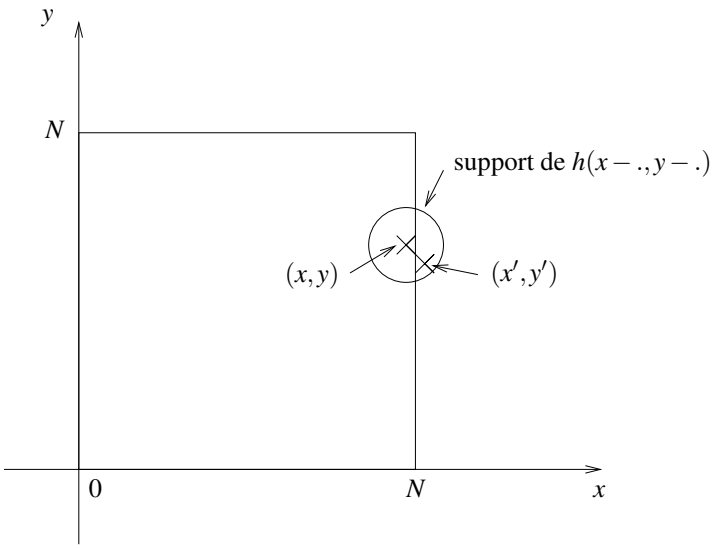


FIGURE 1.1 – La valeur de $h * v(x,y)$ dépend de la valeur de $v(x',y')$.

De même, l'information sur la valeur de $v(x,y)$ est répartie sur les valeurs de $h * v(x',y')$, avec (x',y') tel que $h(x'-x,y'-y) \neq 0$. Si un tel (x',y') est en dehors de $[0,N]^2$, cette information est perdue. Nous sommes donc contraint de faire une hypothèse sur cette information.

Pour remédier à ces problèmes, nous prolongerons $(h*v)\mathbf{1}_{|[0,N]^2}$ par périodisation. Ceci revient à poser

$$h * v(x,y) = h * v(x + t_x N, y + t_y N)$$

avec t_x et t_y tels que $(x+t_xN, y+t_yN) \in [0,N]^2$.

Il y a bien sûr beaucoup d'autres possibilités pour traiter ces *problèmes de bord*.

1.4 L'échantillonnage

L'échantillonnage est bien souvent la partie du processus de création d'une image digitale durant laquelle le plus d'informations est perdue. Elle consiste à ne garder que les valeurs de $(h*v)\mathbf{1}_{|[0,N]^2}$ aux points entiers. Mathématiquement, on obtient une fonction définie sur $\{1,\dots,N\}^2$ définie par

$$h*v(m,n)\mathbf{1}_{|[0,N]^2}(m,n),$$

pour $(m,n) \in \{1,\dots,N\}^2$.

Pour sous-échantillonner, d'un rapport K (pour K un entier strictement positif), une image digitale u définie sur $\{1,\dots,KN\}^2$, pour obtenir une image digitale u' définie sur $\{1,\dots,N\}^2$, on effectue l'opération

$$u'_{m,n} = u_{Km,Kn}.$$

On considère en général que le sous-échantillonnage d'une image digitale approxime bien les effets de l'échantillonnage permettant de créer une image à partir de $K = 3$.

1.5 Le bruit

Un appareil de mesure créant une image digitale génère toujours un bruit. Les causes peuvent venir de plusieurs sources (caractère probabiliste du nombre de photons issus d'une région d'intensité donnée, imperfections électroniques de l'appareil, imperfections des capteurs, ...). Nous ne considérerons ici que les bruits additifs. Il s'agira d'une suite, définie sur $\{1,\dots,N\}^2$, dont la valeur est aléatoire. Bien que cela soit rarement le cas, il est souvent raisonnable de supposer que le bruit b est *blanc* (les valeurs $b_{m,n}$ sont indépendantes les unes des autres). On supposera de plus qu'il est *Gaussien*. Ceci veut dire que chaque $b_{m,n}$ est une réalisation de la loi de probabilité continue, de densité

$$p(t) = \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{t^2}{2\sigma^2}), \tag{1.2}$$

pour $\sigma > 0$. On a introduit $\sigma > 0$, qui représente l'importance du bruit. On obtient alors une fonction définie sur $\{1,\dots,N\}^2$, valant

$$h*v(m,n)\mathbf{1}_{|[0,N]^2}(m,n) + b_{m,n}.$$

Pour un bruit b, comme pour toute variable aléatoire, on peut parler de *l'espérance* d'une fonction $f(b)$. On la définit mathématiquement par

$$\mathbb{E}(f(b)) = \int_{\mathbb{R}} f(t)p(t)dt$$

où $p(t)$ est la densité de la loi de b (ici la loi Gaussienne définie par (1.2)). L'espérance représente la valeur moyenne de $f(b)$ pour un nombre infini de réalisations indépendantes de b. C'est même en utilisant cette propriété que l'on calcule numériquement l'espérance, lorsque l'on sait produire des réalisations (indépendantes) de b.

Par exemple, il n'est pas difficile de voir (ce sont de simples changements de variable) que

$$\mathbb{E}(b) = \frac{1}{\sqrt{2\pi}\sigma}\int_{\mathbb{R}} t\exp(-\frac{t^2}{2\sigma^2})dt = 0.$$

Ceci représente la *valeur moyenne* de b.

On définit aussi la *variance* de b, par

$$\begin{aligned}\mathbb{E}\left((b-\mathbb{E}(b))^2\right) &= \frac{1}{\sqrt{2\pi}\sigma}\int_{\mathbb{R}} t^2\exp(-\frac{t^2}{2\sigma^2})dt \\ &= \frac{\sigma^2}{\sqrt{2\pi}}\int_{\mathbb{R}} t^2\exp(-\frac{t^2}{2})dt \\ &= \sigma^2\end{aligned}$$

La variance (dans le cas d'un bruit Gaussien, simplement, σ^2) nous donne l'écart quadratique moyen entre une réalisation de b et sa valeur moyenne.

1.6 La quantification

La quantification est l'opération qui consiste à traduire les valeurs de

$$h * v(m,n)\mathbf{1}_{|[0,N]^2}(m,n) + b_{m,n}.$$

sous la forme d'une couleur. Pour cela, il faut approximer cette valeur (qui est dans $\mathbb{R}^3$, pour des images couleurs et $\mathbb{R}$ pour des images noir et blanc) de façon à ce qu'elle soit codable dans un ordinateur (on dispose d'un nombre fini de bits).

Si l'on considère le cas d'images noir et blanc avec des niveaux de gris appartenant à un ensemble $\{0,1,\ldots,255\}$, on *quantifie* une valeur $t \in \mathbb{R}$ en l'approximant par la valeur $Ar(t)$, la plus proche de t dans $\{0,1,\ldots,255\}$.

On obtient ainsi enfin notre image digitale sous la forme

$$u_{m,n} = Ar\left(h * v(m,n)\mathbf{1}_{|[0,N]^2}(m,n) + b_{m,n}\right)$$

Remarque : Il est parfois nécessaire de modifier la dynamique de l'image avant la quantification (par exemple : si l'image est trop sombre, une quantification brutale engendrerait trop de perte d'informations ; si l'image est trop claire, beaucoup de points satureraient à la valeur 255). Nous négligerons dans la suite ce *changement de contraste*.

1.7 Autres dégradations

Il y a bien sûr beaucoup d'autres sources possibles de dégradation d'une image. Nous n'avons abordé ici que celles concernant l'appareil de mesure fonctionnant normalement. On peut mentionner par exemple :

- **Le changement de contraste :** Comme nous l'avons dit au chapitre précédent, on a parfois intérêt à modifier le contraste. Les caméras numériques et appareils photo numériques font presque toujours un changement de contraste pour s'adapter aux conditions d'éclairage.
- **La perte d'une partie de l'image :** Il peut arriver aussi qu'une partie de l'image soit perdue (par exemple : sur une photo abîmée, durant la transmission d'une image satellite, vieux films). Dans ce cas, on a un masque M, défini sur $\{1,\ldots,N\}^2$, à valeur dans $\{0,1\}$ et la nouvelle image est donnée par
$$\tilde{u}_{m,n} = M_{m,n}u_{m,n}.$$
- **Des distorsions géométriques :** Certains appareils de mesure ne font pas un échantillonnage sur une grille parfaite. Cela peut être du à des imperfections du capteur, des vibrations d'un satellite, ... On a alors une fonction de déformation $\varphi : \mathbb{R}^2 \to \mathbb{R}^2$ et l'image prend la forme
$$u_{m,n} = Ar\left(h * v\left(\varphi(m,n)\right)\mathbf{1}_{|[0,N]^2}\left(\varphi(m,n)\right) + b_{m,n}\right)$$

– **Les pertes dues à la compression :** Pour stocker ou transmettre une image, on la compresse souvent. Cette compression peut générer des défauts sur l'image reconstruite. Ces défauts dépendent évidement de la méthode et du niveau de compression.

Il existe évidement d'autres sources de dégradations possibles. Nous n'aborderons pas (ou peu) les méthodes visant à réduire les effets de ces dégradations.

1.8 Exemple

Nous montrons dans les images suivantes les résultats pour une image donnée des dégradations définies par un noyau de convolution $h(x,y) = \mathbf{1}_{[[-1,1]]^2}$ et un bruit de variance $\sigma = 4$. (Ce noyau de convolution est réaliste, on rencontre des noyaux ayant le même genre d'effets en imagerie satellite. Le bruit dans un tel cas serait plus faible. Nous l'avons volontairement augmenté afin qu'il soit bien visible.)

FIGURE 1.2 – Image analogique de départ.

FIGURE 1.3 – Image analogique après la convolution.

FIGURE 1.4 – Image analogique après la convolution et le fenêtrage.

FIGURE 1.5 – Image analogique après la convolution, le fenêtrage et l'échantillonnage.

FIGURE 1.6 – Image analogique après la convolution, le fenêtrage, l'échantillonnage et l'ajout d'un bruit.

FIGURE 1.7 – Image analogique après la convolution, le fenêtrage, l'échantillonnage, l'ajout d'un bruit et la quantification. C'est une image digitale.

Chapitre 2

Rappels d'optimisation

Nous rappelons dans cette section les principes de quelques algorithmes d'optimisation que l'on utilise souvent en traitement d'images. Ce sont des algorithmes classiques et génériques d'optimisation qui ont un large spectre d'application et qui seront raisonnablement bien adaptés aux fonctionnelles ainsi qu'à la taille des problèmes que l'on rencontre en traitement d'images. Notamment, ils ne nécessitent pas de calcul du Hessien, d'inversion de matrice ou autres calculs qui s'avéreraient difficilement praticables pour beaucoup de problèmes de traitement d'images.

Enfin, nous n'énoncerons pas toujours de résultats formels concernant la convergence des algorithmes présentés. Il est, en effet, difficile d'avoir un énoncé pouvant s'appliquer à l'ensemble des fonctionnelles rencontrées en traitement d'images. Les résultats de convergence s'établissent en pratique individuellement pour chacune des fonctionnelles rencontrées en traitement d'images en adaptant les résultats génériques de convergence des algorithmes.

Une présentation plus complète de la programmation non-linéaire peut être trouvée dans [1, 3].

2.1 Optimisation sans-contrainte

2.1.1 Nature du problème

En traitement d'images, on rencontre un nombre important de problèmes d'optimisation dans lesquels on recherche un minimiseur d'un problème de la forme

$$\min_{w \in W} E(w),$$

où W est un espace Euclidien sur le corps des réels et E est une fonctionnelle (dans ce cours, on utilisera aussi le terme *énergie*) **convexe**, **coercive**, définie sur W et ne prenant que des valeurs finies.

On sait donc, grâce aux résultats classiques en optimisation non-linéaire, qu'une solution à notre problème existe. On sait aussi que cette solution n'est éventuellement pas unique. On se place de ce point de vue dans un cadre relativement simple de la programmation non-linéaire. Les principales difficultés que posent les fonctionnelles habituellement utilisées en traitement d'images sont

- *La dimension du problème :* L'espace W est souvent de la taille de l'image. Ceci représente en général des tailles de l'ordre du million ou du milliard de variables.
- *Les fonctionnelles sont souvent non différentiables :* Dans le cadre de ce cours : Nous utiliserons, lorsque cela est possible, un algorithme de minimisation adapté ; ou nous nous ramènerons au cas de fonctionnelles différentiables en régularisant la fonctionnelle initiale. Ceci permet d'utiliser des algorithmes utilisant le gradient. Ce gradient reste cependant très instable au voisinage des points

où la fonctionnelle non régularisée n'est pas différentiable. Ceci rend impraticable les algorithmes utilisant la Hessienne de la fonctionnelle.
- *Les fonctionnelles sont souvent "peu elliptiques" :* Nous ne rentrerons pas dans les difficultés propres à la minimisation de ce genre de fonctionnelles.

Ces difficultés étant mentionnées, la minimisation des fonctionnelles rencontrées en traitement d'images se passe généralement bien. Améliorer l'algorithme de minimisation utilisé reste évidemment toujours une piste à envisager pour améliorer un algorithme de traitement d'images.

2.1.2 L'algorithme du gradient (la plus profonde descente)

L'algorithme du gradient est décrit dans l'Algorithme 1. On sait qu'il converge pour une classe assez grande de fonctionnelles E.

Algorithme 1 Algorithme du gradient

Entrée: Les entrées nécessaires au calcul de E et de ∇E
Sortie: Une approximation d'un minimiseur de E : w

Initialisation de w
Tant que l'algorithme n'a pas convergé **faire**
  Calculer $d = \nabla E(w)$
  Calculer un pas $t \geq 0$
  Mettre à jour : $w \leftarrow w - t\, d$
Fin tant que

En pratique, pour implémenter l'Algorithme 1, il faut implémenter une fonction calculant $\nabla E(w)$. La fonction calculant le pas a aussi souvent besoin de calculer $E(w-td)$ (voir la section suivante dans laquelle plusieurs stratégies pour calculer le pas sont décrites). Il faut donc généralement implémenter une fonction calculant $E(w-td)$. Les choix restant à faire sont :

- **le critère de convergence** : en traitement d'images on a rarement besoin d'un très bon niveau de convergence et un utilisateur ne voit généralement pas la différence entre deux images dont certains pixels diffèrent de plusieurs niveaux de gris.
- **la méthode d'initialisation** : Tant que la fonction est convexe et coercive, le choix de la méthode d'initialisation n'empêche pas l'algorithme de converger. En ce sens, on ne peut pas faire d'erreur rédhibitoire concernant l'initialisation. Par contre, une bonne initialisation permet de faire moins d'itérations et donc de gagner du temps de calcul.
- **la stratégie de calcul du pas** : Ce point fera l'objet de la Section 2.1.3. Il sera aussi illustré par le Théorème 1.

On rappelle ci-dessous un résultat bien connu.

Théorème 1 *Si $E : W \longrightarrow \mathbb{R}$ est une fonctionnelle différentiable, s'il existe des constantes $\alpha > 0$ et $L > 0$ telles que*

$$\left\langle \nabla E(w) - \nabla E(w'), w - w' \right\rangle \geq \alpha \|w - w'\|_2^2 \quad \text{, pour tout } w, w' \in W, \tag{2.1}$$

$$\|\nabla E(w) - \nabla E(w')\|_2 \leq L \|w - w'\|_2 \quad \text{, pour tout } w, w' \in W, \tag{2.2}$$

et s'il existe deux nombres a et b tels que, à chaque itération de l'algorithme 1, le pas t vérifie

$$0 < a \leq t \leq b < \frac{2\alpha}{L^2},$$

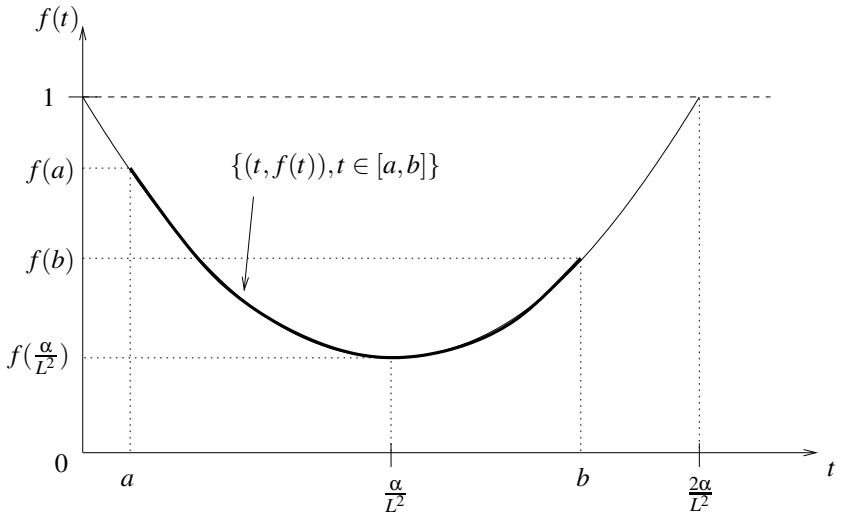


FIGURE 2.1 – Graphe de la fonction $f(t)$ apparaissant dans la preuve de convergence de l'algorithme du gradient.

l'Algorithme 1 converge. De plus, si l'on note w^* sa limite, il existe $\beta < 1$ tel que

$$\|w^k - w^*\|_2 \leq \beta^k \|w^0 - w^*\|_2,$$

où w^k représente l'élément w de l'algorithme 1 à l'itération k.

En particulier, si à chaque itération de l'algorithme du gradient on prend $t = \frac{\alpha}{L^2}$, on a

$$\beta = \sqrt{1 - \frac{\alpha^2}{L^2}}.$$

Preuve. Nous ne le montrerons pas, mais il n'est pas très difficile de voir qu'une fonctionnelle elliptique (i.e. vérifiant (2.1)) est strictement convexe et coercive. Elle a donc un unique minimum global w^*. De plus, comme E est différentiable, ce minimum global vérifie $\nabla E(w^*) = 0$. On a donc

$$\begin{aligned}
\|w^{k+1} - w^*\|_2^2 &= \|w^k - w^* - t(\nabla E(w^k) - \nabla E(w^*))\|_2^2 \\
&= \|w^k - w^*\|_2^2 - 2t\left\langle w^k - w^*, \nabla E(w^k) - \nabla E(w^*)\right\rangle + t^2 \|\nabla E(w^k) - \nabla E(w^*)\|_2^2 \\
&\leq \left(1 - 2\alpha t + L^2 t^2\right) \|w^k - w^*\|_2^2
\end{aligned}$$

On note $f(t) = 1 - 2\alpha t + L^2 t^2$. Le graphe de la fonction $f(t)$ est dessiné sur la Figure 2.1. On voit sur cette figure que si

$$0 < a \leq t \leq b < \frac{2\alpha}{L^2},$$

alors

$$f(t) \leq \max(f(a), f(b)).$$

On a donc pour tout $0 < a \leq t \leq b < \frac{2\alpha}{L^2}$

$$\begin{aligned}
\|w^{k+1} - w^*\|_2 &\leq \sqrt{f(t)} \|w^k - w^*\|_2 \\
&\leq \beta \|w^k - w^*\|_2,
\end{aligned}$$

où l'on a posé $\beta = \sqrt{\max(f(a), f(b))}$.

Par récurrence, on obtient finalement que

$$\|w^{k+1} - w^*\|_2 \leq \beta^{k+1} \|w^0 - w^*\|_2.$$

On déduit la dernière phrase de l'énoncé du fait que $f(\frac{\alpha}{L^2}) = 1 - \frac{\alpha^2}{L^2}$. □

Il existe plusieurs énoncés de convergence différents pour l'algorithme de gradient. Ils diffèrent dans les hypothèses qu'ils imposent à la fonctionnelle, la stratégie adoptée pour fixer le pas t et la qualité de la garantie de convergence obtenue. On a notamment l'énoncé ci-dessous.

Théorème 2 *Si $E : W \longrightarrow \mathbb{R}$ est une fonctionnelle différentiable, convexe et coercive. S'il existe une constante $L > 0$ telle que*

$$\|\nabla E(w) - \nabla E(w')\|_2 \leq L\|w - w'\|_2 \qquad \text{, pour tout } w, w' \in W, \tag{2.3}$$

et si, à chaque itération de l'algorithme 1, le pas $t = \frac{1}{L}$ alors la suite $(E(w^k))_{k\in\mathbb{N}}$, où w^k représente l'élément w de l'algorithme 1 à l'itération k, converge. De plus, si l'on note w^* un minimiseur de E, on a

$$0 \leq E(w^k) - E(w^*) \leq \frac{L}{2k}\|w^0 - w^*\|_2.$$

Preuve. Pour montrer ce résultat, on commence par montrer deux lemmes.

Lemme 1 Création d'un majorant quadratique

Sous les hypothèses du Théorème 2, on a pour tout w et $w' \in W$

$$E(w') \leq E(w) + \big\langle \nabla E(w), w' - w \big\rangle + \frac{L}{2}\|w' - w\|_2^2.$$

Preuve. La preuve repose sur la formule de Taylor avec reste intégrale :

$$E(w') = E(w) + \int_0^1 \big\langle \nabla E(tw + (1-t)w'), w' - w \big\rangle dt.$$

On a donc

$$\begin{aligned}
E(w') - \big(E(w) + \big\langle \nabla E(w), w' - w \big\rangle\big) &= \int_0^1 \big\langle \nabla E(tw + (1-t)w') - \nabla E(w), w' - w \big\rangle dt, \\
&\leq \int_0^1 \|\nabla E(tw + (1-t)w') - \nabla E(w)\|_2 \, \|w' - w\|_2 dt, \\
&\leq \int_0^1 L\|tw + (1-t)w' - w\|_2 \, \|w' - w\|_2 dt, \\
&= L\|w' - w\|_2^2 \int_0^1 (1-t)dt = \frac{L}{2}\|w' - w\|_2^2.
\end{aligned}$$

□

On note, pour $k \geq 1$ et $w \in W$

$$F_k(w) = E(w^{k-1}) + \left\langle \nabla E(w^{k-1}), w - w^{k-1} \right\rangle + \frac{L}{2}\|w - w^{k-1}\|_2^2.$$

On a alors le lemme suivant.

Lemme 2 On minore le majorant quadratique et il décroit significativement

Sous les hypothèses du Théorème 2, on a

$$w^k = \text{Argmin}_{w\in W} F_k(w). \tag{2.4}$$

On a aussi [1] *pour tout w et tout* $w' \in W$

$$F_k(w) = F_k(w') + \frac{L}{2}\|w-w'\|_2^2 + \left\langle \nabla F_k(w'), w-w' \right\rangle \tag{2.5}$$

En particulier, si $w' = w^k$*, on a*

$$F_k(w) = F_k(w^k) + \frac{L}{2}\|w-w^k\|_2^2.$$

Preuve. Ce résultat découle du fait que pour tout $w \in W$

$$\nabla F_k(w) = \nabla E(w^{k-1}) + L(w-w^{k-1}).$$

On a donc en utilisant le fait que $w^k = w^{k-1} - \frac{1}{L}\nabla E(w^{k-1})$

$$\nabla F_k(w^k) = \nabla E(w^{k-1}) + L\left(\left(w^{k-1} - \frac{1}{L}\nabla E(w^{k-1})\right) - w^{k-1}\right) = 0. \tag{2.6}$$

Le point w^k est donc un point critique de F_k qui est une fonction quadratique dont le Hessien est défini positif : w^k est le minimiseur global de F_k.

Pour montrer le second point, on calcule pour w et $w' \in W$

$$\begin{aligned}
F_k(w) - F_k(w') &= E(w^{k-1}) + \left\langle \nabla E(w^{k-1}), w-w^{k-1} \right\rangle + \frac{L}{2}\|w-w^{k-1}\|_2^2 \\
&\quad - \left(E(w^{k-1}) + \left\langle \nabla E(w^{k-1}), w'-w^{k-1} \right\rangle + \frac{L}{2}\|w'-w^{k-1}\|_2^2 \right), \\
&= \left\langle \nabla E(w^{k-1}), w-w' \right\rangle + \frac{L}{2}\left(\|w-w'\|_2^2 + 2\left\langle w-w', w'-w^{k-1} \right\rangle \right), \\
&= \frac{L}{2}\|w-w'\|_2^2 + \left\langle \nabla E(w^{k-1}) + L(w'-w^{k-1}), w-w' \right\rangle, \\
&= \frac{L}{2}\|w-w'\|_2^2 + \left\langle \nabla F_k(w'), w-w' \right\rangle.
\end{aligned}$$

Le troisième point est une conséquence directe de (2.5) et de (2.6).

□

Revenons à la preuve du Théorème 2.

Si l'on note $w^* \in \text{Argmin}_{w\in W} E(w)$, on peut alors facilement établir en utilisant les deux lemmes, puis en utilisant la convexité de E que

$$\begin{aligned}
E(w^k) &\leq F_k(w^k) \\
&= F_k(w^*) - \frac{L}{2}\|w^*-w^k\|_2^2 \\
&= E(w^{k-1}) + \left\langle \nabla E(w^{k-1}), w^*-w^{k-1} \right\rangle + \frac{L}{2}\|w^*-w^{k-1}\|_2^2 - \frac{L}{2}\|w^*-w^k\|_2^2 \\
&\leq E(w^*) + \frac{L}{2}\left(\|w^*-w^{k-1}\|_2^2 - \|w^*-w^k\|_2^2 \right)
\end{aligned}$$

1. Cette propriété est en fait vraie pour toutes les fonctions quadratiques.

En remarquant que du fait du Lemme 1 et de (2.4), on a

$$E(w^k) \leq F_k(w^k) \leq F_k(w^{k-1}) = E(w^{k-1}),$$

et donc $(E(w^k))_{k\in\mathbb{N}}$ décroit. On a pour tout $k' \leq k$

$$E(w^k) - E(w^*) \leq E(w^{k'}) - E(w^*).$$

On conclut finalement

$$\begin{aligned} E(w^k) - E(w^*) &\leq \frac{1}{k}\sum_{k'=1}^{k}\Big(E(w^{k'}) - E(w^*)\Big) \\ &\leq \frac{L}{2k}\sum_{k'=1}^{k}\Big(\|w^* - w^{k'-1}\|_2^2 - \|w^* - w^{k'}\|_2^2\Big) \\ &\leq \frac{L}{2k}\Big(\|w^* - w^0\|_2^2 - \|w^* - w^k\|_2^2\Big) \\ &\leq \frac{L}{2k}\|w^* - w^0\|_2^2 \end{aligned}$$

□

Comme le Théorème 1, le Théorème 2 énonce aussi que l'algorithme 1 converge. Cependant, l'hypothèse faite sur la fonctionnelle est plus faible : on remplace (2.1) par une simple hypothèse de convexité et de coercivité. De ce fait, la garantie de convergence est aussi plus faible : On remplace une garantie sur la convergence de w^k par une garantie sur la convergence de $E(w^k)$; Aussi, la *vitesse de convergence* est en $\frac{1}{k}$, au lieu d'être en β^k, avec $\beta < 1$.

Enfin, l'inégalité (2.3) reste vraie lorsque l'on augmente L. Cependant, dans un tel cas le pas diminue ce qui va ralentir l'algorithme. On a donc intérêt à trouver (en pratique par le calcul) la meilleur constante Lipschitz L possible.

2.1.3 Le calcul du pas de descente

Le pas constant

À chaque itération de l'Algorithme 1, on utilise le même pas $t = t_0$.

C'est la méthode la plus simple et elle permet souvent de construire un algorithme convergeant. Son principal défaut est que le pas que l'on choisit est souvent difficile à ajuster a-priori. Un pas trop faible conduit à un nombre d'itération (et donc un temps de calcul) très important. Un pas trop grand fait, en général, diverger l'algorithme. Les Théorèmes 1 et 2 donnent de bons exemples de pas t_0 permettant de faire converger l'algorithme du gradient lorsque la fonctionnelle E vérifie ses hypothèses et que l'on sait calculer α et M.

Le pas conduisant à la plus grande diminution

Si l'on note w^k l'élément w de l'Algorithme 1 à la $k^{\text{ième}}$ itération, on sait que w^{k+1} est sur la droite paramétrée :

$$\{w^k - \alpha\nabla E(w^k), \alpha \in \mathbb{R}\}.$$

Comme le but de l'Algorithme 1 est de minimiser E, il semble donc naturel de chercher un minimiseur de

$$\min_{\alpha\in\mathbb{R}} E(w^k - \alpha\nabla E(w^k)).$$

On sait que

$$\begin{array}{rcl} \mathbb{R} & \longrightarrow & \mathbb{R} \\ \alpha & \longmapsto & E(w^k - \alpha \nabla E(w^k)) \end{array}$$

est une fonction convexe, coercive décroissante en $\alpha = 0$. On peut donc, par exemple, construire un algorithme de recherche de minimum d'une fonction définie sur $\mathbb{R}$. On construit facilement un algorithme utilisant des intervalles dont la taille est divisée par 2 à chaque itération (algorithme par dichotomie). Un tel algorithme converge donc très rapidement. Il suffit pour l'implémenter de savoir calculer $E(w^k - \alpha \nabla E(w^k))$, en tout $\alpha \in \mathbb{R}$.

Le pas conduisant à la plus grande diminution peut conduire à une convergence lente si la fonctionnelle est mal conditionnée. Dans ce cas, l'algorithme risque en effet de faire des petits pas en zigzaguant sans réellement s'approcher de la solution.

Une autre situation bien connue dans laquelle le pas conduisant à la plus grande diminution n'est pas adapté est le cas de fonctionnelles non-différentiables (avec un algorithme de sous-gradient). On sait, en effet que dans ce genre de situation, l'algorithme de gradient avec un pas conduisant à la plus grande descente peut converger vers un point qui n'est pas le minimum de la fonctionnelle. Nous ne serons jamais dans cette situation puisque nous avons choisi de régulariser les fonctionnelles utilisées. Cependant, on peut craindre que des phénomènes similaires puissent se produire et gênent la convergence de notre algorithme. Notre fonctionnelle sera en effet proche d'une fonctionnelle non différentiable.

La règle de Armijo

Une solution pour contourner le défaut du pas conduisant à la plus grande descente, mentionnée au paragraphe précédent est de ne le calculer qu'approximativement, tout en garantissant une décroissance suffisante.

Dans ce cadre, on fixe un pas maximum $t_{max} > 0$ et un facteur de réduction β tel que $0 < \beta < 1$. On choisit un pas de la forme

$$t_i = \beta^i t_{max}, \text{ pour } i \in \mathbb{N}.$$

Il n'est pas difficile de voir que $(t_i)_{i \in \mathbb{N}}$ est une suite décroissante telle que $\lim_{i \to \infty} t_i = 0$.

La règle d'Armijo consiste à choisir i comme étant le plus petit entier (i.e. le plus grand pas t_i) tel que

$$E(w^k) - E(w^k - t_i \nabla E(w^k)) \geq \sigma \, t_i \|\nabla E(w^k)\|_2^2, \tag{2.7}$$

pour un paramètre $\sigma \in]0,1[$ (par exemple, $\sigma = 0.5$).

Comme pour le pas conduisant à la plus grande descente, on peut facilement implémenter la règle d'Armijo si l'on est capable de calculer $E(w^k - t_i \nabla E(w^k))$, pour les différentes valeurs de t_i. L'existence d'un pas satisfaisant la règle d'Armijo est illustrée sur la Figure 2.2.

