METEYER Paul 4ETI, IMI

BRALET Antoine

Chapitre 1 : Les Méthodes de Descente

1. Contexte

La logique dans laquelle se place l’étude qui suit est la suivante : on cherche à résoudre un problème d’optimisation, à savoir : en connaissance d’une observation dégradé de la vérité de terrain, essayer de trouver une approximation la plus proche possible de cette vérité de terrain. Il est remarquable que tout type de problème lié à l’optimisation peut se ramener à la recherche du minimum global d’une fonction. Pour trouver un tel minimum, les méthodes de descentes sont particulièrement efficaces, à condition de bien choisir les paramètres de départ.

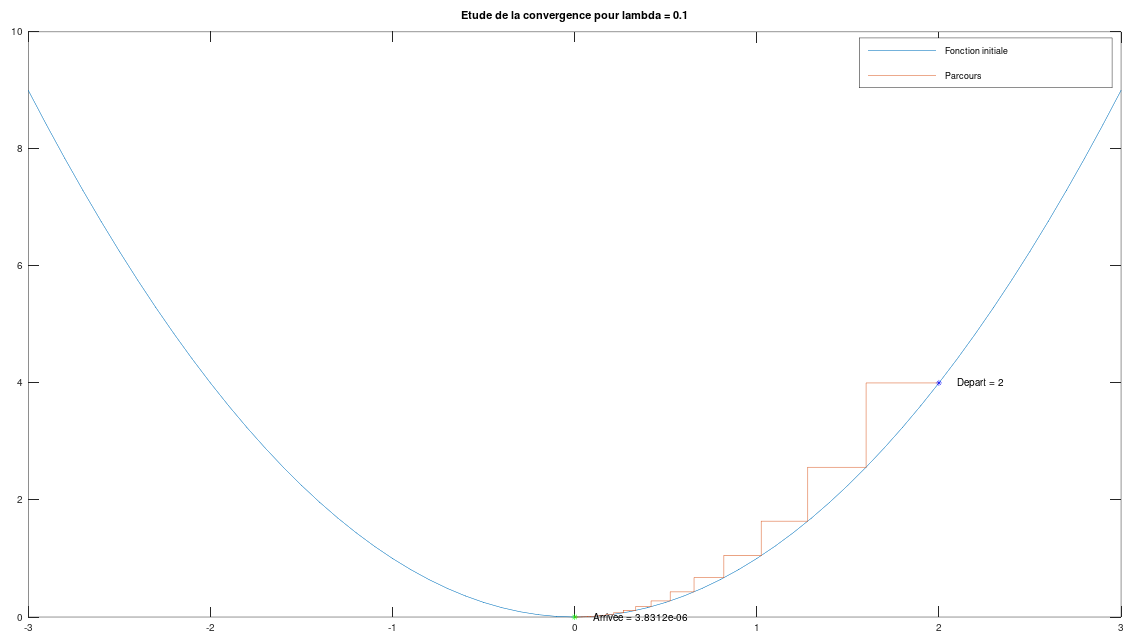
1. La méthode de descente de gradient

|  |
| --- |
| **Point de Cours : La méthode de descente de gradient**  La méthode de descente de gradient est relativement simple : elle consiste à suivre la pente descendante opposée au gradient et ce jusqu’à stabilisation des itérations successives.  Cette méthode est issue d’une constatation simple : on cherche à ce que entre deux itérations successives :  Et pour trouver ce dk, on passe par un développement de Taylor à l’ordre 1 :  Ce qui permet de justifier de l’utilisation du gradient.  Cette méthode dépend de trois paramètres principaux :   * x0 le point de départ * λ le pas de descente * ε la tolérance (l’écart minimal entre deux itérations successives pour considérer que le résultat attendu est le résultat souhaité)   La suite se construit alors de la manière suivante :  Avec f la fonction de x à minimiser. |

**Illustration de la méthode :** Le but est ici de minimiser la fonction .

→ Le minimum de cette fonction est bien connu, l’algorithme doit trouver 0.

En appliquant la relation précédente, pour x0 = 2 et λ = 0.1 (de façon empirique) et avec une tolérance à entre deux échantillons consécutifs, ceci permet d’obtenir :

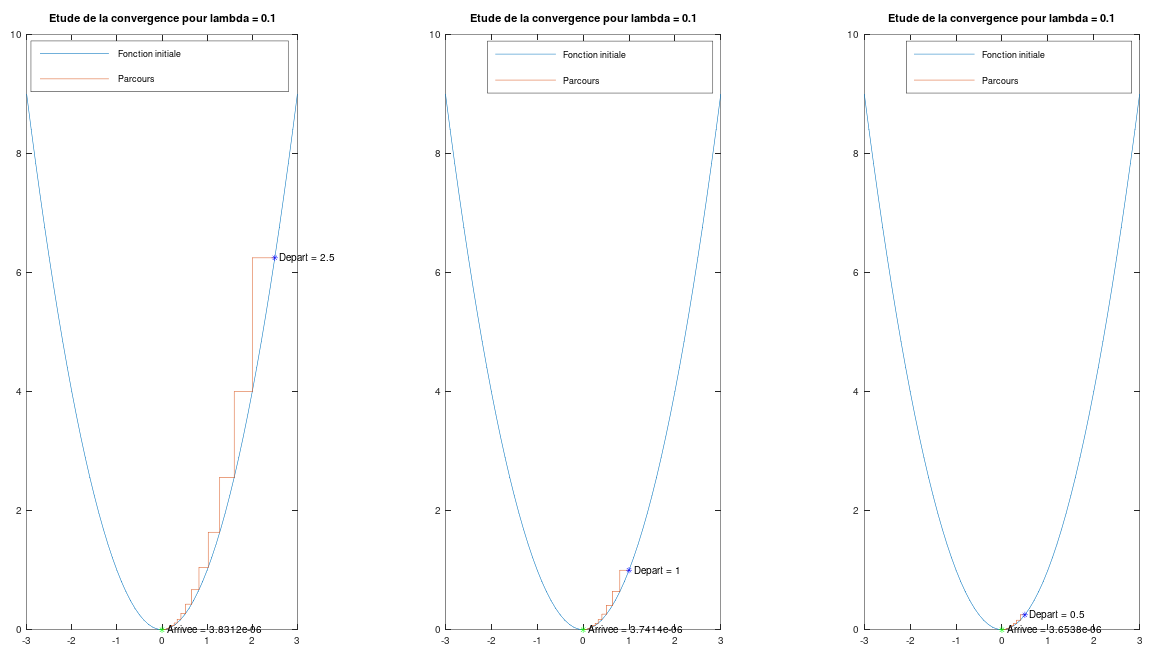


On peut alors bien constater l’évolution de l’algorithme et sa convergence vers le point minimum de la courbe tandis qu'il suit le gradient de la fonction.

|  |
| --- |
| **Point de Cours : Choix du point de départ x0**  Le choix d’un point de départ non pertinent peut avoir plusieurs conséquences sur l’algorithme:   * Augmentation des temps de calculs et du nombre d’étapes * Convergence vers un minimum local et non pas global * Convergence vers un point selle * Divergence si fonction non coercive |

**Illustration :**

En gardant la même fonction que précédemment, il est possible d’observer l’évolution du nombre d’étapes en fonction du point de départ :

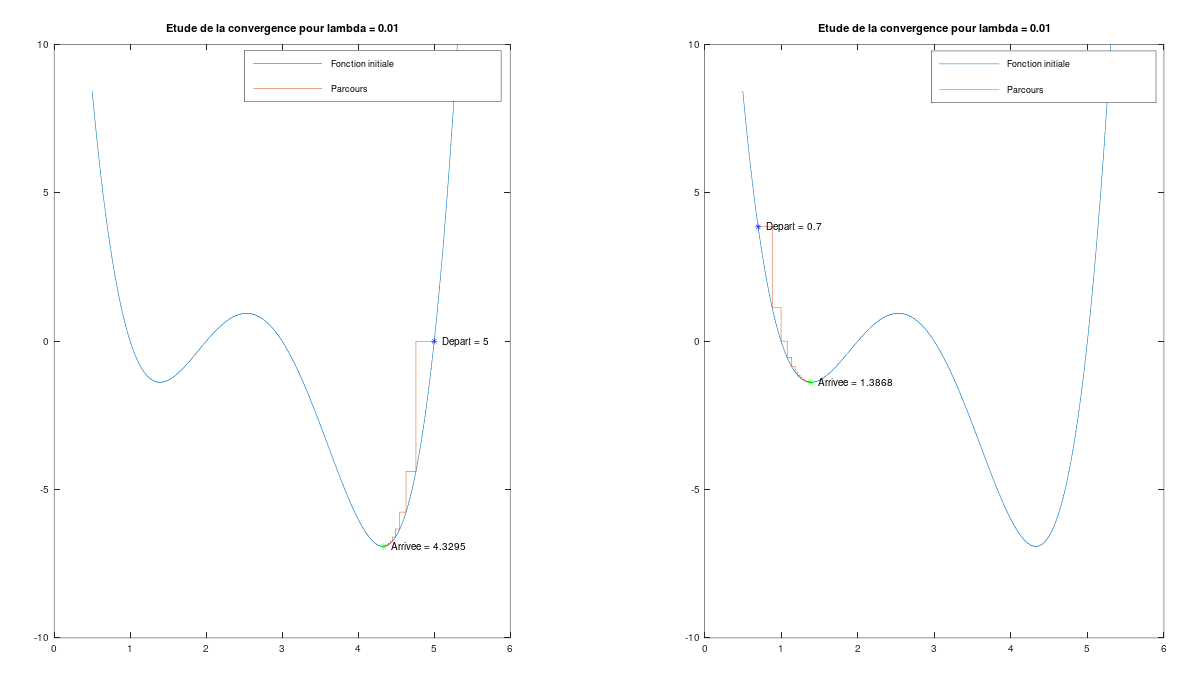


En effet, la première fois, il a fallu 121 étapes pour arriver à une précision inférieure à puis 113 pour la seconde et 107 pour la troisième. Il est certain que, dans le cas, présent cette différence est minime, mais elle peut avoir son importance lorsque l’on travaille sur de très grands intervalles.

L’illustration de l’erreur liée à la détection d’un minimum local n’est pas visible ici, c’est pourquoi la fonction qui illustre l’exemple en question est issu de l’université Paris Descartes [1] à savoir la fonction :

dont la dérivée est :

On observe alors qu’en prenant deux points diamétralement opposés on obtient :



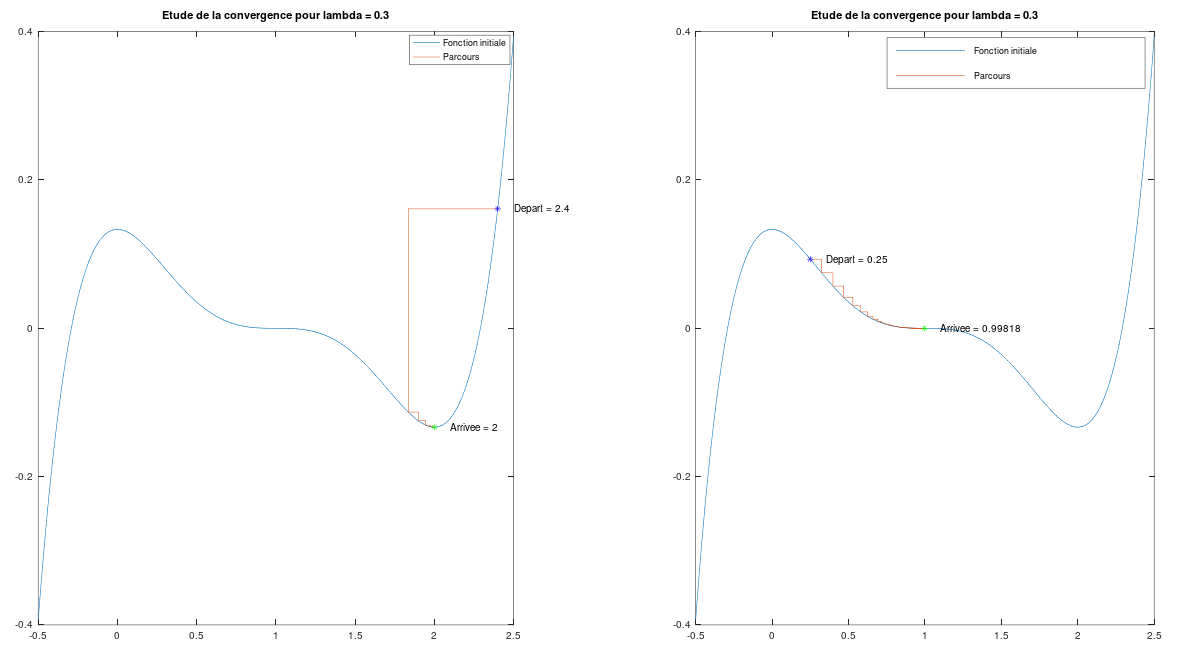
Dans le premier cas où le point de départ a été bien choisi, on se retrouve bien avec le minimum global de la fonction mais dans le second cas où le point de départ a été très mal choisi, on se retrouve alors avec un minimum local et non pas un minimum global.

