CPE LYON

Année universitaire
2022 - 2023

IMI – Optimisation et problèmes inverses – S8
Mini projet

Serge Mazauric
Eric Van Reeth

Objectifs.

- Application des notions d'**optimisation** lisse et non lisse à des problématiques de **traitement d'image 2D et 3D**
- Compréhension des modèles et de leurs limites
- Optimisation multivariée

Déroulement. Ce TP se déroule sur **1 séance** et est à effectuer par binôme sous Matlab/Octave, sur **un seul des sujets au choix**. Vous trouverez sur CPe-campus une archive contenant l'ensemble des fichiers nécessaires à la réalisation de ce TP.

Chaque sujet est proposé au format bureau d'étude, pour compléter votre cours sur les méthodes d'optimisation. Il doit vous permettre de mettre en application les modèles d'optimisation et les algorithmes de résolution abordés aux TP1 et TP2, pour différents traitements d'image en 2D et 3D, et d'étudier les limites de ces approches. Il permet aussi d'aller plus loin dans la compréhension du design d'un problème d'optimisation en fonction de l'application visée. Il est dès lors indispensable que vous preniez le temps d'étudier l'influence des paramètres et, le cas échéant, de vos choix d'implémentation.

Pour chacun des sujets proposés, vous trouverez :

- le contexte,
- le problème d'optimisation considéré,
- une aide pour la résolution du problème.

Configuration. Ce TP nécessite les librairies suivantes :

- Matlab : *Image Processing Toolbox*

- Octave : *Image toolbox* `pkg install -forge image`
  puis, en début de script : `pkg load image`

Compte-rendu. A la fin de la séance, vous devrez avoir rédigé un compte-rendu (max. 15 pages) permettant d'identifier clairement votre démarche pour la résolution du problème choisi :

- contexte, position du problème
- analyse et compréhension du problème
- solution proposée et mise en œuvre
- résultats

L'évaluation de TP portera sur la démarche proposée, et en particulier sur votre argumentaire concernant la solution proposée. Une attention particulière sera portée à la clarté du compte-rendu final.

Il vous est demandé **un compte-rendu par binôme**, correspondant à **votre** restitution des notions. Il en va de même pour le code. Tout travail emprunté à un (ou plusieurs) autre(s) groupe(s) doit être **explicitement identifié**.

Nous rappelons que toute tentative de copie entrainera une sanction de l'ensemble des binômes concernés.

Sujets proposés

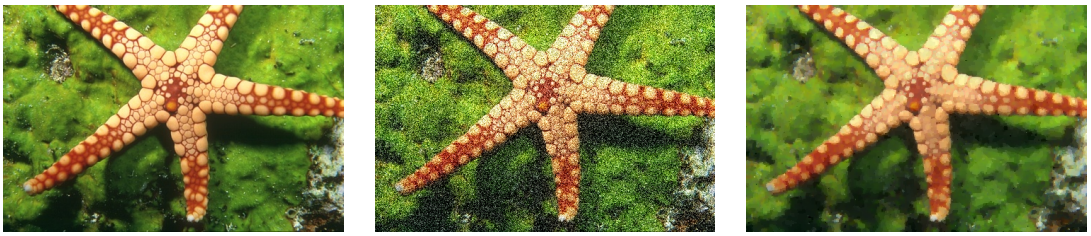
Serge Mazauric
Eric Van Reeth

1 Débruitage d'image couleur ⋆

Nous avons considéré dans les TP1 et TP2 des fonctions de coût de la forme

$$f(x) = \|Hx - z\|_2^2 + \lambda R(x), \tag{1}$$

où R est une régularisation choisie en fonction du type de résultat souhaité.
Cette fonction de coût n'est cependant pas appropriée dans le cas du débruitage d'images couleur. Aussi, on se propose d'étendre cette fonction de coût aux images couleur (ou multicanaux de manière générale).



Vérité terrain $\bar{x}$ Observation z Estimation $\hat{x}$

FIGURE 1 – Débruitage d'image couleur.

Objectifs.