Lorsque l'algorithme du gradient décrit dans l'Algorithme 1 est utilisé avec un pas obéissant à la règle d'Armijo, il converge pour une large classe de fonctionnelles. Notamment, pour les fonctionnelles que l'on rencontrera en traitement d'images. Un avantage par rapport à une stratégie du pas conduisant à la plus grande diminution est qu'un nombre moins important d'évaluations de $E(w^k - \alpha \nabla E(w^k))$ est nécessaire pour calculer chaque pas. Comparé au pas constant, un pas conforme à la règle d'Armijo ne risque pas de faire diverger l'algorithme sans rajouter de sur-coût trop important, puisque peu d'évaluations de $E(w^k - t_i \nabla E(w^k))$ sont nécessaires à son calcul.

2.1.4 Élément d'optimisation non-différentiable : L'algorithme du gradient proximal

Cet algorithme permet de minimiser efficacement certaines fonctionnelles non différentiables. Pour ces fonctionnelles, on ne peut en effet pas calculer le gradient et donc pas appliquer l'algorithme du

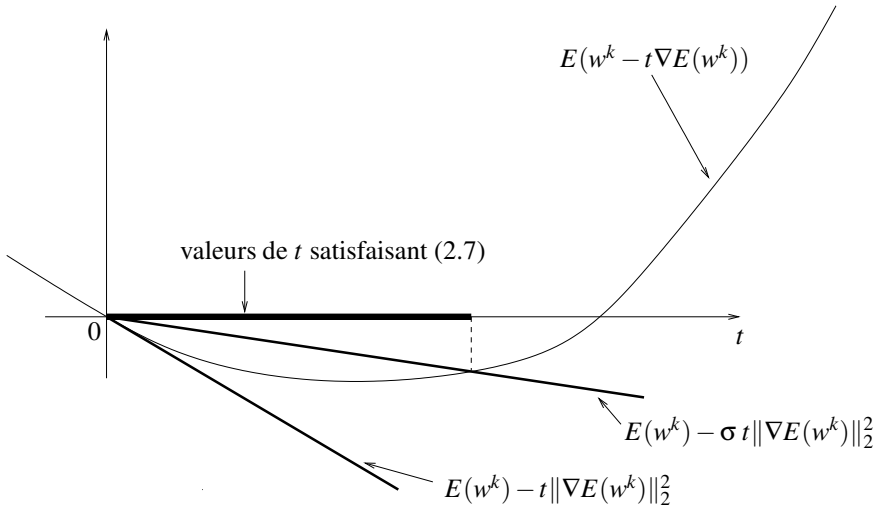


FIGURE 2.2 – Illustration de la règle d'Armijo.

gradient vu au chapitre précédent.

Pour pouvoir réutiliser les calculs fait au chapitre précédent, on considère une fonctionnelle $\mathcal{E}$ définie sur un espace Euclidien W. On suppose qu'elle peut s'écrire sous la forme

$$\mathcal{E}(w) = E(w) + R(w) \qquad \text{, pour tout } w \in W,$$

où E est différentiable, convexe, coercive et à gradient Lipschitz (i.e. E satisfait les hypothèses du Théorème 2) ; et R est semi-continue inférieurement [2], propre [3], convexe mais peut-être non-différentiable. On suppose aussi que l'on sait calculer simplement, pour tout $w' \in W$

$$\text{prox}_R^t(w') = \text{Argmin}_{w \in W} \frac{t}{2}\|w - w'\|_2^2 + R(w).$$

À noter, pour que le calcul de $\text{prox}_R^t(w')$ soit simple, il faut souvent faire un calcul analytique, sur papier, pour établir une formule simple à implémenter. (De la même manière qu'il faut faire un calcul analytique sur papier pour calculer le gradient dans le cas différentiable.)

Ces hypothèses sont un cadre que l'on rencontre souvent en traitement d'image et du signal. Par exemple, R peut être une fonction caractéristique d'un ensemble (i.e. valant 0 pour w dans l'ensemble et l'infini sinon). Dans ce cas, l'opérateur $\text{prox}_R^t(w')$ est simplement la projection de w' sur l'ensemble. On est capable d'obtenir une formule analytique pour une telle projection pour certains ensembles pertinents en traitement du signal et des images. Dans un tel cas, l'algorithme du gradient proximal sera identique à l'algorithme du gradient projeté. Ce dernier est en fait un cas particulier de l'algorithme du gradient proximal. On sait aussi calculer l'opérateur proximal de plusieurs critères R favorisant la parcimonie (nous verrons cela au Chapitre 5).

Si R est différentiable, alors $\text{prox}_R^t(w')$ vérifie

$$t\left(\text{prox}_R^t(w') - w'\right) + \nabla R(\text{prox}_R^t(w')) = 0.$$

On a donc

$$\text{prox}_R^t(w') = w' - \frac{1}{t}\nabla R(\text{prox}_R^t(w')). \tag{2.8}$$

2. I.e. Pour tout $w \in W$ et tout $\varepsilon > 0$, il existe un voisinage U de w tel que pour tout $u \in U$, $E(u) \geq E(w) - \varepsilon$

3. I.e. $\forall w \in W$, on a $E(w) > -\infty$ et il existe $w \in W$ tel que $E(w) < +\infty$.

Cette formule est très proche de

$$w = w' - \frac{1}{t}\nabla R(w')$$

qui correspond à une itération de l'algorithme du gradient. Dans (2.8), on prend le gradient au point d'arrivé. Ceci correspond à un schéma *implicite* avec un pas de taille $\frac{1}{t}$.

L'algorithme du gradient proximal itère en alternant une itération de gradient de E et une itération de l'opérateur proximal de R, avec le même pas. Il est décrit dans l'Algorithme 2. La problématique de l'initialisation est similaire à du cas de l'algorithme du gradient. Enfin, il faut en pratique établir sur le papier une formule pour la constante Lipschitz L, pour pouvoir l'implémenter.

Algorithme 2 Algorithme du gradient proximal

Entrée: Les entrées nécessaires au calcul de ∇E, de $\text{prox}_R^t(w')$ et de L.
Sortie: Une approximation d'un minimiseur de $\mathcal{E}$: w

Initialisation de w
Calcul de L
Tant que l'algorithme n'a pas convergé **faire**
  Calculer $d = \nabla E(w)$
  Mettre à jour : $w' \leftarrow w - \frac{1}{L} d$
  Mettre à jour : $w = \text{prox}_R^L(w')$
Fin tant que

Théorème 3 *Si $E : W \longrightarrow \mathbb{R}$ est une fonctionnelle différentiable, convexe et coercive. S'il existe une constante $L > 0$ telle que*

$$\|\nabla E(w) - \nabla E(w')\|_2 \leq L\|w - w'\|_2 \qquad \text{, pour tout } w, w' \in W$$

Si R est semi-continue inférieurement, propre, convexe et coercive. Si, w^k représente l'élément w de l'algorithme 2 à l'itération k et l'on note $\mathcal{E} = E + R$, alors la suite $(\mathcal{E}(w^k))_{k\in\mathbb{N}}$ converge.

De plus, si l'on note w^* un minimiseur de $\mathcal{E}$, on a

$$\mathcal{E}(w^k) - \mathcal{E}(w^*) \leq \frac{L}{2k}\|w^0 - w^*\|_2.$$

Preuve. La preuve de ce théorème est très proche de celle du Théorème 2. Notamment, comme E satisfait les hypothèses du Théorème 2 le Lemme 1 est valable. On utilise ainsi le même majorant quadratique pour E. On note à nouveau, pour $k \geq 1$ et $w \in W$,

$$F_k(w) = E(w^{k-1}) + \left\langle \nabla E(w^{k-1}), w - w^{k-1} \right\rangle + \frac{L}{2}\|w - w^{k-1}\|_2^2.$$

On a (voir Lemme 1) pour tout $w \in W$

$$E(w) \leq F_k(w). \tag{2.9}$$

Le Lemme 2 doit cependant être adapté.

Lemme 3 On minore le majorant et il décroit significativement

Sous les hypothèses du Théorème 3, on a

$$w^k = \text{Argmin}_{w\in W} F_k(w) + R(w). \tag{2.10}$$

On a aussi pour tout $w \in W$

$$F_k(w) + R(w) \geq F_k(w^k) + R(w^k) + \frac{L}{2}\|w - w^k\|_2^2. \tag{2.11}$$

Preuve. Le premier point est immédiat. On a en effet en utilisant les définitions de l'opérateur proximal et de w^k

$$\begin{aligned} w^k &= \text{prox}_R^L\left(w^{k-1} - \frac{1}{L}\nabla E(w^{k-1})\right), \\ &= \text{Argmin}_{w\in W}\ \frac{L}{2}\|w-w^{k-1} + \frac{1}{L}\nabla E(w^{k-1})\|_2^2 + R(w), \\ &= \text{Argmin}_{w\in W}\ \frac{1}{2L}\|\nabla E(w^{k-1})\|_2^2 + \left\langle w-w^{k-1}, \nabla E(w^{k-1})\right\rangle + \frac{L}{2}\|w-w^{k-1}\|_2^2 + R(w), \\ &= \text{Argmin}_{w\in W}\ F_k(w) + R(w). \end{aligned}$$

Pour montrer le second point, on commence par remarquer que comme (2.10) est vraie, alors on a aussi

$$w^k \in \text{Argmin}_{w\in W}\ \left\langle \nabla F_k(w^k), w-w^k\right\rangle + R(w) \tag{2.12}$$

En effet, sous les hypothèses du théorème, la fonction objectif ci-dessus est convexe et coercive, en w^k, elle a la même sous-différentielle[4] que $F_k(w)+R(w)$. Enfin, on déduit de (2.10) que 0 appartient à cette sous-diférenteelle :

$$0 \in \nabla F_k(w^k) + \partial R(w^k).$$

Ces deux point suffisent à conclure que (2.12) est vraie.

On obtient alors facilement (2.11). Tout d'abord : en utilisant (2.5), on voit que pour tout $w \in W$

$$F_k(w)+R(w)-F_k(w^k)-R(w^k) = \frac{L}{2}\|w-w^k\|_2^2 + \left\langle \nabla F_k(w^k), w-w^k\right\rangle + R(w) - R(w^k).$$

Puis, du fait de (2.12), on a

$$\left\langle \nabla F_k(w^k), w-w^k\right\rangle + R(w) \geq R(w^k)$$

et donc

$$F_k(w)+R(w)-F_k(w^k)-R(w^k) \geq \frac{L}{2}\|w-w^k\|_2^2.$$

□

On peut alors finir la preuve du Théorème 3.

Si l'on note $w^* \in \text{Argmin}_{w\in W}\,\mathcal{E}(w)$, on peut alors facilement établir en utilisant (2.9), le Lemme 3, puis en utilisant la convexité de E que

$$\begin{aligned} \mathcal{E}(w^k) &\leq F_k(w^k)+R(w^k) \\ &\leq F_k(w^*)+R(w^*) - \frac{L}{2}\|w^*-w^k\|_2^2 \\ &= E(w^{k-1}) + \left\langle \nabla E(w^{k-1}), w^*-w^{k-1}\right\rangle + R(w^*) + \frac{L}{2}\|w^*-w^{k-1}\|_2^2 - \frac{L}{2}\|w^*-w^k\|_2^2 \\ &\leq \mathcal{E}(w^*) + \frac{L}{2}\left(\|w^*-w^{k-1}\|_2^2 - \|w^*-w^k\|_2^2\right) \end{aligned}$$

En remarquant que du fait du Lemme 1 de (2.10), on a

$$\mathcal{E}(w^k) \leq F_k(w^k)+R(w^k) \leq F_k(w^{k-1})+R(w^{k-1}) \leq \mathcal{E}(w^{k-1})$$

4. On appelle sous-différentielle de R en w l'ensemble $\partial R(w) = \{g \in W | \forall w' \in W, R(w') \geq R(w) + \langle g, w'-w\rangle\}$.

$(\mathcal{E}(w^k))_{k\in\mathbb{N}}$ décroit, on a pour tout $k' \leq k$

$$\mathcal{E}(w^k) - \mathcal{E}(w^*) \leq \mathcal{E}(w^{k'}) - \mathcal{E}(w^*).$$

On conclut finalement

$$\begin{aligned}
\mathcal{E}(w^k) - \mathcal{E}(w^*) &\leq \frac{1}{k}\sum_{k'=1}^{k}\Big(\mathcal{E}(w^{k'}) - \mathcal{E}(w^*)\Big) \\
&\leq \frac{L}{2k}\sum_{k'=1}^{k}\Big(\|w^* - w^{k'-1}\|_2^2 - \|w^* - w^{k'}\|_2^2\Big) \\
&\leq \frac{L}{2k}\Big(\|w^* - w^0\|_2^2 - \|w^* - w^k\|_2^2\Big) \\
&\leq \frac{L}{2k}\|w^* - w^0\|_2^2
\end{aligned}$$

□

2.2 Optimisation sous contrainte

2.2.1 Nature du problème

Dans cette section, nous nous intéresserons à la recherche d'un minimiseur d'un problème de la forme

$$\begin{cases} \min_{w\in W}\ E(w), \\ \text{sous la contrainte } g_i(w) \leq 0 \quad \text{, pour tout } i \in \{1,\dots,M\}, \end{cases} \tag{2.13}$$

où, comme dans la section précédente, W est un espace Euclidien sur le corps des réels et E est une fonctionnelle **convexe**, **coercive**, définie sur W et ne prenant que des valeurs finies. La différence principale avec la section précédente est que les éléments w doivent satisfaire un ensemble de contraintes définies par des fonctions **convexes** et **différentiables** g_i allant de W dans $\mathbb{R}$.

Si l'on note

$$C = \{w \in W, \forall 1 \leq i \leq M,\ g_i(w) \leq 0\},$$

on vérifie facilement que C est **convexe** et **fermé**. On peut alors ré-écrire (2.13) sous la forme

$$\begin{cases} \min_{w}\ E(w), \\ \text{sous la contrainte } w \in C \subset W. \end{cases} \tag{2.14}$$

Dans la suite, nous utiliserons suivant le contexte l'une ou l'autre des écritures du problème afin de simplifier les notations.

Typiquement, C pourra être un espace affine, une boule l^2, un espace affine dilaté, un hypercube, un polyèdre, etc

Typiquement dans les problèmes que l'on rencontrera en traitement d'images, on sera toujours dans un cas où soit E est coercive, soit C est compact. De ce fait, on sait, grâce aux résultats classiques en programmation non-linéaire, qu'une solution à notre problème existe. On sait aussi que cette solution n'est éventuellement pas unique. On se place encore une fois de ce point de vue dans un cadre relativement simple de la programmation non-linéaire.

On retrouvera, pour les problèmes d'optimisation rencontrés en traitement d'images les mêmes spécificités que dans le chapitre précédent :

– dimension du problème ;
– fonctionnelles approximant des fonctionnelles non différentiables ;
– manque d'ellipticité.

2.2.2 Algorithme avec projection

Il n'est pas difficile de voir que les solutions du problème (2.14) coïncident exactement avec les solutions du problème suivant :

$$\min_{w \in W} E(w) + \chi_C(w)$$

où

$$\chi_C(w) = \begin{cases} 0 & , \text{si } w \in C, \\ +\infty & , \text{sinon.} \end{cases}$$

Dans un grand nombre de cas E et C sont tels que E et $\chi_C(w)$ satisfont les hypothèses du Théorème 3. Dans un tel cas, on a une garantie de convergence pour l'algorithme 2. Celui-ci nécessite de calculer la constante Lipschitz L du gradient de E et $\text{prox}^L_{\chi_C}(w')$.

On voit facilement que $\text{prox}^L_{\chi_C}(w')$ est indépendant de L est la solution de

$$\begin{cases} \min \|w' - w\|_2^2, \\ \text{sous la contrainte } w \in C. \end{cases}$$

Dis en mots, $\text{prox}^L_{\chi_C}(w')$ est la projection $P_C(w')$ de $w' \in W$ sur C. Cette projection consiste en le point de C le plus proche de w'.

Dans certains cas, on peut facilement calculer la projection d'un élément $w' \in W$ sur C. Ce sera (par exemple) généralement le cas lorsque C est une boule l^2, un hypercube ou un espace affine. Dans ce cas, on peut facilement construire un algorithme pour résoudre (2.14) en adaptant l'algorithme du gradient proximal décrit dans l'Algorithme 2.

Algorithme 3 Algorithme du gradient avec projection

Entrée: Les entrées nécessaires au calcul de E, de ∇E et de la projection
Sortie: Une approximation d'un minimiseur de (2.14) : w

Initialisation de w
Calculer L la constante Lipschitz du gradient de E
Tant que l'algorithme n'a pas convergé **faire**
  Calculer $d = \nabla E(w)$
  Mettre à jour : $w' \leftarrow w - \frac{1}{L} d$
  Projeter sur C : $w \leftarrow P_C(w')$
Fin tant que

Le Théorème 3 garantit la convergence avec une vitesse en $\frac{1}{k}$ de l'algorithme du gradient projeté tel que décrit dans l'Algorithme 3. En fait, nous ne le montrerons pas et ne l'énoncerons pas formellement mais l'algorithme du gradient avec projection converge pour une large classe de fonctionnelles, de contraintes et pour de nombreuses stratégies de calcul du pas t. La convergence sera notamment facile à établir pour les fonctionnelles que nous rencontrerons en traitement d'images.

2.2.3 Algorithme de pénalisation

Nous ne parlerons ici que de la méthode de "pénalisation extérieure". Pour simplifier, nous l'appellerons simplement : méthode de pénalisation. Ce choix est justifié par le fait que les fonctionnelles E

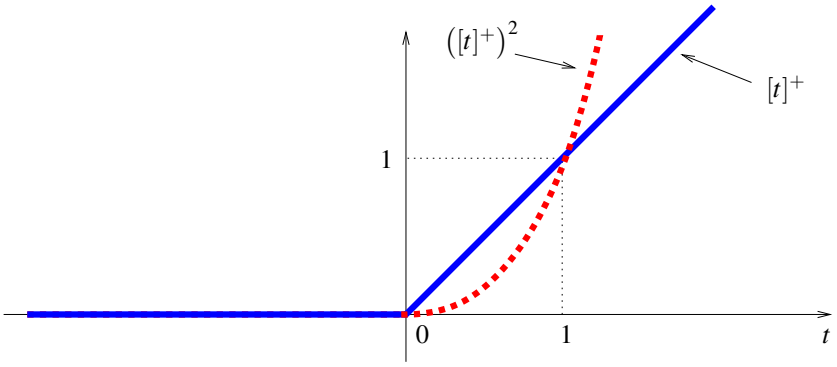


FIGURE 2.3 – Graphe des fonctions $t \longmapsto [t]^+$ et $t \longmapsto \left([t]^+\right)^2$.

rencontrées en traitement d'images sont définies sur tout W. Aussi, bien souvent, la contrainte $w \in C$ peut être satisfaite approximativement sans que cela ne se voit sur le résultat. Il n'est donc pas nécessaire d'utiliser une "pénalisation intérieure". Par ailleurs, certaines contraintes C rencontrées en traitement d'images ont un intérieur vide. Lorsque cela arrive, une pénalisation intérieure ne peut pas être mise en œuvre.

Pour écrire le problème à résoudre par une approche par pénalisation extérieure, on note

$$[t]^+ = \max(t,0) \quad ,\forall t \in \mathbb{R}. \tag{2.15}$$

Il n'est pas difficile de voir que la fonction

$$t \longmapsto \left([t]^+\right)^2$$

est croissante, convexe et différentiable sur $\mathbb{R}$ (voir Figure 2.3).

On obtient alors que, pour tout $\alpha \in [0,1]$ et tout w_1 et $w_2 \in W$ et tout $i \in \{1,\dots,M\}$,

$$\begin{aligned}\left(\left[g_i(\alpha w_1 + (1-\alpha)w_2)\right]^+\right)^2 &\leq \left(\left[\alpha g_i(w_1) + (1-\alpha)g_i(w_2)\right]^+\right)^2, \\ &\leq \alpha\left([g_i(w_1)]^+\right)^2 + (1-\alpha)\left([g_i(w_2)]^+\right)^2.\end{aligned}$$

On en déduit donc que la fonctionnelle

$$\begin{aligned} W &\longrightarrow \mathbb{R}, \\ w &\longmapsto \left([g_i(w)]^+\right)^2, \end{aligned}$$

est convexe. Elle est aussi différentiable (car elle est la composée de fonctions différentiables).

Le problème pénalisé s'écrit alors

$$\min_{w\in W} E(w) + n\sum_{i=1}^{M}\left([g_i(w)]^+\right)^2, \tag{2.16}$$

pour une valeur de $n \in \mathbb{N}$.

Il n'est pas difficile de voir que la fonctionnelle à minimiser est convexe, finie et différentiable sur W. Bien que cela dépende de la fonctionnelle E et de C, elle est, pour les problèmes rencontrés en traitement d'images, généralement coercive. Il n'est, par exemple, pas difficile de voir que ce sera bien le cas lorsque E est coercive ou lorsque C est compact.

On est donc dans un cadre dans lequel les algorithmes vus dans la Section 2.1 convergent. On peut donc le résoudre à l'aide de l'un de ces algorithmes.

Si, pour tout $n \in \mathbb{N}$, on note w_n^* un minimiseur de (2.16). Si l'on note S l'ensemble des minimiseurs de (2.13) et si l'on note pour tout $n \in \mathbb{N}$

$$d_n = \min_{w' \in S} \|w_n^* - w'\|_2,$$

la distance entre w_n^* et S, il n'est pas difficile de voir que

$$\lim_{n \to \infty} d_n = 0.$$

Les étapes pour montrer un résultat de ce genre sont les suivantes :

- Montrer que $(w_n^*)_{n \in \mathbb{N}}$ est bornée. Ceci se déduit généralement facilement de la coercivité de E et/ou de la compacité de C.
- Montrer que tout point d'accumulation de $(w_n^*)_{n \in \mathbb{N}}$ est dans C. On a, en effet, pour tout $w_0 \in C$,

$$0 \leq \sum_{i=1}^{M} \left([g_i(w_n^*)]^+ \right)^2 \leq \frac{E(w_0)}{n}.$$

  Le résultat se déduit alors du fait que les fonctions g_i et $([.]^+)^2$ sont continues.
- En déduire que tout point d'accumulation de $(w_n^*)_{n \in \mathbb{N}}$ est dans S. On considère un point d'accumulation w^* et une suite extraite $(w_o^*)_{o \in \mathbb{N}}$ convergeant vers w^*. Pour tout $o \in \mathbb{N}$ et tout $s \in S$, on a

$$\begin{aligned} E(w_o^*) &\leq E(w_o^*) + o \sum_{i=1}^{M} \left([g_i(w_o^*)]^+ \right)^2, \\ &\leq E(s) \end{aligned}$$

  On en déduit que, comme E est continue,

$$E(w^*) = \lim_{o \to \infty} E(w_o^*) \leq E(s)$$

- Conclure que $\lim_{n \to \infty} d_n = 0$. Si l'on écrit la négation de cet énoncé :

$$\exists \varepsilon > 0, \forall N, \exists n_N \geq N, |d_{n_N}| > \varepsilon$$

  on construit une suite $(w_{n_N}^*)_{N \in \mathbb{N}}$ extraite de $(w_n^*)_{n \in \mathbb{N}}$ dont aucune suite extraite ne peut converger dans S.

Pour finir, bien souvent, on résout itérativement (2.16) pour des valeurs croissantes de n. L'intérêt est que l'on peut utiliser la solution w_n^* pour initialiser l'algorithme de minimisation de (2.16) pour un $n' > n$. Ceci permet généralement d'obtenir une convergence plus rapide si l'on cherche directement à minimiser (2.16) pour n' avec une mauvaise initialisation.

2.2.4 Algorithme de Uzawa (Dual ascent)

Pour introduire l'algorithme de Uzawa, nous devons d'abord définir le Lagrangien du problème (2.13). Celui-ci nous permettra de définir le problème *dual* de (2.13). Nous verrons que ce problème s'exprime comme une maximisation. L'algorithme de Uzawa consiste à faire cette maximisation (d'où le nom anglais *dual ascent*).

Commençons donc par définir le Lagrangien de (2.13). Pour cela, nous notons

$$\mathbb{R}_+^M = \{(\lambda_i)_{1\leq i\leq M} \in \mathbb{R}^M, \forall 1 \leq i \leq M, \lambda_i \geq 0\}.$$

Le Lagrangien est alors défini pour tout $w \in W$ et $(\lambda_i)_{1\leq i\leq M} \in \mathbb{R}_+^M$ par

$$L(w,(\lambda_i)_{1\leq i\leq M}) = E(w) + \sum_{i=1}^{M} \lambda_i\, g_i(w).$$

On rappelle aussi que

$$C = \{w \in W, \forall 1 \leq i \leq M,\ g_i(w) \leq 0\}.$$

Il n'est pas très difficile de voir que, pour tout $w \in W$,

$$\max_{(\lambda_i)_{1\leq i\leq M}\in\mathbb{R}_+^M} L(w,(\lambda_i)_{1\leq i\leq M}) = \begin{cases} E(w) & \text{, si } w \in C \\ +\infty & \text{, si } w \notin C \end{cases} \tag{2.17}$$

Ainsi, on obtient trivialement que les minimiseurs du problème (2.13) (ou de manière équivalente du problème (2.14)) sont les mêmes que les solutions du problème

$$\min_{w\in W} \Big(\max_{(\lambda_i)_{1\leq i\leq M}\in\mathbb{R}_+^M} L(w,(\lambda_i)_{1\leq i\leq M}) \Big). \tag{2.18}$$

Or, de par la construction du Lagrangien, celui-ci est convexe et différentiable en w (à $(\lambda_i)_{1\leq i\leq M}$ fixé) et affine (et donc concave et différentiable) en $(\lambda_i)_{1\leq i\leq M}$ (à w fixé). Un Lagrangien correspondant à un problème ayant suffisamment de coercivité et satisfaisant ces conditions [5] possède un point-selle $(w^*,(\lambda_i^*)_{1\leq i\leq M}) \in \big(W \times \mathbb{R}_+^M\big)$. Celui-ci vérifie donc [6]

$$\max_{(\lambda_i)_{1\leq i\leq M}\in\mathbb{R}_+^M} L(w^*,(\lambda_i)_{1\leq i\leq M}) = L(w^*,(\lambda_i^*)_{1\leq i\leq M}) = \min_{w\in W} L(w,(\lambda_i^*)_{1\leq i\leq M}). \tag{2.19}$$

Ceci entraine que, pour tout $w \in W$,

$$\begin{aligned} \max_{(\lambda_i)_{1\leq i\leq M}\in\mathbb{R}_+^M} L(w^*,(\lambda_i)_{1\leq i\leq M}) &= \min_{w'\in W} L(w',(\lambda_i^*)_{1\leq i\leq M}), \\ &\leq L(w,(\lambda_i^*)_{1\leq i\leq M}), \\ &\leq \max_{(\lambda_i)_{1\leq i\leq M}\in\mathbb{R}_+^M} L(w,(\lambda_i)_{1\leq i\leq M}). \end{aligned}$$

On en déduit donc que le premier argument w^* de ce point-selle est un minimiseur de (2.18) et donc un minimiseur de (2.13).

On a donc ramené la résolution de notre problème de minimisation sous contrainte (2.13) à la recherche du premier argument d'un point-selle d'une fonction définie sur tout $W \times \mathbb{R}_+^M$.

Pour trouver le point-selle, on remarque d'abord que, pour tout point-selle $(w^*,(\lambda_i^*)_{1\leq i\leq M}) \in \big(W \times \mathbb{R}_+^M\big)$ du Lagrangien L, on a forcément

$$\begin{aligned} \max_{(\lambda_i)_{1\leq i\leq M}\in\mathbb{R}_+^M} L(w^*,(\lambda_i)_{1\leq i\leq M}) &\geq \max_{(\lambda_i)_{1\leq i\leq M}\in\mathbb{R}_+^M} \Big(\min_{w\in W} L(w,(\lambda_i)_{1\leq i\leq M})\Big), \\ &\geq \min_{w\in W} L(w,(\lambda_i^*)_{1\leq i\leq M}), \\ &\geq \max_{(\lambda_i)_{1\leq i\leq M}\in\mathbb{R}_+^M} L(w^*,(\lambda_i)_{1\leq i\leq M}). \end{aligned}$$

5. Nous ne rentrerons pas dans les détails des conditions nécessaires à l'existence d'un point-selle.
6. Ceci est la définition d'un point-selle.

On a donc

$$\max_{(\lambda_i)_{1\leq i\leq M}\in\mathbb{R}_+^M} L(w^*,(\lambda_i)_{1\leq i\leq M}) = \max_{(\lambda_i)_{1\leq i\leq M}\in\mathbb{R}_+^M} \Big(\min_{w\in W} L(w,(\lambda_i)_{1\leq i\leq M})\Big). \tag{2.20}$$

On a d'ailleurs de même que

$$\min_{w\in W} L(w,(\lambda_i^*)_{1\leq i\leq M}) = \min_{w\in W} \Big(\max_{(\lambda_i)_{1\leq i\leq M}\in\mathbb{R}_+^M} L(w,(\lambda_i)_{1\leq i\leq M})\Big). \tag{2.21}$$

Il semble donc raisonnable à ce stade de calculer un minimiseur de (2.18) en calculant le second argument d'un point-selle. Pour cela, il suffit de calculer un maximiseur $(\lambda_i')_{1\leq i\leq M}\in\mathbb{R}_+^M$ de

$$\max_{(\lambda_i)_{1\leq i\leq M}\in\mathbb{R}_+^M} \Big(\min_{w\in W} L(w,(\lambda_i)_{1\leq i\leq M})\Big). \tag{2.22}$$

On appelle le problème ci-dessus le problème *dual* de (2.18).

Même si cela n'est pas essentiel, il n'est pas difficile de voir que tout maximiseur de (2.22) est le second argument d'un point-selle. On a, en effet, en utilisant (2.20)

$$\begin{aligned}
L(w^*,(\lambda_i')_{1\leq i\leq M}) &\geq \min_{w\in W} L(w,(\lambda_i')_{1\leq i\leq M}),\\
&\geq \max_{(\lambda_i)_{1\leq i\leq M}\in\mathbb{R}_+^M} \Big(\min_{w\in W} L(w,(\lambda_i)_{1\leq i\leq M})\Big),\\
&\geq \max_{(\lambda_i)_{1\leq i\leq M}\in\mathbb{R}_+^M} L(w^*,(\lambda_i)_{1\leq i\leq M}),\\
&\geq L(w^*,(\lambda_i')_{1\leq i\leq M}).
\end{aligned}$$

On en déduit donc que $(w^*,(\lambda_i')_{1\leq i\leq M})$ satisfait (2.19) et est donc un point-selle du Lagrangien.

La maximisation de (2.22) constitue le principe de l'algorithme de Uzawa. Plus précisément, si l'on note pour $(\lambda_i)_{1\leq i\leq M}\in\mathbb{R}_+^M$

$$\mathcal{L}\left((\lambda_i)_{1\leq i\leq M}\right) = \min_{w\in W} L(w,(\lambda_i)_{1\leq i\leq M}),$$

l'algorithme de Uzawa consiste à appliquer un algorithme de gradient avec projection avec un pas constant, pour la maximisation de $\mathcal{L}$ sur $\mathbb{R}_+^M$.