Il est également nécessaire d’utiliser une nouvelle fonction afin de visualiser les problèmes de point selle. Un point selle est un point d’une fonction dont la dérivée s’annule mais qui ne correspond ni à un minimum local ni à un maximum local. Une fonction particulièrement adaptée pour le voir est la fonction :

dont la dérivée est :

On retrouve alors bien ici que 1 est bien un zéro de la dérivée, et il se trouve que celui ci est également le-dit point selle.

De nouveau, en considérant deux points d’initialisation différents, on constate que plusieurs résultats sont possibles :



On constate alors que l’algorithme reste alors bloqué sur le point selle dans le deuxième cas parce que l’initialisation a de nouveau été mal choisie.



|  |
| --- |
| **Point de Cours : Critère de minimum global unique**  Comme précédemment illustré, il est possible que certaines fonctions possèdent des particularités qui peuvent mettre en déroute l’algorithme de descente de gradient. Néanmoins, il est possible de trouver des fonctions pour lesquelles le choix de l’initialisation n’importe pas du tout dans la résultat final. Ces fonctions doivent vérifier les conditions suivantes :   * f est une fonction propre : elle est définie sur un intervalle non nul. * f est une fonction semi-continue inférieure : son épigraphe est une surface fermée. * f est une fonction coercive : ses limites en +∞ et en -∞ sont +∞. * f est strictement convexe : f est convexe et sans parties ‘plates’. |

|  |
| --- |
| **Point de Cours : Choix du pas de descente λ**  Le choix d’un pas de descente est particulièrement important, plus encore que le point de départ car un mauvais point de départ avec un pas de descente adapté peut tout de même converger vers le bon minimum global.  Le pas de descente caractérise le poids que l’on veut donner au gradient par rapport à la valeur en laquelle il est calculé.  Un mauvais pas de descente peut provoquer de nombreux problèmes tels que :   * Rallongement des temps de calculs et démultiplication des étapes * Divergence de l’algorithme * Oscillation de l’algorithme entre plusieurs valeurs différentes du minimum global   Il est notable que pour éviter de rencontrer les deux derniers problèmes, il est important que le pas de calcul respecte une contrainte forte :  En considérant f comme étant β-lipschitzienne alors : |

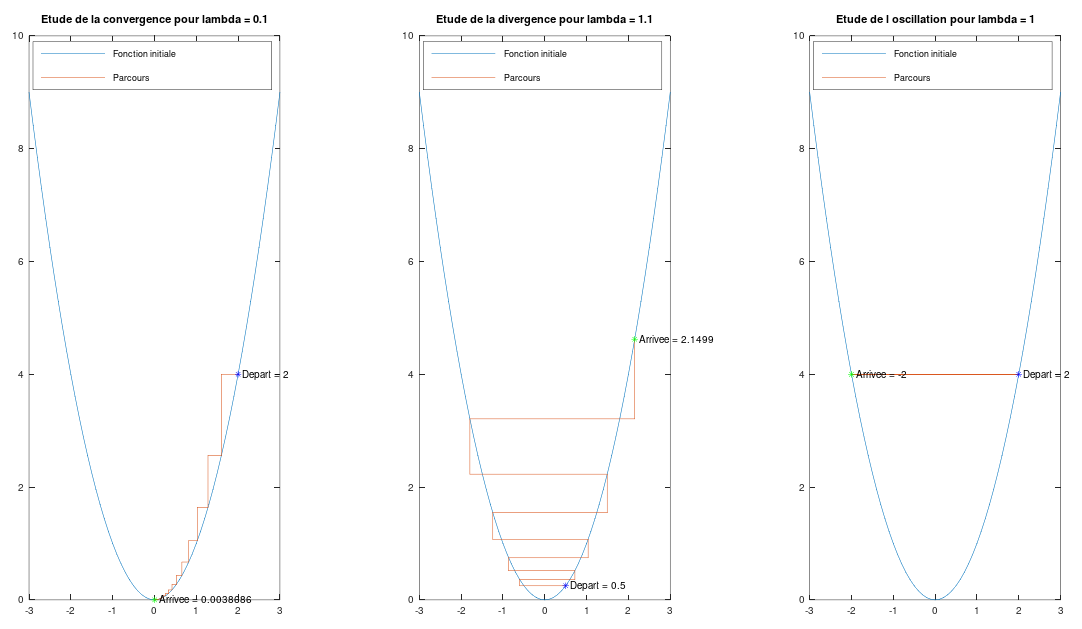
**Illustration :**

Pour cette étude, la fonction convient parfaitement.

Tout d’abord il est notable que la fonction f est 2-lipschitzienne, en effet :

soit :

Afin de vérifier que l’hypothèse sur le pas de descente est bien nécessaire pour qu’il y ait convergence, il suffit de faire fonctionner l’algorithme avec des pas de descentes qui ne respectent pas cette condition :

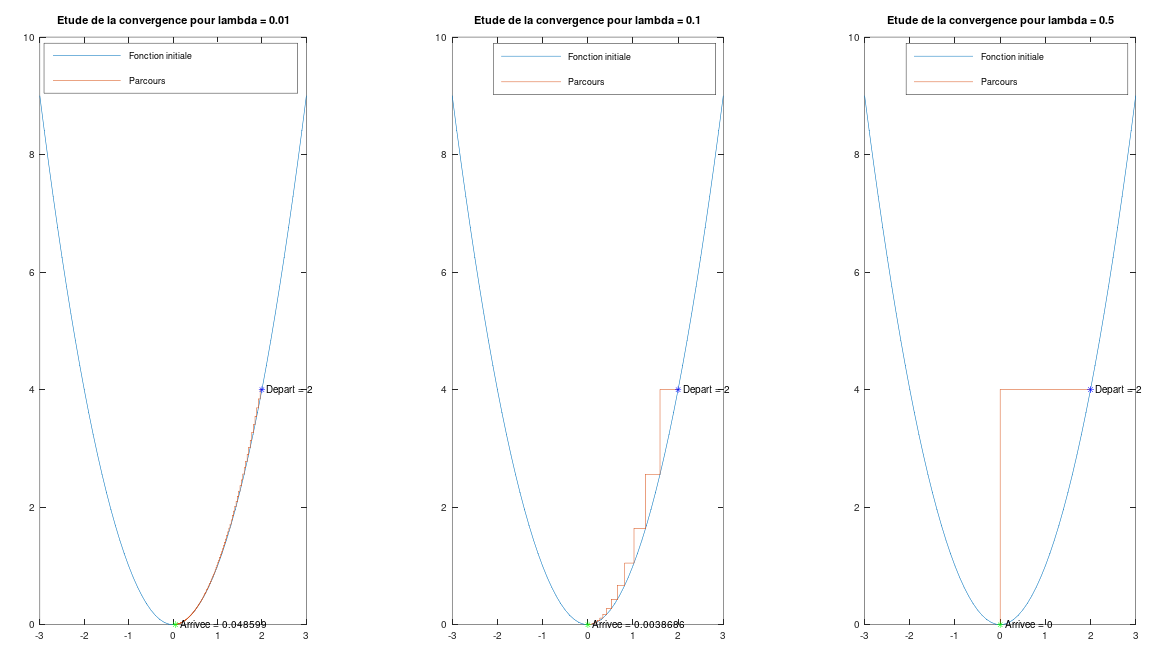


On constate alors ici que la condition est réellement nécessaire, en effet, dans le premier cas, la condition est respectée donc tout va bien, mais dans les deux autres cas, la condition n’est pas respectée ce qui provoque deux phénomènes distincts :

* Lorsque le pas est supérieur à 2/β, alors l’algorithme diverge complètement
* Lorsque le pas est égal à 2/β, alors l’algorithme oscille entre deux valeurs qui n’ont aucun lien direct avec la valeur du minimum global.



Malgré tout, même si le pas de descente respecte la condition, des problèmes d’optimisation de l'algorithme peuvent surgir :



On constate alors que pour cette fonction, le pas de descente a un grand rôle sur le nombre d’étapes et le temps de calcul. En effet, dans le premier cas où le pas de descente n’est pas du tout optimisé, il faut 1053 étapes pour atteindre le minimum global ; dans le second cas il en faut 119 alors que dans le troisième cas 1 seule étape est nécessaire pour pouvoir trouver le minimum global de la fonction.

|  |
| --- |
| **Point de Cours : Conclusion**  La méthode de descente de gradient est particulièrement efficace et peut s’avérer très rapide dans les cas où les paramètres éligibles sont bien choisis. Malgré tout, elle peut être sujette à des limites dont il faut avoir conscience lors de l’utilisation de celle ci :   * Sensible aux minima locaux d’une fonction * Dépendante du caractère dérivable de la fonction * Dépendante du caractère β-lipschitzien sur une partie de la fonction * Risque de divergence ou d’oscillations * Besoin de choisir deux paramètres indépendants   Il est également à noter que l’approximation obtenue par la méthode de gradient est issue d’un développement de Taylor à l’ordre 1 mais que d’autres méthodes permettent d’obtenir une approximation bien meilleure car elles se basent sur un développement de Taylor à l’ordre 2, c’est le cas par exemple de la méthode de Newton. Soit il ne s’agit plus de trouver des droites tangentes mais des fonctions polynomiales d’ordre 2 qui “ressemblerait le plus” à la fonction. Cette méthode serait donc particulièrement efficace pour la fonction x → x2 puisque c’est exactement une fonction polynomiale d’ordre 2. |

1. Résolution du modèle des moindres carrés

|  |
| --- |
| **Point de Cours : Le modèle des moindres carrés**  Ce modèle est principalement basé sur une explication probabiliste : considérons que seule l’observation z est connue. Alors :  avec la réalité et n un bruit.  **1er a priori :** on considère que n suit une loi normale centrée et d’écart type σ.  **2nd a priori :** on considère que .  **3ie a priori :** on considère que P(z) suit une loi uniforme  On cherche alors à maximiser notre connaissance de alors que l’on ne connaît que z, ce qui revient à maximiser la probabilité P(|z) (soit maximiser le maximum de vraisemblance). Or d’après Bayes :  Car :  Sachant que l’on cherche à maximiser cette probabilité et avec n = z- :  On cherche à minimiser :  Ainsi donc en ne considérant pas de régularisation dans un premier temps soit en considérant que , le problème d’optimisation se transforme en la résolution du modèle des moindres carrés soit trouver l’image (ou le signal) x qui minimise l’expression :  Avec H le flou et z l’observation de l’image (soit l’image dégradée). |

**Illustration :**

Afin d’illustrer le type d’observation qui peut être obtenue, voici par exemple une sinusoïde bruitée :



Aussi il est assez évident que le but final de l’étude revient à retrouver le signal ci dessus en bleu soit la sinusoïde à proprement parler.