- extension multicanaux
- résolution multivariée

Problème. Soit $\bar{x} = (\bar{x}_1, \bar{x}_2, ..., \bar{x}_C) \in \mathbb{R}^{N \times C}$ une image de taille $N = h \times w$ possédant C canaux. En particulier,

- si $\bar{x}$ est une image en niveau de gris, $C = 1$;
- si $\bar{x}$ est une image RGB, $C = 3$ et $\bar{x} = (\bar{x}_1, \bar{x}_2, \bar{x}_3) = (r, g, b)$;

Soit $n \in \mathbb{R}^N \sim \mathcal{N}(0, \sigma)$ un bruit blanc Gaussien additif. On considère le modèle de dégradation :

$$\forall c = 1, ..., C, \quad z_c = \bar{x}_c + n \tag{2}$$

(Le cas avec flou pourra être traité dans un second temps, en fonction du choix de la régularisation R).

Étant donnée l'observation $z \in \mathbb{R}^{N \times C}$, on cherche une approximation $\hat{x} \in \mathbb{R}^{N \times C}$ de $\bar{x}$, solution du problème d'optimisation :

$$\hat{x} = (\hat{x}_1, ..., \hat{x}_C) \in \underset{x \in \mathbb{R}^{N \times C}}{\operatorname{argmin}} \underbrace{\sum_{c=1}^{C} \|x_c - z_c\|_2^2 + \lambda R(x_c)}_{f(x) = f(x_1, ..., x_C)} \tag{3}$$

Serge Mazauric
Eric Van Reeth

Difficulté. Pour minimiser numériquement une telle fonction de coût, on adopte très souvent une stratégie de minimisation alternée. Autrement dit, à chaque itération, on fixe les variables x_j pour $j \neq i$ et on minimise par rapport à x_i. Dans le cas RGB, cela conduit aux trois sous-problèmes suivants :

$$r_{k+1} = \text{argmin}_{r \in \mathbb{R}^N} f(r, g_k, b_k) \quad \text{(R)}$$
$$g_{k+1} = \text{argmin}_{g \in \mathbb{R}^N} f(r_{k+1}, g, b_k) \quad \text{(G)}$$
$$b_{k+1} = \text{argmin}_{b \in \mathbb{R}^N} f(r_{k+1}, g_{k+1}, b) \quad \text{(B)}$$

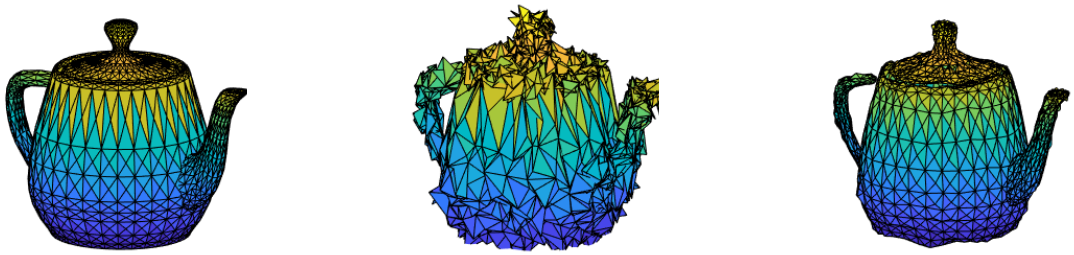
L'algorithme qui en résulte est le suivant :

```
Initialiser r0, g0, b0;
Choix des paramètres pour la résolution de (R), (G), (B);
For k=1:niter
   Résoudre (R);
   Résoudre (G);
   Résoudre (B);
end
```

2 Débruitage de maillage ⋆

Nous avons considéré dans les TP1 et TP2 des applications en 1D et 2D. Toutefois, les modèles considérés sont valables en N dimensions. Il est possible d'utiliser les modèles étudiés pour régulariser des objets de l'espace : surfaces de l'espace, nuages de points, volumes IRM, etc.
Ainsi, le domaine de la synthèse d'images nécessite régulièrement d'utiliser des techniques de traitement d'images originellement pensées en 2D ; en particulier, les techniques de débruitage et de lissage.

Aussi, on se propose de s'intéresser au débruitage/lissage d'un maillage triangulé, ce dernier étant une surface de l'espace. La particularité dans ce cas vient du fait qu'un point ne possède pas un nombre constant de voisins.



Vérité terrain $\bar{x}$ Observation z Estimation $\hat{x}$

FIGURE 2 – Débruitage d'un maillage.

Objectifs.

- extension 3D, résolution multivariée
- extension graphe

Problème. Soit $\bar{x} \in \mathbb{R}^{N\times 3}$ un maillage triangulé à N points, chaque point étant un point dans l'espace.

