CPE LYON

Année universitaire
2022 - 2023

IMI – Optimisation et problèmes inverses – S8

TP1 – Optimisation différentiable

Marion Foare
Serge Mazauric
Eric Van Reeth

Objectifs.

- Compréhension de l'**algorithme de descente de gradient**
- Compréhension de l'apport de la méthode numérique par rapport à une résolution directe
- Compréhension des limites de l'algorithme

- Compréhension du **modèle des moindres carrés**
- Compréhension du **modèle de Tikhonov**
- Compréhension des limites des modèles
- Implémentation de l'algorithme de descente de gradient pour la minimisation des modèles des moindres carrés et de Tikhonov en 1D et en 2D

Déroulement. Ce TP se déroule sur **2 séances** et est à effectuer par binôme sous Matlab/Octave. Vous trouverez sur CPe-campus une archive contenant l'ensemble des fichiers nécessaires à la réalisation de ce TP.

Ce TP fait partie intégrante de la construction de votre cours sur les méthodes de descente, et les modèles différentiables. Il doit vous permettre à la fois de comprendre le fonctionnement de ces méthodes, d'illustrer votre cours, mais aussi de mettre en évidence l'importance de certaines hypothèses, l'apport de ces méthodes par rapport à des méthodes de résolution classiques, et les limites de ces méthodes. 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.

Bien qu'il soit relativement guidé, n'hésitez donc pas à sortir des sentier battus, à vous poser vos propres questions et à prendre du recul sur les pistes de réflexion proposées dans ce TP.

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`

Évaluation. Aucun compte-rendu n'est à rendre pour ce TP, mais il est fortement conseillé de rédiger des notes personnelles pour synthétiser les notions abordées.
Chaque binôme sera évalué à l'oral lors de la deuxième séance de TP. L'évaluation pourra porter sur les notions générales vues en cours, les résultats obtenus en TP, et votre capacité d'analyser ces résultats.

Marion Foare
Serge Mazauric
Eric Van Reeth

Contexte

On se concentrera dans ce module sur des exemples en débruitage et déconvolution 1D (signal) et 2D (image). Pour autant, nous rappelons que l'optimisation est un domaine transversal, et permet, en fonction du choix de la fonction de coût, de traiter de nombreuses applications.
En particulier, on s'intéresse au débruitage de l'image ci-dessous :

On cherche pour cela à résoudre un problème d'optimisation (partie 2), défini sous la forme :

$$\hat{x} \in \underset{x \in \mathbb{R}}{\operatorname{argmin}}\ f(x)$$

Comme évoqué en cours, l'une des principales questions liées à l'optimisation est l'**implémentation d'un algorithme** (partie 1) qui nous permette d'approcher la solution même lorsqu'aucune forme explicite n'est envisageable, autrement dit de construire une suite $\{x_n\}_n$ qui converge vers $\hat{x}$.
Bien que chaque problème d'optimisation soit unique, et qu'il n'existe pas de méthode idéale permettant de résoudre efficacement tous les problèmes d'optimisation, les **méthodes dites "de descente"** font partie des algorithmes (itératifs) les plus classiques pour approcher numériquement le minimiseur $\hat{x}$. L'idée générale repose sur le fait de rechercher, à chaque itération, x_{k+1} tel que $f(x_{k+1}) < f(x_k)$.

1 Algorithme de descente de gradient

Cette section doit vous permettre de commencer à établir un plan de votre cours sur les méthodes de descente, et en particulier de la méthode de descente de gradient, que vous pourrez ensuite compléter au fur et à mesure avec des exemples issus de la section 2.

1.1 Prise en main de la méthode

Pour étudier le principe de la méthode de descente de gradient, on se propose d'appliquer l'algorithme pour minimiser la fonction $f(x) = x^2$, i.e. résoudre le problème de minimisation 1D

$$\hat{x} = \underset{x \in \mathbb{R}}{\operatorname{argmin}}\ x^2 \tag{1}$$

Pistes de réflexion :

- résultat attendu ?
- solution analytique du problème (1)
- implémentation de la méthode de descente de gradient
- quel choix pour l'initialisation ?
- quel choix du pas de descente ?
- étude de la convergence de la méthode