Il est notable que la résolution de cette équation possède une solution analytique :

Il est également possible de rechercher cette solution à l’aide de la méthode de descente de gradient précédemment évoquée avec :

Ce qui permet d’obtenir :

Il est ici remarquable que la solution optimale trouvée (donc analytique : en violet) apparaît comme étant fortement bruitée et pas du tout fiable alors que le signal numérique avec une précision à 10^-3 près semblerait meilleur. Mais lorsque la précision est alors bien augmentée, on constate alors que le signal numérique vient se coller au signal analytique. Donc il apparaîtrait que la solution analytique et donc numérique précise coïncident et sont donc bien les meilleures approximations que l’on puisse avoir du signal d’origine.

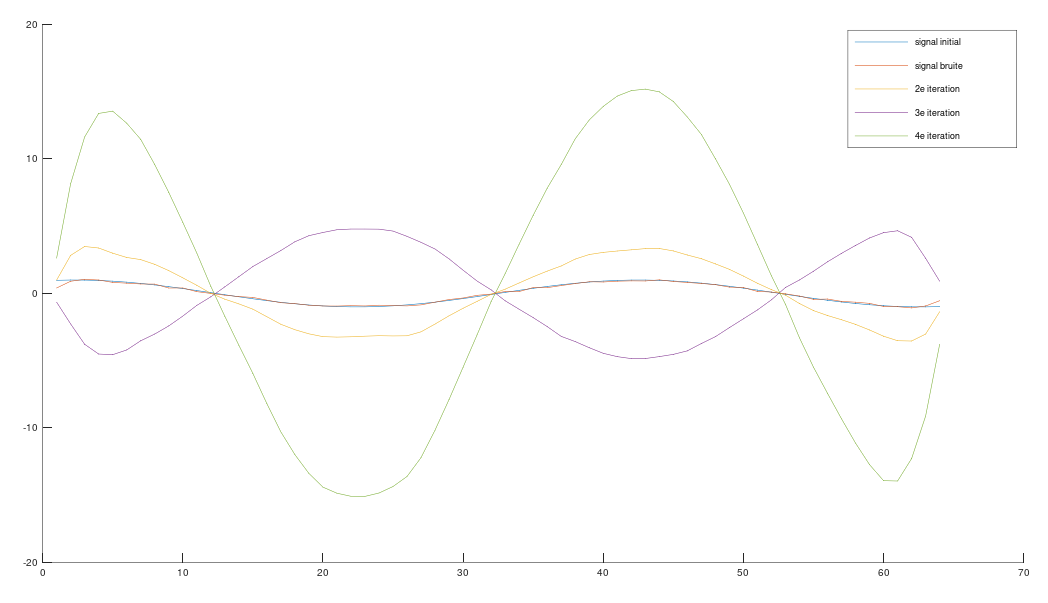
|  |
| --- |
| **Point de Cours : Conditionnement de la matrice H**  Lors de la résolution analytique de la méthode des moindre carrés, ceci a permi de faire surgir la solution :  car    Ainsi la solution analytique correspond donc bien à la solution originale à laquelle s’ajoute une distorsion du bruit. À cet instant deux possibilités :   * H est bien conditionnée ce qui implique une réduction du bruit * H est mal conditionnée ce qui implique une amplification du bruit |

Les résultats précédents témoignent d’un mauvais conditionnement de la matrice H et c’est pourquoi une régularisation est donc nécessaire. Cependant avant de traiter ladite régularisation, il peut être particulièrement intéressant de jouer avec les paramètres de la descente de gradient qui rappelons le, sont fondamentaux.

* Modification du pas de descente :

Les résultats suivants sont obtenus pour différents pas de descente compris entre 0 et 1. En effet dans le cas contraire :





On observe bien une divergence de la méthode.

Aussi, en restant dans l’intervalle précédemment décrit, on obtient :

Il n’est pas évident à première vue de trouver de réelles différences sur le résultat final, la principale différence se joue sur le temps de calcul, en effet il a été beaucoup plus rapide d’atteindre la solution finale lorsque λ = 0.75 que pour les deux autres : 

4367 étapes lorsque λ = 0.1 ; 2205 étapes lorsque λ = 0.5 ; 1328 étapes lorsque λ = 0.75

* Modification de l’initialisation

Afin de déterminer le point d’initialisation optimal, plusieurs initialisations ont été choisies :



Il n'apparaît aucun réel changement aussi bien sur le résultat final que sur les temps de calcul car :

1441 étapes lorsque x0 = vecteur nul; 1406 étapes lorsque x0 = xbar; 1458 étapes lorsque x0 = z

Ce résultat était assez prévisible dans le sens où notre terme d’attache aux données respectent toutes les conditions permettant de statuer sur l’existence et l’unicité du minimum trouvé (soit fonction propre, coercive, sci et strictement convexe).

|  |
| --- |
| **Point de Cours : Conclusion**  On constate que le bon conditionnement de la matrice H joue un rôle crucial dans la bonne résolution du problème. En effet si dans un premier temps, à une faible précision le résultat obtenu numériquement paraît plus satisfaisant, il apparaît qu’en augmentant la précision, un phénomène de sur-apprentissage qui amplifie grandement le bruit se produit et fausse complètement l’approximation finale. C’est pourquoi, il est nécessaire d’introduire une régularisation afin de pallier ce problème. |

1. Résolution du modèle de Tikhonov 1D

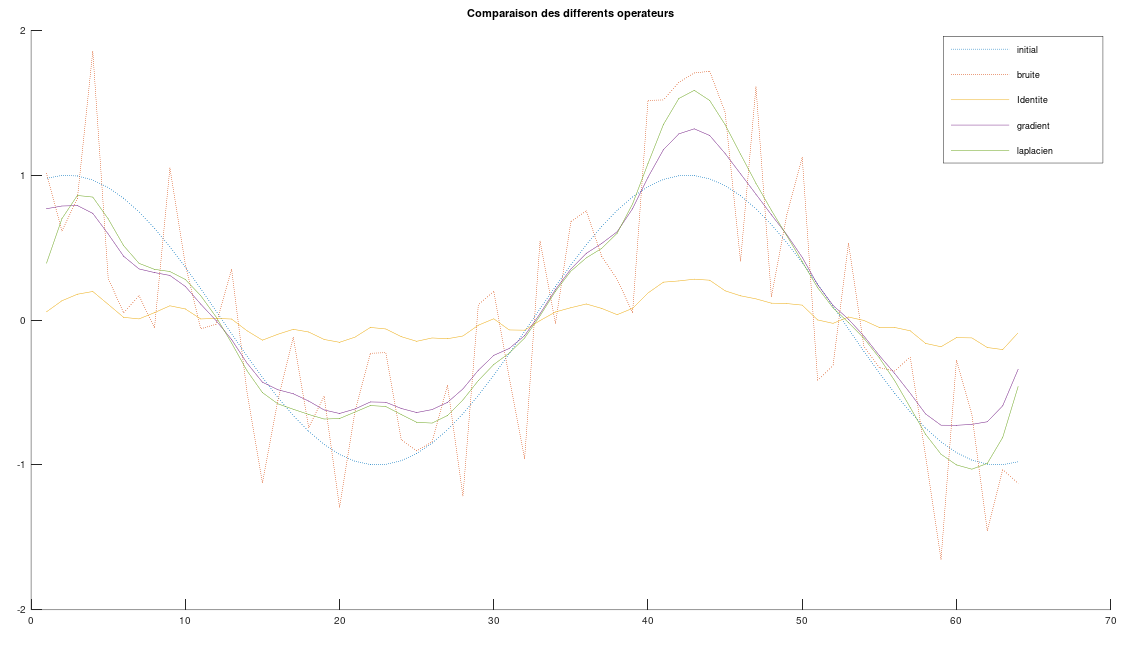
|  |
| --- |
| **Point de Cours : La régularisation de Tikhonov**  Cette méthode de régularisation est définie de la manière à minimiser l’expression suivante :  Avec Γ un opérateur du type identité, gradient ou Laplacien et λ le facteur de “poids” de la régularisation. |

**Illustration de la méthode :**

Le calcul de la descente de gradient sera nettement changé car désormais l’on considère :

Et la solution analytique optimale elle aussi a été modifiée :

Suite à ces changements, il est alors possible d’utiliser les trois opérateurs avec les mêmes paramètres afin d’évaluer leurs différences :



|  |
| --- |
| **Point de Cours : Les opérateurs**   * L’opérateur Identité :   On peut remarquer que l’opérateur identité a tendance à être très drastique, et force le signal à se rapprocher de la droite nulle. Il s’agit donc d’un opérateur particulièrement efficace si l’on sait que notre signal initial est composé en grande majorité de valeurs faibles.   * L’opérateur Gradient :   On peut remarquer que l’opérateur gradient permet une meilleure approximation de la sinusoïde. En effet, celui ci permet de forcer les dérivées du signal à s’annuler ce qui justifie les nombreuses “ondulation” du signal.   * L’opérateur Laplacien :   On peut remarquer que l’opérateur Laplacien permet une approximation qui se rapproche plus du signal d’observation que les autres tout en reprenant les caractéristiques du signal que l’on souhaiterait. Cet opérateur permet lui de minimiser les différences au sein d’un même voisinage de chaque point.  Ainsi donc le choix de l’opérateur dépend essentiellement de la connaissance a priori du signal que l’on souhaite avoir, en effet, si l’on sait que le signal possède de nombreuses valeurs faibles, l’opérateur identité paraît alors le plus adapté. Si l’on sait que le signal possède de nombreuses dérivées nulles, alors il faudra plus s’orienter sur l’opérateur gradient. Enfin si l’on sait que le voisinage de chaque point lui est très similaire, alors le Laplacien apparaît comme le meilleur choix. |

|  |
| --- |
| **Point de Cours : Rapport entre le poids λ et le bruit**  Il est à noter que le poids λ quantifie la confiance que l’on a en notre observation initiale. En effet, si l’on sait que notre observation est très bruitée, dans ce cas il paraît nécessaire d’utiliser un poids relativement fort sur la régularisation. À l’inverse, si le signal observé est relativement peu bruité, on aura plutôt tendance à faire confiance à cette observation et donc mettre un poids plus faible sur la régularisation. |

**Illustration :**

En reprenant la sinusoïde traitée précédemment et ce avec un bruit d’amplitude +/-0.1 puis d’amplitude +/-1, il est constatable que :



On remarque alors bien que lorsque le bruit est très fort, on ne peut pas se fier aux valeurs de poids trop faibles parce qu’elles sont beaucoup trop perturbées par le bruit alors que lorsque le bruit est faible alors il est plus aisé de se fier aux résultats obtenus même si la valeur du poids est faible. Au contraire, on constate que peu importe si le bruit est fort ou faible, le résultat obtenu avec un poids fort reste relativement identique ce qui est très naturel puisque le terme d’attache aux donnée n’est presque pas considéré. Aussi, sachant que l’on ne peut pas avoir confiance en l’attache aux données lorsque le bruit est fort, l’approximation obtenu apparaît comme relativement correct. Mais quand l’observation est relativement fiable ; donc peu de bruit ; on constate que le résultat est bien en deçà de ce que l’on obtient lorsque le poids était faible.

Ceci confirme donc bien le lien fort qui s’exerce entre le bruit et le poids que l’on donne à la régularisation.

|  |
| --- |
| **Point de Cours : Conclusion**  La méthode de Tikhonov est particulièrement efficace pour traiter aussi bien des signaux très dégradés comme des signaux peu dégradés. Le problème majeur de cette méthode est la nécessité de connaître des informations sur le résultat à obtenir :   * la forte présence de valeurs faibles dans le signal ? (particulièrement utile en Machine Learning) * de nombreuses dérivées faibles ? (Un créneau, une porte…) * un signal très lisse et donc une invariance du voisinage ? (Un sinus, une gaussienne…)   La méthode peut donner de bons résultats pour chacun de ces critères. Mais pour cela, il faut connaître les vrais critères.  De plus il faut arriver à jauger à quel point on peut avoir confiance en la régularisation à l’aide du paramètre ce qui amène le problème du compromis entre l’attache aux données et la régularisation. |

1. Résolution de la méthode de Tikhonov en 2D

|  |
| --- |
| **Point de Cours : Et Ailleurs ?**  La méthode des moindres carrés et la régularisation de Tikhonov est parfaitement généralisables dans toutes les dimensions et il en est de même pour les problèmes liés à chacune d’entre elle. Mais elles peuvent très bien être utilisées pour des signaux, des images mais aussi des modèles 3D et ce jusqu’en dimension n. |

**Illustration :**

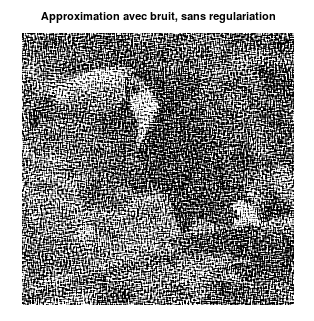
On va chercher ici à travailler sur des images et refaire le même travail que précédemment pour constater que les résultats obtenus en 1D sont également valables en 2D.