Soit $n \in \mathbb{R}^{N\times 3} \sim \mathcal{N}(0,\sigma)$ un bruit blanc Gaussien additif. On considère le modèle de dégradation :

$$z = \bar{x} + n \tag{4}$$

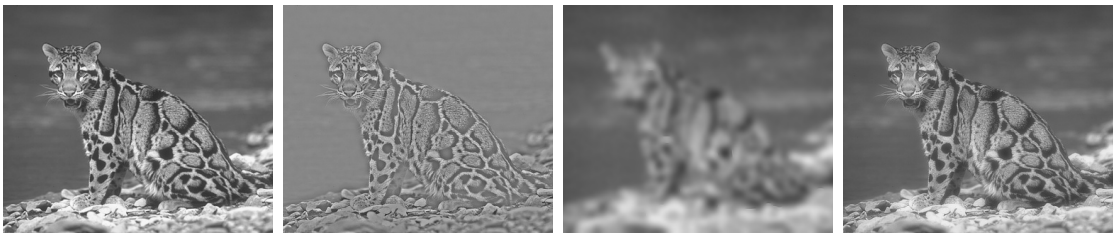
Étant donnée l'observation $z \in \mathbb{R}^{N\times 3}$, on cherche une approximation $\hat{x} \in \mathbb{R}^{N\times 3}$ de $\bar{x}$, solution du problème d'optimisation :

$$\hat{x} \in \underset{x\in\mathbb{R}^{N\times 3}}{\operatorname{argmin}} \underbrace{\|x-z\|_2^2 + \lambda R(x)}_{f(x)} \tag{5}$$

Difficulté. Les maillages triangulés fournis pour ce sujet sont encodés au format `.off`. Des fonctions de lecture (`loadOff.m`) et d'écriture (`exportOff.m`) spécifiques pour ce format sont fournies.
L'affichage se fait au moyen de la fonction `trisurf(TRI,X,Y,Z)`.
De plus, il faut porter une attention particulière au choix des paramètres en 3D, le maillage étant très sensible à des changements d'amplitude trop importants.

3 Décomposition cartoon + texture ⋆⋆

Étant donnée une image, on souhaite pouvoir la décomposer en une image de texture et une image "cartoon", qui donne uniquement l'information d'intensité. Dit autrement, on souhaiterait séparer les informations de texture (hautes fréquences), de l'intensité des régions constantes (plutôt basse fréquence) de l'image.



Donnée $z = \bar{x}$ Texture estimée $\hat{u}$ Cartoon estimé $\hat{v}$ Estimation $\hat{x} = \hat{u} + \hat{v}$

FIGURE 3 – Décomposition cartoon+texture.

Objectifs.

- résolution multivariée

Problème. Soit $\bar{x} \in \mathbb{R}^N$ une image de taille $N = h \times w$. Pour cette application, l'observation n'est pas dégradée par rapport à la vérité terrain, aussi

$$z = \bar{x} \tag{6}$$

Étant donnée l'observation $z \in \mathbb{R}^N$, on cherche une approximation $\hat{x} \in \mathbb{R}^N$ de $\bar{x}$, telle que $\hat{x} = \hat{u} + \hat{v}$. Les composantes u et v représentent respectivement la texture et la partie cartoon de l'image. On se propose de les estimer via le problème d'optimisation :

$$(\hat{u}, \hat{v}) \in \underset{(u,v)\in\mathbb{R}^N}{\operatorname{argmin}} \ \frac{1}{2} \underbrace{\|u + v - z\|_2^2 + \lambda \|Bu\|_2^2 + \mu \|Dv\|_1}_{f(u,v)} \tag{7}$$

Les opérateurs `B` et `Badj` spécifiques à ce sujet sont disponibles dans le dossier "opérateurs".