Pour cela, on admettra que

$$\nabla\mathcal{L}\left((\lambda_i)_{1\leq i\leq M}\right) = \Big(g_i(w_{(\lambda_i)_{1\leq i\leq M}})\Big)_{1\leq i\leq M},$$

où $w_{(\lambda_i)_{1\leq i\leq M}}$ est un minimiseur de

$$\min_{w\in W} L(w,(\lambda_i)_{1\leq i\leq M}). \tag{2.23}$$

On remarque aussi que la projection orthogonale sur $\mathbb{R}_+^M$ de $(\lambda_i)_{1\leq i\leq M}\in\mathbb{R}^M$ est simplement définie par

$$\lambda_i \leftarrow \max\big(0,\lambda_i\big),\quad \forall i\in\{1,\ldots,M\}.$$

On obtient alors l'algorithme décrit dans l'Algorithme 4.

Au-delà des arguments intuitifs décrits ci-dessus pour expliquer le principe de la construction de l'algorithme de Uzawa, nous ne rentrerons pas dans la justification rigoureuse de sa convergence. Celle-ci fait intervenir la caractérisation des solutions de (2.13) à l'aide des conditions de Karush-Kuhn-Tucker. Cette justification alourdirait (encore) considérablement la description de l'algorithme de Uzawa.

Par ailleurs, au regard de l'Algorithme 4, il n'est pas très difficile de se convaincre qu'il converge, tant que (2.23) a une solution. En effet, cet algorithme consiste à calculer un w minimisant L, pour un certain $(\lambda_i)_{1\leq i\leq M}$, puis à :

Algorithme 4 Algorithme de Uzawa

Entrée: Les entrées nécessaires au calcul de E, de ∇E et de g_i, pour tout $i \in \{1,\dots,M\}$
Sortie: Une approximation d'un minimiseur de (2.14) : w

Initialisation de w et de $(\lambda_i)_{1\leq i\leq M}$
Choix d'un pas t_0
Tant que l'algorithme n'a pas convergé **faire**
  Calculer un minimiseur w de (2.23) avec un algorithme adapté
  Mettre à jour, pour tout $i \in \{1,\dots,M\}$: $\lambda_i \leftarrow \max\big(0, \lambda_i + t_0 g_i(w)\big)$
Fin tant que

- augmenter les valeurs de λ_i pour les indices i ne satisfaisant pas la contrainte $g_i(w) \leq 0$: ainsi, on les pénalise de plus en plus au fur et à mesure du processus itératif jusqu'à ce que la contrainte soit satisfaite, comme le ferait un algorithme de pénalisation ;
- diminuer les λ_i pour les indices i satisfaisant la contrainte $g_i(w) \leq 0$: ainsi, on pénalise de moins en moins les contraintes satisfaites par notre solution w jusqu'à éventuellement ne plus les pénaliser du tout, du fait de la projection.

En pratique, cet algorithme peut être instable si le Lagrangien manque de coercivité en w. Cela peut, par exemple, arriver si les contraintes g_i sont linéaires et E a une croissance linéaire lorsque $\|w\|$ croit à l'infini.

Par ailleurs, le choix de l'algorithme permettant de minimiser (2.23) est surtout important lors des premières itérations. Ensuite, comme on peut l'initialiser avec le w précédent, il est donc très bien initialisé et le choix de l'algorithme n'a pas beaucoup d'importance.

Comme pour l'algorithme du gradient, une bonne initialisation de l'Algorithme 4 permet de gagner du temps. Cependant, une mauvaise initialisation ne remet pas en cause la convergence de l'algorithme.

Enfin, le choix du pas t_0 obéit aux même règles que dans un algorithme de descente (montée) à pas constant. Des éléments permettant de fixer ce pas sont donnés dans la Section 2.1.3 et le Théorème 1.

2.2.5 Principe des méthodes du Lagrangien augmenté

La méthode du Lagrangien augmenté peut être vue comme un compromis entre la méthode de pénalisation et la méthode de Uzawa. On définit le Lagrangien augmenté du problème (2.13) au point $w \in W$ et $(\lambda_i)_{1\leq i\leq M} \in \mathbb{R}_+^M$ par

$$L_\mu(w,(\lambda_i)_{1\leq i\leq M}) = E(w) + \sum_{i=1}^{M} \lambda_i\, g_i(w) + \mu \sum_{i=1}^{M} \big([g_i(w)]^+\big)^2,$$

où $\mu \geq 0$ et la fonction $[.]^+$ est définie par (2.15).

On utilise alors un algorithme similaire à l'algorithme de Uzawa décrit dans l'Algorithme 4, mais en remplaçant dans (2.23) le Lagrangien L par le Lagrangien augmenté L_μ. On fait aussi généralement augmenter μ au cours du processus itératif.

Il n'est pas difficile de se convaincre que cet algorithme converge vers une solution de (2.13). En effet, il peut-être vu comme un algorithme de Uzawa appliqué à la résolution du problème

$$\begin{cases} \min_{w\in W}\ E(w) + \mu \sum_{i=1}^{M} \big([g_i(w)]^+\big)^2, \\ \text{sous la contrainte } g_i(w) \leq 0 \quad \text{, pour tout } i \in \{1,\dots,M\}. \end{cases}$$

On obtient immédiatement que les solutions de ce problème sont les mêmes que celles de (2.13) car, pour tout $i \in \{1,\dots,M\}$ et tout w tel que $g_i(w) \leq 0$, on a $[g_i(w)]^+ = 0$.

On a ainsi deux façons d'imposer les contraintes :

- Soit, si μ est petit, l'algorithme impose les contraintes à la manière de l'algorithme de Uzawa.
- Soit, si μ est grand, l'algorithme impose les contraintes à la manière de l'algorithme de pénalisation.

Le paramètre μ sert donc à gérer le compromis entre les comportements de ces deux algorithmes. Typiquement, on peut stabiliser le mauvais comportement éventuel de l'algorithme de Uzawa en prenant un paramètre μ suffisamment grand. Comparer à la méthode de pénalisation, on utilise des valeurs de μ plus faibles, ce qui améliore généralement la structure de la fonctionnelle en lui évitant d'avoir des gradients trop forts et simplifie ainsi la phase de minimisation en w.

Chapitre 3

La restauration d'images

3.1 Introduction à la restauration d'images

La restauration d'images consiste à retrouver une image représentant au mieux la scène observée v, à partir d'une donnée mesurée $u \in \mathbb{R}^{N^2}$. Le plus souvent l'approximation de v est représentée par une image $w^* \in \mathbb{R}^{N'^2}$. L'entier N' est a priori différent de l'entier N. Cependant, lorsque cela n'est pas ambigu, nous considérerons pour simplifier que $N' = N$.

Pour simplifier et généraliser ce que nous avons dit dans le Chapitre 1, on modélise notre appareil de mesure par un opérateur (que, pour simplifier, nous supposons linéaire)

$$H : \mathbb{R}^{N'^2} \to \mathbb{R}^{N^2}$$

et nous modélisons l'imperfection de la mesure par un bruit blanc Gaussien $b \in \mathbb{R}^{N^2}$ d'écart type σ. Le problème de la restauration d'images consiste alors à chercher une image aussi proche que possible de v, à partir d'une mesure bruitée :

$$u = Hv + b.$$

Ce problème est évidemment plus ou moins difficile en fonction du conditionnement de l'opérateur H, voir de la dimension de son noyau et du niveau de bruit σ.

3.1.1 Un point de vue pragmatique

Les méthodes d'optimisation permettant de construire une telle approximation de v utiliseront toutes un *apriori* sur la régularité des images. Typiquement, on minimisera un critère de régularité convexe

$$R : \mathbb{R}^{N'^2} \to \mathbb{R}$$

sous la contrainte que l'image obtenue soit cohérente avec la donnée observée. Pour cela, on impose à

$$Hw^* - u$$

d'être conforme à ce que l'on attend du bruit b. La difficulté est que, pour se prêter à l'optimisation, il faut aussi être convexe. Par exemple (et c'est de loin le plus classique), on impose que $\|Hv - u\|_2 \simeq N\sigma$.

On formalise ceci en cherchant un minimiseur au problème

$$\begin{cases} \min_w R(w) \\ \text{sous la contrainte } \|Hw - u\|_2 \leq \tau \end{cases} \tag{3.1}$$

pour un paramètre τ proche de $N\sigma$.

En fait, pour éviter la difficulté pratique supplémentaire liée à la formulation d'un problème d'optimisation sous contrainte, on se contente souvent de prendre un minimiseur du problème

$$\min_{w\in\mathbb{R}^{N'^2}} R(w) + \lambda\|Hw-u\|_2^2. \tag{3.2}$$

On sait en effet, par des arguments de dualité Lagrangienne, que pour tout τ et toute solution w^* de (3.1), il existe un λ tel que w^* soit aussi solution de (3.2) pour ce λ. Inversement, pour tout λ et tout minimiseur w^* de (3.2), w^* est aussi solution du problème (3.1) pour $\tau = \|Hw^*-u\|_2$. Ainsi, considérer (3.1) ou (3.2) se résume à changer le paramètre τ par un paramètre λ, ce qui est souvent raisonnable.

Par ailleurs, on a vu dans les Chapitres 2.2.4 et 2.2.5 que la minimisation d'un problème non contraint du type (3.2) est en fait une étape de la résolution du problème contraint (3.1).

Dans la suite, nous décrirons la variation totale qui est un critère de régularité très utilisé pour résoudre différents problèmes de restauration d'images. Nous verrons aussi plusieurs modèles d'optimisation utilisant la variation totale pour résoudre différents problèmes de restauration d'images.

3.1.2 Un point de vue Bayesien

La modélisation Bayesienne fournit un bon moyen de modéliser des problèmes lorsque ceux-ci sont complexes. L'estimateur de *Maximum A Posteriori (MAP)* conduit alors à un problème d'optimisation dont la résolution fournit une estimation de la solution de notre problème inverse.

Dans le cas, relativement simple décrit ci-dessus, on procède comme suit.

Comme le bruit ajouté à l'image est supposé indépendant et identiquement distribué (i.i.d.) Gaussien centré d'écart type $\sigma > 0$, on a

$$\mathbb{P}(b) \propto e^{-\frac{\|b\|_2^2}{2\sigma^2}}.$$

On a donc, pour u obtenue à partir de l'image w

$$\mathbb{P}(u|w) \propto e^{-\frac{\|u-Hw\|_2^2}{2\sigma^2}}.$$

Par ailleurs, on suppose que les images suivent une loi, que l'on appelle loi à priori,

$$\mathbb{P}(w) \propto e^{-\lambda R(w)}.$$

On peut alors appliquer les loi de *Bayes* pour en déduire la loi à posteriori

$$\begin{aligned}\mathbb{P}(w|u) &= \frac{\mathbb{P}(u|w)\mathbb{P}(w)}{\mathbb{P}(u)},\\ &\propto \mathbb{P}(u|w)\mathbb{P}(w),\\ &\propto e^{-\frac{\|u-Hw\|_2^2}{2\sigma^2}} e^{-\lambda R(w)}.\end{aligned}$$

L'estimateur de Maximum a posteriori maximise cette loi à posteriori $\mathbb{P}(w|u)$. De manière équivalente, il minimise $-\log(\mathbb{P}(w|u))$. On retrouve alors un problème d'optimisation ressemblant aux problèmes vues à la section précédente :

$$\min_{w\in W} \frac{\|u-Hw\|_2^2}{2\sigma^2} + \lambda R(w).$$

La modélisation Bayesienne peut cependant être très pratique pour modéliser des problèmes dans lesquels un nombre important de variables interagissent. Cette modélisation peut aussi fournir des solutions pour estimer des paramètres.

3.1.3 Un point de vue "problème inverse"

Dans certains cas, les modèles d'optimisation peuvent être justifiés théoriquement par le calcul d'une borne supérieure de la norme de l'erreur $\|w^* - v\|$, pour une "famille" d'images v et pour w^* la solution d'un problème d'optimisation bien choisi. Nous verrons quelques exemples de résultats de ce type dans les Chapitres 5.1.3 et 5.2.2.

Lorsqu'un tel résultat est possible, il s'agit bien sûr de la meilleure garantie qui puisse être que la solution envisagée permet bien de résoudre le problème que l'on a.

3.2 La variation totale

3.2.1 Dans le domaine continu

Pour toute cette partie, le lecteur pourra trouver des compléments et des détails dans [4].

Pour simplifier, nous considérons dans cette sous-section une fonction w définie sur le carré $[0,N]^2$. Plus précisément, on suppose que $w \in L^1\left([0,N]^2\right)$. Nous définissons la variation totale par dualité : nous dirons que w est à *variation bornée* si

$$\sup\left\{\int_{[0,N]^2} w \ \mathrm{div}\,\varphi \ dxdy \ \ , \ \ \varphi \in C^1\left(]0,N[^2,\mathbb{R}^2\right) \text{ et } |\varphi| \leq 1\right\} < \infty,$$

où $C^1\left(]0,N[^2,\mathbb{R}^2\right)$ désigne l'ensemble des fonctions continûment dérivable allant de $]0,N[^2$ dans $\mathbb{R}^2$ et $|\varphi|$ désigne la norme euclidienne de φ en (x,y) (les dépendances en (x,y), n'est pas explicitée pour simplifier les notations).

On note $w \in BV\left([0,N]^2\right)$ pour indiquer qu'une fonction w est à variation bornée et on définit alors

$$TV(w) = \sup\left\{\int_{[0,N]^2} w \ \mathrm{div}\,\varphi \ dxdy \ \ , \ \ \varphi \in C^1\left(]0,N[^2,\mathbb{R}^2\right) \text{ et } |\varphi| \leq 1\right\}. \tag{3.3}$$

Exemple 1 : si $w \in BV\left([0,N]^2\right)$ et w est C^1 :

On obtient alors en intégrant par partie que, pour tout $\varphi = (\varphi_1,\varphi_2) \in C^1\left(]0,N[^2,\mathbb{R}^2\right)$, tel que $|\varphi| \leq 1$

$$\begin{aligned}
\int_{[0,N]^2} w \ \mathrm{div}\,\varphi \ dxdy &= \int_{[0,N]^2} w\left(\frac{\partial \varphi_1}{\partial x} + \frac{\partial \varphi_2}{\partial y}\right) dxdy, \\
&= -\int_{[0,N]^2} \frac{\partial w}{\partial x}\,\varphi_1 + \frac{\partial w}{\partial y}\,\varphi_2 \ dxdy, \\
&= -\int_{[0,N]^2} \nabla w \,.\, \varphi \ dxdy,
\end{aligned} \tag{3.4}$$

où $\nabla w \,.\, \varphi$ désigne le produit scalaire usuel dans $\mathbb{R}^2$. En remarquant maintenant que, comme $|\varphi| \leq 1$, on a

$$-\nabla w \,.\, \varphi \leq |\nabla w|,$$

on a finalement

$$\int_{[0,N]^2} w \ \mathrm{div}\,\varphi \ dxdy \leq \int_{[0,N]^2} |\nabla w| \ dxdy \qquad ,\forall \varphi \in C^1\left(]0,N[^2,\mathbb{R}^2\right) \text{ tel que } |\varphi| \leq 1.$$

Donc

$$TV(w) \leq \int_{[0,N]^2} |\nabla w| \ dxdy. \tag{3.5}$$

Pour prouver que l'on a en fait l'égalité entre ces deux termes, il suffirait de construire une suite de fonctions $\varphi_k \in C^1\left(]0,N[^2,\mathbb{R}^2\right)$ telles que $|\varphi_k| \leq 1$ approximant de mieux en mieux, lorsque k tend vers l'infini, la fonction

$$\vartheta = \begin{cases} -\frac{\nabla w}{|\nabla w|} & \text{, si } |\nabla w| \neq 0 \\ 0 & \text{, sinon.} \end{cases}$$

En effet, formellement, on a grâce à (3.4)

$$\int_{[0,N]^2} w \ \mathrm{div}\,\vartheta \ dxdy = -\int_{[0,N]^2} \nabla w \ . \ \vartheta \ dxdy = \int_{[0,N]^2} |\nabla w| dxdy.$$

Ainsi, si $(\varphi_k)_{k\in\mathbb{N}}$ est convenablement construite, on obtient que pour tout $\varepsilon > 0$, il existe k tel que

$$\int_{[0,N]^2} |\nabla w| \ dxdy - \varepsilon \leq \int_{[0,N]^2} w \ \mathrm{div}\,\varphi_k \ dxdy \leq TV(w).$$

Avec (3.5), ceci permet finalement de conclure que

$$TV(w) = \int_{[0,N]^2} |\nabla w| \ dxdy. \tag{3.6}$$

On peut en fait construire $(\varphi_k)_{k\in\mathbb{N}}$ assez facilement à l'aide de convolutions et multiplications de ϑ par des fonctions régulières. Les détails de cette construction sont cependant assez éloignés des objectifs de ce cours et sont laissés en exercice.

Exemple 2 : si w est une fonction indicatrice d'un ensemble régulier : formule de la co-aire

Nous ne le montrerons pas mais la preuve du résultat suivant est dans [4].

Théorème 4 *Si $E \subset [0,N]^2$ est une ensemble ouvert dont la frontière est régulière (par exemple Lipschitz) et si $w = \mathbf{1}_{|E}$ alors $w \in BV\left([0,N]^2\right)$ et*

$$TV(w) = \mathcal{H}^1(\partial E),$$

où $\mathcal{H}^1$ désigne la mesure de Hausdorff[1] *de dimension* 1.

En d'autres termes, $TV(w)$ est égale au périmètre de E. Ceci donne un exemple de fonction discontinue dont la variation totale est finie. Ceci montre l'intérêt de la définition de la variation totale par dualité. En effet, une définition basée sur une simple intégrale du module du gradient aurait contraint les fonctions à variation bornée à avoir un gradient.

Avant d'introduire la formule de la co-aire, nous donnons la définition suivante : Pour une fonction $w \in L^1\left([0,N]^2\right)$ et $t \in \mathbb{R}$, on note l'ensemble de niveau t de la fonction w :

$$\mathcal{L}_w(t) = \{(x,y) \in [0,N]^2, w(x,y) \geq t\}.$$

Théorème 5 (Formule de la co-aire) *Si $w \in BV\left([0,N]^2\right)$, alors*

- *pour presque tout $t \in \mathbb{R}$,*
$$TV\left(\mathbf{1}_{\mathcal{L}_w(t)}\right) < \infty;$$
- *la fonction*
$$t \longmapsto TV\left(\mathbf{1}_{\mathcal{L}_w(t)}\right) \qquad \textit{est mesurable ;}$$
- *et*
$$TV(w) = \int_{-\infty}^{\infty} TV\left(\mathbf{1}_{\mathcal{L}_w(t)}\right) dt. \tag{3.7}$$

Ce théorème est illustré sur la Figure 3.1.

1. La mesure de Hausedorff de dimension 1 d'un ensemble correspond à la longueur de cet ensemble s'il "est 1D" ; vaut 0 s'il est de dimension inférieure à 1 ; et vaut $+\infty$ s'il est de dimension supérieure à 1.

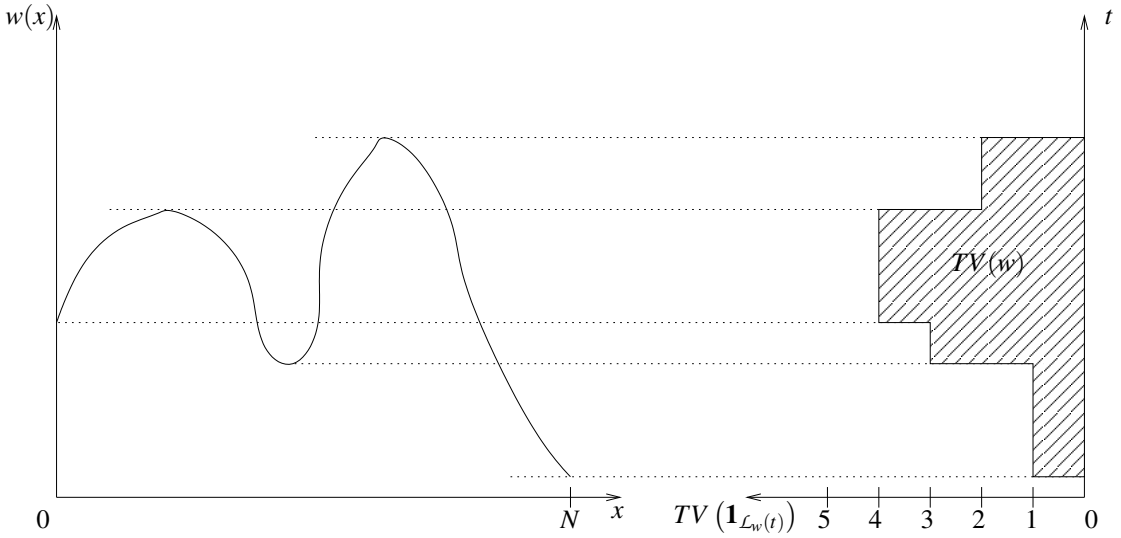


FIGURE 3.1 – Illustration de la formule de la co-aire en dimension 1: à gauche, le graphe d'une fonction $w \in TV([0,N])$; à droite, le graphe (tourné) de la fonction $t \longmapsto TV\left(\mathbf{1}_{\mathcal{L}_w(t)}\right)$.

Exemple 3 : en dimension 1, si w est croissante

Si $w \in BV([0,N])$ est croissante et C^1, en utilisant un analogue 1d de (3.6), on a

$$\begin{aligned} TV(w) &= \int_0^N |w'|\,dx \\ &= \int_0^N w'\,dx \\ &= w(N) - w(0). \end{aligned}$$

En fait, on peut facilement vérifier en utilisant la formule de la co-aire que ce résultat est encore vrai lorsque w contient des discontinuïtés. Ce résultat est illustré sur la Figure 3.2.

Ceci est un argument fort en faveur de l'utilisation de la variation totale comme critère de régularité en traitement d'images. Les images contiennent en effet souvent des discontinuïtés. En pratique, le fait que le coût d'un bord discontinu soit identique au coût d'un bord très régulier permet de laisser le terme d'attache aux données déterminer la régularité de la solution. Ce raisonnement est en fait un peu simpliste, mais justifie tout de même que les discontinuïtés ne sont pas impossibles.

Il faut finalement noter que cette qualité implique que la variation totale n'a pas de bonnes propriétés de type ellipticité, ce qui rend sa minimisation délicate. Pour mémoire, une fonctionnelle C^1 et elliptique f est telle qu'il existe $\alpha > 0$ tel que pour tout w' et w on ait

$$\left\langle \nabla f(w') - \nabla f(w), w' - w \right\rangle \geq \alpha \|w' - w\|_2.$$

On peut montrer qu'une telle fonctionnelle est strictement convexe.

On peut construire un contre-exemple simple, en considérant deux fonctions différentes w_1' et w_2', croissantes sur $[0,N]$ et telles que $w_1'(0) = w_2'(0) = 0$ et $w_1'(N) = w_2'(N) = 1$. Si l'on considère $w = \frac{w_1' + w_2'}{2}$, on voit facilement que $TV(w_1') = TV(w_2') = TV(w)$ dont on déduit que la variation totale n'est pas strictement convexe.

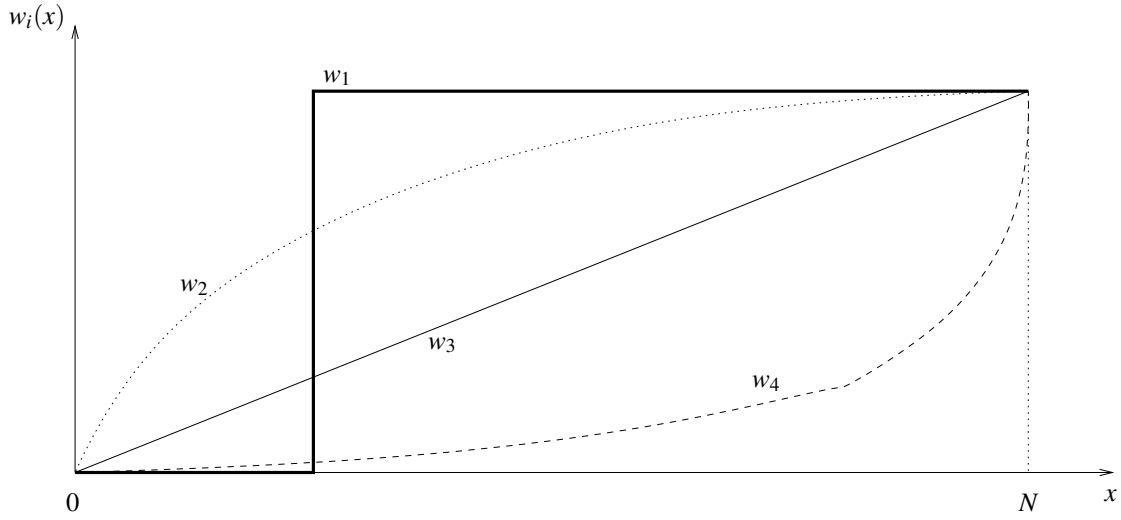


FIGURE 3.2 – Les fonctions w_1, w_2, w_3 et w_4 ont toutes la même variation totale.

3.2.2 Sur la grille

Définition et premières propriétés

Il existe plusieurs façons de définir la variation totale d'une image $w \in \mathbb{R}^{N^2}$. On pourrait notamment discrétiser la variation totale en utilisant la formule de la co-aire et une notion de périmètre de sous-ensembles de $\{1,\dots,N\}^2$ (voir (3.7)). On pourrait aussi utiliser la définition utilisant la dualité entre fonctions définies sur $\{0,\dots,N\}^2$ (voir (3.3)). Nous choisirons ici de la définir en utilisant un analogue discret de (3.6).

On définit ainsi la variation totale de $w \in \mathbb{R}^{N^2}$ par

$$TV(w) = \sum_{m,n=1}^{N} |\nabla w_{m,n}|, \tag{3.8}$$

où $|.|$ est la norme euclidienne dans $\mathbb{R}^2$,

$$\nabla w_{m,n} = \begin{pmatrix} \partial_m w_{m,n} \\ \partial_n w_{m,n} \end{pmatrix} = \begin{pmatrix} w_{m+1,n} - w_{m,n} \\ w_{m,n+1} - w_{m,n}, \end{pmatrix} \tag{3.9}$$

et où l'on étend w en dehors de son support (par exemple) par périodisation.

On note ci-dessus

$$\begin{aligned} \partial_m : \mathbb{R}^{N^2} &\longrightarrow \mathbb{R}^{N^2}, \\ (w_{m,n})_{1\le m,n\le N} &\longmapsto (w_{m+1,n} - w_{m,n})_{1\le m,n\le N}, \end{aligned}$$

et

$$\begin{aligned} \partial_n : \mathbb{R}^{N^2} &\longrightarrow \mathbb{R}^{N^2}, \\ (w_{m,n})_{1\le m,n\le N} &\longmapsto (w_{m,n+1} - w_{m,n})_{1\le m,n\le N}. \end{aligned}$$

Il n'est pas difficile de voir que la variation totale ainsi définie est :

- **continue**, car elle est obtenue comme une somme de fonctions qui sont la composition de fonctions continues.
- **convexe**, car elle est obtenue comme une somme de fonctions qui sont trivialement convexes.
- **non-coercive** sur $\mathbb{R}^{N^2}$, car pour tout $w \in \mathbb{R}^{N^2}$ et tout $c \in \mathbb{R}$, $TV(w+c) = TV(w)$. Ainsi, on peut choisir c de manière à avoir $\|w+c\|$ aussi grand que l'on veut sans que $TV(w+c)$ ne tende vers l'infini. Ceci peut poser problème lorsque l'on cherchera à minimiser la variation totale.
- **est une semi-norme** sur $\mathbb{R}^{N^2}$. Ceci est laissé en exercice.
- **est une norme** sur $\{w \in \mathbb{R}^{N^2}, \sum_{m,n=1}^{N} w_{m,n} = 0\}$. Il n'est pas difficile de voir que si $w \in \mathbb{R}^{N^2}$ est telle que $TV(w) = 0$, alors w est constante [2]. Ainsi, si w vérifie aussi $\sum_{m,n=1}^{N} w_{m,n} = 0$, on a forcément $w \equiv 0$.
  Ces deux dernières propriétés garantissent en pratique que les problèmes d'optimisation convexe faisant intervenir la variation totale n'auront pas de solution de norme arbitrairement grande si et seulement si un autre terme (par exemple : le terme d'attache aux données) contraint la moyenne du minimum . Ce sera en pratique toujours le cas.
- **non-différentiable**, dès qu'il existe $(m,n) \in \{1,\dots,N\}^2$ tel que $|\nabla w_{m,n}| = 0$, car $|.|$ n'est pas différentiable en $(0,0)$. Ceci est à prendre en compte lors du choix d'algorithme pour minimiser une fonctionnelle contenant un terme de variation totale. Une modification simple permettant de contourner ce problème consiste à remplacer la variation totale par

$$TV_\varepsilon(w) = \sum_{m,n=1}^{N} \sqrt{(\partial_m w_{m,n})^2 + (\partial_n w_{m,n})^2 + \varepsilon},$$

  où $\varepsilon > 0$ est petit.

Calcul du gradient de TV_ε

Pour simplifier les calculs, on note

$$\varphi_\varepsilon(t) = \sqrt{t+\varepsilon} \quad ,\forall t > 0.$$

On a évidemment

$$\varphi'_\varepsilon(t) = \frac{1}{2\sqrt{t+\varepsilon}} \quad ,\forall t > 0. \tag{3.10}$$

Nous allons, tout d'abord, calculer l'adjoint ∂_m^* de ∂_m. Pour cela, nous considérons w et w' dans $\mathbb{R}^{N^2}$. Nous avons

$$\begin{aligned}
\langle w, \partial_m w' \rangle &= \sum_{m,n=1}^{N} w_{m,n} \left(w'_{m+1,n} - w'_{m,n} \right), \\
&= \sum_{m,n=1}^{N} w_{m,n} w'_{m+1,n} - \sum_{m,n=1}^{N} w_{m,n} w'_{m,n}, \\
&= \sum_{m,n=1}^{N} w_{m-1,n} w'_{m,n} - \sum_{m,n=1}^{N} w_{m,n} w'_{m,n}, \\
&= \sum_{m,n=1}^{N} \left(w_{m-1,n} - w_{m,n} \right) w'_{m,n}, \\
&= \langle \partial_m^* w, w' \rangle.
\end{aligned}$$

2. On peut le montrer simplement en faisant deux récurrences : une pour montrer que pour tout m, $w_{m,1} = w_{1,1}$, une autre pour montrer qu'à m fixé, pour tout n, $w_{m,n} = w_{m,1}$.