* Image initiale et Observation :



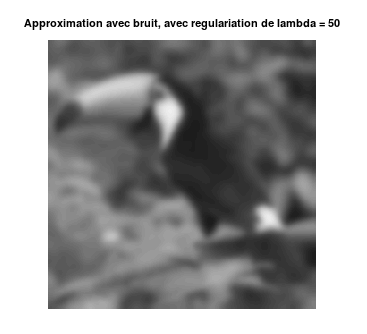
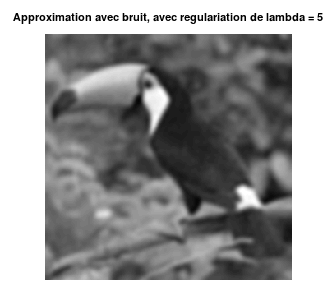
* Résultat sans régularisation :

Comme précédemment il faut comparer deux états de l’images : lorsqu’elle est bruitée mais aussi lorsqu’elle n’est que floutée et ce avec une précision à 5.10^-4 :



De nouveau on peut constater que sans bruit la méthode des moindres carrés fonctionne très bien mais que dès que du bruit y est appliqué, alors l’image subit elle aussi le phénomène de sur-apprentissage qui provient du mauvais conditionnement de la matrice H.

* Résultats avec régularisation :



On peut alors ici constater le même résultat que précédemment : lorsque le poids de la régularisation est trop faible, le bruit reste manifestement apparent (même si ici le résultat est relativement satisfaisant du fait du bruit initial très faible) et quand le poids est trop élevé, l’image est totalement débruitée mais tellement approximée que les principales informations ont disparu et en premier lieu les textures : **la méthode des moindres carrés ne conserve donc pas les textures.**

1. Références

[1] Chapitre sur la descente de gradient, Bruno Bouzy <http://helios.mi.parisdescartes.fr/~bouzy/Doc/AA1/DescenteGradient.pdf>

Chapitre 2 : Optimisation convexe non-lisse

1. Contexte

Les résultats du chapitre précédent ne sont pas particulièrement satisfaisant surtout concernant le traitement sur les images. Aussi pour améliorer ce résultat, un changement de régularisation va être nécessaire. Cette régularisation ne prendra plus en compte la norme 2 mais la norme 1, aussi le problème d’optimisation vise à trouver le minimum de l’expression suivante :

Dans le cadre de ce chapitre, le but est d’étudier le comportement de la régularisation, l’étude portera donc essentiellement sur ce terme et non pas sur le terme d’attache aux données. De plus en une dimension, il est possible de réduire ce problème à la résolution du problème suivant :

1. Sous-différentielle, sous-gradient et algorithme de sous-gradient

|  |
| --- |
| **Point de Cours : Sous-Gradient et Sous-Différentielles**  Afin de généraliser la différentiabilité d’une fonction aux fonctions qui ne sont pas différentiables mais tout de même continues et convexes, il est possible d’introduire la notion de sous-gradient et de sous-différentielle.  Ainsi, un sous-gradient d’une fonction en un point de non dérivabilité est un vecteur donc la direction est comprise entre le gradient à gauche de la fonction en ce point et le gradient à droite de la fonction en ce point.  Ce qui permet d’introduire la notion de sous-différentielle d’une fonction en un point et notée , celle ci est définie comme l’ensemble des sous-gradients de ce point. |

**Illustration :**

En considérant les deux fonctions suivantes :



Il est possible de déterminer aussi bien les différents sous-gradients ainsi que les sous-différentielles de chaque fonctions et ce de manière graphique mais aussi de manière numérique :



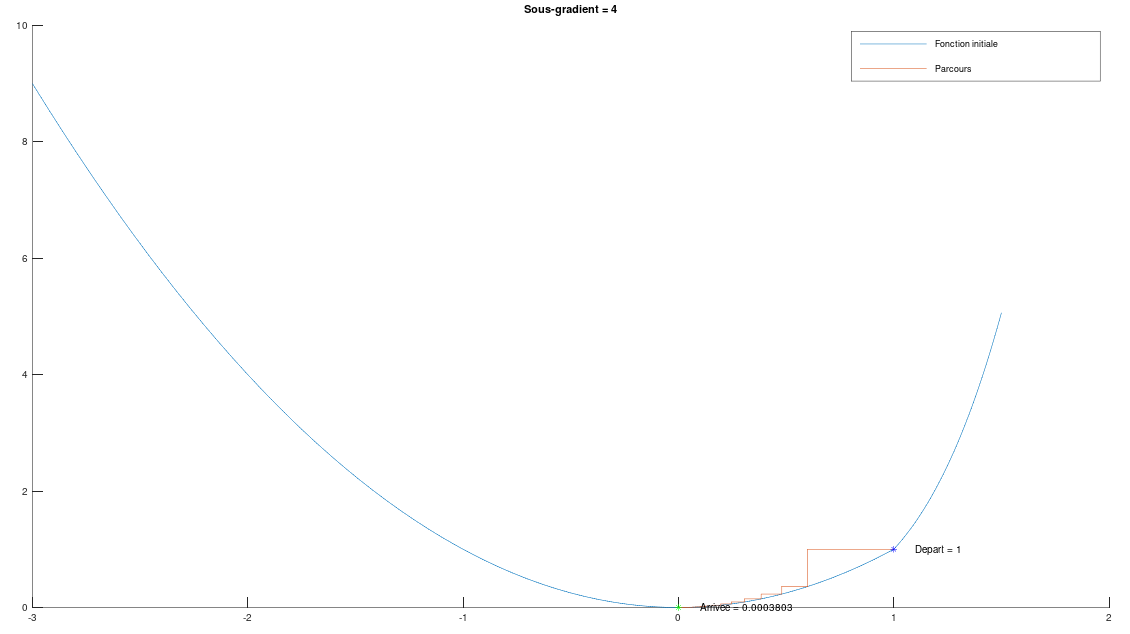
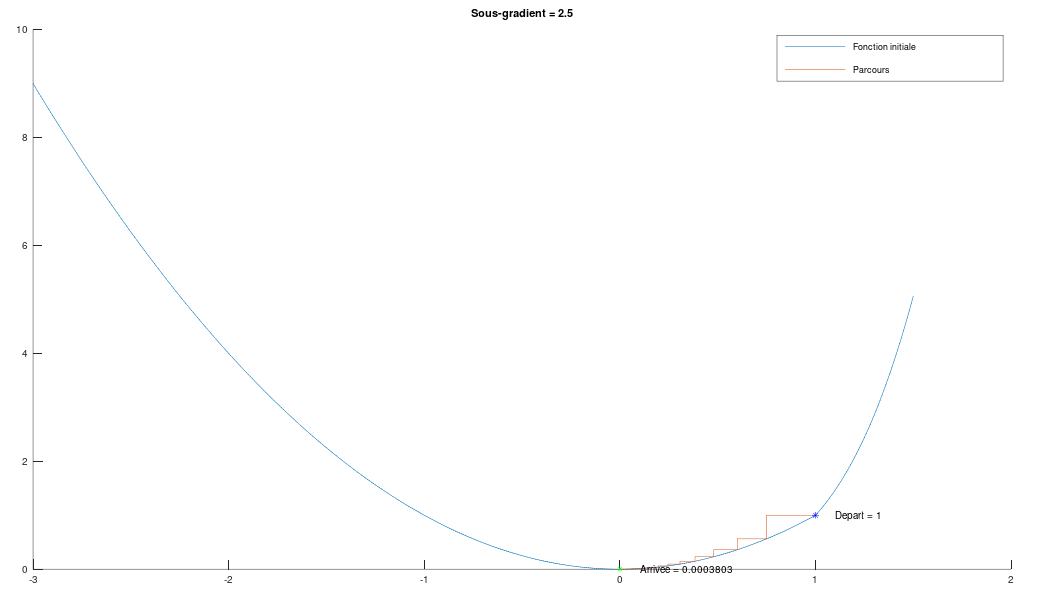
On en déduit numériquement :



|  |
| --- |
| **Point de Cours : Algorithme de sous-gradient**  De la même manière que l’algorithme de descente de gradient, il est possible de déterminer un algorithme de descente de sous-gradient pour les fonctions non dérivables en certains points mais pourtant convexes et continues.  Le principe est le même que pour le chapitre précédent à ceci près que l’on doit “choisir arbitrairement” un sous-gradient comme gradient en les points non dérivables. Ceci ne doit avoir aucun impact sur la méthode de descente puisque peu importe le sous-gradient choisi, celui ci nous amènera nécessairement sur une meilleure approximation.  De plus une condition est nécessaire afin que cet algorithme fonctionne : il faut que 0 appartienne à l’ensemble des sous gradients dans le cas où le point de “cassure” est le point recherché par l’algorithme. |

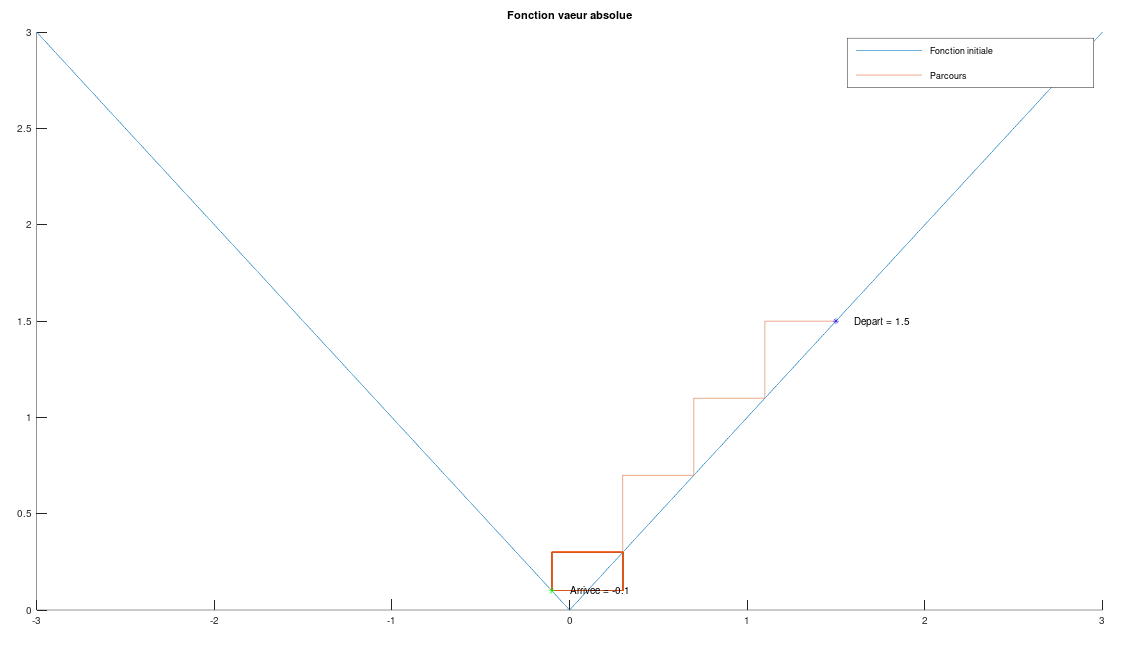
**Illustration de la méthode :**

Dans un premier temps, pour illustrer le bon fonctionnement de la méthode et que le point de non dérivabilité ne pose pas de problèmes, il est intéressant de se pencher sur la fonction g précédemment définie.