Difficulté. Pour minimiser numériquement une telle fonction de coût, on adopte très souvent une stratégie de minimisation alternée. Autrement dit, à chaque itération, on fixe une valeur de v et on minimise par rapport à u, puis inversement. Cela conduit aux deux sous-problèmes suivants :

$$u_{k+1} = \operatorname{argmin}_{u\in\mathbb{R}^N} f(u, v_k) \tag{T}$$
$$v_{k+1} = \operatorname{argmin}_{v\in\mathbb{R}^N} f(u_{k+1}, v) \tag{C}$$

L'algorithme qui en résulte est le suivant :

```
Initialiser u0, v0;
Choix des paramètres pour la résolution de (T);
Choix des paramètres pour la résolution de (C);
For k=1:niter
   Résoudre (T);
   Résoudre (C);
end
```

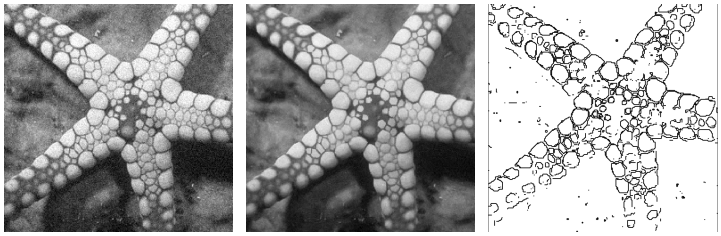

Serge Mazauric
Eric Van Reeth

Remarque : la résolution du sous-problème (C) pourra elle-même demander la décomposition en deux sous-problèmes (cf. TP2).

Serge Mazauric
Eric Van Reeth

4 Débruitage et détection de contours ⋆⋆

Dans ce projet, on souhaite débruiter une image en niveaux de gris et détecter ses contours. Ces deux tâches seront formalisées dans un unique problème d'optimisation.



Donnée z Image débruitée $\hat{x}$ Contours $\hat{e}$

FIGURE 4 – Débruitage et segmentation conjoints.

Objectifs.

- résolution multivariée

Problème. Soit $\bar{x} \in \mathbb{R}^N$ une image de taille $N = h \times w$. Et soient $H \in \mathbb{R}^{N \times N}$ un opérateur de dégradation linéaire (e.g. un flou), et $n \in \mathbb{R}^N \sim \mathcal{N}(0, \sigma)$ un bruit blanc Gaussien additif. On considère le modèle de dégradation :

$$z = H\bar{x} + n \tag{8}$$

Étant donnée l'observation $z \in \mathbb{R}^N$, on cherche à la fois une approximation $\hat{x} \in \mathbb{R}^N$ de $\bar{x}$, et les contours $\hat{e} \in \mathbb{R}^{2N}$ (tel que $\hat{e} = 1$ lorsqu'il y a un contour, et 0 sinon), solutions du problème d'optimisation :

$$(\hat{x}, \hat{e}) \in \underset{x \in \mathbb{R}^N, e \in \mathbb{R}^{2N}}{\operatorname{argmin}} \; \|Hx - z\|_2^2 + \beta \|(1-e) \odot (Dx)\|_2^2 + \lambda \|e\|_1 \tag{9}$$

où $\odot$ représente le produit de Hadamard (produit terme à terme).

Difficulté. Pour minimiser numériquement une telle fonction de coût, on adopte très souvent une stratégie de minimisation alternée. Autrement dit, à chaque itération, on fixe une valeur de e et on minimise par rapport à x, puis inversement. Cela conduit aux deux sous-problèmes suivants :

$$x_{k+1} = \operatorname{argmin}_{x \in \mathbb{R}^N} f(x, e_k) \tag{X}$$
$$e_{k+1} = \operatorname{argmin}_{e \in \mathbb{R}^{2N}} f(x_{k+1}, e) \tag{E}$$

L'algorithme qui en résulte est le suivant :

```
Initialiser u0, v0;
Choix des paramètres pour la résolution de (X);
Choix des paramètres pour la résolution de (E);
For k=1:niter
  Résoudre (X);
  Résoudre (E);
end
```


Serge Mazauric
Eric Van Reeth

A Mesure de la qualité des résultats

Lorsque l'on essaie d'estimer un phénomène physique à partir d'une observation dégradée de ce dernier (z), il peut être intéressant de quantifier objectivement la qualité de la reconstruction $\hat{x}$ en sortie de traitement.
Cela permet notamment d'aider au paramétrage d'un modèle. On trace la courbe d'une métrique en fonction de différentes valeurs de paramètres, et on extrait le paramètre optimisant le score choisi.

Il existe différents scores, dont trois sont particulièrement utilisés en image :

- l'Erreur Quadratique Moyenne (MSE),
- le Rapport Signal à Bruit (SNR),
- la Structural SIMilarity (SSIM).

Ces scores nécessitent en général d'**avoir accès à une référence** (on choisira souvent la vérité terrain $\bar{x}$).