Marion Foare
Serge Mazauric
Eric Van Reeth

1.2 Généralisation au cas N-D

Généraliser le code au cas d'un signal $x \in \mathbb{R}^N$, permettant de résoudre le problème d'optimisation suivant :

$$\hat{x} = \underset{x \in \mathbb{R}^N}{\operatorname{argmin}} \ \|x\|_2^2$$

2 Utilisation de l'algorithme de descente de gradient pour la résolution de problèmes d'optimisation lisses

Dans cette section, on se propose d'utiliser la méthode de descente de gradient pour résoudre différents modèles de débruitage.
On rappelle que tous ces modèles sont construits à partir du modèle direct suivant :

$$z = H\bar{x} + n,$$

avec :

✔	$z \in \mathbb{R}^M$	observation
✔	$H \in \mathbb{R}^{M \times N}$	dégradation linéaire, e.g. un flou
✔		densité de probabilité du bruit
🔍	$\hat{x} \in \mathbb{R}^N$	le plus proche possible de $\bar{x}$ (inconnu)

Création des données :

- création d'une vérité terrain 1D $\bar{x}$ au choix (e.g. Heaviside, polynôme, etc.)
- création d'une observation z associée

2.1 Résolution du modèle des moindres carrés

Nous avons vu en cours que les fonctions de coût en image sont de la forme

$$f(x) = \mathcal{L}(Hx; z) + \lambda R(x)$$

où $\mathcal{L}$ est l'attache aux données (choisie en fonction du type de dégradation) et R est une régularisation (choisie en fonction du type de résultat souhaité).
Afin d'étudier l'influence de chaque terme, on se concentre dans un premier temps sur une fonction de coût sans régularisation (i.e. $f(x) = \mathcal{L}(Hx; z)$). Dans le cas d'une dégradation par un flou H et un bruit gaussien, le problème de minimisation est donc réduit au **modèle des moindres carrés**, défini par :

$$\hat{x} = \underset{x \in \mathbb{R}^N}{\operatorname{argmin}} \ \|Hx - z\|_2^2 \tag{2}$$

On pourra se limiter dans un premier temps au choix $H = Id$. Une fois la méthode implémentée et validée, on pourra introduire une dégradation de type flou et/ou décimation. Pour cela, vous pouvez utiliser les fonctions fournies (cf annexe B).

Attention. On veillera à choisir un signal de taille raisonnable ($N \leq 256$), afin que le temps de calcul reste raisonnable.

Pistes de réflexion :

- solution attendue ?
- solution analytique du problème (2)

Marion Foare
Serge Mazauric
Eric Van Reeth

- résolution numérique par descente de gradient du problème (2)
- comparaison des solutions obtenues par les deux méthodes (valeurs de la fonction de coût)
- influence du pas de descente de gradient
- influence du choix de l'initialisation
- limites de chacune des deux approches ?

2.2 Résolution du modèle de Tikhonov 1D

On souhaite étudier l'influence de la régularisation R. L'un des modèles les plus connus (utilisé par exemple pour la reconstruction tomographique) en traitement d'image est le modèle de Tikhonov, défini de la manière suivante :

$$\hat{x} = \underset{x \in \mathbb{R}^N}{\operatorname{argmin}} \ \|Hx - z\|_2^2 + \lambda \|\Gamma x\|_2^2 \tag{3}$$

où Γ modélise un opérateur de votre choix (identité, gradient, Laplacien...).

Pistes de réflexion :

- solution attendue ?
- solution analytique du problème (3)
- résolution numérique par descente de gradient du problème (3) pour le même signal 1D que dans la partie 2.1
- influence du choix de l'opérateur Γ
- comparaison des solutions obtenues par les deux méthodes (valeurs de la fonction de coût)
- apport de la régularisation ?
- influence du choix de λ

2.3 Résolution du modèle de Tikhonov 2D

L'extension aux images est presque immédiate si l'on considère x le vectorisé de l'image. Les seuls changements à effectuer concernent les matrices H et Γ (gradient 2D, Laplacien 2D). Attention cependant à ne pas choisir des images trop grandes : on rappelle que pour une image de taille $N \times N$, Γ est de taille $N^2 \times N^2$...
Pour pallier cela, on pourra utiliser les opérateurs `D`, `L` et `H` fournis, ainsi que leurs adjoints `Dadj`, `Ladj` et `Hadj`, plutôt que les matrices. Dans ce cas, les opérateurs s'appliquent à l'image **non vectorisée**.

Pistes de réflexion :

- résolution numérique par descente de gradient du problème (3) pour une image
- influence du choix de l'opérateur Γ
- influence du choix de λ
- commentaires

Marion Foare
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 :

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

Marion Foare
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)`)

Opérateurs (à privilégier en Octave).

`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)

Marion Foare
Serge Mazauric
Eric Van Reeth