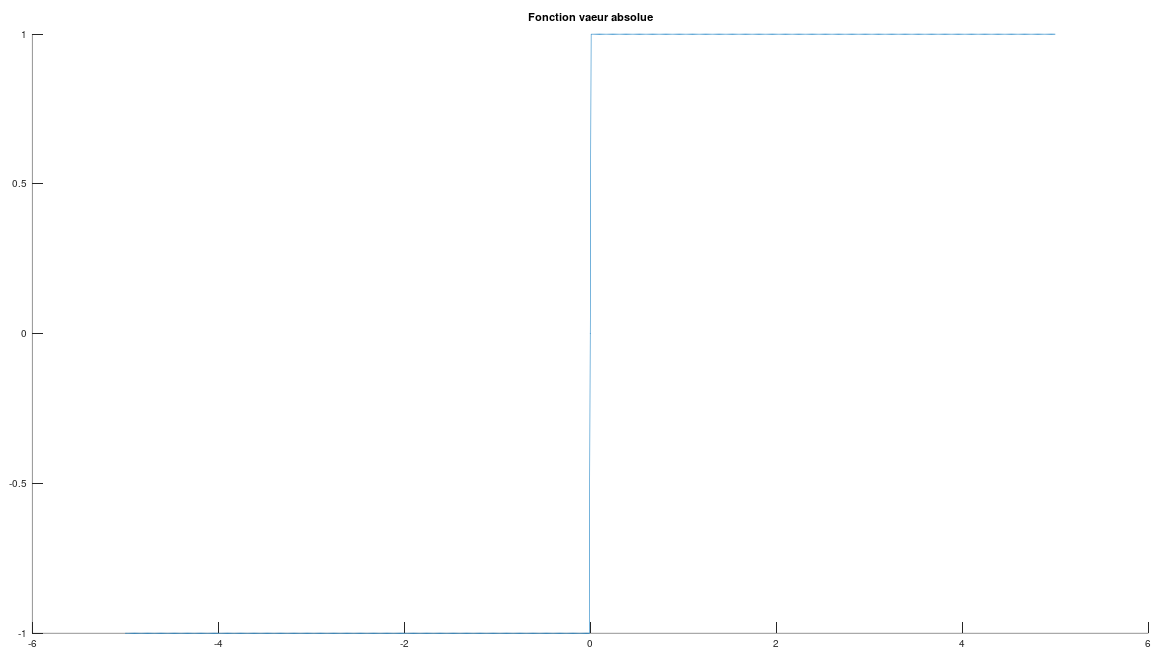


On peut ainsi bien remarquer que le point de non-dérivabilité ne pose pas de problème lors de la méthode de descente de sous-gradient et ce peu importe quel sous-gradient a été choisi.

Il est également possible de regarder le résultat fourni par la fonction f précédente en prenant arbitrairement un sous-gradient en le point de non dérivabilité :



On constate alors que pour tout sous-gradient en ce point, à moins de tomber exactement sur la solution exacte, le résultat fini par osciller, ceci est dû à la valeur du gradient de part et d’autre de la discontinuité :



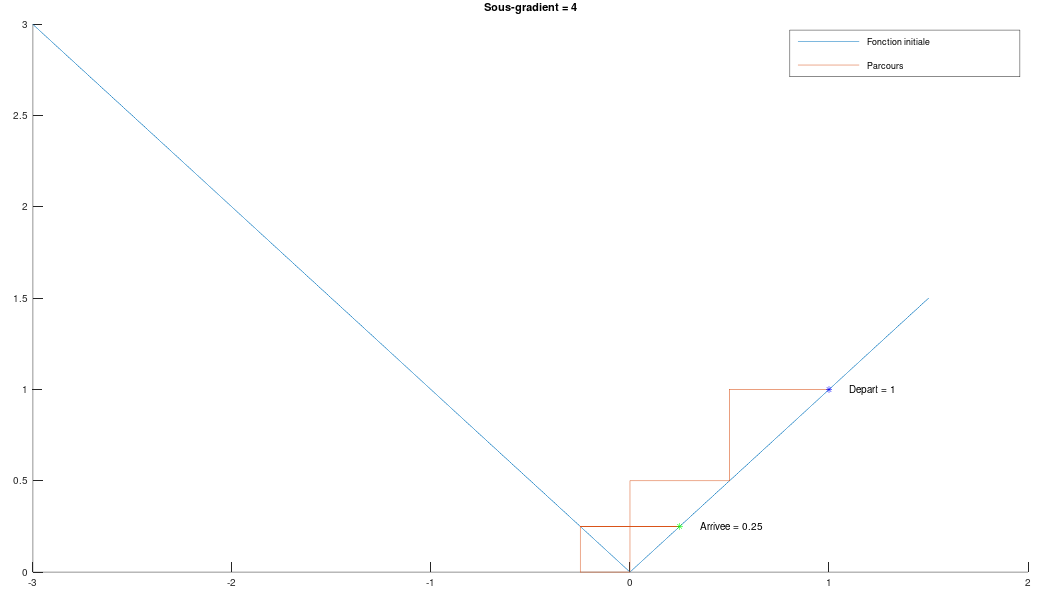
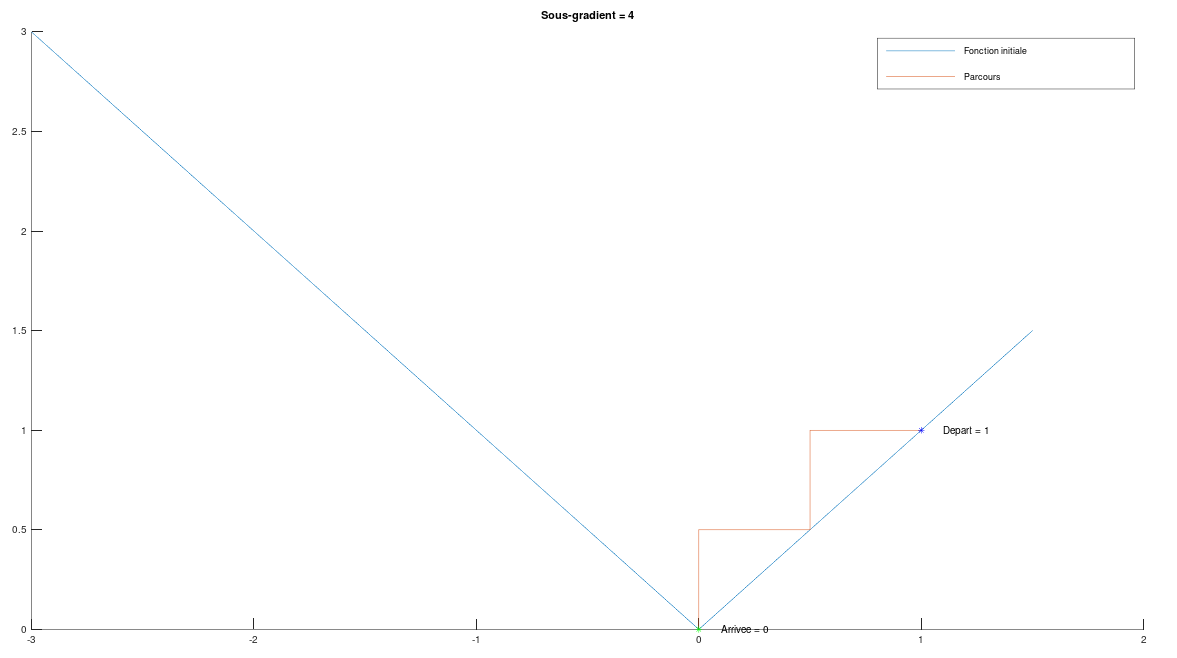
Cette figure illustre particulièrement bien le problème : on peut s’approcher autant que l’on veut du minimum recherché, si on ne tombe pas exactement dessus, alors le gradient est constant et donc ne permet pas de se rapprocher du minimum mais finit par aller sur la pente opposée qui elle même va “renvoyer la balle” à la pente initiale. Ce problème est dû à la non continuité du gradient qui ne permet donc pas de tomber sur le résultat souhaité.

|  |
| --- |
| **Point de Cours : Limites de l’algorithme de sous-gradient**  L’algorithme de sous-gradient possède exactement les mêmes limites que l’algorithme de descente de gradient mais à celles-ci s’ajoute un nouveau problème : si le point de non dérivabilité est exactement le minimum de la fonction, alors l’algorithme ne permettra de le déterminer qu’à une seule condition :   * le pas est réglé de telle sorte que l’algorithme tombe exactement sur le point de non dérivabilité * le sous-gradient défini en ce point est nul   Ces conditions sont particulièrement difficiles à obtenir dans le cas non différentiable car le gradient de la fonction change du tout au tout dès que l’on dépasse le point de recherche : il n’y a pas de réduction progressive et continue du gradient ce qui implique la création d’oscillations. |

**Illustration :**

Dans le cas de la fonction valeur absolue, cet exemple est particulièrement probant, en effet en considérant le pas de descente à 0.5 ce qui permet de tomber directement sur le point de non dérivabilité alors :

Si le sous-gradient est nul Si le sous-gradient est non nul



1. Enveloppe de Moreau et opérateur proximal

|  |
| --- |
| **Point de Cours : L’enveloppe de Moreau**  L’enveloppe de Moreau est définie pour toute fonction f de la manière suivante :  Cette fonction a pour majeure particularité de conserver exactement le même minimum que la fonction initiale. Aussi, une fois cette fonction déterminée, celle-ci étant dérivable en tout point, il est alors possible de faire une descente de gradient sur cette fonction et ainsi de déterminer le minimum de la fonction f en trouvant le minimum de la fonction Mf .  NB : Cette fonction peut également être vue comme le résultat de la dilatation de la fonction initiale par la fonction g précédente. |

|  |
| --- |
| **Point de Cours : L’opérateur proximal**  Afin de déterminer l’enveloppe de Moreau plus facilement, il peut être particulièrement efficace de passer par l’utilisation de l’opérateur proximal. Celui-ci va de pair avec l’enveloppe de Moreau car il consiste à rechercher l’argument y pour lequel la fonction g est minimale. Cet opérateur se défini de la manière suivante : |

|  |
| --- |
| **Point de Cours : Enveloppe de Moreau et opérateur proximal**  Pour déterminer l’enveloppe de Moreau, il suffit de mettre en oeuvre l’équation suivante : |

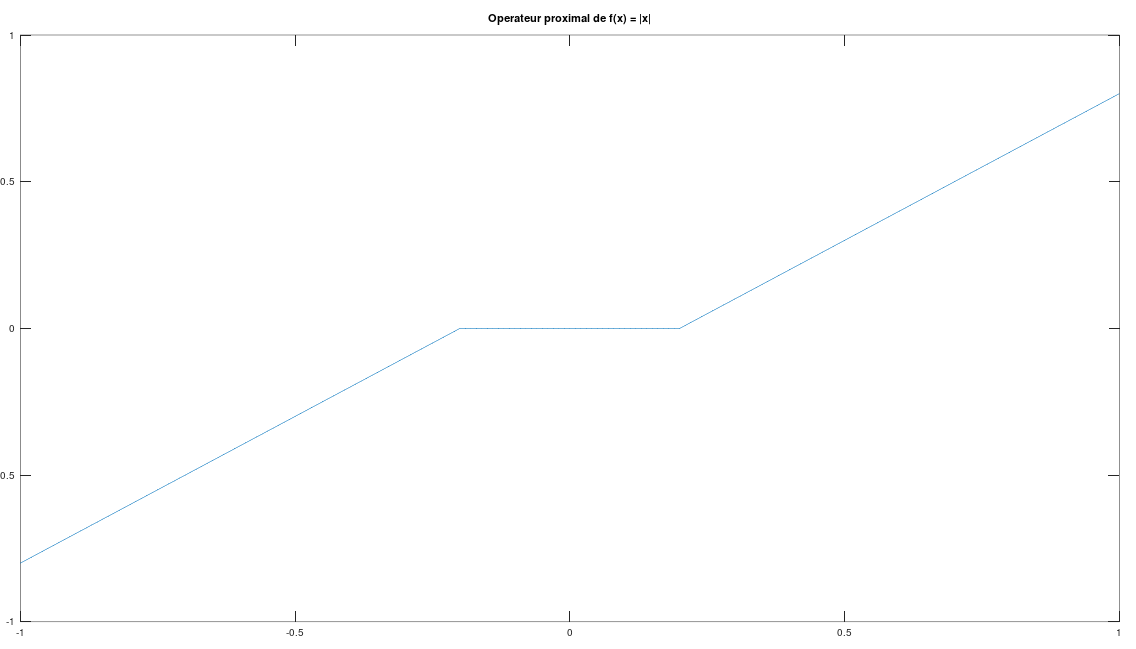
**Illustration :**

De nouveau, il est pertinent de s’intéresser à la fonction valeur absolue. En effet, pour cette fonction, le calcul de l’opérateur proximal devient :