MSE : Mean Squared Error. La MSE repose sur une différence de reconstruction pixel à pixel, entre l'image de référence $\bar{x}$ et l'image reconstruite $\hat{x}$:

$$MSE(I) = \|\hat{x} - \bar{x}\|_2^2.$$

SNR : Signal to Noise Ratio. Le SNR permet de quantifier la dégradation de l'image (a priori porteuse d'information utile) par un bruit (non informatif).
Le SNR (en dB) est défini comme le rapport de la puissance du signal P_s sur celle du bruit P_B :

$$SNR(s) = 10\ \log_{10}\left(\frac{P_s}{P_B}\right)\ dB.$$

Dans le cadre d'un problème inverse, nous n'avons en général par accès à ces deux informations séparément. Aussi, on calcule le SNR du signal estimé $\hat{x}$ relativement à une référence $\bar{x}$:

$$SNR(s) = 10\ \log_{10}\left(\frac{\|\bar{x}\|_2^2}{\|\hat{x} - \bar{x}\|_2^2}\right)\ dB.$$

SSIM : Structural SIMilarity. Le SSIM permet quant à lui de mesurer la similarité de structure par rapport à une image de référence $\bar{x}$. Il essaie de mieux tenir compte de la perception de l'oeil humain, a priori plus sensible aux changements de structure que de couleur.

Ce score est calculé sur plusieurs patchs de l'image. Il repose sur trois métriques de comparaison : la luminance l, le contraste c et la structure s. Pour chaque patch p, le SSIM est défini comme le produit de ces trois valeurs :

$$SSIM(p_{\hat{x}}, p_{\bar{x}}) = l(p_{\hat{x}}, p_{\bar{x}}) \cdot c(p_{\hat{x}}, p_{\bar{x}}) \cdot s(p_{\hat{x}}, p_{\bar{x}}).$$

Serge Mazauric
Eric Van Reeth

B Index des fonctions

Normes.

Si `x` est un **vecteur** :

`norm(x)`	calcule la norme 2 d'un vecteur $\|x\|_2 = \sqrt{\sum_i x_i^2}$
`norm(x,2)`	calcule la norme 2 d'un vecteur $\|x\|_2 = \sqrt{\sum_i x_i^2}$

Si A est une **matrice** :

`norm(A)`	calcule la norme d'opérateur $\|\|\|A\|\|\| = \max(\mathrm{SVD}(A))$
`norm(A,2)`	calcule la norme d'opérateur $\|\|\|A\|\|\| = \max(\mathrm{SVD}(A))$
`norm(A,'fro')`	calcule la norme de Frobenius $\|A\|_2 = \sqrt{\sum_{i,j} A_{ij}^2}$

Remarque : dans le cas d'une matrice, `norm(A,'fro')` = `norm(A(:),2)`.

Matrices.

`H = matH(size(x),type,N)`	Crée la matrice H de flou de type `type` et de taille `N`
`D = matGamma(s,'gradient')`	Crée la matrice D de gradient (`s = size(x)`)
`L = matGamma(s,'laplacian')`	Crée la matrice L de Laplacien (`s = size(x)`)
`D = matGamma3D(C,F,'gradient')`	Crée la matrice D du gradient d'un maillage (C,F)

Opérateurs (à privilégier).

`y = H(x,type)`	Calcule l'opération matricielle Hx (x **non** vectorisé)
`y = Hadj(x,type)`	Calcule l'opération matricielle $H^\top x$ (x **non** vectorisé)
`y = D(x)`	Calcule l'opération matricielle Dx (x **non** vectorisé)
`y = Dadj(x)`	Calcule l'opération matricielle $D^\top x$ (x **non** vectorisé)
`y = L(x)`	Calcule l'opération matricielle Lx (x **non** vectorisé)
`y = Ladj(x)`	Calcule l'opération matricielle $L^\top x$ (x **non** vectorisé)
`y = B(x)`	Calcule l'opération matricielle Bx (x **non** vectorisé)
`y = Badj(x)`	Calcule l'opération matricielle $B^\top x$ (x **non** vectorisé)

Import/Export de fichier `.off`.

`[C,F] = loadOff(filename)`	Importe le maillage (C,F) du fichier `filename`
`exportOff(filename,C,F)`	Exporte le maillage (C,F) dans le fichier `filename`