On a donc pour tout $w \in \mathbb{R}^{N^2}$

$$\partial_m^* w_{m,n} = w_{m-1,n} - w_{m,n} \quad ,\forall (m,n) \in \{1,\dots,N\}^2. \tag{3.11}$$

On obtient de même que l'adjoint ∂_n^* de ∂_n satisfait pour tout $w \in \mathbb{R}^{N^2}$

$$\partial_n^* w_{m,n} = w_{m,n-1} - w_{m,n} \quad ,\forall (m,n) \in \{1,\dots,N\}^2. \tag{3.12}$$

Nous pouvons maintenant commencer le calcul du gradient de TV_ε. Considérons w et w' dans $\mathbb{R}^{N^2}$. Nous avons

$$\begin{aligned} TV_\varepsilon(w+w') - TV_\varepsilon(w) &= \sum_{m,n=1}^{N} \varphi_\varepsilon\Big(|\nabla(w+w')_{m,n}|^2\Big) - \varphi_\varepsilon\Big(|\nabla w_{m,n}|^2\Big), \\ &= \sum_{m,n=1}^{N} \varphi_\varepsilon\Big(|\nabla w_{m,n}|^2 + 2(\partial_m w_{m,n}\partial_m w'_{m,n} + \partial_n w_{m,n}\partial_n w'_{m,n}) + o(|\nabla w'_{m,n}|)\Big) - \varphi_\varepsilon\Big(|\nabla w_{m,n}|^2\Big), \\ &= \sum_{m,n=1}^{N} 2\,\varphi'_\varepsilon\Big(|\nabla w_{m,n}|^2\Big)(\partial_m w_{m,n}\partial_m w'_{m,n} + \partial_n w_{m,n}\partial_n w'_{m,n}) + o(|\nabla w'_{m,n}|). \end{aligned}$$

Si l'on note $X \in \mathbb{R}^{N^2}$ et $Y \in \mathbb{R}^{N^2}$ tels que pour tout $(m,n) \in \{1,\dots,N\}^2$ on ait

$$X_{m,n} = 2\,\varphi'_\varepsilon\Big(|\nabla w_{m,n}|^2\Big)\,\partial_m w_{m,n} \quad \text{et} \quad Y_{m,n} = 2\,\varphi'_\varepsilon\Big(|\nabla w_{m,n}|^2\Big)\,\partial_n w_{m,n}, \tag{3.13}$$

on obtient finalement

$$\begin{aligned} TV_\varepsilon(w+w') - TV_\varepsilon(w) &= \sum_{m,n=1}^{N} X_{m,n}\partial_m w'_{m,n} + \sum_{m,n=1}^{N} Y_{m,n}\partial_n w'_{m,n} + \sum_{m,n=1}^{N} o(|\nabla w'_{m,n}|) \\ &= \big\langle X, \partial_m w' \big\rangle + \big\langle Y, \partial_n w' \big\rangle + o(\|\nabla w'\|_1), \\ &= \big\langle \partial_m^* X + \partial_n^* Y, w' \big\rangle + o(\|w'\|_2). \end{aligned}$$

Ci-dessus, on voit facilement qu'un terme négligeable devant $\|\nabla w'\|_1$ est aussi négligeable devant $\|w'\|_2$ car

$$\sum_{m,n=1}^{N} |\nabla w'_{m,n}| \le 4 \sum_{m,n=1}^{N} |w'_{m,n}| = 4\big\langle w', signe(w')\big\rangle \le 4N\|w'\|_2,$$

où $signe(w')$ est un élément de $\mathbb{R}^{N^2}$ ne contenant que des 1 et des -1, suivant le signe de w'.

On peut donc conclure que, pour tout $w \in \mathbb{R}^{N^2}$,

$$\nabla TV_\varepsilon(w) = \partial_m^* X + \partial_n^* Y, \tag{3.14}$$

où X et Y sont définis par (3.13).

On remarque que la formule ci-dessus correspond à une divergence discrète de $(X_{m,n}, Y_{m,n})_{1\le m,n\le N}$. La divergence discrète est en fait l'opérateur adjoint de ∇. On voit aussi en utilisant (3.10) dans (3.13) que lorsque ε tend vers 0, on a

$$X_{m,n} \simeq \frac{\partial_m w_{m,n}}{|\nabla w_{m,n}|} \quad \text{et} \quad Y_{m,n} \simeq \frac{\partial_n w_{m,n}}{|\nabla w_{m,n}|}.$$

On a ainsi

$$\nabla TV_\varepsilon(w) \simeq \operatorname{div}\left(\frac{\nabla w}{|\nabla w|}\right). \tag{3.15}$$

Dans (3.15), le terme de droite correspond à une écriture que l'on rencontre couramment pour définir le gradient de la variation totale.

3.3 Le débruitage

Comme nous l'avons vu au Chapitre 1, les images contiennent généralement un bruit que l'on modélise souvent en le supposant additif, blanc et Gaussien. On a ainsi en chaque $(m,n) \in \{1,\dots,N\}^2$,

$$u_{m,n} = v_{m,n} + b_{m,n},$$

où u est l'image dont on dispose, v l'image que l'on aimerait connaitre et b est un bruit blanc Gaussien.

Comme nous l'avons déjà indiqué dans la Section 3.1, une solution pour retrouver une image débruitée proche de v consiste à trouver un minimiseur du problème

$$\min_{w \in \mathbb{R}^{N^2}} \; E(w), \tag{3.16}$$

où

$$E(w) \;=\; TV_\varepsilon(w) + \lambda \|w-u\|_2^2. \tag{3.17}$$

Pour minimiser E, on utilise par exemple un algorithme de gradient tel que nous les avons vu dans le Chapitre 2 et utilisant [3]

$$\nabla E(w) = \nabla TV_\varepsilon(w) + 2\lambda(w-u),$$

où $\nabla TV_\varepsilon(w)$ est donné dans (3.14), (3.11), (3.12) et (3.13).

La nature des résultats que l'on peut obtenir avec cet algorithme est illustrée sur la Figure 3.3. On présente sur cette figure l'image non bruitée, des images bruitées pour des bruits d'écart type 10, 20 et 30 ainsi que les images solution de (3.16) correspondantes. Dans tous les cas, on a pris $\varepsilon = 1$ et on a pris $\lambda = 0.05$, 0.017 et 0.0125 respectivement pour les bruits d'écart type 10, 20 et 30.

3. Le calcul conduisant à ∇E ne présente pas de difficulté et est laissé en exercice.

FIGURE 3.3 – En haut, l'image souhaitée puis de haut an bas: à gauche l'image bruitée et à droite l'image débruitée pour des bruits d'écart type 10, 20, 30.

Sur la Figure 3.4, on montre les résultats pour débruiter l'image contenant un bruit d'écart type 20, avec $\varepsilon = 1$ et $\lambda = 0.5$, 0.05, 0.017 et 0.005. On voit que, plus λ est grand, plus l'image est bruitée. Inversement, plus λ est petit, plus l'image est lisse. Ceci correspond au fait que, dans (3.17), lorsque λ est petit, le terme de régularisation TV_ε a plus d'influence sur l'énergie et donc sur la position de son minimiseur. La situation inverse se produit lorsque λ est grand.

FIGURE 3.4 – En haut, à gauche l'image souhaitée et à droite l'image bruitée avec un bruit d'écart type 20. Puis, de haut en bas et de gauche à droite, l'image débruitée pour $\lambda = 0.5$, 0.05, 0.017 et 0.005.

Enfin, on remarque sur la Figure 3.4 que même lorsque l'image est trop régularisée (i.e. pour des valeurs de λ très faible), le résultat contient des discontinuïtés. C'est une illustration, dans un cas extrême, de l'intérêt de la variation totale en traitement d'images. De telles discontinuïtés sont en effet souvent présentes dans des images.

3.4 L'inversion d'opérateurs/la déconvolution

Comme nous l'avons vu au Chapitre 1, en plus du bruit, les images sont généralement acquises par un appareil de mesure qui restitue imparfaitement le contenu de l'image idéale que l'on voudrait obtenir. Ainsi, on mesure en chaque pixel $(m,n) \in \{1,\dots,N\}^2$,

$$u_{m,n} = Hv_{m,n} + b_{m,n},$$

où u est l'image dont on dispose, v l'image que l'on aimerait connaitre, H est un opérateur linéaire connu et b est un bruit blanc Gaussien. Pour simplifier, nous supposerons que H va de $\mathbb{R}^{N^2}$ dans $\mathbb{R}^{N^2}$ mais, comme nous l'avons vu au Chapitre 3.1, il est parfois utile de supposer que H va de $\mathbb{R}^{N'^2}$ dans $\mathbb{R}^{N^2}$, avec $N' \neq N$. Lorsque H est invariant par translation, il s'agit d'une convolution (voir Chapitre 1). L'opération consistant à estimer v à partir de u s'appelle dans ce cas une déconvolution. En général, H dépend de l'appareil de mesure et peut prendre plusieurs formes. Il est souvent très mal conditionné et a même souvent un nombre important de valeurs propres valant 0.

Là encore (voir la Section 3.1), une solution pour retrouver une image proche de v consiste à trouver un minimiseur du problème

$$\min_{w \in \mathbb{R}^{N^2}} \; E(w), \tag{3.18}$$

où

$$E(w) \;=\; TV_\varepsilon(w) + \lambda \|Hw - u\|_2^2. \tag{3.19}$$

Comme dans la section précédente, on peut utiliser pour résoudre ce problème un algorithme de gradient tel que nous les avons vus dans le Chapitre 2 et utilisant [4]

$$\nabla E(w) = \nabla TV_\varepsilon(w) + 2\lambda H^t(Hw - u),$$

où H^t est l'opérateur adjoint de H et $\nabla TV_\varepsilon(w)$ est donné dans (3.14), (3.11), (3.12) et (3.13).

On présente sur la Figure 3.5, les résultats d'un problème de déconvolution. Dans cette expérience, l'opérateur H correspond à une convolution avec un noyau Gaussien de variance connue et le bruit additif blanc Gaussien est d'écart type 2. On montre l'image idéale recherchée, l'image floue et les minimiseurs de E pour différentes valeurs de λ. On voit que lorsque λ est grand, l'image contient beaucoup de détails mais contient aussi du bruit, alors que pour un lambda faible, l'image contient moins de détails mais est bien régulière. Des valeurs de λ intermédiaires permettent d'obtenir un compromis entre ces deux extrêmes.

4. Là encore, le calcul conduisant à ∇E ne présente pas de difficulté et est laissé en exercice.

FIGURE 3.5 – Expérience sur la déconvolution. De haut en bas, puis de gauche à droite: l'image idéale que l'on recherche, l'image floue puis les images restaurées pour des valeurs de $\lambda = 100$, 10 et 0.01.

On présente sur la Figure 3.6, les résultats d'un problème d'interpolation. Dans cette expérience, on ne connait qu'une version bruitée (il s'agit d'un bruit additif blanc Gaussien d'écart type 10) de certains pixels de l'image. Ceci correspond à un opérateur H défini comme une multiplication par un masque (connu) valant 1 pour les pixels connus et 0 pour les pixels inconnus. Remarquez que l'on suppose ici que les bords de l'image sont inconnus. C'est un artifice que l'on emploie parfois pour gérer les problèmes de bords. Nous profitons de cette expérience pour l'illustrer. On voit sur la Figure 3.6, l'image recherchée, l'image dans laquelle les pixels manquants sont en noir puis les minimiseurs de E pour des valeurs décroissantes de λ. Là encore, lorsque λ diminue, l'image obtenue est plus régulière mais contient moins de détails.

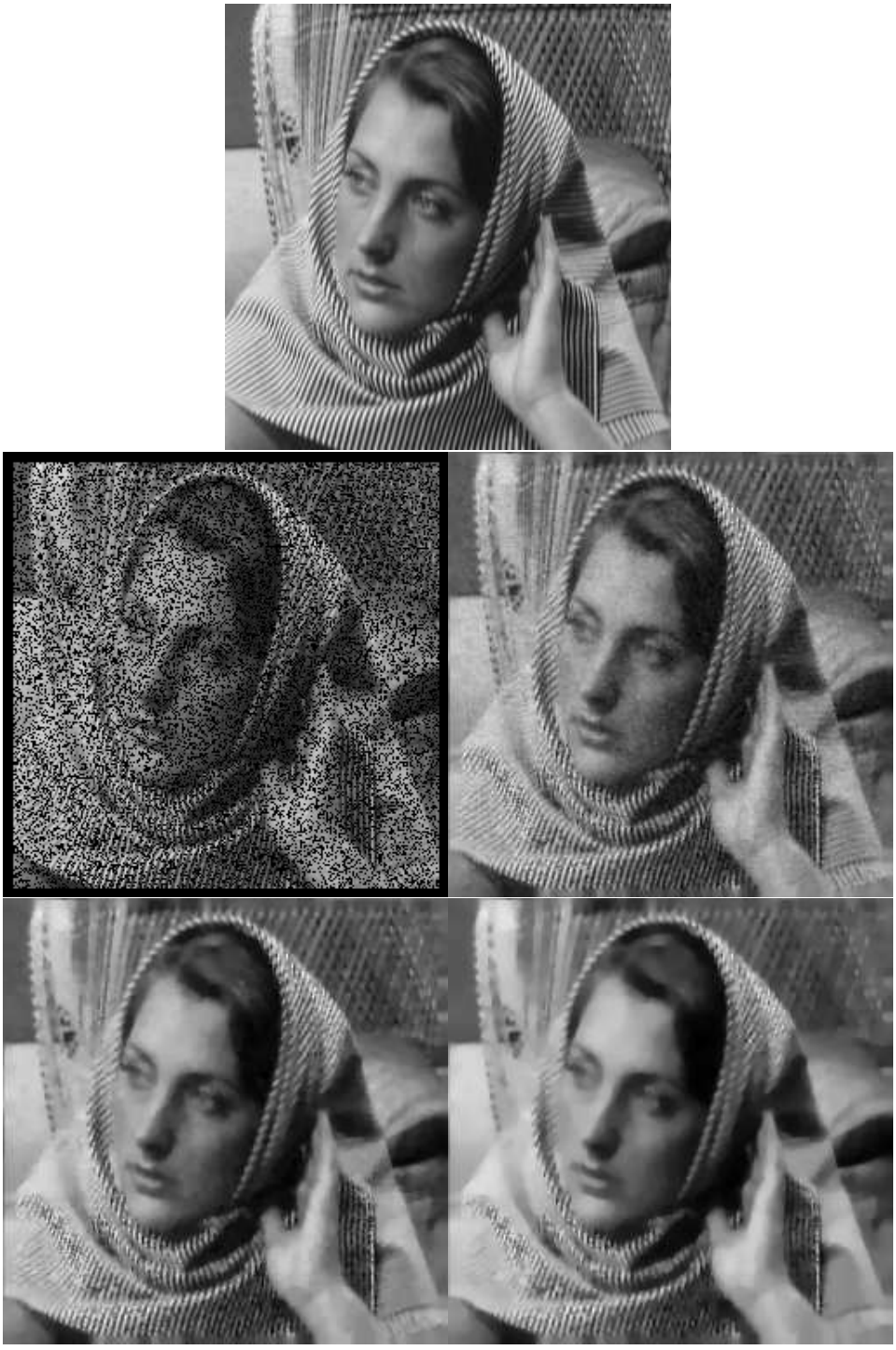


FIGURE 3.6 – Expérience sur l'interpolation et le débruitage. De haut en bas, puis de gauche à droite: l'image idéale que l'on recherche, l'image à interpoler puis les images restaurées pour des valeurs décroissantes de λ.

Nous présentons sur la Figure 3.7, les résultats pour un problème de zoom. Dans cette expérience, l'opérateur H correspond à la succession d'une convolution et d'un échantillonnage. (Ici, on a typiquement $N' > N$.) Dans cette expérience, le bruit est d'écart type 1. On présente sur la Figure 3.7, l'image de départ, l'image obtenue en répliquant la valeur des pixels, l'image obtenue en minimisant E pour $\lambda = 10$. Notez que, dans cette expérience, nous n'avons pas traité les problèmes de bord. On peut voir des oscillations qui se propagent assez loin des bords de l'image. On voit sur cette expérience que pour ce genre d'application, il est très important de bien gérer cet aspect. Par contre, on voit que l'image ne contient pas la "pixellisation" que l'on peut observer sur le zoom par réplication.

FIGURE 3.7 – Expérience sur le zoom de niveau 4 d'images. De haut en bas: l'image de départ; l'image obtenue en répliquant les valeurs des pixels; l'image obtenue en régularisant avec la variation totale.

3.5 La restauration d'image compressée

L'objectif de cette section n'est évidemment pas de donner un exposé complet des algorithmes de compression d'images tels que JPEG et JPEG 2000. Ce qu'il nous suffit de savoir, c'est que ces deux algorithmes :

- transforment l'image à compresser u sous la forme d'un jeu de coordonnées $(\langle u,\varphi\rangle)_{\varphi\in\mathcal{B}}$, où $\mathcal{B}$ est une base orthogonale de $\mathbb{R}^{N^2}$;
- quantifie ces coordonnées pour obtenir $\left(Q_\varphi(\langle u,\varphi\rangle)\right)_{\varphi\in\mathcal{B}}$ où, pour tout $\varphi\in\mathcal{B}$, $Q_\varphi:\mathbb{R}\longrightarrow\mathbb{Z}$ est une quantification ;
- puis font un codage sans perte du résultat.

Ainsi, si l'on note q_φ le pas de quantification de la coordonnée suivant φ, on sait que

$$q_\varphi\left(Q_\varphi(\langle u,\varphi\rangle)-\frac{1}{2}\right)\leq\langle u,\varphi\rangle\leq q_\varphi\left(Q_\varphi(\langle u,\varphi\rangle)+\frac{1}{2}\right),\quad,\forall\varphi\in\mathcal{B} \tag{3.20}$$

où u est l'image de départ et $\left(Q_\varphi(\langle u,\varphi\rangle)\right)_{\varphi\in\mathcal{B}}$ représente ses coordonnées quantifiées. Pour restaurer l'image compressée, on va donc chercher le "meilleur" élément de $\mathbb{R}^{N^2}$ satisfaisant (3.20).

Finalement, une façon naturelle de restaurer une image compressée u, exprimée sous la forme de ses coordonnées quantifiées $\left(Q_\varphi(\langle u,\varphi\rangle)\right)_{\varphi\in\mathcal{B}}$, consiste à chercher une solution du problème suivant :

$$\begin{cases}\min TV_\varepsilon(w)\\ \text{sous les contraintes } q_\varphi\left(Q_\varphi(\langle u,\varphi\rangle)-\frac{1}{2}\right)\leq\langle w,\varphi\rangle\leq q_\varphi\left(Q_\varphi(\langle u,\varphi\rangle)+\frac{1}{2}\right)\quad,\forall\varphi\in\mathcal{B}.\end{cases} \tag{3.21}$$

Il faut noter que l'ensemble C des $w\in\mathbb{R}^{N^2}$ satisfaisant les contraintes de (3.21) a une forme d'hypercube dont les côtés sont de tailles différentes. Il est donc convexe mais son bord est non-différentiable. La projection orthogonale sur cet ensemble est par ailleurs facile, puisque la base $\mathcal{B}$ est orthogonale.

Si l'on note $P_C(w)$ la projection orthogonale de $w\in\mathbb{R}^{N^2}$, sur C, on a immédiatement

$$\langle P_C(w),\varphi\rangle=\begin{cases}q_\varphi^+ & ,\text{ si }\langle w,\varphi\rangle\geq q_\varphi^+,\\ \langle w,\varphi\rangle & ,\text{ si }q_\varphi^+\geq\langle w,\varphi\rangle\geq q_\varphi^-,\\ q_\varphi^- & ,\text{ si }q_\varphi^-\geq\langle w,\varphi\rangle,\end{cases}$$

où l'on a noté

$$q_\varphi^+=q_\varphi\left(Q_\varphi(\langle u,\varphi\rangle)+\frac{1}{2}\right),$$

et

$$q_\varphi^-=q_\varphi\left(Q_\varphi(\langle u,\varphi\rangle)-\frac{1}{2}\right).$$

On peut donc (par exemple) construire un algorithme de gradient utilisant la formule du gradient de TV_ε décrite dans (3.14), (3.11), (3.12) et (3.13) et la forme de la projection ci-dessus pour construire un algorithme de gradient résolvant (3.21).

On illustre ce modèle sur la Figure 3.8. Sur cette figure, on présente différents niveaux de compression, avec pour chaque image compressée, l'image restaurée correspondante. L'image restaurée est obtenue en résolvant (3.21).

FIGURE 3.8 – Exemples d'images dans lesquelles les défauts dus à la compression ont été corrigés. A gauche: les images compressées; à droite: les images restaurées. De haut en bas: l'expérience est faite pour un taux de compression de plus en plus important.

Chapitre 4

Le recalage

4.1 Principe des méthodes d'optimisation pour le recalage d'images

Le but du *recalage* d'images ("*registration*" ou "*optical flow*", en anglais) est d'établir une correspondance entre les points représentant un même objet dans deux images d'une même scène. De telles images se rencontrent fréquemment (deux images successives dans un film, images satellite d'une même région ...). On peut aussi vouloir comparer une image à une image de référence (en imagerie médicale, notamment).

Plus précisément, on dispose de deux images v^1 et $v^2 \in \mathbb{R}^{N^2}$ et on cherche un champ de vecteurs $t = (t_{m,n})_{1\leq m,n\leq N} = (t^1_{m,n}, t^2_{m,n})_{1\leq m,n\leq N} \in (\mathbb{R}^2)^{N^2}$ tel que l'objet représenté dans v^1 en $(m,n)+t_{m,n}$ soit le même que l'objet représenté dans v^2 au pixel (m,n).

Le recalage d'images est l'un des grands domaines du traitement d'images et de la vision par ordinateur. Il existe un nombre très important de méthodes pour résoudre ce problème. Les méthodes d'optimisation consistent généralement à minimiser une énergie définie pour $t \in (\mathbb{R}^2)^{N^2}$ et qui est la somme

- d'un terme favorisant la cohérence entre le niveau de gris observé dans v^1 au point $(m,n)+t_{m,n}$ et $v^2_{m,n}$ et
- d'un terme de régularisation de t (ou même une contrainte sur t).

La première difficulté vient du terme d'attache aux données entre v^1 au point $(m,n)+t_{m,n}$ et $v^2_{m,n}$. Ce terme pose deux problèmes :

- On doit en effet définir v^1 en un point non entier et donc interpoler v^1. Dans la suite, on appelle $\widetilde{v^1}$ une version de v^1 interpolée sur tout $[0,N]^2$. On peut par exemple penser aux solutions suivantes :
  - L'interpolation au plus proche voisin : on définit alors

$$\widetilde{v^1}\big((m,n)+t_{m,n}\big) = v^1(\lfloor (m,n)+t_{m,n} \rfloor),$$

    où la fonction $\lfloor . \rfloor$ fait un arrondi à l'entier le plus proche. Cette interpolation conduit cependant à des fonctionnelles non convexes et non-différentiables (sa dérivée vaut 0 presque partout).
  - Une façon simple d'obtenir une fonctionnelle convexe et différentiable (en t) est de considérer une approximation affine de v^1 au voisinage de (m,n). On pose alors

$$\widetilde{v^1}\big((m,n)+t_{m,n}\big) = v^1_{m,n} + \langle \nabla v^1_{m,n}, t_{m,n} \rangle. \tag{4.1}$$

    Une telle approximation n'est cependant généralement valable que dans un petit voisinage autour de (m,n). Cette approximation sera particulièrement grossière pour les points situés sur un contour de l'image.

- On peut aussi utiliser une interpolation linéaire utilisant les valeurs des pixels dans un voisinage de $(m,n)+t_{m,n}$. On peut, par exemple choisir un noyau $h \in L^1([0,N]^2)$ (périodisé) et poser

$$\widetilde{v^1}(x,y) = \sum_{m,n=1}^{N} h(x-m,y-n)\, v^1_{m,n}.$$

  Là encore, ce choix conduira à des fonctionnelles non convexes en t.

– On doit définir un critère de similarité entre $\widetilde{v^1}\big((m,n)+t_{m,n}\big)$ et $v^2_{m,n}$. On utilise souvent
  - un critère de ressemblance supposant que la couleur du pixel correspondant à un même objet est identique dans les deux images. On utilise alors typiquement :

$$\sum_{m,n=1}^{N} \Big(\widetilde{v^1}\big((m,n)+t_{m,n}\big) - v^2_{m,n}\Big)^2. \tag{4.2}$$

  - Une variante importante est :

$$\sum_{m,n=1}^{N} \Big|\widetilde{v^1}\big((m,n)+t_{m,n}\big) - v^2_{m,n}\Big|.$$

    Elle permet de moins tenir compte des changements d'occlusions entre les deux images. Une occlusion correspond à un objet uniquement présent dans l'une des deux images. Les pixels de cet objet sont généralement peu nombreux, mais chacun de ces pixels conduit à une forte valeur de $\Big|\widetilde{v^1}\big((m,n)+t_{m,n}\big) - v^2_{m,n}\Big|$. Par rapport à (4.2), l'utilisation d'un terme l^1 permet d'atténuer leur influence dans le terme d'attache aux données, tout en restant convexe.
  - Le contraste peut cependant avoir changé entre les deux prises de vues. Ce sera par exemple le cas si la lumière a changé. Une façon d'atténuer ce défaut est de chercher à recaler les gradients des images. Intuitivement, on cherche alors à mettre les bords des objets en correspondance. On prend alors

$$\sum_{m,n=1}^{N} \Big(\nabla\widetilde{v^1}\big((m,n)+t_{m,n}\big) - \nabla v^2_{m,n}\Big)^2.$$

  - On cherche même parfois (notamment en imagerie médicale) à recaler des images acquises à l'aide de capteurs différents et ne permettant pas de voir les mêmes objets. On utilise alors généralement un critère utilisant l'information mutuelle. On minimise alors un terme ayant la forme

$$I(\widetilde{v^1}, v^2) = -\sum_{a,b} \mathbb{P}_{\widetilde{v^1},v^2}(a,b) \log \frac{\mathbb{P}_{\widetilde{v^1},v^2}(a,b)}{\mathbb{P}_{\widetilde{v^1}}(a)\mathbb{P}_{v^2}(b)}, \tag{4.3}$$

    où

$$\mathbb{P}_{\widetilde{v^1},v^2}(a,b) = \mathbb{P}\Big(\widetilde{v^1}\big((m,n)+t_{m,n}\big) = a \text{ et } v^2_{m,n} = b\Big)$$

    est la probabilité (sur (m,n)) jointe de $\widetilde{v^1}$ et v^2 et

$$\mathbb{P}_{\widetilde{v^1}}(a) = \mathbb{P}\Big(\widetilde{v^1}\big((m,n)+t_{m,n}\big) = a\Big) \quad \text{et} \quad \mathbb{P}_{v^2}(b) = \mathbb{P}\big(v^2_{m,n} = b\big).$$

  Le calcul des lois de probabilité se fait typiquement à l'aide d'histogrammes normalisés. Il existe évidemment toute une gamme de solutions pour leur estimation.
  Là encore, cela conduit à des fonctionnelles non convexes et à de nombreux problèmes numériques.

exemple : Si v^1 se déduit de v^2 par un changement de contraste g (i.e. : $\forall(m,n), v^1_{m,n} = g(v^2_{m,n})$), on a, pour $t \equiv 0$

$$\mathbb{P}_{\widetilde{v^1}}(g(b)) = \mathbb{P}_{v^2}(b)$$

et

$$\mathbb{P}_{\widetilde{v^1},v^2}(a,b) = \begin{cases} \mathbb{P}_{v^2}(b) & \text{, si } a = g(b) \\ 0 & \text{, sinon.} \end{cases}$$

On a donc

$$\begin{aligned} I(\widetilde{v^1}, v^2) &= -\sum_b \mathbb{P}_{v^2}(b) \log \frac{\mathbb{P}_{v^2}(b)}{\mathbb{P}_{v^2}(b)^2}, \\ &= \sum_b \mathbb{P}_{v^2}(b) \log\, \mathbb{P}_{v^2}(b). \end{aligned}$$

– Concernant les critères de régularisation sur le champ de vecteur t, on distingue typiquement les approches suivantes :
  - On impose à t d'appartenir à une classe de déformation dont les éléments sont définis par un nombre restreint de paramètres. Par exemple, on peut parfois se restreindre à considérer uniquement les translations, les rotations.... On optimise alors le terme d'attache aux données sur les vecteurs de translations possibles ou les paramètres d'une rotation.
  - On favorise le fait qu'un critère de régularité tel que

$$\sum_{m,n=1}^{N} |\nabla t^1_{m,n}|^2 + |\nabla t^2_{m,n}|^2, \tag{4.4}$$

ou

$$\sum_{m,n=1}^{N} \sqrt{|\nabla t^1_{m,n}|^2 + |\nabla t^2_{m,n}|^2},$$

soit petit.

Il faut noter à ce stade, que les modèles que nous avons envisagés dans cette section ne tiennent que partiellement compte des occlusions. On peut rajouter un terme permettant de représenter ces occlusions en utilisant de l'optimisation de forme.

4.2 Exemples de méthodes de recalage : La méthode de Horn et Schunck

Le modèle de Horn et Schunk correspond parfaitement au cadre décrit dans le chapitre précédent. Il consiste à minimiser une fonctionnelle dont le terme d'attache aux données est défini par (4.2) avec une interpolation définie par (4.1) et utilise le critère de régularité défini par (4.4). On aboutit alors à la minimisation en t d'une fonctionnelle de la forme

$$\sum_{m,n=1}^{N} \left(v^1_{m,n} + \langle \nabla v^1_{m,n}, t_{m,n} \rangle - v^2_{m,n}\right)^2 + \lambda \left(|\nabla t^1_{m,n}|^2 + |\nabla t^2_{m,n}|^2\right), \tag{4.5}$$

avec

$$\nabla v_{m,n} = (v_{m,n} - v_{m-1,n}\ ,\ v_{m,n} - v_{m,n-1}),$$

où v désigne v^1, t^1 ou t^2 ; ceux-ci étant périodisés en dehors de leur support.

Il est possible de montrer que ce problème a une solution dès que ∇v^1 peut prendre plusieurs directions. En effet, cette fonctionnelle est trivialement continue. Il n'est pas difficile de se convaincre qu'elle

est aussi coercive car t ne peut pas être à la fois constant et avoir des produits scalaires nuls avec tous les $\nabla v^1_{m,n}$ si ceux-ci changent de direction. Cette solution est de plus unique, sauf si le champ de vecteur constant est solution.

Le rôle du paramètre λ est simple à comprendre :

- plus λ est grand, plus le terme
$$\sum_{m,n=1}^{N} \left(|\nabla t^1_{m,n}|^2 + |\nabla t^2_{m,n}|^2\right)$$
a d'importance (et donc t est régulier) ;
- plus λ est petit, plus
$$\sum_{m,n=1}^{N} \left(v^1_{m,n} + \langle \nabla v^1_{m,n}, t_{m,n}\rangle - v^2_{m,n}\right)^2$$
a d'importance (au risque d'avoir des problèmes pour recaler les objets dont l'intensité a changé).

Des exemples de résultats de recalages avec la méthode de Horn et Schunck sont présentés sur la Figure 4.1. On voit sur cette figure deux images différentes d'une même scène et les résultats de la minimisation de (4.5) pour $\lambda = 100$ (ligne du milieu) et $\lambda = 1000$ (ligne du bas). Sur chaque ligne, on représente : sur l'image de gauche, le champ de vecteurs t pour une grille de pixels, lorsque t est suffisamment grand ; sur l'image de droite, le champ de vecteurs t pour une grille de pixels, multiplié par un facteur 10.

4.3 Applications du recalage

4.3.1 L'augmentation de résolution

Cette application est souvent utilisée pour retrouver des détails sur un film de mauvaise qualité. En effet, en recalant deux images successives (on peut en fait recaler, sans difficulté, toute une séquence du film), on obtient une image définie sur la grille irrégulière

$$\left\{(x,y) \in [0,N]^2, \text{ il existe } (m,n) \in \{1,\dots,N\}^2, \text{ tel que } (x,y) = (m,n) \text{ ou } (x,y) = (m,n) + t_{m,n}\right\},$$

où t représente le résultat du recalage entre les deux images de départ.