Rechercher l’argument minimal revient à rechercher y tel que la dérivée de la fonction s’annule soit :

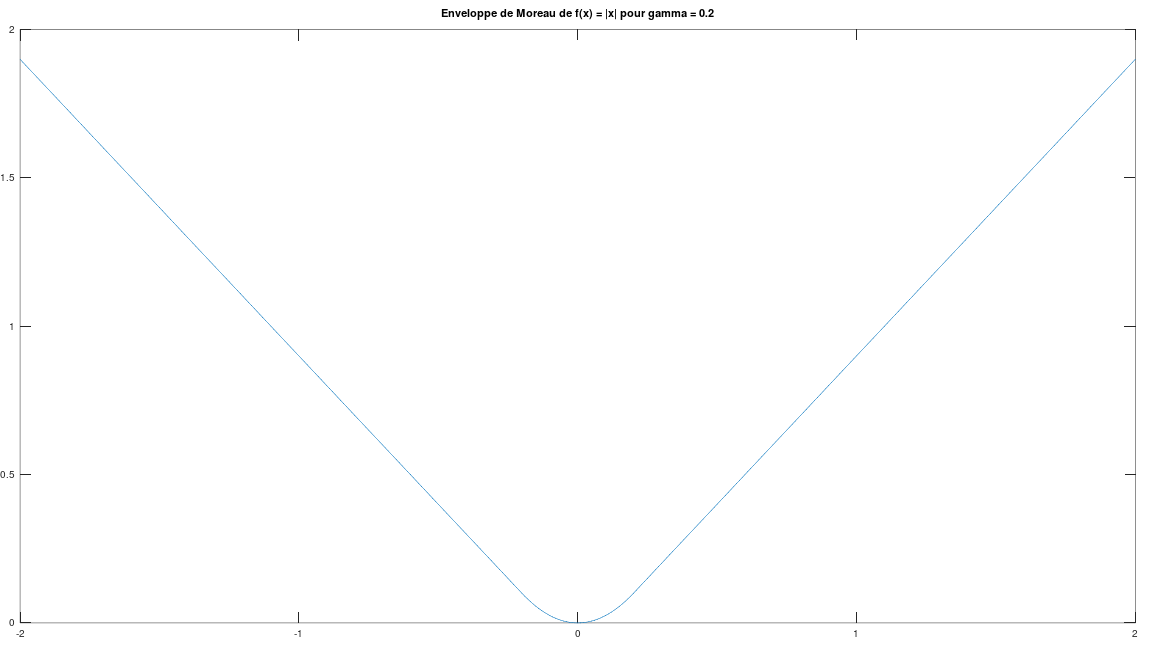
|  |  |  |
| --- | --- | --- |
| si y < 0  Et donc pour respecter y < 0 : | si y > 0  Et donc pour respecter y > 0 : | sinon y = 0  Et donc : |

Il est alors possible de visualiser le-dit opérateur proximal :



Ce qui permet d’obtenir aisément l’enveloppe de Moreau du signal :

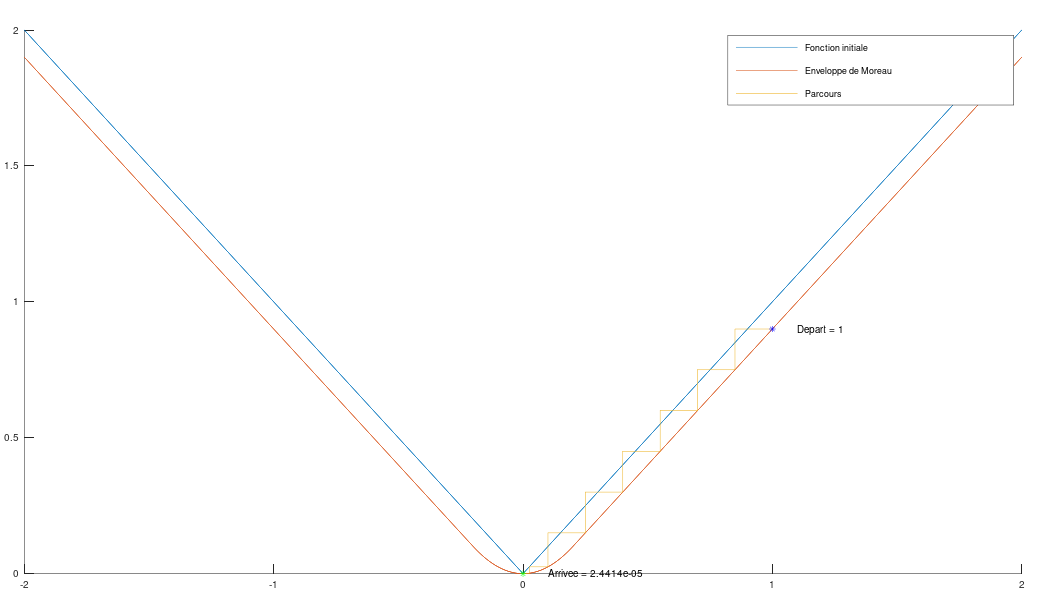




Il est alors notable que cette fonction possède plusieurs caractéristiques notables :

* Sa dérivabilité en tous points
* Sa convexité, sa coercivité
* Il s’agit d’une fonction propre
* **Son minimum est le même que le minimum de la fonction initiale**

Aussi, en réitérant l’algorithme de descente de gradient non plus à la fonction initiale mais à son enveloppe de Moreau, il est possible d’obtenir le minimum de la fonction initiale :



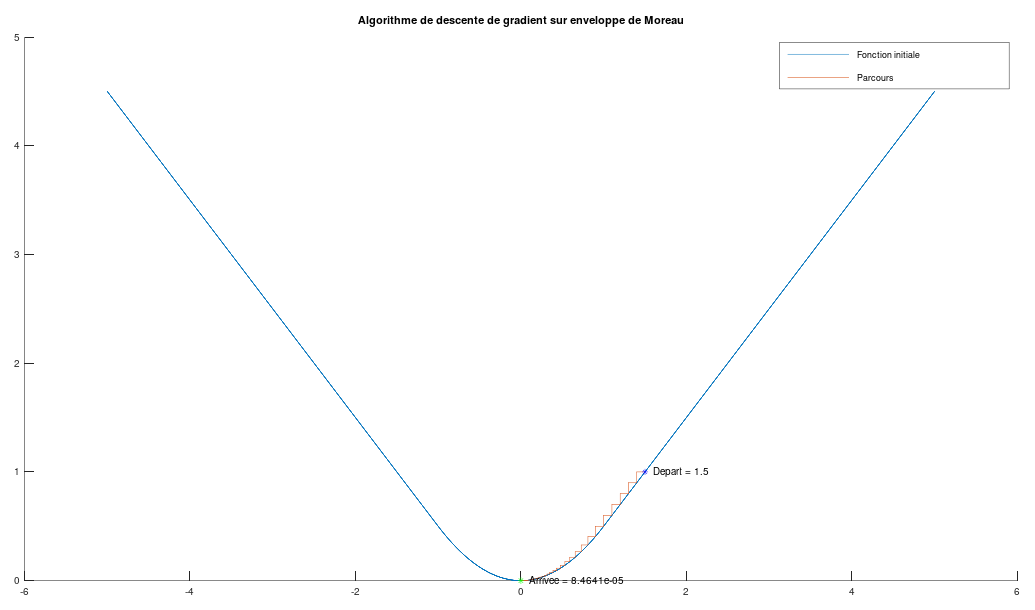
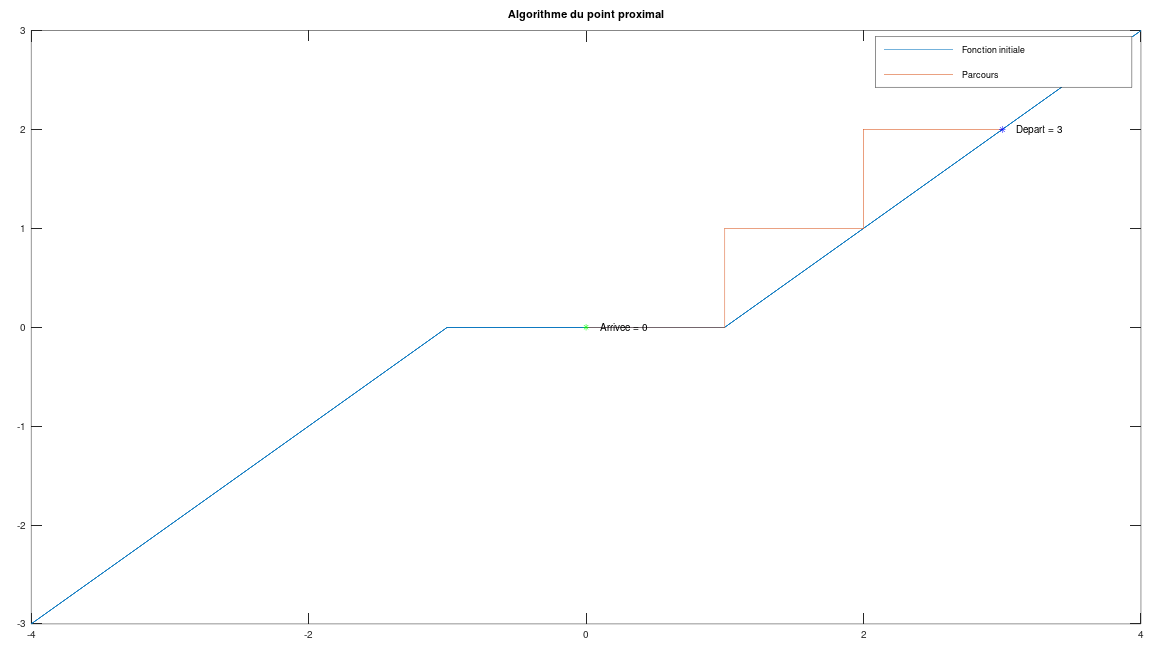
Dans le cas présent, la précision étant de 10-5, il est constatable que l’algorithme finit bien par se rapprocher du minimum en question.

1. Algorithme du point proximal

|  |
| --- |
| **Point de Cours : Algorithme du point proximal**  L’algorithme du point proximal se base sur une relation que l’on admettra qui est la suivante :  Il est alors possible de retrouver le principe même de l’algorithme de descente de gradient mais appliqué à l’enveloppe de Moreau. Cette relation permet ainsi de calculer beaucoup plus simplement et beaucoup plus rapidement le minimum recherché.  De plus, cette relation signifie qu’il n’y a pas besoin de connaître l’enveloppe de Moreau pour pouvoir obtenir le résultat recherché. Seule la connaissance de l’opérateur proximal de f est nécessaire pour pouvoir mener à bien cet algorithme. |

**Illustration de l’algorithme :**

Afin de comparer la vraie méthode de descente de gradient directement sur l’enveloppe de Moreau et la méthode du point proximal, on considère :



Il est alors remarquable que l’algorithme de point proximal atteint son objectif en 4 itérations seulement alors que l’algorithme de descente de gradient classique atteint son objectif en 94 itérations. De plus, il semble évident qu’en utilisant un encore plus grand, l’algorithme verra son efficacité accrue, ce qui est facilement perceptible à l’aide de la fonction valeur absolue : une augmentation du paramètre signifie un élargissement du plateau central de l’opérateur proximal et par conséquent l’obtention presque instantanée de la valeur recherchée (en 1 seule itération si le point initial est compris entre - et ).

1. Algorithme du gradient proximal

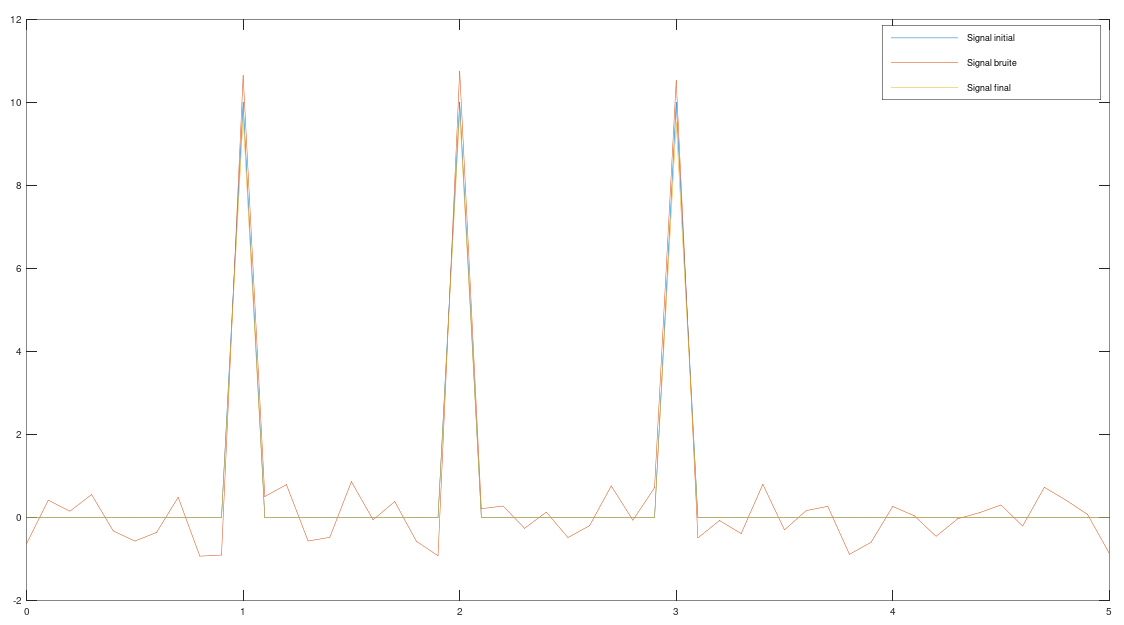
|  |
| --- |
| **Point de Cours : Algorithme du gradient proximal**  L’algorithme du gradient proximal permet de trouver une solution à un problème d’optimisation dans son ensemble. En effet, l’algorithme de point proximal est très efficace lorsque l’on peut déterminer aisément l’opérateur proximal d’une fonction. Or, le problème initial étant de minimiser l’expression suivante :    Aussi, pour pouvoir résoudre ce problème à l’aide de l’algorithme de point proximal, en considérant dans un premier temps et que f est une fonction différentiable, il est possible d’utiliser l’algorithme du gradient proximal qui s’organise sous la forme suivante : |

**Illustration de l’algorithme :**

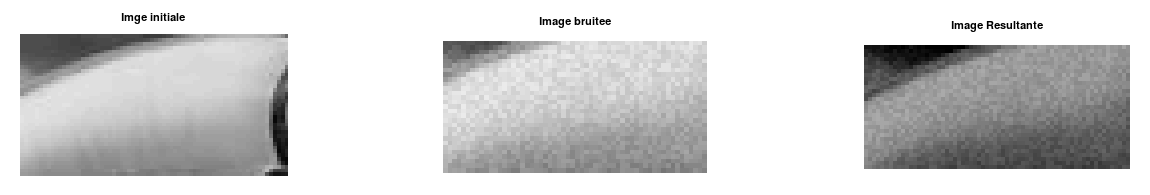
Sachant que la norme 1 équivaut à une somme de valeurs absolues membre à membre, alors il s’agit ici de minimiser chacun des membres. Aussi, l’opérateur proximal est modifié de par la présence du et devient donc :

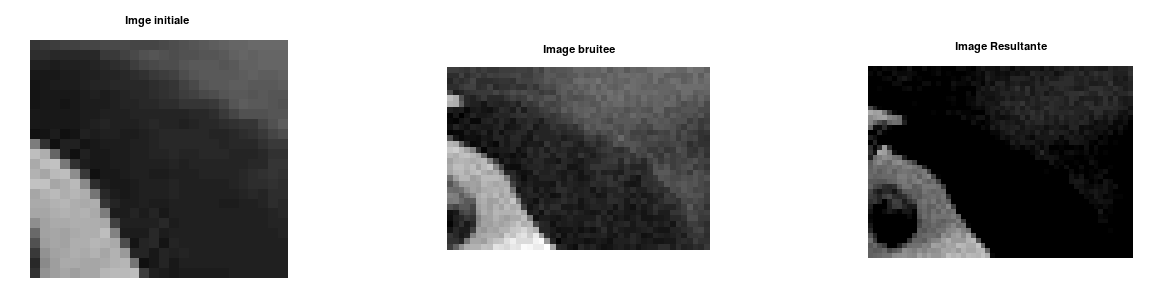


Tout comme la résolution du problème de Tikhonov, puisque ici l’opérateur utilisé est l’identité, cette méthode convient donc pour un signal comportant de nombreuses valeurs très faibles. Aussi, considérons une observation d’un signal nul comportant des pics de valeurs à 10 en 1,2 et 3 bruité. Alors, le passage de cette observation dans l’algorithme permet d’obtenir :



Cette méthode est donc particulièrement efficace pour les signaux en 1D à l’aide de l’opérateur identité. Il est à noter que les opérateurs gradient et Laplacien ne peuvent pas actuellement être mis en place pour des raisons d’adaptation de l’opérateur proximal. Ces deux opérateurs seront traités dans la partie suivante. Avant de passer à cette partie, il pourrait être intéressant de voir le comportement de l’algorithme lorsqu’il s’agit de la 2D soit avec une image :





On peut alors ici constater la tendance de chacun des pixels à se rapprocher des valeurs sombres, ce qui est lié à l’utilisation de l’opérateur identité. De nouveau, cet opérateur se caractérise bien par sa capacité à retrouver des signaux/images dont la majorité des valeurs sont nulles. Aussi, on peut remarquer que les zones initialement claires ont été foncées et que le bruit a été très peu diminué (ce qui est particulièrement bien visible sur le bec) tandis que les zones initialement foncées ont bien été débruitées voire même ont perdu la texture de départ (ce qui est bien visible dans le plumage de l’oiseau).

1. Algorithme de Forward-Backward : Régularisation avec opérateur.

|  |
| --- |
| **Point de Cours : Fonction conjuguée**  La fonction conjuguée aussi appelée transformée de Legendre-Fenchel est caractérisée par la définition suivante :  En considérant une fonction f, sa fonction conjuguée f\* est : |

**Illustration :**

Il est possible de calculer la fonction conjuguée de plusieurs fonctions de références :

→

→

|  |
| --- |
| **Point de Cours : Problème primal et problème dual**  Afin de résoudre le problème d’optimisation avec opérateur, il est possible de passer par le problème dit dual. Le problème primal étant :    Alors le problème dit dual est le suivant :  Il s’avère alors que de résoudre le premier problème revient exactement à résoudre le second puis à le repasser dans le domaine primal. |

|  |
| --- |
| **Point de Cours : Opérateur proximal d’une fonction conjuguée**  Jusqu’à présent, l’opérateur proximal a été particulièrement utilisé et il le sera encore par la suite, il est alors à noter que : |

**Illustration :**

Revenons au problème initial et donc primal :

Par passage au dual on obtient alors :

Puis, en passant par l’algorithme de gradient proximal, il convient de mettre en place l’algorithme suivant :

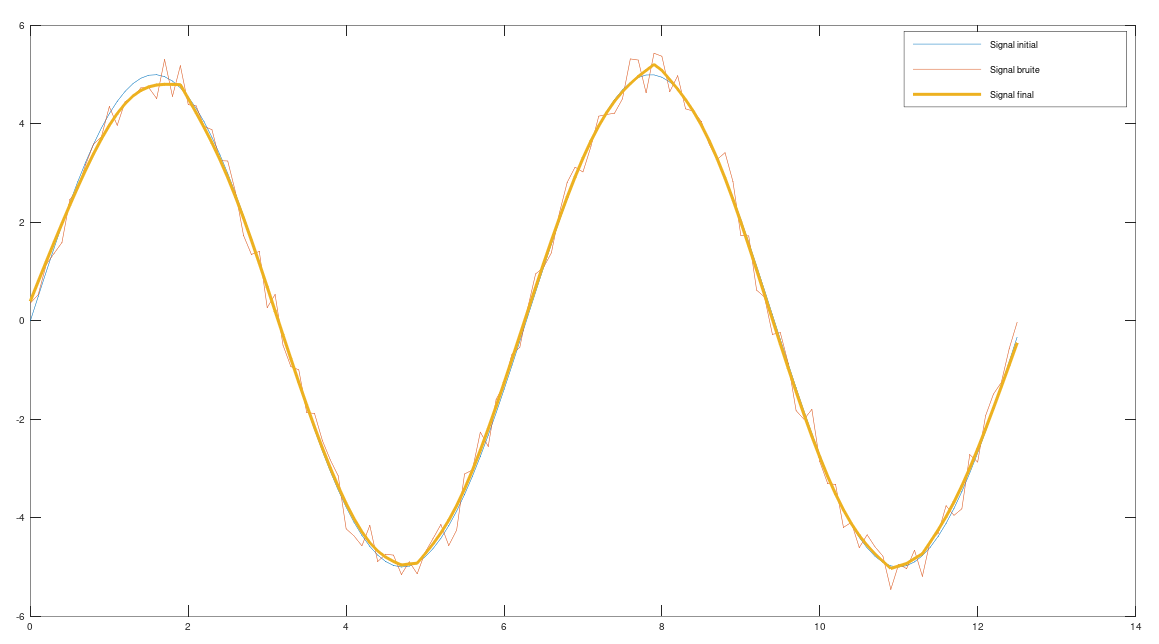
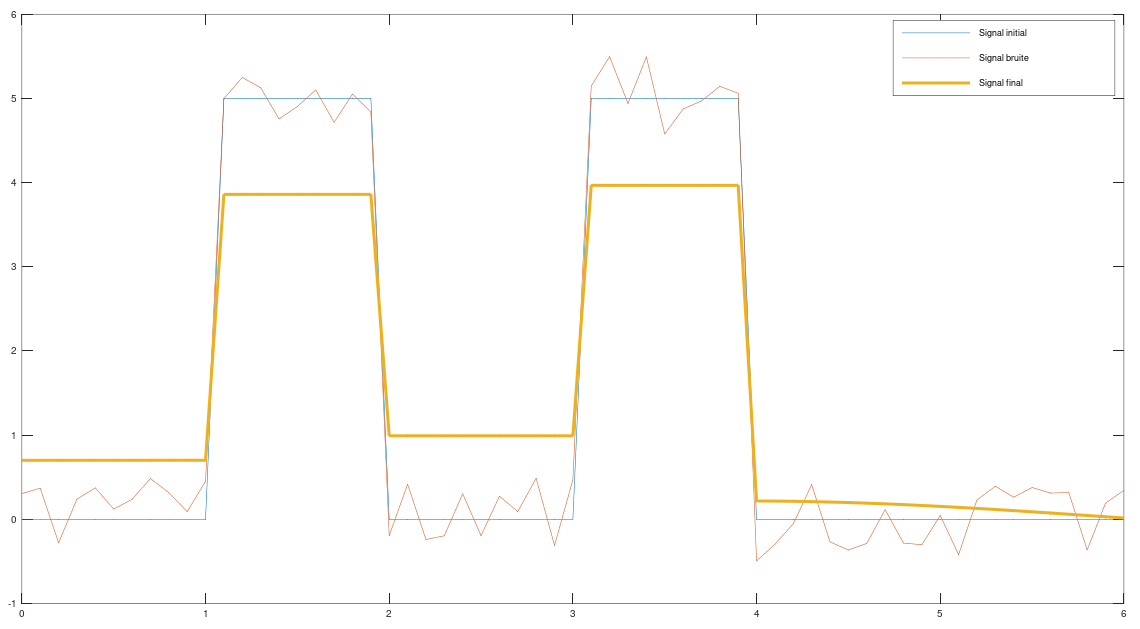
Avec :

|  |
| --- |
| **Point de Cours : Choix de l’opérateur**  Différents opérateurs sont possibles comme précédemment et chacun possède ses avantages et ses inconvénients. Il est à noter que le choix de l’opérateur dépend fortement du résultat que l’on souhaite obtenir :   * Identité : si le signal (image) contient beaucoup de 0 comme les Diracs. * Gradient : si le signal (image) contient de nombreuses pentes nulles tel un signal créneau ou la fonction partie entière. * Laplacien : si le signal (image) ne possède que de très faibles variations de voisinage tel un signal sinusoïdal. |