Il faut encore restaurer cette image de manière à la ré-échantillonner sur une nouvelle grille régulière (par exemple $\{1,\dots,2N\}^2$). On peut aussi profiter de cette étape pour déconvoluer et débruiter l'image en adaptant une des méthodes décrites dans le Chapitre 3.

4.3.2 Le codage de vidéos

Le principe de ce genre de codage est assez simple. Il part du constat que dans un film, les images successives sont proches les unes des autres. De plus, entre deux images successives, un objet solide (ou le fond de l'image) a un déplacement facilement codable (par exemple : entre deux images, l'objet, sur le film, est simplement translaté). Ainsi, le résultat du recalage entre ces deux images devrait être facile à coder (par exemple : t est constant sur plusieurs régions de l'image).

De ce fait, pour compresser une séquence de film, il semble moins coûteux de coder la première image et de coder ensuite les différents recalages entre les images de la séquence.

On voit ici une autre application du recalage qui est de *segmenter* (séparer en plusieurs éléments) une image en objets distincts (ceux qui subissent plus ou moins le même déplacement entre les différentes images d'un film). Ceci peut notamment être utile pour suivre un objet dans une scène.

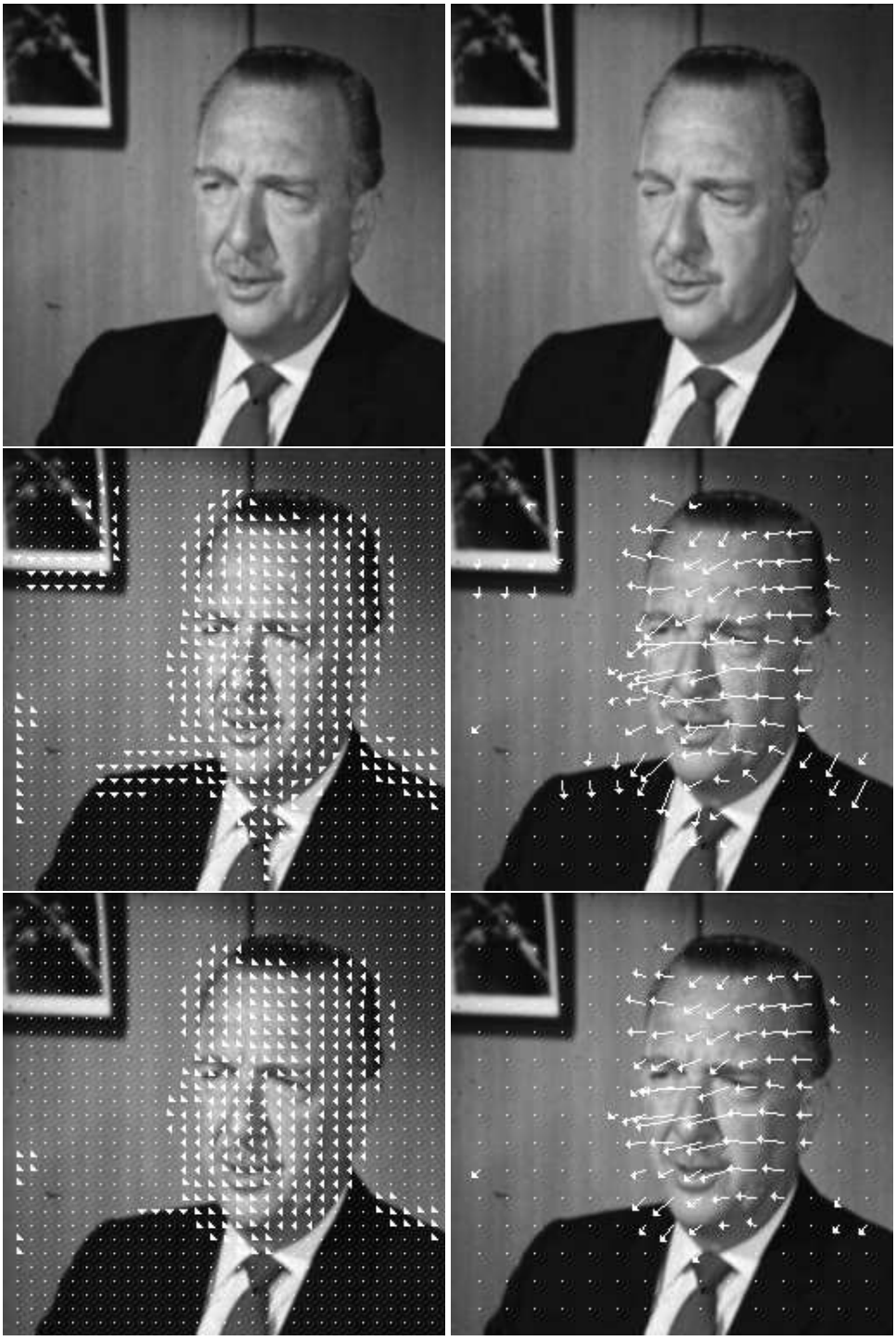


FIGURE 4.1 – Exemples de résultats obtenus avec la méthode de Horn et Schunck.

4.3.3 La stabilisation de films

En effet, dans beaucoup de films amateurs, la caméra n'est pas fixe et le film présente des oscillations. Il est possible d'enlever ces oscillations à l'aide du recalage. Dans un tel cas, le décalage entre deux images prend la forme

$$v^2_{m,n} \sim \widetilde{v^1}\left((m,n)+t'+t(x,y)\right)$$

où t' est le déplacement dû aux mouvements de la caméra entre les deux prises de vues et t est le déplacement dans l'image dû aux mouvements des objets filmés. (On suppose ici qu'il y a simplement une translation, le mouvement est donc uniforme sur l'image. On pourrait supposer d'autres types de mouvement sans que cela pose de réelle difficulté.)

On peut alors stabiliser le film en calculant la moyenne du déplacement sur l'image. On peut aussi (c'est souvent plus réaliste) restreindre cette moyenne à des points ayant un déplacement proche (il est même possible ici de demander à un utilisateur quelle est la partie de l'image qu'il souhaite garder immobile). Une fois calculée une approximation de t', il ne reste plus qu'à translater la deuxième image. Pour un meilleur rendu, il est souhaitable d'enlever la partie du bord des images ne se trouvant pas dans toutes les images.

4.3.4 La stéréoscopie

La stéréoscopie consiste à reconstruire le relief à partir de deux vues, prises de deux points de vues différents d'une même scène. Cette application est importante en robotique et en géographie (pour faire des cartes du relief à partir de photos satellites).

Avec un modèle de *camera pinhole* ("trou d'épingle" en français), et dans un cas où tous les paramètres de calibration des appareils sont connus (il faut les déterminer dans le cas général), on peut déterminer la distance entre les appareils de mesure et l'objet observé.

Nous allons le détailler dans le cas simple : en dimension 1, quand les appareils de mesure sont orientés dans une même direction, sont tous les deux situés dans un plan orthogonal à cette direction et ont la même distance focale (qui est connue). Cette situation est représentée sur la Figure 4.2.

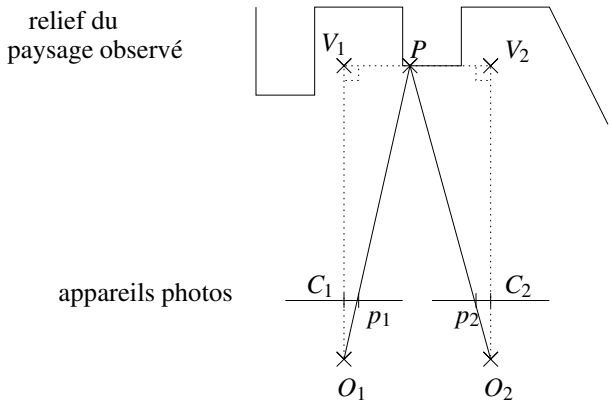


FIGURE 4.2 – Stéréoscopie (dessin en dimension 1): un même point est vu par deux appareils de mesure.

Dans la suite, on prend les notations décrites sur la Figure 4.2, où P désigne le point observé, p_1 et p_2 la position de P sur les images, O_1 et O_2 sont les centres des appareils de mesure produisant les deux images ; et C_1 et C_2 sont les centres de la ligne de l'image contenant p_1 et p_2.

La Figure 4.2 est en dimension 1. Pour passer à la dimension 2, il faut réaliser qu'en stéréoscopie, comme les deux images sont supposées acquises simultanément, le déplacement $C_2p_2 - C_1p_1$ est horizontal (i.e. parallèle à O_1O_2). Les points p_1 et p_2 sont donc situés sur une même ligne.

On a alors en appliquant le Théorème de Thalès

$$\frac{\overline{O_1V_1}}{\overline{O_1C_1}} = \frac{\overline{V_1P}}{\overline{C_1p_1}},$$

donc

$$\overline{O_1V_1}\,\overline{C_1p_1} = \overline{V_1P}\,\overline{O_1C_1}.$$

De même, on a

$$\overline{O_2V_2}\,\overline{C_2p_2} = \overline{V_2P}\,\overline{O_2C_2}.$$

On a donc, comme $\overline{O_1V_1} = \overline{O_2V_2}$ (même altitude) et $\overline{O_1C_1} = \overline{O_2C_2}$ (même distance focale)

$$\begin{aligned}\overline{O_1V_1}\left(\overline{C_1p_1} - \overline{C_2p_2}\right) &= \overline{O_1C_1}\left(\overline{V_1P} - \overline{V_2P}\right)\\ &= \overline{O_1C_1}\,\overline{V_1V_2}\\ &= \overline{O_1C_1}\,\overline{O_1O_2}.\end{aligned}$$

Ainsi, comme $\overline{C_1p_1} - \overline{C_2p_2}$ représente précisément le décalage calculé lors du recalage (modulo le fait que nous sommes ici en dimension 1), on obtient que $\overline{O_1V_1}$, qui représente l'altitude du point P, vaut

$$\overline{O_1V_1} = \frac{\overline{O_1C_1}\,\overline{O_1O_2}}{\overline{C_1p_1} - \overline{C_2p_2}},$$

$\overline{O_1C_1}$ et $\overline{O_1O_2}$ étant supposés connus.

Chapitre 5

Représentation parcimonieuse dans un dictionnaire d'atomes

5.1 Introduction : la minimisation ℓ^0

5.1.1 De l'approximation dans une base à la représentation parcimonieuse dans un dictionnaire

Dans le cours sur la transformée de Fourier et les ondelettes, nous avons envisagé d'exprimer notre donnée dans une base. Ce faisant, nous avons vu plusieurs bases. Certaines bases étaient bien adaptées pour représenter des images/signaux réguliers (Fourier), des signaux ou des images contenant des objets de petite taille (les ondelettes), des textures périodiques (certains paquets d'ondelettes). Nous avons aussi mentionné des bases (ou dictionnaires) permettant de représenter des bords étendus. Nous avons aussi vu que pour l'approximation, la compression ou le débruitage, il est primordial d'utiliser une base permettant d'avoir une bonne approximation de l'image ou du signal en n'utilisant qu'un nombre restreint de d'éléments de la base.

En fait ces différentes structures (bords, textures, zones régulières ...) sont souvent présentes sur une même donnée. Par exemple, sur l'image *Barbara* (voir Figure 5.1) on a simultanément des bords (par exemple à la frontière entre le pied de la table et le sol) des textures périodiques (par exemple sur le pantalon), des petits objets (par exemple le point blanc au bas du pied de la table ou les marques des livres). On voudrait donc idéalement représenter chaque zone de l'image avec les éléments de la base la mieux adaptée à chaque zone.

Par exemple, si l'on a deux bases $\mathcal{B}$ et $\mathcal{B}'$ de $\mathbb{R}^{N^2}$, on cherche à approximer un signal ou une image $u \in \mathbb{R}^{N^2}$ en utilisant le moins de coordonnées possible dans $\mathcal{B} \cup \mathcal{B}'$. On veut donc idéalement résoudre le problème d'optimisation suivant :

$$\begin{cases} \text{Min } l^0\left((\lambda_\varphi)_{\varphi\in(\mathcal{B}\cup\mathcal{B}')}\right) \\ \|\sum_{\varphi\in(\mathcal{B}\cup\mathcal{B}')}\lambda_\varphi\varphi - u\| \leq \tau, \end{cases}$$

où $\tau \geq 0$,

$$\|v\| = \sqrt{\sum_{i,j=0}^{N-1} |v_{i,j}|^2}, \quad \forall v \in \mathbb{R}^{N^2}$$

et

$$l^0\left((\lambda_\varphi)_{\varphi\in(\mathcal{B}\cup\mathcal{B}')}\right) = \left|\{\varphi \in (\mathcal{B}\cup\mathcal{B}'), \lambda_\varphi \neq 0\}\right|,$$

FIGURE 5.1 – Exemple d'image contenant des structures ponctuelles, des bords et des textures périodiques.

représente le nombre de coordonnées λ_φ non nulles. (On rappelle que $|.|$ désigne le cardinal d'un ensemble.)

Ci dessus, on n'a considéré que le cas où le dictionnaire est constitué de deux bases. Plus généralement, si on considère un dictionnaire $\mathcal{D}$ (i.e. un système linéaire fini générant $\mathbb{R}^{N^2}$), on cherche un jeu de coordonnées résolvant le problème suivant :

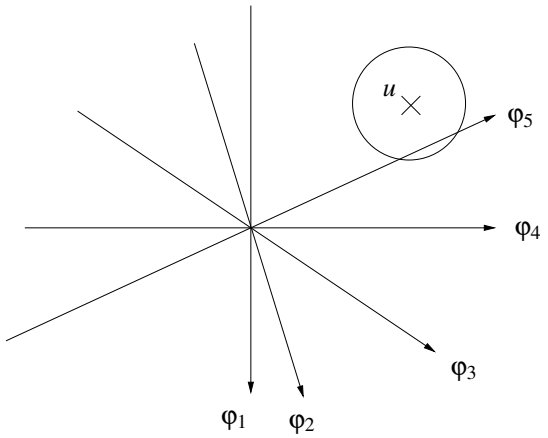
$$\begin{cases} \text{Min } l^0\left((\lambda_\varphi)_{\varphi\in\mathcal{D}}\right) \\ \|\sum_{\varphi\in\mathcal{D}}\lambda_\varphi\varphi - u\| \leq \tau, \end{cases} \tag{5.1}$$

pour $\tau \geq 0$.

Un dessin décrivant le problème (5.1) est fourni sur la Figure 5.2.

Proposition 1 *Pour tout dictionnaire $\mathcal{D}$ engendrant $\mathbb{R}^{N^2}$ et tout $u \in \mathbb{R}^{N^2}$, le problème d'optimisation* (5.1) *a une solution. Cette solution n'est généralement pas unique (voir la preuve de la démonstration pour plus de précisions).*

Preuve. Il n'est pas très difficile de voir que le problème (5.1) a une solution et qu'il atteint cette solution. En effet, comme $\mathcal{D}$ génère $\mathbb{R}^{N^2}$, il existe (au moins) un jeu de coordonnées permettant de construire u.



$$\begin{cases} \min_{(\lambda_i)_{1\leq i\leq 5}\in\mathbb{R}^P} \|(\lambda_i)_{1\leq i\leq 5}\|_0 \\ \text{sous } \|\sum_{i=1}^5 \lambda_i\varphi_i - u\| \leq \tau \end{cases}$$

FIGURE 5.2 – Dessin pour $N = 2$, $K = 1$. Le nombre de support de taille K est $C_P^K = \frac{P(P-1)...(P-K+1)}{K!}$.

Ainsi

$$\left\{(\lambda_\varphi)_{\varphi\in\mathcal{D}} \in \mathbb{R}^{\mathcal{D}}, \|\sum_{\varphi\in\mathcal{D}} \lambda_\varphi\varphi - u\| \leq \tau\right\} \neq \emptyset.$$

Par ailleurs, comme la fonction l^0 est à valeur dans $\mathbb{N}$ (un ensemble minoré), $l^0\left((\lambda_\varphi)_{\varphi\in\mathcal{D}}\right)$ a une valeur minimum sur cet ensemble. On la note $val(u,\mathcal{D},\tau)$. On remarque que $val(u,\mathcal{D},\tau) \in \mathbb{N}$.

On peut donc construire un jeu de coordonnées $(\lambda_\varphi^*)_{\varphi\in\mathcal{D}}$ tel que

$$\begin{cases} l^0\left((\lambda_\varphi^*)_{\varphi\in\mathcal{D}}\right) \leq val(u,\mathcal{D},\tau) + \frac{1}{2} \\ \|\sum_{\varphi\in\mathcal{D}} \lambda_\varphi^*\varphi - u\| \leq \tau. \end{cases}$$

Comme $l^0(.)$ est à valeur dans $\mathbb{N}$, on a donc forcément

$$l^0\left((\lambda_\varphi^*)_{\varphi\in\mathcal{D}}\right) = val(u,\mathcal{D},\tau),$$

et $(\lambda_\varphi^*)_{\varphi\in\mathcal{D}}$ est donc un minimiseur de (5.1).

Enfin, il n'est pas non plus très difficile de voir que le problème (5.1) n'a généralement pas une unique solution. En effet, si $\mathcal{S} \subset \mathcal{D}$ est tel qu'il existe un minimiseur de (5.1) dont le support est inclus dans $\mathcal{S}$, alors

$$\text{Vect}(\mathcal{S}) \bigcap \{v \in \mathbb{R}^{N^2}, \|v - u\| \leq \tau\}$$

où

$$\text{Vect}(\mathcal{S}) = \left\{\sum_{\varphi\in\mathcal{S}} \lambda_\varphi\varphi, (\lambda_\varphi)_{\varphi\in\mathcal{S}} \in \mathbb{R}^{\mathcal{S}}\right\},$$

est une boule l^2 de $\text{Vect}(\mathcal{S})$. □

5.1.2 Passage aux notations matricielles

Introduction

En général, nous avons besoin de notations différentes pour les différentes bases que nous considérons. Nous allons dans ce chapitre introduire des notations matricielles, car elles permettent d'abstraire la

base ou le dictionnaire. Elles permettent aussi d'introduire un éventuel opérateur de dégradation du signal ou de l'image.

Les notations matricielles permettent enfin de ne plus distinguer les signaux et les images. Ceux-ci sont vus comme de simples vecteurs d'un espace Euclidien. Cela est immédiat pour un signal. Pour une image $u \in \mathbb{R}^{N^2}$, on construit un vecteur en faisant la concaténation de toutes les colonnes de la u. On dit que l'on vectorise l'image. On vectorise de même tous les éléments du dictionnaire $\mathcal{D}$ et on range le vecteur ainsi obtenu dans une colonne d'une matrice A. Dis autrement, la matrice A est obtenue en mettant côte à côte la vectorisation de tous les éléments du dictionnaire. La matrice A est donc une matrice de taille $N^2 \times \#\mathcal{D}$ (c'est une matrice rectangulaire avec plus de colonnes que de lignes). Si on met les coordonnées $(\lambda_\varphi)_{\varphi\in\mathcal{D}} \in \mathbb{R}^{\mathcal{D}}$ dans un vecteur colonne x, on a alors

$$\sum_{\varphi\in\mathcal{D}} \lambda_\varphi \varphi = Ax.$$

Avec ces notations, le problème ℓ^0 (5.1) s'écrit

$$\begin{cases} \text{Min } l^0(x) \\ \|Ax-u\| \leq \tau, \end{cases} \tag{5.2}$$

pour $\tau \geq 0$.

Un dessin de la situation dans laquelle on se trouve est présentée sur la Figure 5.3.

Les différents algorithmes que nous allons voir [1] utilisent deux opérateurs mettant en jeu le dictionnaire :

- la multiplication Ax, pour un vecteur $x \in \mathbb{R}^{\#\mathcal{D}}$
- la multiplication $A^t v$, pour un vecteur $v \in \mathbb{R}^{N^2}$ et pour A^t la transposée de A. Il n'est pas difficile de voir que l'on a
$$A^t v = \left(\langle v, \varphi\rangle\right)_{\varphi\in\mathcal{D}},$$
où $\langle v, \varphi\rangle$ désigne le produit scalaire entre v et φ.

Le temps de calcul nécessaire à un algorithme construisant une approximation parcimonieuse est souvent majoritairement passé à faire ces multiplications. Nous détaillons ci-dessous comment faire ces calculs pour les bases de Fourier, d'ondelettes, de paquets d'ondelettes ou des dictionnaires constitués obtenus en faisant l'union de telles bases.

Le cas des bases

Dans le cours sur la transforméee de Fourier et sur les ondelettes, nous avons vu plusieurs bases. Si l'on utilise les notation matricielles, on sait calculer, pour chacune de ces bases Ax et son inverse $A^{-1}v$. Par exemple, Ax peut correspondre à la transformée de Fourier inverse de x, auquel cas $A^{-1}v$ correspond à la tranformée de Fourier de v. Ax peut correspondre à la reconstruction de la tranformée en ondelettes, auquel cas $A^{-1}v$ correspond à la décomposition en coefficients d'ondelettes.

Nous détaillons ci-dessous comment calculer les opérateurs Ax et $A^t v$ pour ces différentes bases.

- Pour les **bases orthonormales** : C'est la cas des bases d'ondelettes et de paquets d'ondelettes orthogonales. On sait que pour une base orthonormée, on a
$$\forall v \in \mathbb{R}^{N^2}, \quad v = \sum_{\varphi\in\mathcal{D}} \langle v, \varphi\rangle \varphi$$
  - Nous pouvons calculer Ax en utilisant l'algorithme de reconstruction.

1. Comme tous les algorithmes dans ce domaine

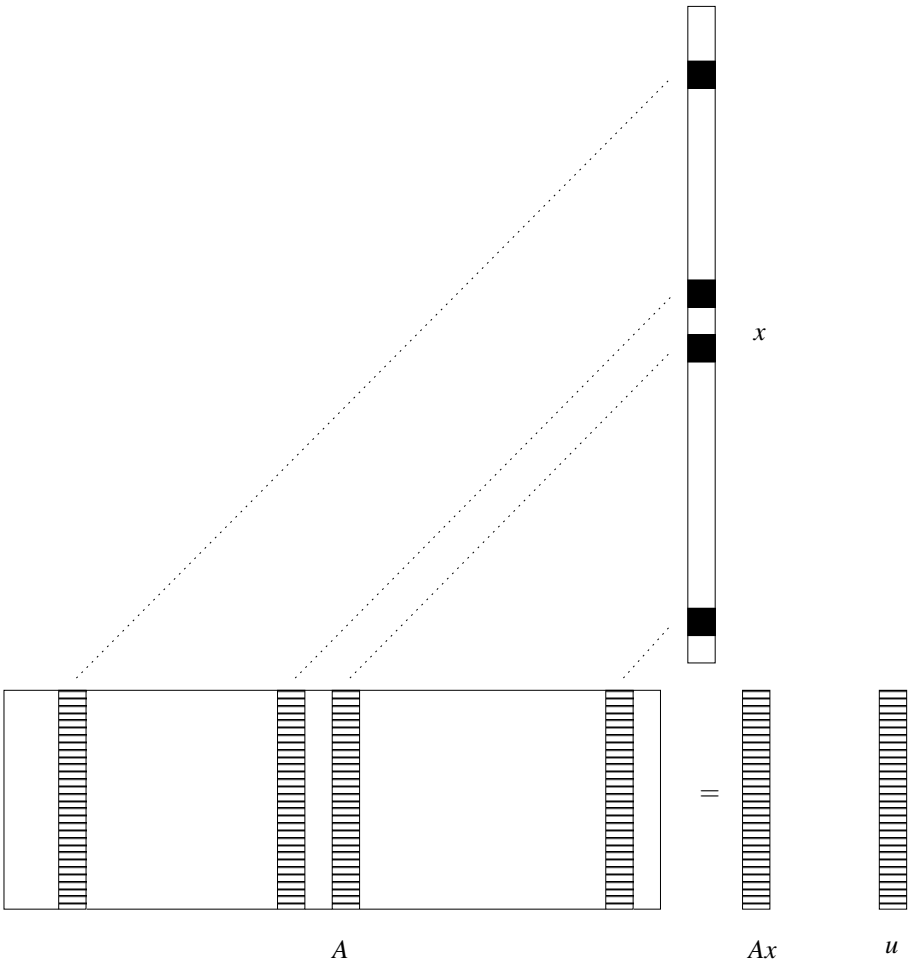


FIGURE 5.3 – Représentation des matrices et vecteurs manipulés dans le problème (5.2). Les cases noires de x représentent les coordonnées non-nulles.

- Nous pouvons donc calculer les coordonnées $A^t v$ en appliquant à v l'algorithme calculant la décomposition.

– Pour les **bases orthogonales dont tous les éléments ont la même norme** δ : C'est le cas de la base de Fourier (avec $\delta = \frac{1}{N}$), de la base de cosinus, de la base de cosinus locaux. Il n'est pas difficile de voir que l'on a alors

$$\forall v \in \mathbb{R}^{N^2}, \quad v = \frac{1}{\delta^2} \sum_{\varphi \in \mathcal{D}} \langle v, \varphi \rangle \varphi$$

Par ailleurs, l'algorithme de reconstruction habituel comprend la multiplication par $\frac{1}{\delta^2}$.

- On peut donc calculer Ax en multipliant le résultat de l'algorithme de reconstruction habituel par δ^2.
- On peut calculer $A^t v$ en utilisant l'algorithme de décomposition habituel.

– Pour les **bases biorthogonale** : Dans le cas particulier des bases d'ondelettes et de paquets d'ondelettes biorthogonales, on calcule la décomposition et son inverse à l'aide de filtres $(h, g, \tilde{h}, \tilde{g})$.

On obtient alors deux bases bi-orthogonales $(\varphi_i)_{1\leq i\leq N^2}$ et $(\tilde{\varphi}_i)_{1\leq i\leq N^2}$ qui vérifient

$$\begin{aligned}\forall v\in\mathbb{R}^{N^2},\quad v &= \sum_{i=1}^{N^2}\langle v,\varphi_i\rangle\tilde{\varphi}_i,\\ &= \sum_{i=1}^{N^2}\langle v,\tilde{\varphi}_i\rangle\varphi_i.\end{aligned}$$

- Pour calculer Ax, on recompose les coefficients avec les filtres $\tilde{h}$ et $\tilde{g}$ (respectivement h et g).
- Pour calculer $A^t v$ on décompose avec les filtres $\tilde{h}$ et $\tilde{g}$ (respectivement h et g).

Ainsi, contrairement à ce que l'on fait pour calculer une décomposition en ondelette bi-orthogonale et son inverse, on n'utilise que deux filtres pour les calculs de Ax et $A^t v$: soit $\tilde{h}$ et $\tilde{g}$; soit h et g.

L'union de bases

Ici, on considère un dictionnaire constitué de l'union de plusieurs bases $\mathcal{D}=\mathcal{B}_1\cup\ldots\cup\mathcal{B}_K$. On considère, pour $k=1,\ldots,K$ et pour chaque base $\mathcal{B}_k$, la matrice A_k correspondante. La matrice A correspondant à $\mathcal{D}$ est alors la concaténation (en ligne) des matrices A_k, pour $k=1,\ldots,K$. Le calcul de Ax, où x est la concaténation des vecteurs colonnes $x_1,\ldots,x_K$ se fait donc avec la formule suivante :

$$Ax = A_1x_1 + A_2x_2 + \ldots + A_Kx_K.$$

Le calcul de $A^t v$, pour $v\in\mathbb{R}^{N^2}$ se fait en concaténant (en colonne) tous les vecteurs colonnes $A_k^t v$, pour $k=1,\ldots,K$.

Application d'un opérateur de dégradation

Nous verrons dans la section suivante qu'il est tentant de proposer un modèle de restauration d'image en utilisant un apriori de parcimonie. Si la donnée a subi une dégradation avec un opérateur linéaire [2] $H:\mathbb{R}^{N^2}\longrightarrow\mathbb{R}^{N^2}$, on veut alors résoudre le problème

$$\left\{\begin{array}{l}\text{Min }\ell^0(x)\\ \|HAx-u\|\leq\tau,\end{array}\right. \tag{5.3}$$

pour $\tau\geq 0$ bien choisi. Ce problème a toujours la même forme et peut s'écrire

$$\left\{\begin{array}{l}\text{Min }\ell^0(x)\\ \|Dx-u\|\leq\tau,\end{array}\right.$$

où la matrice D vaut HA. Les résultats et analyses que nous faisons sur (5.2) sont donc aussi vrai pour (5.3).

Il faut enfin noter que (trivialement) le calcul numérique de Dx se fait en appliquant la fonction calculant H au résultat Ax ; et que le calcul de l'opérateur adjoint $D^t v$ se fait en appliquant la fonction calculant l'opérateur A^t au résultat de $H^t v$ (i.e. : en appliquant $D^t=(HA)^t=A^tH^t$).

5.1.3 L'échantillonnage compressé, cas de la minimisation ℓ^0

Pour pouvoir inclure un éventuel opérateur de dégradation (éventuellement non inversible), on considère le problème

$$\left\{\begin{array}{l}\text{Min }\ell^0(x)\\ \|Dx-u\|\leq\tau.\end{array}\right. \tag{5.4}$$

2. Nous supposons que l'espace d'arrivé est le même que l'espace de départ par simplicité, mais cela n'est pas impératif.

Pour simplifier les notations, on suppose que $x \in \mathbb{R}^P$, D est une matrice $N \times P$ (i.e. N lignes et P colonnes) et $u \in \mathbb{R}^N$. Nous pouvons alors énoncer un premier résultat d'échantillonnage compressé. Il considère la reconstruction de coordonnées x_0, à l'aide de la résolution du problème (5.4), lorsqu'il n'y a pas de bruit.

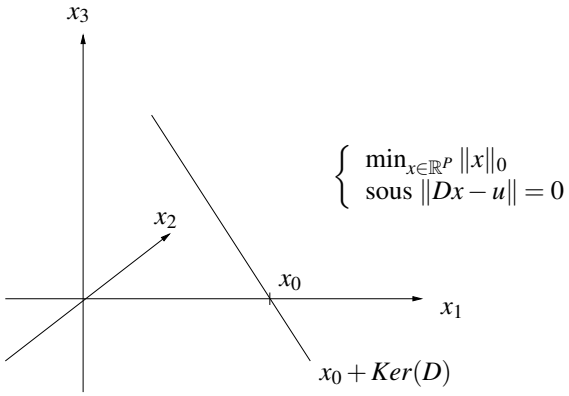


FIGURE 5.4 – Dessin pour $P = 3$, $K = 1$, $\tau = 0$. On suppose qu'il existe x_0 tel que $\ell^0(x_0) \leq K$ et $u = Dx_0$. On a $\dim(Ker(D)) = P - N$ et $K \ll N$. Leur intersection est donc très improbable.