**Illustration :**

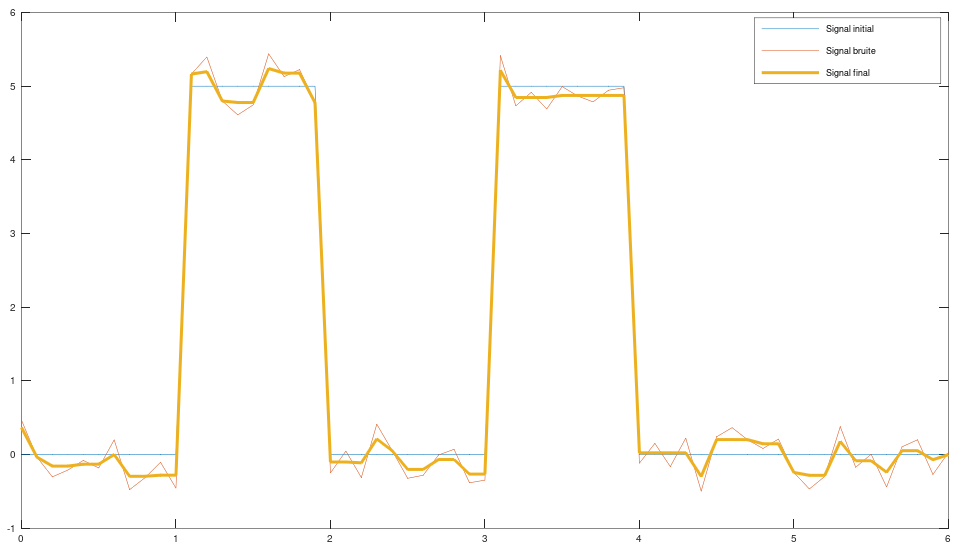
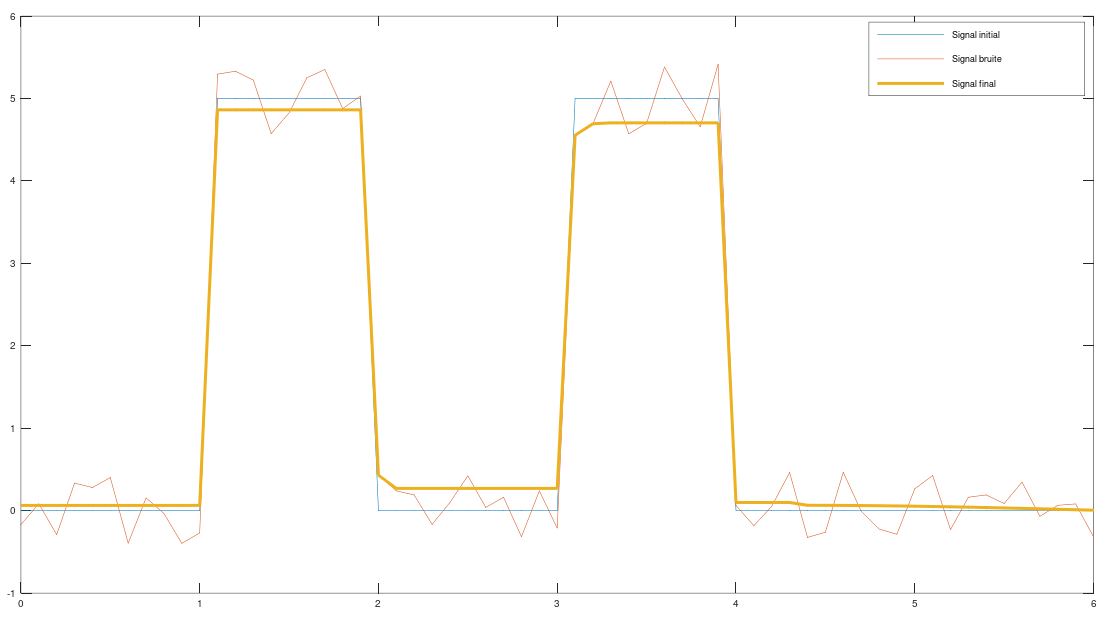
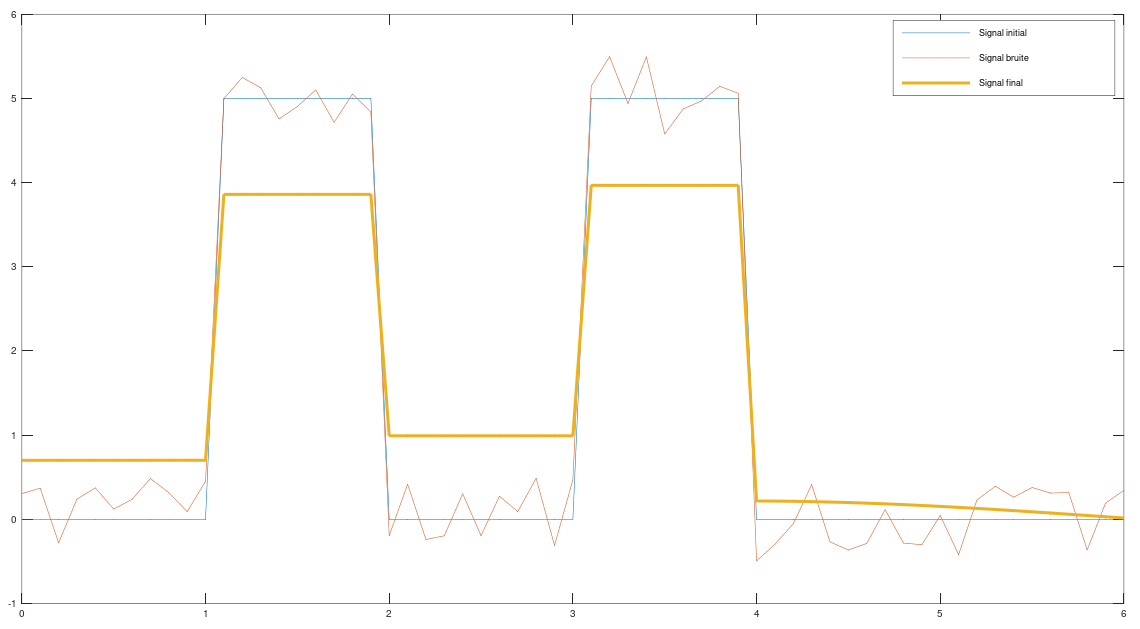
L’illustration concernant l’opérateur identité a déjà été vu plus haut, en ce qui concerne les deux autres opérateurs :

Le Gradient Le Laplacien



Il est alors remarquable que ces deux opérateurs conviennent bien aux signaux précédemment décrits de part leurs caractéristiques propres. Néanmoins, il est possible de constater que certains biais existent, en effet, en ce qui concerne le gradient tout particulièrement, on remarque que toutes les valeurs tendent à s’aligner suivant une seule et unique droite. Ce qui implique des différences avec le signal de départ au niveau des amplitudes. Ceci est normal, puisque lorsque l’on passe d’un niveau bas à un niveau haut, la pente est infinie. Or, la norme 1 utilisée avec le gradient tend à réduire au maximum les pentes, ce qui explique donc la volonté de réduire ce temps de “transition” d’un état à l’autre.

Il est tout de même possible de diminuer cet écart. Pour ce faire, il suffit de faire plus confiance au signal initial, soit diminuer quelque peu la valeur de . Ce qui permet d’obtenir :



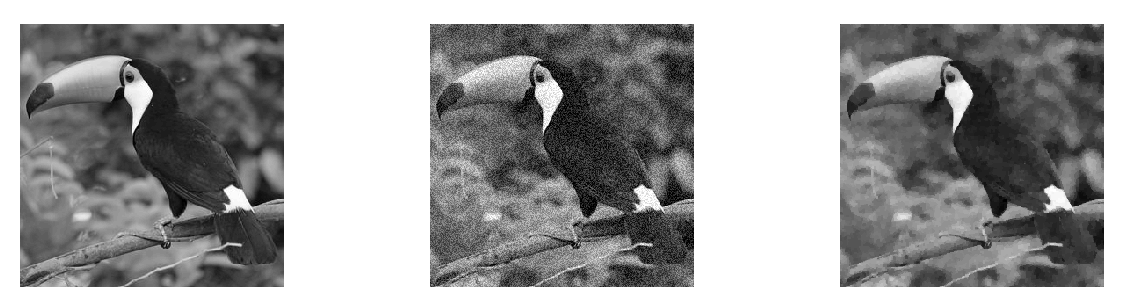
Aussi, il est remarquable que de faire trop confiance au signal initial permet ainsi au bruit de retrouver sa place, même diminué. De plus, même si le signal semble bien meilleur, il faut comme toujours faire preuve de confiance avec parcimonie.

Ce travail peut également être effectué pour le traitement d’une image, ainsi donc en reprenant le toucan précédent, il faut alors rappeler que l’opérateur identité ternissait particulièrement l’image, quand est-il des deux autres ?

Le Gradient



Le Laplacien



On peut alors remarquer que ces deux opérateurs ont tendance à conserver les zones homogènes et même à les accentuer. En effet, il suffit de diminuer encore la confiance que l’on a en notre image initiale pour obtenir :



Il est alors remarquable que les zones les plus homogènes ont été uniformisées et que l’ensemble des textures ont été floutées. On en déduit donc que ces deux opérateurs sont très efficaces pour uniformiser les zones homogènes tout en conservant les contours marqués de l’image mais qu’ils ne permettent pas de conserver les textures. Celles-ci restent malheureusement encore très difficilement conservables.

1. Conclusion

|  |
| --- |
| **Point de Cours : Conclusion générale**  Suite à cette étude, il est alors possible de tirer les conclusions suivantes :   * la norme 1 rend parcimonieuse, c’est-à-dire qu’elle cherche à trouver le plus de 0 possible. Celle-ci permet ainsi de rester proche d’un maximum de point quitte à s’éloigner de certains. * la norme 2 cherche à être “proche de tout le monde”, soit à minimiser la “distance” entre le signal observé et le signal obtenu à la fin |

**Illustration :**

Pour illustrer ce phénomène il est particulièrement intéressant de visualiser un nuage de point et de déterminer la droite issue des normes 1 et 2 :



1. Références

[1] Cours de l’INSA de Toulouse sur l’introduction à l’optimisation, Aude RONDEPIERRE,

<http://www.math.univ-toulouse.fr/~rondep/CoursTD/poly4GMM_nondiff.pdf>

[2] Cours de l’ENS de Lyon sur l’optimisation convexe non lisse, Nelly PUSTELNIK,

<http://pbil.univ-lyon1.fr/members/fpicard/franckpicard_fichiers/master/cours_lyonOptim_main.pdf>