Théorème 6 *Si il existe $K \leq N$ et x_0 tel que $\ell^0(x_0) \leq K$ et $u = Dx_0$. Si de plus, D est tel que, pour tout x tel que $\ell^0(x) \leq 2K$, on a $Dx \neq 0$. Alors x_0 est l'unique minimiseur de* (5.4)*, lorsque $\tau = 0$.*

Preuve. En raisonnant par l'absurde, s'il existe un minimiseur x_1 de (5.4) (avec $\tau = 0$) différent de x_0, on a $\ell^0(x_1) \leq K$ et $\ell^0(x_0) \leq K$ avec $Dx_1 = Dx_0$. On a alors $\ell^0(x_1 - x_0) \leq 2K$ et $D(x_1 - x_0) = 0$. Ceci contredit la seconde hypothèse du théorème. □

Malgré la simplicité de sa preuve, ce théorème dit que, même lorsque $P \geq N$, on peut inverser l'opérateur D à condition qu'il existe un x_0 de départ parcimonieux. Si on considère H, cela revient à dire que l'on peut inverser H, en résolvant (5.4), si le signal ou l'image de départ Ax_0 s'exprime de façon parcimonieuse.

Un point important est que la seconde condition (celle sur le noyau, $\ker(D)$, de D) est en fait presque toujours vérifiée lorsque $2K < N$. En effet, lorsque D est surjectif (ce qui est généralement le cas) $\ker(D)$ est un espace vectoriel de dimension $P - N$ et

$$\Sigma_{2K} = \{x \in \mathbb{R}^P, \ell^0(x) \leq 2K\}$$

est une union finie d'espaces vectoriels de dimension $2K$. Leur intersection est donc presque sûrement réduite à 0 dès que $P > \dim(\ker(D)) + 2K = P - N + 2K$, c'est à dire dès que $2K < N$.

Finalement, on obtient que pour presque tout D, si une donnée signal ou image peut être exprimée de manière parcimonieuse à l'aide de A, on est capable d'inverser l'opérateur H (même s'il est a priori non inversible). Ce résultat ne garantit cependant pas la stabilité de l'inversion. Pour la garantir, nous avons besoin d'une hypothèse plus forte sur H. Par exemple, la propriété d'isométrie restreinte.

Définition 1 *On suppose les colonnes de la matrice D de norme 1. On dit que la matrice D vérifie la propriété d'isométrie restreinte ("Restricted Isometry Property" (RIP)) au niveau K, pour $K \in \{1, \ldots, N\}$, si il existe $\delta_K \in [0, 1)$ tel que*

$$(1 - \delta_K)\|x\|^2 \leq \|D_K x\|^2 \leq (1 + \delta_K)\|x\|^2 \qquad , \forall x \in \mathbb{R}^K$$

pour toute matrice D_K de taille $N \times K$ dont les colonnes sont extraites de D.

Cette propriété signifie que quelque-soit l'ensemble de K colonnes de D que l'on considère, cet ensemble est constitué de vecteurs presque orthogonaux entre eux. Bien sûr, si $K > \mathrm{rk}(D)$, D ne peut pas satisfaire le K-RIP.

Si D vérifie cette propriété on obtient la stabilité de l'inversion de D, pour les données compressibles.

Théorème 7 *Si il existe $K \leq N$ et x_0 tel que $\ell^0(x_0) \leq K$ et si l'on considère la donnée*

$$u = Dx_0 + b.$$

Si D vérifie le $2K$-RIP pour la constante δ_{2K}, alors

$$\|x_0 - x^*\| \leq \frac{2}{\sqrt{1-\delta_{2K}}}\tau$$

où x^* est une solution quelconque de (5.4)*, lorsque $\|b\| \leq \tau$.*

Preuve. On remarque tout d'abord que comme, lorsque $\|b\| \leq \tau$, x_0 satisfait les contraintes de (5.4), on a pour x^* solution de (5.4)

$$\ell^0(x^*) \leq \ell^0(x_0) \leq K.$$

On a alors, pour la matrice D_{2K} dont les colonnes correspondent aux entrées non nulles de x_0 et x^*,

$$\begin{array}{rcll}
(1-\delta_{2K})\|x_0 - x^*\|^2 & \leq & \|D_{2K}(x_0 - x^*)\|^2 & \text{, car } D \text{ vérifie le } 2K\text{-RIP} \\
& = & \|D(x_0 - x^*)\|^2 & \text{, du fait de la construction de } D_{2K} \\
& = & \|(Dx_0 - u) - (Dx^* - u)\|^2 & \\
& \leq & (\|Dx_0 - u\| + \|Dx^* - u\|)^2 & \text{, inégalité triangulaire vérifiée par } \|.\| \\
& \leq & (2\tau)^2 & \text{, car } \|Dx_0 - u\| = \|b\| \leq \tau \text{ et } \|Dx^* - u\| \leq \tau
\end{array}$$

□

Dans le théorème ci-dessus, le terme b contient typiquement une composante correspondant à l'erreur d'approximation par un signal-une image parcimonieuse auquel s'ajoute un éventuel terme de bruit.

Lorsque D contient un opérateur H à inverser, ce théorème signifie que, sous l'hypothèse de parcimonie, on peut inverser H et que l'erreur après inversion est du même ordre de grandeur que $\|b\|$ (i.e. l'erreur faite lorsque l'on à approximé le signal/l'image de départ). L'opérateur D doit pour cela satisfaire le $2K$-RIP. On doit donc notamment avoir $2K \leq \mathrm{rk}(D)$, où l'on rappelle que K est la parcimonie de la donnée.

5.1.4 Résolution numérique : cas d'une base orthonormée

Le principal défaut de (5.1) est qu'il n'est ni convexe ni différentiable. Ceci rend sa résolution approchée par des algorithmes itératifs très difficile. Il est par ailleurs NP-difficile. Ceci signifie que (en général) le calcul d'une solution exacte de (5.1) est (et sera toujours) trop long pour être réalisé.

Il existe évidemment un certain nombre de situations dans lesquels on sait calculer une solution de (5.1). Par exemple, si $\mathcal{D}$ est une base orthonormale, il suffit généralement de seuiller les coordonnées de la donnée à la bonne valeur pour obtenir une solution.

Proposition 2 *On considère une base orthonormée $\mathcal{D}$ et on suppose que $u \in \mathbb{R}^{N^2}$ est tel que*

$$u_\varphi \neq u_{\varphi'} \quad \text{, pour tout } \varphi \neq \varphi' \in \mathcal{D} \tag{5.5}$$

où $(u_\varphi)_{\varphi\in\mathcal{D}} \in \mathbb{R}^{\mathcal{D}}$ est l'unique jeu de coordonnées tel que

$$u = \sum_{\varphi\in\mathcal{D}} u_\varphi \varphi.$$

On définit, pour $\alpha \geq 0$*, la fonction de seuillage*

$$T_\alpha(t) = \begin{cases} 0 & \text{, si } |t| < \alpha, \\ t & \text{, sinon.} \end{cases}$$

Il existe $\alpha^* \geq 0$ *(dont la construction est fournie dans la preuve) tel que*

$$\left(T_{\alpha^*}(u_\varphi)\right)_{\varphi \in \mathcal{D}}$$

soit un minimiseur de (5.1)*.*

Preuve. On remarque tout d'abord que si $\tau \geq \|u\|$, les coordonnées uniformément nulles sont une solution triviale de (5.1). L'énoncé de la proposition est donc vrai pour tout $\alpha^* > \max\{|u_\varphi|, \varphi \in \mathcal{D}\}$.

Supposons maintenant que $\tau < \|u\|$.

En utilisant les notations de la proposition, on note pour tout $\alpha \geq 0$

$$f(\alpha) = \sum_{\varphi : |u_\varphi| < \alpha} u_\varphi^2,$$

il n'est pas difficile de voir que f est croissante, constante par morceaux et semi-continue inférieurement. On voit aussi qu'elle vaut 0 en 0 et $\|u\|^2$ pour $\alpha > \max\{|u_\varphi|, \varphi \in \mathcal{D}\}$.

On note aussi $\alpha^* = \max\{\alpha \geq 0, f(\alpha) \leq \tau^2\}$ et

$$\mathcal{S}^* = \{\varphi \in \mathcal{D}, |u_\varphi| < \alpha^*\}.$$

Soit $(\lambda_\varphi)_{\varphi \in \mathcal{D}} \in \mathbb{R}^{\mathcal{D}}$ satisfaisant $\#\mathcal{S} > \#\mathcal{S}^*$ pour $\mathcal{S} = \{\varphi \in \mathcal{D}, \lambda_\varphi = 0\}$, on a

$$\| \sum_{\varphi \in \mathcal{D}} \lambda_\varphi \varphi - u\|^2 = \sum_{\varphi \in \mathcal{D}} (\lambda_\varphi - u_\varphi)^2 \geq \sum_{\varphi \in \mathcal{S}} u_\varphi^2. \tag{5.6}$$

Par ailleurs, comme $\#\mathcal{S} > \#\mathcal{S}^*$, on sait qu'il existe $\varphi_0 \in \mathcal{S} \setminus \mathcal{S}^*$. Mais, du fait de la construction de $\mathcal{S}^*$,

$$\sum_{\varphi \in \mathcal{S} \setminus \{\varphi_0\}} u_\varphi^2 \geq \sum_{\varphi \in \mathcal{S}^*} u_\varphi^2.$$

On ajoute en effet plus de termes et ceux de $\mathcal{S}^*$ sont plus petits. On a donc finalement que, du fait de (5.5), de la structure de f^* et de la définition de α^*, on a forcément

$$\sum_{\varphi \in \mathcal{S}} u_\varphi^2 \geq \sum_{\varphi \in \mathcal{S}^*} u_\varphi^2 + u_{\varphi_0}^2, \tag{5.7}$$

Remarquez que

$$\forall \varphi_0 \in \mathcal{D} \setminus \mathcal{S}^*, \quad \sum_{\varphi \in \mathcal{S}^*} u_\varphi^2 + u_{\varphi_0}^2 \geq f(\alpha^*) + \alpha^{*2}. \tag{5.8}$$

On a par ailleurs forcément, du fait de la définition de α^*,

$$f(\alpha^*) + \alpha^{*2} > \tau^2. \tag{5.9}$$

On déduit donc finalement de (5.6), (5.7), (5.8), (5.9), que

$$\| \sum_{\varphi \in \mathcal{D}} \lambda_\varphi \varphi - u\|^2 > \tau^2.$$

Ainsi, pour tout $(\lambda_\varphi)_{\varphi\in\mathcal{D}} \in \mathbb{R}^{\mathcal{D}}$

$$l^0\left((\lambda_\varphi)_{\varphi\in\mathcal{D}}\right) < N^2 - \#\mathcal{S}^* \quad \Longrightarrow \quad \| \sum_{\varphi\in\mathcal{D}} \lambda_\varphi \varphi - u\|^2 > \tau^2$$

et donc que

$$val(u,\mathcal{D},\tau) \geq N^2 - \#\mathcal{S}^*.$$

Enfin, on a par ailleurs que $(T_{\alpha^*}(u_\varphi))_{\varphi\in\mathcal{D}}$ vérifie

$$\begin{cases} l^0\left((T_{\alpha^*}(u_\varphi))_{\varphi\in\mathcal{D}}\right) = N^2 - \#\mathcal{S}^*, \\ \sum_{\varphi\in\mathcal{D}} (T_{\alpha^*}(u_\varphi) - u_\varphi)^2 = \sum_{\varphi\in\mathcal{S}^*} u_\varphi^2 \leq \tau^2. \end{cases}$$

On en conclut donc que $(T_{\alpha^*}(u_\varphi))_{\varphi\in\mathcal{D}}$ est un minimiseur de (5.1). □

Remarquez que la condition (5.5) n'exclut qu'un ensemble de mesure nulle de données u. Ce n'est donc pas une restriction trop importante. On pourrait par ailleurs l'éviter en rafinant un peu le choix des coordonnées à seuiller.

5.2 La minimisation ℓ^1

5.2.1 Introduction

Comme nous l'avons vu, le problème (5.4) est trop difficile pour être résolu. Une alternative possédant quelques similarités avec le problème (5.4) consiste à remplacer le terme ℓ^0 de (5.4) par un terme ℓ^1. On obtient alors le problème

$$\begin{cases} \text{Min } \|x\|_1 \\ \|Dx - u\| \leq \tau, \end{cases} \tag{5.10}$$

pour $\tau \geq 0$ et pour la norme ℓ^1 définie par

$$\|x\|_1 = \sum_{i=1}^{P} |x_i|.$$

L'avantage de ce problème est que l'on peut calculer ses solutions. Il est, en effet, convexe et coercif (mais non différentiable). Nous donnerons différents éléments sur la résolution de (5.10) et décrirons un algorithme permettant de le résoudre dans la Section 5.2.3. Avant cela, nous énonçons des résultats de type "échantillonnage compressé" dans la Section 5.2.2.

5.2.2 L'échantillonnage compressé, cas de la minimisation ℓ^1

Théorème 8 *Soit $K \leq N$ et x_0 tel que $\ell^0(x_0) \leq K$. On considère la donnée*

$$u = Dx_0 + b.$$

Si D vérifie le $4K$-RIP et que les constantes δ_{3K} et δ_{4K} vérifient $\delta_{3K} + 3\delta_{4K} < 2$, alors on a

$$\|x_0 - x^*\| \leq C_K \tau$$

où x^* est la solution de (5.10) *pour n'importe quel τ tel que $\|b\| \leq \tau$. Une constante C_K est donnée dans la preuve.*

Comme pour les théorèmes précédents, celui-ci dit que sous réserve que le signal/l'image originale puisse être exprimée de manière parcimonieuse et que D satisfasse les bonnes propriétés, la distance entre la solution de (5.10) et x_0 est comparable à l'erreur initial. À nouveau, b contient typiquement un terme d'erreur commis lors de l'approximation parcimonieuse du signal/de l'image, auquel s'ajoute un éventuel terme de bruit.

Avant de donner la preuve du Théorème 8, nous énonçons différents lemmes. Dans les lemmes et la preuve, on note

$$x^* = x_0 + h.$$

On note aussi T_0 le support de x_0 (i.e. les indices $i \in \{1,\dots,P\}$ auxquels les coordonnées de x_0 sont non-nuls : $(x_0)_i \neq 0$.) Pour tout vecteur de $y \in \mathbb{R}^P$, et tout ensemble $T \subset \{1,\dots,P\}$, on note y^T la restriction de y à T :

$$y_i^T = \begin{cases} y_i & \text{, si } i \in T \\ 0 & \text{, sinon.} \end{cases} \qquad \text{,} \forall i \in \{1,\dots,P\}.$$

Il faut noter que l'on a pour tout y en utilisant l'inégalité de Cauchy-Schwarz

$$\|y^T\|_1 = \sum_{i\in T} |y_i^T| = \sum_{i\in T} y_i^T \operatorname{sign}\left(y_i^T\right) \leq \|y^T\| \sqrt{|T|} \tag{5.11}$$

On note T^c l'ensemble complémentaire de T : $T^c = \{1,\dots,P\} \setminus T$.

Lemme 4 *Sous les hypothèses de Théorème 8, on a*

$$\|Dh\| \leq 2\tau, \tag{5.12}$$

$$\|x^*\|_1 \leq \|x_0\|_1$$

et

$$\|h^{T_0^c}\|_1 \leq \|h^{T_0}\|_1. \tag{5.13}$$

Ce lemme contraint x^* à être dans la boule ℓ^1 de rayon $\|x_0\|$. De plus, comme x^* est proche de x_0 (dans le sens où $\|Dh\|$ est petit) et x_0 est parcimonieux, x^* est contraint à être proche d'une facette de petite dimension de la sphère ℓ^1 (i.e. dans un "coin" de la boule ℓ^1). L'équation (5.13) est une première conséquence des deux premières propriétés et dit que l'erreur h est majoritairement (au sens ℓ^1) sur les éléments du support T_0 de x_0.

Preuve. On a en effet,

$$\begin{aligned} \|Dh\| = \|D(x^* - x_0)\| &= \|(Dx^* - u) - (Dx_0 - u)\| \\ &\leq \|Dx^* - u\| + \|Dx_0 - u\| \\ &\leq 2\tau \end{aligned}$$

car $\|Dx^* - u\| \leq \tau$ et $\|Dx_0 - u\| = \|b\| \leq \tau$.

Pour montrer la seconde inégalité, on remarque que, comme x^* est solution de (5.10) pour τ tel que $\|b\| \leq \tau$ et que x_0 satisfait la contrainte de (5.10), on a

$$\|x^*\|_1 \leq \|x_0\|_1.$$

Par ailleurs, comme x_0 est à support dans T_0, on a

$$\begin{aligned} \|x^*\|_1 = \|x_0 + h\|_1 &= \sum_{i\in T_0} |(x_0)_i + h_i| + \sum_{i\notin T_0} |h_i|, \\ &\geq \sum_{i\in T_0} \left(|(x_0)_i| - |h_i|\right) + \|h^{T_0^c}\|_1, \\ &\geq \|x_0\|_1 - \|h^{T_0}\|_1 + \|h^{T_0^c}\|_1, \\ &\geq \|x^*\|_1 - \|h^{T_0}\|_1 + \|h^{T_0^c}\|_1. \end{aligned}$$

Dont on déduit la troisième inégalité. □

Avant d'énoncer le second lemme, on décompose pour $M > 0$

$$T_0^c = T_1 \cup T_2 \cup \ldots \cup T_L$$

où, pour tout $l = 1..L-1$, $|T_l| = M$ et $|T_L| \leq M$; et pour tout $l = 1..L-1$, tout $i \in T_l$ et tout $i' \in T_{l+1}$

$$|h_i| \geq |h_{i'}|.$$

Dis en mots, T_1 contient les indices des M plus grandes coordonnées (en valeur absolue) de $h^{T_0^c}$,T_2 les M suivantes, etc

On note aussi $T_{01} = T_0 \cup T_1$.

Lemme 5 *Sous les hypothèses de Théorème 8, on a*

$$\|h^{T_{01}^c}\|^2 \leq \|h^{T_0}\|^2 \frac{|T_0|}{M}$$

dont on déduit

$$\|h\| \leq \sqrt{1 + \frac{|T_0|}{M}} \|h^{T_{01}}\|. \tag{5.14}$$

Ce lemme dit que $h^{T_{01}}$ contient l'essentiel de l'énergie de h (en normeℓ^2), lorsque $\frac{|T_0|}{M}$ est petit. Dis autrement, l'erreur est concentrée sur les indices de T_0 et sur un ensemble dont la taille est comparable à celle de T_0. Intuitivement, cette propriété exploite le fait que la boule ℓ^1 est "pointue", en grande dimension. Il implique que pour majorer $\|h\|$, il suffit de majorer $\|h^{T_{01}}\|$.

Preuve. On a en effet que, pour $k \leq P$, la $k^{\text{ième}}$ plus grande valeur de $|h^{T_0^c}|$ (noté $|h^{T_0^c}|_{(k)}$) vérifie

$$\|h^{T_0^c}\|_1 \geq \sum_{k'=1}^{k} |h^{T_0^c}|_{(k')} \geq \sum_{k'=1}^{k} |h^{T_0^c}|_{(k)} = k\, |h^{T_0^c}|_{(k)}.$$

On a donc

$$\|h^{T_{01}^c}\|^2 = \sum_{k=M+1}^{P} |h^{T_0^c}|_{(k)}^2 \leq \sum_{k=M+1}^{P} \frac{\|h^{T_0^c}\|_1^2}{k^2} \leq \frac{\|h^{T_0^c}\|_1^2}{M},$$

où la dernière inégalité est due au fait que

$$\sum_{k=M+1}^{P} \frac{1}{k^2} \leq \int_M^{+\infty} \frac{1}{t^2} dt = \left[-\frac{1}{t}\right]_M^{+\infty} = \frac{1}{M}.$$

En utilisant (5.13) et (5.11), on obtient alors la première inégalité

$$\|h^{T_{01}^c}\|^2 \leq \frac{\|h^{T_0}\|_1^2}{M} \leq \|h^{T_0}\|^2 \frac{|T_0|}{M}.$$

Finalement, en remarquant que cela implique que

$$\|h^{T_{01}^c}\|^2 \leq \frac{|T_0|}{M} \|h^{T_{01}}\|^2$$

on obtient

$$\|h\|^2 = \|h^{T_{01}}\|^2 + \|h^{T_{01}^c}\|^2 \leq \left(1 + \frac{|T_0|}{M}\right) \|h^{T_{01}}\|^2$$

dont on déduit l'énoncé du lemme. □

Lemme 6 *Sous les hypothèses de Théorème 8, on a*

$$\sum_{l=2}^{L} \|h^{T_l}\| \leq \sqrt{\frac{K}{M}} \|h^{T_0}\| \tag{5.15}$$

Ce Lemme exprime que l'énergie de h (au sens ℓ^2) est concentrée sur le support T_0 de x_0.
Preuve. Soient $l \in \{1,\ldots,L-1\}$, $n \in T_{l+1}$ et $n' \in T_l$. On a, du fait de la définition des T_l

$$|h_n| \leq |h_{n'}|.$$

Donc

$$M|h_n| = \sum_{n' \in T_l} |h_n| \leq \sum_{n' \in T_l} |h_{n'}| = \|h^{T_l}\|_1$$

et

$$\|h^{T_{l+1}}\|^2 = \sum_{n \in T_{l+1}} |h_n|^2 \leq \sum_{n \in T_{l+1}} \frac{\|h^{T_l}\|_1^2}{M^2} = \frac{\|h^{T_l}\|_1^2}{M}.$$

On a donc

$$\begin{aligned} \sum_{l=2}^{L} \|h^{T_l}\| = \sum_{l=1}^{L-1} \|h^{T_{l+1}}\| &\leq \sum_{l=1}^{L-1} \frac{\|h^{T_l}\|_1}{\sqrt{M}} \\ &\leq \frac{\|h^{T_0^c}\|_1}{\sqrt{M}} \leq \frac{\|h^{T_0}\|_1}{\sqrt{M}} \leq \sqrt{\frac{K}{M}} \|h^{T_0}\| \end{aligned}$$

□

où les deux dernières inégalités sont respectivement obtenus en utilisant (5.13) et (5.11).

Lemme 7 *Sous les hypothèses de Théorème 8 et si $M \leq 3K$ on a*

$$\|Dh\| \geq C_{K,M} \|h^{T_{01}}\|, \tag{5.16}$$

avec

$$C_{K,M} = \sqrt{1-\delta_{M+K}} - \sqrt{1+\delta_M}\sqrt{\frac{K}{M}}.$$

Ce lemme combine les propriétés de RIP et les résultats précédents pour obtenir le fait que $\|h^{T_{01}}\|$ est petit (et pas seulement $\|Dh\|$).
Preuve. We have indeed,

$$\begin{array}{rcll} \|Dh\| & = & \|Dh^{T_{01}} + Dh^{T_{01}^c}\| & \\ & \geq & \|Dh^{T_{01}}\| - \|\sum_{l=2}^{L} Dh^{T_l}\| & \text{(inégalité triangulaire)} \\ & \geq & \|Dh^{T_{01}}\| - \sum_{l=2}^{L} \|Dh^{T_l}\| & \text{(inégalité triangulaire)} \\ & \geq & \sqrt{1-\delta_{M+K}}\|h^{T_{01}}\| - \sqrt{1+\delta_M}\sum_{l=2}^{L} \|h^{T_l}\| & ((M+K)\text{-RIP et } M\text{-RIP}) \\ & \geq & \sqrt{1-\delta_{M+K}}\|h^{T_{01}}\| - \sqrt{\frac{K}{M}}\sqrt{1+\delta_M}\|h^{T_0}\| & \text{(du fait de (5.15))} \end{array}$$

L'énoncé découle alors immédiatement du fait que $\|h^{T_0}\| \leq \|h^{T_{01}}\|$.

□

Preuve du théorème 8 : Pour achever la preuve du Théorème 8, on remarque simplement que si $M = 3K$ et si, comme cela est supposé dans le Théorème 8, $\delta_{3K} + 3\delta_{4K} < 2$, on a $1 + \delta_{3K} < 3(1 - \delta_{4K})$ et donc

$$C_{K,3K} = \sqrt{1 - \delta_{4K}} - \frac{1}{\sqrt{3}}\sqrt{1 + \delta_{3K}} > 0.$$

On a alors

$$\begin{aligned} \|h\| &\leq \sqrt{1 + \tfrac{|T_0|}{M}}\|h^{T_{01}}\| \quad \text{(en appliquant (5.14) du Lemme 5)} \\ &\leq \frac{\sqrt{1 + \frac{|T_0|}{M}}}{C_{K,M}}\|Dh\| \quad \text{(en appliquant (5.16) du Lemme 7)} \\ &\leq 2\frac{\sqrt{1 + \frac{|T_0|}{M}}}{C_{K,M}}\ \tau \quad \text{(en appliquant (5.12) du Lemme 4)} \end{aligned}$$

Le théorème est donc vrai avec la constante

$$C_K = 2\frac{\sqrt{1 + \frac{1}{3}}}{\sqrt{1 - \delta_{4K}} - \frac{1}{\sqrt{3}}\sqrt{1 + \delta_{3K}}} = \frac{4}{\sqrt{3 - 3\delta_{4K}} - \sqrt{1 + \delta_{3K}}}.$$

□

À ce stade, il faut noter que la constante C_K reste raisonnable si δ_{4K} n'est pas proche de 1. Il faut d'ailleurs aussi noter que dans ce cas δ_{3K} n'est pas non plus proche de 1, car on peut toujours prendre $\delta_{3K} \leq \delta_{4K}$.

5.2.3 Résolution numérique : l'algorithme du gradient proximal

Le problème (5.10) peut, en fait, être résolute de manière exacte par des méthodes d'homotopie en exploitant la structure de ce problème. Par contre, ce calcul n'est faisable en un temps acceptable que dans les cas où sa solution reste très parcimonieuse (i.e. à très peu de coordonnées non nulles). Lorsque ce n'est pas le cas, on peut utiliser un algorithme itératif pour approcher une solution de (5.10).

Un nombre important d'algorithmes a été proposé pour résoudre (5.10). Ils reposent souvent sur un argument de dualité lagrangienne qui permet de récrire (5.10) sous la forme

$$\|x\|_1 + \frac{\beta}{2}\|Dx - u\|^2, \tag{5.17}$$

pour une valeur de β bien choisie. Ce problème est convexe, coercif mais non-différentiable. Il n'est cependant pas difficile de voir qu'il peut être résolu avec l'algorithme du gradient proximal vu au Chapitre 2.1.4.

Pour le mettre en œuvre, il faut être capable de calculer la plus grande valeur singulière de D (i.e. son rayon spectral) et l'opérateur proximal $\text{prox}^L_{\|.\|_1}$. On peut généralement facilement calculer la plus grande valeur singulière de D à l'aide de l'algorithme des puissances. Il n'est pas difficile d'établir une formule explicite pour le calcul de $\text{prox}^L_{\|.\|_1}$.

On a en effet

$$\begin{aligned} \text{prox}^L_{\|.\|_1}(w') &= \text{Argmin}_{w \in W}\ \frac{L}{2}\|w - w'\|_2^2 + \|w\|_1, \\ &= \text{Argmin}_{w \in W} \sum_{i=1}^{P} \left(\frac{L}{2}(w_i - w'_i)^2 + |w_i|\right). \end{aligned}$$

On en déduit donc facilement que chaque coordonnée i de $\text{prox}^L_{\|.\|_1}(w')$ est le minimiseur de

$$\text{prox}^L_{\|.\|_1}(w')_i = \text{Argmin}_{w_i \in \mathbb{R}} \frac{L}{2}(w_i - w'_i)^2 + |w_i|.$$

Il n'est alors pas difficile de voir (en distinguant chacun des trois cas) que

$$\text{prox}^L_{\|.\|_1}(w')_i = \begin{cases} w'_i + \frac{1}{L} & \text{, si } w'_i < -\frac{1}{L}, \\ 0 & \text{, si } -\frac{1}{L} \leq w'_i \leq \frac{1}{L}, \\ w'_i - \frac{1}{L} & \text{, si } w'_i > \frac{1}{L}. \end{cases}$$

Ceci correspond à un simple seuillage doux de chacune des coordonnées de w'.

Finalement, on obtient un algorithme dont on peut prouver la convergence et pour lequel on a une vitesse de convergence (sur la fonction objectif) en $\frac{1}{k}$ (voir Théorème 3).

5.3 Les algorithmes de type "Orthogonal Matching Pursuit"

Une alternative purement algorithmique au problème (5.2) est l'orthogonal matching pursuit (OMP). En fait, il y a toute une famille d'algorithmes se ressemblant : Matching Pursuit, stagewise OMP, CO-SAMP...

Le point commun entre ces algorithmes est de sélectionner à chaque itération une ou plusieurs coordonnées parmi les plus grandes coordonnées (en valeur absolue) de $D^t r$, où $r = u - Dx$ et x est la solution courante. Intuitivement, r contient la partie de u qui n'est pas encore présente dans Dx. Chaque coordonnée de $D^t r$ représente donc la corrélation entre un élément du dictionnaire et ce qu'il reste à exprimer. Cela revient à choisir les vecteurs "pointant" le plus dans la direction de r.

Une fois qu'un ensemble d'éléments du dictionnaire est sélectionné, on construit la solution en projetant u orthogonalement sur le sous-espace vectoriel engendré par ces éléments. Plus précisément, on note

- $\mathcal{S} \subset \{1, \ldots, P\}$,
- $\text{Vect}(\mathcal{S})$ le sous-espace vectoriel de $\mathbb{R}^N$ engendré par les colonnes de D dont l'indice est dans $\mathcal{S}$
- $P_{\text{Vect}(\mathcal{S})}(v)$ la projection orthogonale d'un élément $v \in \mathbb{R}^{N^2}$ sur $\text{Vect}(\mathcal{S})$ et est défini par

$$P_{\text{Vect}(\mathcal{S})}(v)_i = \begin{cases} \tilde{x}_i & \text{, si } i \in \mathcal{S} \\ 0 & \text{, sinon,} \end{cases} \tag{5.18}$$

où

$$\tilde{x} = \text{Argmin}_{x \in \mathbb{R}^{\mathcal{S}}} \|v - D^{\mathcal{S}} x\|_2^2, \tag{5.19}$$

dans lequel $D^{\mathcal{S}}$ désigne la matrice obtenue en extrayant de D les colonnes dont l'indice est dans $\mathcal{S}$.

Dans les algorithmes de la famille Orthogonal Matching Pursuit, on construit itérativement $\mathcal{S}$ comme indiqué ci-dessus et on calcule les coordonnées de $P_{\text{Vect}(\mathcal{S})}(u)$ à l'aide de (5.18). Il est important de noter que dans (5.19) le calcul de $\tilde{x}$, n'est pas très difficile. En effet, il s'agit d'un simple problème de minimisation d'une fonction quadratique dont le gradient vaut

$$-(D^{\mathcal{S}})^T(v - D^{\mathcal{S}} x).$$

Si l'algorithme OMP est arrêté suffisamment tôt, un point critique

$$\tilde{x} = [(D^{\mathcal{S}})^T D^{\mathcal{S}}]^{\dagger} (D^{\mathcal{S}})^T v,$$

est aussi une solution de (5.19). Cette dernière expression est facile a calculer lorsque $|\mathcal{S}|$ est petit. La matrice $(D^{\mathcal{S}})^T D^{\mathcal{S}}$ est en effet de taille $|\mathcal{S}| \times |\mathcal{S}|$. Cependant, dans certains cas (notamment lorsque la solution n'est pas très parcimonieuse), le calcul de $\tilde{x}$ peut être un peu long. Bien souvent, une algorithmique particulière est nécessaire pour accélérer ce calcul.

Nous présentons l'Orthogonal Matching Pursuit dans la Table 5.1.

- Entrées : La donnée u, une matrice D, un paramètre de critère d'arrêt.
- Sortie : Les coordonnées x
- L'algorithme :
  - On initialise $\mathcal{S} = \emptyset$, $r = u$
  - Tant que le critère d'arrêt n'est pas satisfait :
    - On calcule $D^t r$
    - On cherche i correspondant au maximum de $|D^t r|$
    - On met à jour $\mathcal{S} \leftarrow \mathcal{S} \cup \{i\}$
    - On calcule les coordonnées x de la projection $P_{\text{Vect}(\mathcal{S})}(u)$ et le nouveau résidu $r = u - P_{\text{Vect}(\mathcal{S})}(u)$

TABLE 5.1 – L'algorithme Orthogonal Matching Pursuit.

Chapitre 6

Éléments d'optimisation de formes

L'optimisation de formes est un domaine important des mathématiques appliquées. Il a un nombre important d'applications telles que :

- la segmentation d'images ;
- le design d'objets : par exemple, construire un objet supportant des contraintes physiques ;
- l'infographie : par exemple, pour construire ou représenter la surface d'un objet que l'on veut visualiser ;
- etc

Il existe un nombre important de méthodes d'optimisation de formes et nous ne verrons ici que deux exemples parmi beaucoup d'autres.

Il faut noter que l'optimisation de formes ne suffit pas à minimiser les énergies que nous rencontrons dans la section 7 (voir (7.5) et (7.6)). La résolution de ces deux problèmes est généralement faite en itérant la succession

- d'une étape de minimisation d'une énergie (souvent convexe) sur des paramètres appartenant à un espace de Hilbert (typiquement : w pour (7.5) ; c_1 et c_2 pour (7.6)).
- et d'une étape d'optimisation de la forme Ω.

C'est cette seconde étape qui nous intéresse ici.

6.1 Par des ensembles de niveau

6.1.1 La représentation de Ω

Les méthodes basées sur les ensembles de niveau représentent l'ensemble $\Omega \subset \{1,\dots,N\}^2$ comme un ensemble de niveau d'une image $\varphi \in \mathbb{R}^{N^2}$. On pose donc

$$\Omega = \left\{(m,n) \in \{1,\dots,N\}^2, \varphi_{m,n} \geq 0\right\}.$$

Ainsi, faire évoluer Ω afin de minimiser une énergie revient à faire évoluer l'image φ pour minimiser une énergie analogue mais portant sur φ. La construction des énergies en φ est décrite ci-dessous.

On ré-écrit le terme de la forme

$$\sum_{(m,n)\in\Omega} a_{m,n}$$

de (7.5), (7.6) ou (7.7), sous la forme

$$\sum_{m,n=0}^{N} a_{m,n}\mathbf{1}_{|\varphi_{m,n}\geq 0}, \tag{6.1}$$

où, pour tout $(m,n) \in \{1,\dots,N\}^2$

$$\mathbf{1}_{|\varphi_{m,n}\geq 0} = \begin{cases} 1 & \text{, si } \varphi_{m,n} \geq 0, \\ 0 & \text{, si } \varphi_{m,n} < 0. \end{cases}$$

De même, on exprime dans (7.5), (7.6) ou (7.7) le terme de la forme

$$\sum_{(m,n)\notin\Omega} a_{m,n}$$

à l'aide de

$$\sum_{m,n=0}^{N} a_{m,n}\left(1-\mathbf{1}_{|\varphi_{m,n}\geq 0}\right). \tag{6.2}$$

Enfin, pour le terme de (7.5), (7.6) ou (7.7) correspondant au périmètre de Ω, on a (du fait de (7.3) et (7.4))

$$\sum_{((m,n),(m',n'))\in\partial\Omega} dl\left((m,n),(m',n')\right) = \sum_{\substack{(m,n)\in\Omega \\ (m',n')\notin\Omega}} dl\left((m,n),(m',n')\right).$$

On utilise donc une expression de la forme

$$\sum_{m,n=0}^{N}\sum_{m',n'=0}^{N} dl\left((m,n),(m',n')\right)\mathbf{1}_{|\varphi_{m,n}\geq 0}\left(1-\mathbf{1}_{|\varphi_{m',n'}\geq 0}\right). \tag{6.3}$$

Les ré-écritures ci-dessus ne se prêtent pas encore à l'optimisation car elles ne sont pas différentiables. Pour contourner ce problème, on approxime dans (6.1), (6.2) et (6.3) la fonction

$$\mathbf{1}_{|\varphi_{m,n}\geq 0}$$

par

$$H_\varepsilon\left(\varphi_{m,n}\right),$$

avec

$$H_\varepsilon(t) = \begin{cases} 1 & \text{, si } t \geq \varepsilon, \\ \frac{1}{2}\left(1+\frac{t}{\varepsilon}+\frac{1}{\pi}\sin\left(\frac{\pi t}{\varepsilon}\right)\right) & \text{, si } -\varepsilon \leq t \leq \varepsilon, \\ 0 & \text{, si } t \leq -\varepsilon. \end{cases}$$

On représente sur la Figure 6.1 des exemples de graphe de fonction H_ε, pour $\varepsilon = 0.1$, 1 et 5. Les fonctions H_ε approximent la fonction de Heaviside et sont dérivables. On peut d'ailleurs calculer leur dérivée qui vaut

$$H'_\varepsilon(t) = \begin{cases} 0 & \text{, si } t \geq \varepsilon, \\ \frac{1}{2\varepsilon}+\frac{1}{2\varepsilon}\cos\left(\frac{\pi t}{\varepsilon}\right) & \text{, si } -\varepsilon \leq t \leq \varepsilon, \\ 0 & \text{, si } t \leq -\varepsilon. \end{cases}$$

6.1.2 Faire évoluer Ω

On fait évoluer φ (et donc Ω) en appliquant un algorithme de gradient à la fonctionnelle que l'on cherche à minimiser. Il faut noter que tout ce que l'on peut espérer est de trouver un minimum local de l'énergie car celle-ci n'est pas convexe. Ainsi, comme souvent en optimisation de formes, il est très important de partir d'une bonne initialisation de la forme.

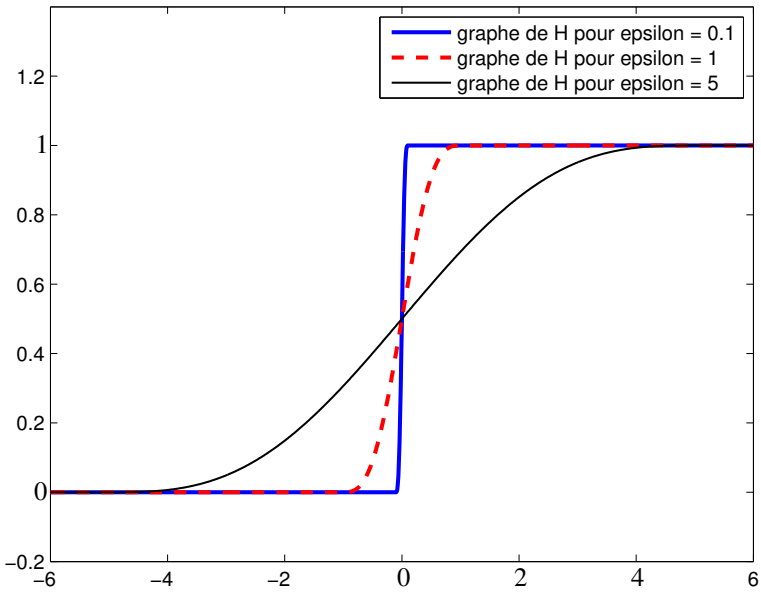


FIGURE 6.1 – Exemples de graphes des fonctions H_ε.

On voit facilement que la fonction E_1 (qui correspond à (6.1)) ci-dessous est différentiable et on calcule facilement le gradient du terme

$$E_1((\varphi_{m,n})_{1\leq m,n\leq N}) = \sum_{m,n=0}^{N} a_{m,n} H_\varepsilon(\varphi_{m,n}),$$

en $\varphi = (\varphi_{m,n})_{1\leq m,n\leq N} \in \mathbb{R}^{N^2}$. On a en effet, pour tout $h = (h_{m,n})_{1\leq m,n\leq N} \in \mathbb{R}^{N^2}$,

$$\begin{aligned} E_1(\varphi+h) - E_1(\varphi) &= \sum_{m,n=0}^{N} a_{m,n}\left(H_\varepsilon(\varphi_{m,n}+h_{m,n}) - H_\varepsilon(\varphi_{m,n})\right), \\ &= \sum_{m,n=0}^{N} a_{m,n}\left(H'_\varepsilon(\varphi_{m,n})\, h_{m,n} + o(|h_{m,n}|)\right), \\ &= \sum_{m,n=0}^{N} a_{m,n} H'_\varepsilon(\varphi_{m,n})\, h_{m,n} + o(\|h\|_2), \\ &= \langle \nabla E_1(\varphi), h\rangle + o(\|h\|_2). \end{aligned}$$

En identifiant les deux dernières lignes du calcul ci-dessus, on obtient

$$\nabla E_1(\varphi) = \left(a_{m,n} H'_\varepsilon(\varphi_{m,n})\right)_{1\leq m,n\leq N}.$$

On voit facilement que la fonctionnelle correspondant à (6.2)

$$E_2((\varphi_{m,n})_{1\leq m,n\leq N}) = \sum_{m,n=0}^{N} a_{m,n}\left(1 - H_\varepsilon(\varphi_{m,n})\right),$$

est différentiable et on déduit son gradient en $\varphi = (\varphi_{m,n})_{1\leq m,n\leq N} \in \mathbb{R}^{N^2}$ du calcul du gradient de E_1. On trouve

$$\nabla E_2(\varphi) = \left(-a_{m,n} H'_\varepsilon(\varphi_{m,n})\right)_{1\leq m,n\leq N}.$$

Enfin, le terme de périmètre (voir (6.3))

$$E_3((\varphi_{m,n})_{1\leq m,n\leq N}) = \sum_{m,n=0}^{N} \sum_{m',n'=0}^{N} dl\left((m,n),(m',n')\right) H_\varepsilon(\varphi_{m,n}) \left(1 - H_\varepsilon\left(\varphi_{m',n'}\right)\right).$$

est aussi différentiable et on peut calculer son gradient en $\varphi = (\varphi_{m,n})_{1\leq m,n\leq N} \in \mathbb{R}^{N^2}$. On a en effet

$$\begin{aligned}
E_3(\varphi+h) &= \sum_{m,n=0}^{N} \sum_{m',n'=0}^{N} dl\left((m,n),(m',n')\right) H_\varepsilon(\varphi_{m,n}+h_{m,n}) \left(1 - H_\varepsilon\left(\varphi_{m',n'}+h_{m',n'}\right)\right), \\
&= \sum_{m,n=0}^{N} \sum_{m',n'=0}^{N} dl\left((m,n),(m',n')\right) \left[H_\varepsilon(\varphi_{m,n}) + H'_\varepsilon(\varphi_{m,n}) h_{m,n} + o(|h_{m,n}|)\right] \\
&\qquad \left[1 - H_\varepsilon\left(\varphi_{m',n'}\right) - H'_\varepsilon\left(\varphi_{m',n'}\right) h_{m',n'} + o\left(|h_{m',n'}|\right)\right], \\
&= E_3(\varphi) + o(\|h\|_2) \\
&+ \sum_{m,n=0}^{N} \sum_{m',n'=0}^{N} dl\left((m,n),(m',n')\right) \left[-H_\varepsilon(\varphi_{m,n}) H'_\varepsilon\left(\varphi_{m',n'}\right) h_{m',n'} + H'_\varepsilon(\varphi_{m,n}) h_{m,n}\left(1 - H_\varepsilon\left(\varphi_{m',n'}\right)\right)\right].
\end{aligned}$$

On intervertit les noms des variables du premier terme de la somme pour obtenir

$$\begin{aligned}
&\sum_{m,n=0}^{N} \sum_{m',n'=0}^{N} -dl\left((m,n),(m',n')\right) H_\varepsilon(\varphi_{m,n}) H'_\varepsilon\left(\varphi_{m',n'}\right) h_{m',n'} = \\
&\qquad \sum_{m',n'=0}^{N} \sum_{m,n=0}^{N} -dl\left((m',n'),(m,n)\right) H_\varepsilon\left(\varphi_{m',n'}\right) H'_\varepsilon(\varphi_{m,n}) h_{m,n}.
\end{aligned}$$

Dans un second temps, on change l'ordre des sommes et on utilise (7.3) pour finalement avoir

$$\begin{aligned}
&\sum_{m,n=0}^{N} \sum_{m',n'=0}^{N} -dl\left((m,n),(m',n')\right) H_\varepsilon(\varphi_{m,n}) H'_\varepsilon\left(\varphi_{m',n'}\right) h_{m',n'} = \\
&\qquad \sum_{m,n=0}^{N} \sum_{m',n'=0}^{N} -dl\left((m,n),(m',n')\right) H_\varepsilon\left(\varphi_{m',n'}\right) H'_\varepsilon(\varphi_{m,n}) h_{m,n}.
\end{aligned}$$

On obtient alors

$$\begin{aligned}
\langle \nabla E_3(\varphi), h\rangle = \sum_{m,n=0}^{N} \sum_{m',n'=0}^{N} &\left[-dl\left((m,n),(m',n')\right) H_\varepsilon\left(\varphi_{m',n'}\right) H'_\varepsilon(\varphi_{m,n}) h_{m,n}\right] \\
&+ \left[dl\left((m,n),(m',n')\right) H'_\varepsilon(\varphi_{m,n}) h_{m,n}\left(1 - H_\varepsilon\left(\varphi_{m',n'}\right)\right)\right].
\end{aligned}$$

On en déduit que pour tout $\varphi = (\varphi_{m,n})_{1\leq m,n\leq N} \in \mathbb{R}^{N^2}$

$$\begin{aligned}
\nabla E_3(\varphi) &= \left(\sum_{m',n'=0}^{N} dl\left((m,n),(m',n')\right) \left(-H_\varepsilon\left(\varphi_{m',n'}\right) H'_\varepsilon(\varphi_{m,n}) + H'_\varepsilon(\varphi_{m,n}) \left(1 - H_\varepsilon\left(\varphi_{m',n'}\right)\right)\right)\right)_{1\leq m,n\leq N}, \\
&= \left(\sum_{m',n'=0}^{N} dl\left((m,n),(m',n')\right) H'_\varepsilon(\varphi_{m,n}) \left(1 - 2H_\varepsilon\left(\varphi_{m',n'}\right)\right)\right)_{1\leq m,n\leq N}.
\end{aligned}$$

6.2 Algorithmique des graphes

L'algorithmique des graphes est un outil très utilisé en segmentation d'images. Dans certains cas, il présente en effet l'intérêt de fournir rapidement un minimiseur (global) de l'énergie minimisée. Nous présenterons ici les éléments d'algorithmique des graphes nécessaires pour faire cette optimisation. Il s'agit donc d'une sous partie très restreinte de l'algorithmique des graphes.

6.2.1 Définition des graphes considérés

Dans la suite, nous considérons un ensemble fini de sommets $\mathcal{V}$ (on note $\mathcal{V}$ pour "vertices") et un ensemble d'arcs $\mathcal{E} \subset (\mathcal{V} \times \mathcal{V})$. On suppose que, pour tout $(s,t) \in \mathcal{E}$, on a $s \neq t$. On remarque aussi qu'il n'y a pas d'arc multiple.

On définit et on note l'ensemble des successeurs d'un sommet $s \in \mathcal{V}$ par

$$\omega^+(s) = \{t \in \mathcal{V}, (s,t) \in \mathcal{E}\}.$$

On considère une valuation positive sur les arcs que l'on note

$$c : \mathcal{E} \to \mathbb{R}^+.$$

Suivant le contexte, la notation c correspondra à un coût (pour les algorithmes de plus court chemin) ou une capacité (pour les algorithmes de flot maximum).

On définit ainsi le graphe valué simple (il n'y a pas d'arc multiple) et sans boucle

$$\mathcal{G} = (\mathcal{V}, \mathcal{E}, c).$$

Un exemple de graphe est illustré sur la Figure 6.2.

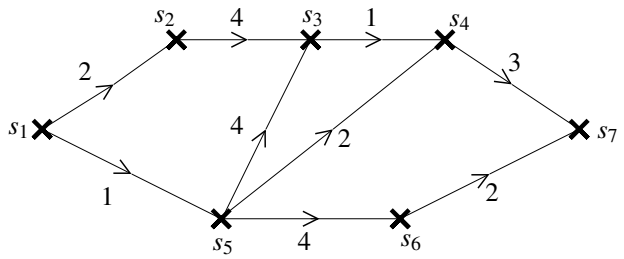


FIGURE 6.2 – Exemple d'un graphe $\mathcal{G} = (\mathcal{V}, \mathcal{E}, c)$.

Dans ce chapitre, nous verrons tout d'abord des algorithmes élémentaires comme les parcours dans des graphes. Dans un second temps, nous exposerons les notions élémentaires sur le problème du flot maximum dans un graphe. Enfin, nous exposerons comment construire des problèmes dans des graphes pour minimiser les fonctionnelles rencontrées en traitement d'images.

6.2.2 Parcours d'un graphe

Les parcours sont des opérations de base en algorithmique des graphes. Ils sont par exemple utilisés pour faire une opération sur chaque sommet d'un graphe, déterminer des parties connexes, etc. L'algorithme de Dijkstra est un parcours dont l'intérêt est de calculer les plus courts chemins (ou le chemin de moindre coût) à un sommet donné, lorsque la valuation c est interprétée comme une longueur (ou un coût).

Parcours en profondeur

Le parcours en profondeur (Depth first search ou DFS) cherche un nouveau sommet (non parcouru) parmi les successeurs du dernier sommet parcouru. S'il n'y en a pas, il cherche parmi les sommets de l'avant dernier sommet parcouru... et ainsi de suite. Si tous les successeurs des sommets déjà parcourus, il cherche un nouveau sommet d'où partir et s'il n'en trouve pas, s'arrête.

Il est décrit dans les Algorithmes 5 et 6. L'Algorithme 6 permet de parcourir tous les sommets qui peuvent être atteints à partir de s. L'Algorithme 5 appelle l'Algorithme 6 tant qu'il reste des sommets à parcourir. La complexité pour un parcours de l'ensemble des sommets (généralement implémenter à l'aide d'une pile) est $O(|\mathcal{V}|+|\mathcal{E}|)$.

Algorithme 5 Parcours en profondeur

Procédure : $DFS(\mathcal{G})$

Initialisation : $\forall s \in \mathcal{V}, marque(s) \leftarrow 0$
for $s \in \mathcal{V}$ **faire**
  if $marque(s) = 0$ **then**
    $DFS\_loc(\mathcal{G}, s)$
  end if
end for

Algorithme 6 Parcours en profondeur à partir de s

Procédure : $DFS\_loc(\mathcal{G}, s)$

$marque(s) \leftarrow 1$
for $t \in \omega^+(s)$ **faire**
  if $marque(t) = 0$ **then**
    $DFS\_loc(\mathcal{G}, t)$
  end if
end for

Ce parcours est par exemple utile pour déterminer les parties connexes d'un graphe ou des partie connexe d'une segmentation d'une image.

Il existe aussi un parcours en largeur en $O(|\mathcal{V}|+|\mathcal{E}|)$. Ces deux parcours présentent des intérêts différents que nous n'aborderons pas ici.

Dijkstra : Parcours fournissant le meilleur chemin

Dans un graphe $\mathcal{G} = (\mathcal{V}, \mathcal{E}, c)$, on appelle chemin une suite d'arcs (s_0, s_1), (s_1, s_2),..., (s_{K-1}, s_K), où, pour tout $k \in \{1, \ldots, K\}$, $(s_{k-1}, s_k) \in \mathcal{E}$. On note un tel chemin $s_0 - s_1 - \ldots - s_K$. On définit le coût d'un chemin $s_0 - s_1 - \ldots - s_K$ dans un graphe $\mathcal{G} = (\mathcal{V}, \mathcal{E}, c)$, par

$$c(s_0 - s_1 - \ldots - s_K) = \sum_{k=1}^{K} c_{(s_{k-1}, s_k)}.$$

Lorsque le coût d'un arc représente une distance, le coût d'un chemin représente sa longueur.

Étant donné un sommet $s_i \in \mathcal{V}$ fixé, l'algorithme de Dijkstra permet de parcourir le graphe $\mathcal{G}$ en commençant par les sommets les plus proches du sommet s_i. Il permet aussi de calculer les distances,

et un plus court chemin, entre s_i et chaque sommet de $\mathcal{V}$. Comme on peut s'y attendre, les sommets ne pouvant être atteints depuis s_i sont à une distance infinie de s_i. L'algorithme de Dijkstra est décrit dans l'Algorithme 7. Sa complexité dépend de la stratégie utilisée pour trouver s_{min}. Des implémentations optimisées, utilisant des "tas" ("heaps" en anglais) pour rechercher l'argmin, permettent d'obtenir des algorithmes en $O(|\mathcal{E}|+|\mathcal{V}|ln|\mathcal{V}|)$.

Algorithme 7 Algorithme de Dijkstra

Procedure : $Dijkstra(\mathcal{G}, s_i)$

Initialisation : $\forall s \in \mathcal{V}, distance(s) \leftarrow \infty$
Initialisation : $distance(s_i) \leftarrow 0$
Initialisation : $\forall s \in \mathcal{V}, predecesseur(s) \leftarrow s$
Initialisation : $\forall s \in \mathcal{V}, marque(s) \leftarrow 0$
Tant que il existe s, tel que $marque(s) = 0$ **faire**
  Choisir $s_{min} \in \text{Argmin}_{s:marque(s)=0} distance(s)$
  $marque(s_{min}) \leftarrow 1$
  for $s \in \omega^+(s_{min})$ et $marque(s) = 0$ **faire**
    if $distance(s) > distance(s_{min}) + c_{(s_{min},s)}$ **then**
      $distance(s) \leftarrow distance(s_{min}) + c_{(s_{min},s)}$
      $predecesseur(s) \leftarrow s_{min}$
    end if
  end for
Fin tant que

Il est important de remarquer que si un ensemble S de sommets est initialisé avec une distance égale à 0, alors l'algorithme calcule la distance à S et construit le plus court chemin pour atteindre S.

L'algorithme de Dijkstra est utilisé en segmentation d'images. Les algorithmes l'utilisant sont en général appelés "Fast Marching". Le principe est de construire une distance (un coût) c telle que les pixels de l'objet à segmenter sont proches entre eux mais sont loin des pixels du fond.

6.2.3 Flot maximum/coupe minimale dans un graphe

Principes du problème de flot max

Pour un problème de flot maximum, on considère un graphe

$$\mathcal{G} = (\mathcal{V} \cup \{S,T\}, \mathcal{E}, c),$$

où

$$\mathcal{E} \subset (\mathcal{V} \cup \{S\}) \times (\mathcal{V} \cup \{T\}),$$

et dans lequel, pour $(s,s') \in \mathcal{E}$, la quantité $c_{(s,s')}$ est interprétée comme la capacité de l'arc (s,s'). Les sommets S et T s'appellent respectivement la source et le puits. On considère un flot

$$f : \mathcal{E} \to \mathbb{R}^+.$$

Intuitivement le flot coule, le long des arcs, de la source S jusqu'au puits T.

Un flot valide vérifie les deux conditions suivantes :

– la contrainte de capacité :

$$\forall (s,s') \in \mathcal{E}, \quad 0 \leq f_{(s,s')} \leq c_{(s,s')};$$

– la conservation du flot en chaque sommet :

$$\forall s \in \mathcal{V}, \quad \sum_{s' \in \omega^+(s)} f_{(s,s')} = \sum_{s' : s \in \omega^+(s')} f_{(s',s)}.$$

Il n'est pas difficile de voir que l'ensemble des flots valides forme un polygone compact de $\mathbb{R}^{\mathcal{E}}$.

Le problème du flot maximum consiste à maximiser le flot sortant du sommet S, parmi les flots valides. Il s'écrit sous la forme du programme linéaire

$$\begin{cases} \max_{f \in \mathbb{R}^{\mathcal{E}}} & \sum_{s \in \omega^+(S)} f_{(S,s)}, \\ \forall (s,s') \in \mathcal{E}, & 0 \leq f_{(s,s')} \leq c_{(s,s')}, \\ \forall s \in \mathcal{V}, & \sum_{s' \in \omega^+(s)} f_{(s,s')} = \sum_{s' : s \in \omega^+(s')} f_{(s',s)}. \end{cases} \tag{6.4}$$

Ce problème a naturellement une solution car il consiste à maximiser une fonction linéaire (donc continue et même concave) sur un ensemble compact. Il existe de nombreux algorithmes de maximisation pour le résoudre. Les principales approches sont les algorithmes de flot augmentant et les algorithmes de "push-relabel". Ils fournissent une solution exacte en un temps polynomial en fonction de la taille du graphe. La complexité (pire cas) des meilleurs algorithmes calculant un flot maximum est $O(|\mathcal{V}||\mathcal{E}|\ln|\mathcal{V}|)$ (une borne inférieure de la complexité est $O(|\mathcal{V}||\mathcal{E}|)$). Aussi, il existe des algorithmes et des implémentations spécialement dédiées aux graphes rencontrés en traitement d'images (voir [6]). La complexité observée, pour ces graphes, est typiquement $O(|\mathcal{V}|)$.

Le problème du flot maximum est le dual du problème de coupe minimale. Pour définir ce dernier, on définit une S-T-coupe $(\mathcal{S},\mathcal{T})$ de $\mathcal{G}$ comme une partition de $\mathcal{V}$ telle que $S \in \mathcal{S}$ et $T \in \mathcal{T}$. On définit la valeur d'une S-T-coupe $(\mathcal{S},\mathcal{T})$, par

$$\mathrm{Val}(\mathcal{S},\mathcal{T}) = \sum_{(s,t) \in \mathcal{E} \cap (\mathcal{S} \times \mathcal{T})} c_{(s,t)}.$$

La quantité $\mathrm{Val}(\mathcal{S},\mathcal{T})$ représente donc la quantité maximale de flot pouvant passer de $\mathcal{S}$ à $\mathcal{T}$. Le théorème suivant fait le lien entre le flot maximum et la coupe minimale.

Théorème 9 (Ford Fulkerson) *La valeur du flot maximum dans le graphe $\mathcal{G}$ est égale à la valeur des S-T-coupes de valeur minimale.*

En fait, on a même plus que cela puisque tout flot maximisant (6.4) sature des arcs de manière à ce qu'il ne soit plus possible de faire passer du flot de S à T. Ce faisant, il définit une S-T-coupe minimale. Ainsi, les algorithmes permettant le calcul d'un flot maximum fournissent également une S-T-coupe minimale.

Minimisation d'un Champ de Markov aléatoire binaire

Cette partie reformule essentiellement des résultats présentés dans [5]. On considère un champ de Markov aléatoire binaire défini pour tout $w \in \{0,1\}^{\mathcal{V}}$ par

$$E(w) = \sum_{s \in \mathcal{V}} E_s(w_s) + \sum_{(s,s') \in \mathcal{E}_n} E_{(s,s')}(w_s, w_{s'}),$$

où $\mathcal{E}_n \subset (\mathcal{V} \times \mathcal{V})$. Le but de cette section est de montrer le résultat suivant :

Théorème 10 (Kolmogorov-Zabih) *Si pour tout $(s,s') \in \mathcal{E}_n$*

$$E_{(s,s')}(0,0) + E_{(s,s')}(1,1) \leq E_{(s,s')}(0,1) + E_{(s,s')}(1,0) \tag{6.5}$$

il est possible de minimiser E de manière exacte en un temps polynomial, à l'aide d'un algorithme de flot maximum dans un graphe $\mathcal{G}$ dont le nombre de sommets est $|\mathcal{V}|+2$ et le nombre d'arcs est majoré par $|\mathcal{E}_n|+|\mathcal{V}|$.

On dit qu'une énergie E satisfaisant (6.5) *est sous-modulaire ("submodular" en anglais).*

Preuve. Pour prouver ce résultat, nous commençons par réécrire E sous une forme équivalente. Puis nous construisons un graphe $\mathcal{G}$ tel que les éléments de $\{0,1\}^{\mathcal{V}}$ et les S-T-coupes de $\mathcal{G}$ soient en correspondance univoque ; et tel que (modulo une constante additive) la valeur de E pour un élément de $\{0,1\}^{\mathcal{V}}$ soit égale à la valeur de la S-T-coupe correspondante. Ainsi, on aura montré que minimiser E est équivalent à chercher une coupe minimale dans $\mathcal{G}$; ce que l'on sait faire en un temps polynomial grâce à un algorithme de flot maximum.

Commençons donc par reformuler E : Si l'on note, pour tout $(s,s') \in \mathcal{E}_n$,

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix} = \begin{pmatrix} E_{(s,s')}(0,0) & E_{(s,s')}(0,1) \\ E_{(s,s')}(1,0) & E_{(s,s')}(1,1) \end{pmatrix},$$

on remarque que

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix} = \begin{pmatrix} A & A \\ C & C \end{pmatrix} + \begin{pmatrix} C & D \\ C & D \end{pmatrix} + \begin{pmatrix} 0 & B+C-A-D \\ 0 & 0 \end{pmatrix} - C.$$

Remarquez aussi que, grâce à l'hypothèse (6.5), on $B+C-A-D \geq 0$.

Ainsi, on peut réécrire, pour tout $(w_s, w_{s'}) \in \{0,1\}^2$,

$$E_{(s,s')}(w_s, w_{s'}) = E_s^{(s,s')}(w_s) + E_{s'}^{(s,s')}(w_{s'}) + E'_{(s,s')}(w_s, w_{s'}) - C,$$

avec

$$E_s^{(s,s')}(w_s) = \begin{cases} E_{(s,s')}(0,0) & \text{, si } w_s = 0, \\ E_{(s,s')}(1,0) & \text{, si } w_s = 1, \end{cases} \quad \text{et} \quad E_{s'}^{(s,s')}(w_{s'}) = \begin{cases} E_{(s,s')}(1,0) & \text{, si } w_{s'} = 0, \\ E_{(s,s')}(1,1) & \text{, si } w_{s'} = 1, \end{cases}$$

et

$$E'_{(s,s')}(w_s, w_{s'}) = \begin{cases} E_{(s,s')}(1,0) + E_{(s,s')}(0,1) - E_{(s,s')}(0,0) - E_{(s,s')}(1,1) & \text{, si } (w_s, w_{s'}) = (0,1), \\ 0 & \text{, sinon.} \end{cases} \tag{6.6}$$

Si l'on fait cela pour tout $(s,s') \in \mathcal{E}_n$, on obtient finalement pour tout $w \in \{0,1\}^{\mathcal{V}}$,

$$\begin{array}{rcl} E(w) & = & \sum_{s \in \mathcal{V}} \left(E_s(w_s) + \sum_{s' : (s,s') \in \mathcal{E}_n} E_s^{(s,s')}(w_s) + \sum_{s' : (s',s) \in \mathcal{E}_n} E_s^{(s',s)}(w_s) \right) \\ & + & \sum_{(s,s') \in \mathcal{E}_n} E'_{(s,s')}(w_s, w_{s'}) \\ & - & \sum_{(s,s') \in \mathcal{E}_n} E_{(s,s')}(1,0). \end{array}$$

Nous notons maintenant, pour tout $s \in \mathcal{V}$,

$$E''_s(w_s) = E_s(w_s) + \sum_{s' : (s,s') \in \mathcal{E}_n} E_s^{(s,s')}(w_s) + \sum_{s' : (s',s) \in \mathcal{E}_n} E_s^{(s',s)}(w_s).$$

Pour tout $s \in \mathcal{V}$, on note aussi :

$$\text{si } E''_s(1) \geq E''_s(0), \quad E'_s(w_s) = \begin{cases} E''_s(1) - E''_s(0) & \text{, si } w_s = 1, \\ 0 & \text{, si } w_s = 0, \end{cases} \tag{6.7}$$

et

$$\text{si } E''_s(1) < E''_s(0), \quad E'_s(w_s) = \begin{cases} 0 & \text{, si } w_s = 1, \\ E''_s(0) - E''_s(1) & \text{, si } w_s = 0. \end{cases} \tag{6.8}$$

On note enfin

$$K = - \sum_{(s,s') \in \mathcal{E}_n} E_{(s,s')}(1,0) + \sum_{s \in \mathcal{V}} \min \left(E''_s(0), E''_s(1) \right).$$

Il n'est alors pas difficile de voir que l'on a, pour tout $w \in \{0,1\}^{\mathcal{V}}$,

$$E(w) = \sum_{s\in\mathcal{V}} E'_s(w_s) + \sum_{(s,s')\in\mathcal{E}_n} E'_{(s,s')}(w_s, w_{s'}) + K,$$

où E' est définie par (6.7), (6.8) et (6.6).

Remarquez que les termes composant cette écriture de E sont tels que

- E'_s est soit positive, soit nulle ;
- et $E'_{(s,s')}$ est positive en $(0,1)$, et nulle sinon.

C'est grâce à ces deux propriétés que nous pourrons construire le graphe $\mathcal{G}$ permettant de minimiser E. Plus précisément, nous construisons maintenant un graphe $\mathcal{G} = (\mathcal{V} \cup \{S,T\}, \mathcal{E}, c)$ tel que pour toute S-T-coupe $(\mathcal{S},\mathcal{T})$ de $\mathcal{G}$, on ait

$$\mathrm{Val}(\mathcal{S},\mathcal{T}) = E(w^{(\mathcal{S},\mathcal{T})}) - K = \sum_{s\in\mathcal{V}} E'_s(w_s^{(\mathcal{S},\mathcal{T})}) + \sum_{(s,s')\in\mathcal{E}_n} E'_{(s,s')}(w_s^{(\mathcal{S},\mathcal{T})}, w_{s'}^{(\mathcal{S},\mathcal{T})}),$$

où

$$w_s^{(\mathcal{S},\mathcal{T})} = \begin{cases} 0 & \text{, si } s \in \mathcal{S}, \\ 1 & \text{, si } s \in \mathcal{T}. \end{cases} \tag{6.9}$$

Remarquez, tout d'abord que $(\mathcal{S},\mathcal{T}) \mapsto w^{(\mathcal{S},\mathcal{T})}$ fait une correspondance univoque entre les S-T-coupes de $\mathcal{G}$ et les éléments de $\{0,1\}^{\mathcal{V}}$. Ainsi, manipuler des S-T-coupes de $\mathcal{G}$ est équivalent à manipuler des éléments de $\{0,1\}^{\mathcal{V}}$.

Pour construire les arcs $\mathcal{E}$ de $\mathcal{G}$, on pose

$$\begin{array}{ll} \mathcal{E} = \mathcal{E}_n & \cup\{(S,s), \text{ pour } s \in \mathcal{V} \text{ tel que } E'_s(1) > 0\} \\ & \cup\{(s,T), \text{ pour } s \in \mathcal{V} \text{ tel que } E'_s(0) > 0\} \end{array} \tag{6.10}$$

On définit les capacités c de $\mathcal{G}$, par

$$\begin{cases} c_{s,s'} = E'_{(s,s')}(0,1) & \forall (s,s') \in \mathcal{E}_n, \\ c_{S,s} = E'_s(1) & \forall s \in \mathcal{V} \text{ tel que } (S,s) \in \mathcal{E}, \\ c_{s,T} = E'_s(0) & \forall s \in \mathcal{V} \text{ tel que } (s,T) \in \mathcal{E}. \end{cases} \tag{6.11}$$

Les différentes situations sont représentées sur la Figure 6.3.

Par exemple, pour le sommet s le plus à gauche, aux vues de (6.10) et (6.11), on est dans la situation où $E_s(1) = 0$ et $E_s(0) \geq 0$. On a bien, en effet, que

- si une S-T-coupe $(\mathcal{S},\mathcal{T})$ est telle que $s \in \mathcal{T}$ (la coupe passe au dessus de s), on ne rajoute rien à la valeur de la coupe, ce qui est bien ce qu'il faut faire dans cette situation, puisque $w_s^{(\mathcal{S},\mathcal{T})} = 1$ (du fait de (6.9)).
- Inversement, si une S-T-coupe $(\mathcal{S},\mathcal{T})$ est telle que $s \in \mathcal{S}$ (la coupe passe au dessous de s), on rajoute $E'_s(0)$ à la valeur de la coupe, ce qui est bien ce qu'il faut faire dans cette situation, puisque $w_s^{(\mathcal{S},\mathcal{T})} = 0$ (du fait de (6.9)).

On peut faire le même raisonnement dans le cas du sommet s au centre de la Figure 6.3.

Enfin, pour les sommets s et s', tels qu'ils sont représentés sur le graphe le plus à droite de la Figure 6.3, les seules coupes pour lesquelles l'arc (s,s') contribue à la valeur de la coupe est lorsque $s \in \mathcal{S}$ et $s' \in \mathcal{T}$. On est donc, du fait de (6.9), dans un cas où $w_s^{(\mathcal{S},\mathcal{T})} = 0$ et $w_{s'}^{(\mathcal{S},\mathcal{T})} = 1$. Il faut donc bien rajouter $E'_{(s,s')}(0,1)$ pour coïncider avec E. Enfin, pour toutes les autres configurations d'une S-T-coupe, rien n'est ajouté à la valeur de la coupe ; ceci correspond bien à ce que fait $E'_{s,s'}$ lorsque $(w_s, w_{s'}) = (0,0)$, $(1,0)$ ou $(1,1)$.

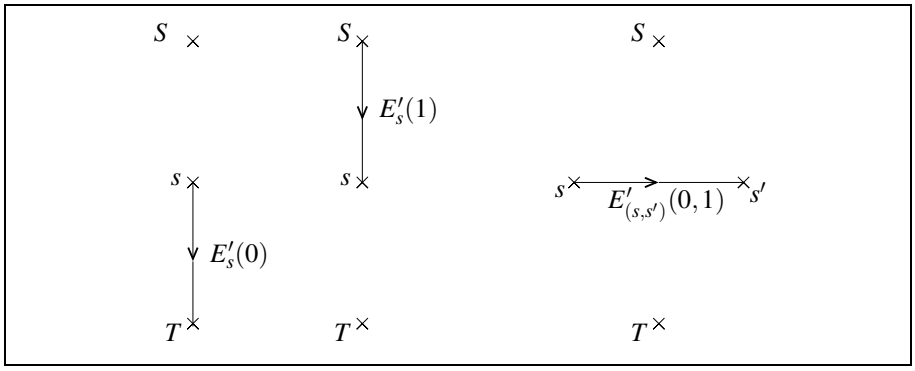


FIGURE 6.3 – Illustration des contributions des sommets s et s' en fonction de la coupe. Pour comprendre cette figure, il faut se représenter que, du fait de (6.9), $w_s = 0$ est équivalent au fait que la coupe passe entre s et T; et $w_s = 1$ est équivalent au fait que la coupe passe entre S et s.

□

Le résultat du Théorème 10, peut-être adapté et étendu aux cas de champs de Markov binaires contenant des termes d'interactions entre trois sommets s, s' et s".

On peut montrer que si E contient des termes qui ne sont pas sous-modulaires, alors la minimisation de E est un problème NP.

Minimisation approchée d'un Champ de Markov aléatoire multi-labels

Cette section reprend des algorithmes présentés dans [2] en intégrant la construction de [5]. On considère maintenant un champ de Markov aléatoire multi-labels défini pour tout $w \in \{0,1,\ldots,L\}^{\mathcal{V}}$, pour $L \geq 2$ par

$$E(w) = \sum_{s\in\mathcal{V}} E_s(w_s) + \sum_{(s,s')\in\mathcal{E}_n} E_{(s,s')}(w_s, w_{s'}),$$

où $\mathcal{E}_n \subset (\mathcal{V}\times\mathcal{V})$. La minimisation d'un tel champ de Markov est malheureusement un problème NP. Nous ne présenterons donc que des heuristiques pour le résoudre. Nous présenterons l'α-expansion et donnerons le principe de l'algorithme de l'α-β-swap.

L'algorithme α-expansion consiste à proposer, itérativement sur les labels $\alpha \in \{0,\ldots,L\}$, à tous les sommets $s \in \mathcal{V}$ de passer à la valeur α. Dans la suite, nous décrivons une étape, pour un α donné. On note w, l'élément de $\{0,1,\ldots,L\}^{\mathcal{V}}$ initial et $\tilde{w}$ de $\{0,1,\ldots,L\}^{\mathcal{V}}$, l'élément dont certains sommets $s \in \mathcal{V}$ ont pris la valeur α. On utilise un élément $u \in \{0,1\}^{\mathcal{V}}$ pour déterminer si un sommet $s \in \mathcal{V}$ passe à la valeur α. On définit pour cela, pour tout $s \in \mathcal{V}$,

$$w_s^u = \begin{cases} \alpha & \text{, si } u_s = 1, \\ w_s & \text{, si } u_s = 0. \end{cases}$$

On peut alors facilement construire un champ de Markov binaire E' tel que, pour tout $u \in \{0,1\}^{\mathcal{V}}$, $E'(u) = E(w^u)$ en posant, pour tout $u \in \{0,1\}^{\mathcal{V}}$:

$$E'(u) = \sum_{s\in\mathcal{V}} E'_s(u_s) + \sum_{(s,s')\in\mathcal{E}_n} E'_{(s,s')}(u_s, u_{s'}), \tag{6.12}$$

avec

$$E'_s(0) = E_s(w_s) \quad \text{et} \quad E'_s(1) = E_s(\alpha),$$

et

$$\begin{pmatrix} E'_{(s,s')}(0,0) & E'_{(s,s')}(0,1) \\ E'_{(s,s')}(1,0) & E'_{(s,s')}(1,1) \end{pmatrix} = \begin{pmatrix} E_{(s,s')}(w_s,w_{s'}) & E_{(s,s')}(w_s,\alpha) \\ E_{(s,s')}(\alpha,w_{s'}) & E_{(s,s')}(\alpha,\alpha) \end{pmatrix}.$$

Si le champ de Markov aléatoire binaire E' ainsi construit est sous-modulaire, c'est à dire si

$$E_{(s,s')}(w_s,w_{s'}) + E_{(s,s')}(\alpha,\alpha) \leq E_{(s,s')}(w_s,\alpha) + E_{(s,s')}(\alpha,w_{s'}), \tag{6.13}$$

on peut minimiser $E'(u)$ (et donc $E(w^u)$) en un temps polynomial grâce un algorithme de coupe dans un graphe.

En itérant sur α, on obtient finalement l'algorithme décrit dans l'Algorithme 8.

Algorithme 8 Algorithme d'α-expansion

Entrée : Les éléments permettant le calcul de E
Sortie: Une approximation d'un minimiseur $w \in \{0,\dots,L\}^{\mathcal{V}}$ de E

Initialisation de w
Tant que l'algorithme n'a pas convergé **faire**
  for all $\alpha = 0,\dots,L$ **faire**
    Résoudre $u \in \text{Argmin}_u E'(u)$, où E' est définie par (6.12)
    $\forall s \in \mathcal{V}$, si $u_s = 1$, faire $w_s \leftarrow \alpha$
  end for
Fin tant que

Cet algorithme marche bien car à chaque pas, on prend un w minimum parmi tous les w d'un voisinage dont la taille est grande. D'un point de vue qualitatif, il permet de faire passer en un seul pas toute une zone de l'image au label α dans des situations où faire passer les sommets s l'un après l'autre aurait (en cours de route) obligé à augmenter E.

Cet algorithme ne peut, par contre, être appliqué que lorsque (6.13) est vraie quels que soient w_s, $w_{s'}$ et α. Une situation dans laquelle (6.13) est facile à comprendre est lorsque $\mathcal{E}_{(s,s')}(\alpha,\alpha) = 0$ (ce qui est d'ailleurs souvent le cas). Elle prend alors la forme d'une inégalité triangulaire.

Comme son nom l'indique, l'algorithme de l'α-β-swap propose itérativement (en α et β) aux sommets ayant les labels α et β de changer pour l'autre label. Sa construction est similaire à celle de l'α-expansion et est laissée en exercice. Cet algorithme est cependant généralement moins efficace que l'α-expansion. Il peut par contre être appliqué pour un nombre plus important d'énergies E car l'analogue de la condition (6.13) est plus faible.

Chapitre 7

La segmentation d'images

7.1 Principe de la segmentation et notion de périmètre

La segmentation d'images est l'un des grands domaines du traitement d'images et de la vision par ordinateur. Le but d'une segmentation est de marquer les pixels correspondant à un objet dans une image. On présente sur la Figure 7.1 deux exemples d'images segmentées. L'un correspond à une application en édition de photographie, l'autre à la segmentation d'une tumeur pulmonaire dans une image scanner 3D.

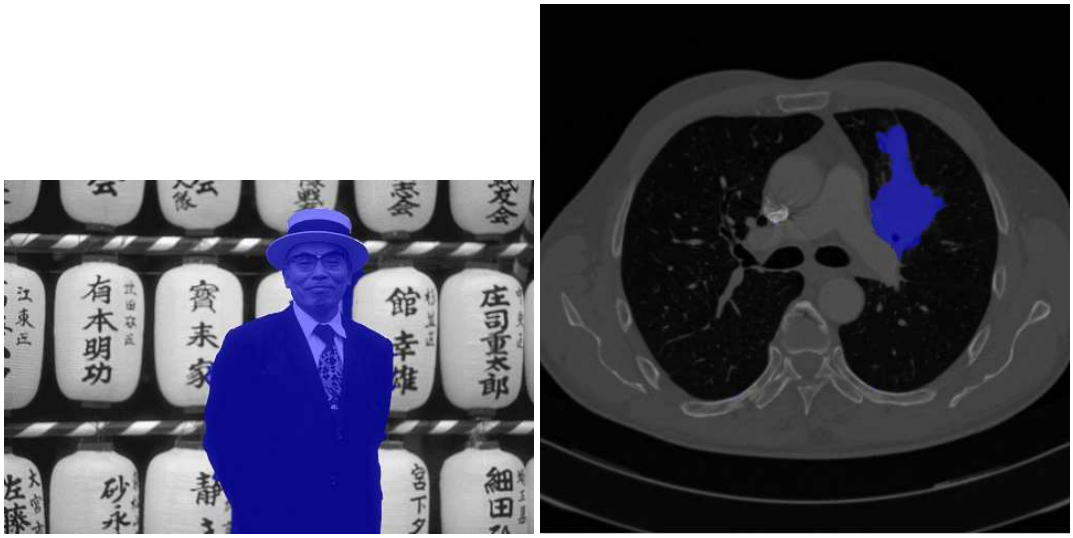


FIGURE 7.1 – Exemples d'images segmentées.

Les méthodes d'optimisation pour la segmentation d'images construisent donc une région Ω incluse dans le support de l'image :

$$\Omega \subset \{1,\dots,N\}^2.$$

Ces méthodes consistent à minimiser une énergie comprenant typiquement un ou plusieurs termes définissant une attache aux données et un terme correspondant au périmètre de Ω. Pour définir ce périmètre, nous considérerons dans la suite un système de voisinages $\sigma(m,n)$ de chaque pixel $(m,n) \in \{1,\dots,N\}^2$.

On peut, par exemple, choisir

$$\sigma(m,n) = \left\{(m',n') \in \{1,\dots,N\}^2, |m-m'|+|n-n'| = 1\right\}, \tag{7.1}$$

qui correspond à ce que l'on appelle la 4-connexité ; ou

$$\sigma(m,n) = \left\{(m',n') \in \{1,\dots,N\}^2, \max\left(|m-m'|,|n-n'|\right) = 1\right\},$$

qui correspond à ce que l'on appelle la 8-connexité.

On définit alors la frontière de $\Omega \subset \{1,\dots,N\}^2$ comme l'ensemble

$$\partial\Omega = \left\{\left((m,n),(m',n')\right) \in (\{1,\dots,N\}^2)^2, (m',n') \in \sigma(m,n) \text{ et } (m,n) \in \Omega \text{ et } (m',n') \notin \Omega\right\}. \tag{7.2}$$

Les notions de voisinage et de frontière sont illustrées sur la Figure 7.2.

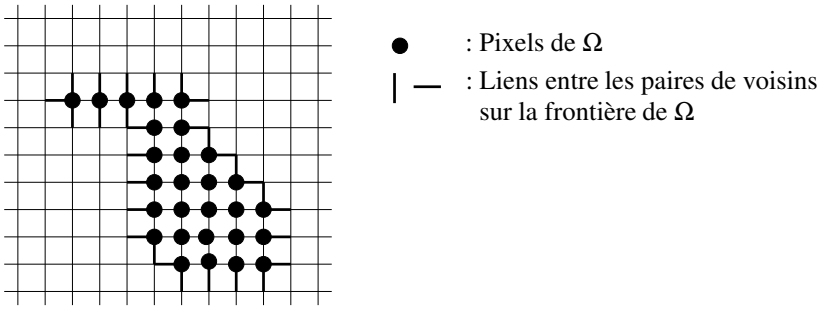


FIGURE 7.2 – Exemple d'un ensemble $\Omega \subset \{1,\dots,N\}^2$ et de sa frontière, pour des voisinages définis par 4-connexité.

On définit alors une notion de périmètre de Ω par

$$\mathbf{P}(\Omega) = \sum_{((m,n),(m',n'))\in\partial\Omega} dl\left((m,n),(m',n')\right),$$

où $dl\left((m,n),(m',n')\right) \geq 0$ sont des éléments de longueur. Ceci correspond bien à une notion de longueur de contour. En effet, comme les $dl\left((m,n),(m',n')\right)$ sont positifs, plus la frontière de Ω contient d'éléments, plus la longueur du contour sera grande. Pour simplifier, nous supposerons par la suite que pour tout $\left((m,n),(m',n')\right) \in (\{1,\dots,N\}^2)^2$

$$dl\left((m,n),(m',n')\right) = dl\left((m',n'),(m,n)\right). \tag{7.3}$$

On suppose aussi que

$$\text{si} \quad (m',n') \notin \sigma(m,n) \quad \text{alors} \quad dl\left((m,n),(m',n')\right) = 0. \tag{7.4}$$

En général, on construit les éléments de longueur de manière à avoir une notion de longueur la plus isotrope possible. On les utilise aussi parfois pour favoriser une localisation de la frontière. Par exemple, pour favoriser le fait que la frontière de Ω corresponde à des bords contrastés dans l'images, on prendra, pour les couples $\left((m,n),(m',n')\right)$ de fort contraste, un petit élément de longueur $dl\left((m,n),(m',n')\right)$. On verra un exemple comme celui-ci dans le Chapitre 7.2.3.

Exemple : On a vu dans le Chapitre 3.2.1 que la variation totale de la fonction caractéristique de $\Omega \subset \{1,\dots,N\}^2$,

$$\mathbf{1}_{|\Omega}(m,n) = \begin{cases} 1 & \text{, si } (m,n) \in \Omega, \\ 0 & \text{, sinon,} \end{cases}$$

correspond à son périmètre. Il semble donc raisonnable de définir une notion de périmètre d'un ensemble correspondant à la variation totale de la fonction caractéristique de cet ensemble.

En utilisant la discrétisation de la variation totale définie dans (3.8), la discrétisation des dérivées partielles de l'image définie dans (3.9) et si l'on considère la norme l^1 du gradient dans $\mathbb{R}^2$, on aboutit à

$$\begin{aligned} TV\left(\mathbf{1}_{|\Omega}\right) &= \sum_{m,n=1}^{N} |\nabla \mathbf{1}_{|\Omega}(m,n)| \\ &= \sum_{m,n=1}^{N} |\mathbf{1}_{|\Omega}(m+1,n) - \mathbf{1}_{|\Omega}(m,n)| + |\mathbf{1}_{|\Omega}(m,n+1) - \mathbf{1}_{|\Omega}(m,n)| \\ &= \sum_{m,n=1}^{N} \Big(\mathbf{1}_{|(m,n)\in\Omega \text{ et } (m+1,n)\notin\Omega} + \mathbf{1}_{|(m,n)\notin\Omega \text{ et } (m+1,n)\in\Omega} \\ &\qquad + \mathbf{1}_{|(m,n)\in\Omega \text{ et } (m,n+1)\notin\Omega} + \mathbf{1}_{|(m,n)\notin\Omega \text{ et } (m,n+1)\in\Omega} \Big) \\ &= \sum_{((m,n),(m',n'))\in\partial\Omega} 1, \end{aligned}$$

où $\partial\Omega$ est défini par (7.2), à l'aide le 4-connexité (voir (7.1)). Cette notion de périmètre consiste à prendre les dl tous égaux à 1.

7.2 Modèles d'optimisation pour la segmentation d'images

Nous allons dans ce chapitre détailler plusieurs modèles standards utilisés pour la segmentation d'images. Il en existe bien entendu de nombreux autres.

7.2.1 Le modèle de Mumford-Shah

Le modèle de Mumford-Shah est très étudié et consiste à minimiser en $\Omega \subset \{1,\dots,N\}^2$ et en $w \in \mathbb{R}^{N^2}$

$$\mathbf{P}(\Omega) + \lambda \sum_{\substack{m,n=1 \\ ((m,n),(m+1,n))\notin\partial\Omega \\ ((m,n),(m,n+1))\notin\partial\Omega}}^{N} |\nabla w_{m,n}|^2 + \mu \, \|w-u\|_2^2, \tag{7.5}$$

où $\lambda \geq 0$, $\mu \geq 0$ sont des paramètres et $u \in \mathbb{R}^{N^2}$ est l'image à segmenter.

Ce modèle crée un ensemble Ω régulier dans le sens où sa longueur n'est pas trop importante. Le second terme est très proche d'un terme H^1

$$\sum_{m,n=1}^{N} |\nabla w_{m,n}|^2,$$

sauf que l'on retire de la somme les points dont le calcul du gradient fait intervenir un voisin de l'autre côté de la frontière de Ω. Ainsi, comme le terme H^1 pénalise fortement les forts gradients, les points ayant

un fort gradient tendent donc à appartenir à la frontière de Ω. Enfin, le dernier terme constitue l'attache aux données.

Il faut enfin noter que l'image w créée au cours du calcul de la segmentation est une version débruitée de u.

7.2.2 Le modèle de Chan-Vese

Le modèle de Chan-Vese suppose que l'objet et le fond ont une distribution de couleurs gaussienne. L'objet et le fond sont séparés par une frontière régulière. Il est par exemple utilisé pour des segmentation d'images biologiques de noyaux cellulaires qui contiennent des noyaux de cellules claires sur un fond sombre.

Le modèle de Chan-Vese consiste à minimiser en $(c_1,c_2) \in \mathbb{R}^2$ et $\Omega \subset \{1,\dots,N\}^2$ l'énergie

$$\mathbf{P}(\Omega) + \mu_1 \sum_{(m,n)\in\Omega} |u_{m,n} - c_1|^2 + \mu_2 \sum_{(m,n)\notin\Omega} |u_{m,n} - c_2|^2, \tag{7.6}$$

où $\mu_1 \geq 0$ et $\mu_2 \geq 0$ sont des paramètres.

Un des intérêts de ce modèle est qu'il ne favorise pas le fait que la frontière entre l'objet et le fond corresponde à des pixels auxquels l'image a un fort gradient. Il permet donc, par exemple, de segmenter des groupes d'objets ayant une couleur similaire. On parle de "segmentation basée région".

Un autre aspect important de ce modèle est qu'il "apprend" le modèle de couleur de l'objet et du fond tout en calculant la segmentation. Concernant cet aspect, il faut remarquer que lorsque Ω est fixé, on calcule facilement les valeurs de c_1 et c_2 minimisant l'énergie ci-dessus. L'énergie ci-dessus est en effet quadratique en c_1 et c_2. Il suffit donc de la dériver et de chercher ses points critiques pour obtenir [1]

$$c_1 = \frac{\sum_{(m,n)\in\Omega} u_{m,n}}{\#\Omega}$$

et

$$c_2 = \frac{\sum_{(m,n)\notin\Omega} u_{m,n}}{N^2 - \#\Omega},$$

où $\#\Omega$ désigne le cardinal de Ω.

7.2.3 Le modèle de Boykov-Jolly

Le modèle de Boykov-Jolly tire bénéfice d'une modélisation des distributions des couleurs plus souple que les modèles précédents. Il peut aussi, suivant le réglage des paramètres favoriser ou pas le fait que la frontière entre l'objet et le fond corresponde aux points de fort gradient. Il s'écrit sous la forme de la minimisation en $\Omega \subset \{1,\dots,N\}^2$ de

$$\mathbf{P}(\Omega) + \sum_{(m,n)\in\Omega} -\log\left(\mathbb{P}(u_{m,n}|(m,n)\in O)\right) + \sum_{(m,n)\notin\Omega} -\log\left(\mathbb{P}(u_{m,n}|(m,n)\in \mathcal{F})\right), \tag{7.7}$$

où $O \subset \{1,\dots,N\}^2$ représente l'objet et $\mathcal{F} \subset \{1,\dots,N\}^2$ représente le fond. Les densités des lois de distribution des couleurs de l'objet $\mathbb{P}(u_{m,n}|(m,n)\in O)$ et du fond $\mathbb{P}(u_{m,n}|(m,n)\in \mathcal{F})$ sont supposées connues. Elles décrivent respectivement la probabilité d'observer la couleur $u_{m,n}$ pour les pixels de l'objet et du fond. Comme pour les densités des lois de probabilité utilisées pour définir l'information mutuelle (voir (4.3)), on peut par exemple les estimer à l'aide d'histogrammes normalisés. Souvent, ces densités sont estimées à partir d'une "graine" saisie à la souris par l'utilisateur (voir Figure 7.3).

Par exemple, supposons que l'on segmente

1. Attention, il faut éventuellement adapter ces formules si, comme proposé dans le Chapitre 6, on utilise des ensembles de niveau pour faire notre optimisation de forme.

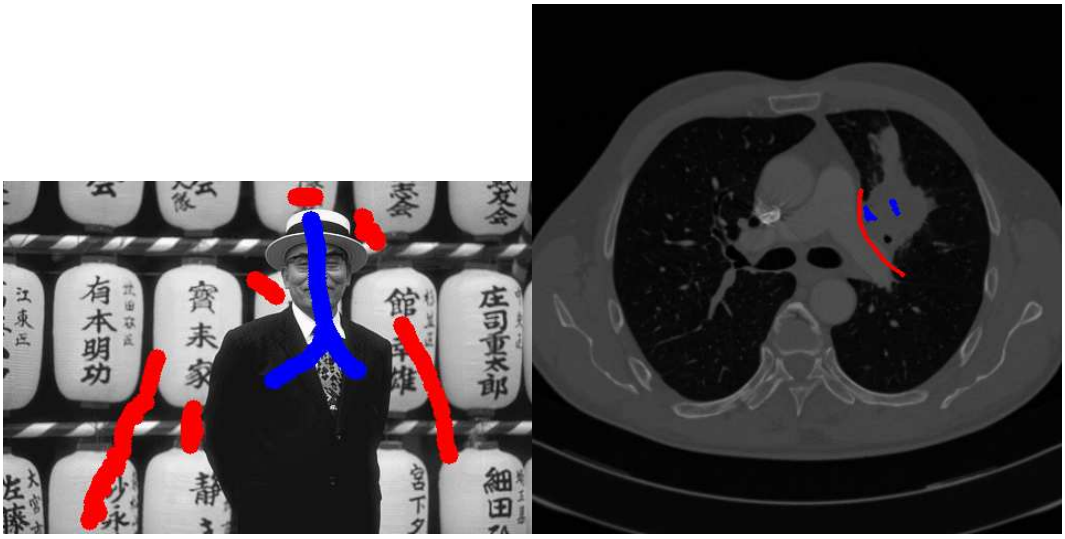


FIGURE 7.3 – Exemples de graines objet (en bleue) et de fond (en rouge) saisies par l'utilisateur.

- un objet sombre : i.e. $\mathbb{P}(c|(m,n) \in O)$ est grande pour c petit et petite pour c grand ;
- sur un fond clair : i.e. $\mathbb{P}(c|(m,n) \in \mathcal{F})$ est grande pour c grand et petite pour c petit.

En un pixel (m,n) tel que $u_{m,n}$ est (par exemple) petit, on trouve que

- pour le terme décrivant l'objet : la densité $\mathbb{P}(u_{m,n}|(m,n) \in O)$ est grande et donc

$$-\log\left(\mathbb{P}(u_{m,n}|(m,n) \in O)\right)$$

est petit ;

- pour le terme décrivant le fond : la densité $\mathbb{P}(u_{m,n}|(m,n) \in \mathcal{F})$ est petite (proche de 0) et donc

$$-\log\left(\mathbb{P}(u_{m,n}|(m,n) \in \mathcal{F})\right)$$

est grand.

Ainsi, comme on peut s'y attendre, les deux termes d'attache aux données de (7.7) vont favoriser le fait que $(m,n) \in \Omega$.

Enfin, dans (7.7), on utilise pour définir $\mathbf{P}(\Omega)$ des éléments de longueurs définis par

$$dl\left((m,n),(m',n')\right) = \frac{e^{-\frac{(u_{m,n}-u_{m',n'})^2}{2\sigma^2}}}{\sqrt{(m-m')^2+(n-n')^2}},$$

où u est l'image à segmenter et $\sigma \geq 0$ est un paramètre.

Typiquement, le dénominateur vise à améliorer l'isotropie de la notion de longueur. Suivant la valeur de σ, le numérateur favorisera les ensembles Ω dont la frontière correspond aux bords contrastés de l'image ou pas. En effet,

- si σ est petit, l'énergie pénalise moins fortement les couples $((m,n),(m',n'))$ fortement contrastés que ceux faiblement contrastés ;
- si σ est grand, l'énergie pénalise de manière similaire tous les couples $((m,n),(m',n'))$. On retrouve alors une notion de périmètre analogue à celle définie à l'aide de la variation totale page 89.

Bibliographie

[1] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, second edition edition, 2003.

[2] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE, trans. on Pattern analysis and Machine intelligence*, 23(11) :1222–1239, Nov. 2001.

[3] Ciarlet. *Analyse numerique matricielle et optimisation*. Dunod, Paris, 1987.

[4] L.C. Evans and R. F. Gariepy. *Measure Theory and Fine Properties of Functions*. Studies in Advanced Mathematics. CRC Press, Boca Raton, 1992.

[5] V. Kolmogorov and R. Zabih. What energy can be minimized via graph cuts. *IEEE, trans. on Pattern Analysis and Machine Intelligence*, 26(2) :147–159, 2004.

[6] V. Kolmogorov Y. Boykov. An experimental comparison of mincut/ max-flow algorithms for energy minimization in vision. *IEEE Transactions on PAMI*, 26(9) :1124–1137, 2004.