

**Optimisation linéaire**

Récupération d’une image floutée (deblurring)

Gatta Nicolas MINFO

Schellekens Arnaud MINFO

UMONS – Catégorie Sciences informatique – 2022 - 2023

# Introduction :

L’objectif de ce projet est l’étude du problème de minimisation qui permettra de reconstruire de manière approchée une image non flottée et non bruitée en partant d’une image floutée et bruitée.

# Question 1 : Modélisation du problème

Il faut écrire le problème ci-dessus sous forme d’un problème d’optimisation linéaire

**(1)**

Nous savons que la norme 1 d’un vecteur est la somme des valeurs absolues de ses composantes, on applique cela dans la formule (1) ce qui nous donne:

Grâce au la n-ième ligne de ***A***, la n-ième ligne de x et n le nombre de lignes de ***A*** et . En effet, ces matrices doivent avoir le même nombre de lignes et de colonnes dans le cas d’une addition/soustraction et doivent avoir le même nombre de lignes pour une multiplication pour que celles-ci puissent se produire. Enfin, en traitant les valeurs absolues, on obtient :

# Question 2 : Forme standard

Nous devons écrire le problème d’optimisation linéaire formulé à la question 1 sous forme standard.

# Question 3 : déflouté d’une image

Pour déflouter une image, on utilise comme demander dans la question la fonction glpk() de Octave. Cette fonction va nous permettre de résoudre le problème d’optimisation linéaire suivant :

Pour faire fonctionner la fonction de ***glpk***, il a fallu remplacer les deux contraintes par des matrices de 1 colonne et n lignes (correspondant à la taille de xtilde) remplie de 0 pour ***lb*** (borne inférieure) et remplie de 1 pour ***ub*** (borne supérieure).

En plus de cela, il a fallu créer une nouvelle variable contenant les types de contraintes utilisés pour chaque ligne. Vu que nous avons pris le temps de convertir toutes les contraintes «  », la variable ***cty*** ( type de contrainte) va être remplie de « U » correspondant à contrainte avec des bornes supérieures.

Enfin, les n derniers éléments de la solution renvoyer par la fonction ***glpk*** vont être sélectionnés pour afficher l’image défloutée et débruités, car les n premiers éléments correspondent aux et les n derniers correspondent aux

# Question 4 : le polyèdre

Il faut déterminer si la solution obtenue est un sommet du polyèdre c'est-à-dire si la solution est une solution admissible de base. La solution x est une solution admissible de base s’il y a n contraintes actives et linéairement indépendantes en x. Comme vu en cours, la solution du problème est un sommet du polyèdre. Cette réponse peut être obtenue via l’algorithme ***estSommet*** fonctionnant de la manière suivante :

* Il crée une matrice vide (***Ac***) qui regroupera, les contraintes actives.
* Pour chaque contrainte, il vérifie si elle est active en x et si elle l’est, il la rajoute dans la matrice.
* Il calcule le rang de la matrice qui est le nombre de contraintes linéairement indépendantes. On a alors le nombre de contraintes actives et linéairement indépendantes.
* On a besoin de trouver 2n-m contraintes actives et linéairement indépendantes. Nous cherchons ces contraintes dans les contraintes de positivité et de .
* Si le nombre de contraintes actives et linéairement indépendantes est plus grand ou égal à 2n alors la solution est donc un sommet du polyèdre.

# Question 5 : sensibilité de la solution

Pour cette question, il faut étudier la sensibilité de la solution en fonction de la valeur de pour les images des Example0 et Example1. (Cependant, Example1 n’est pas faisable celui-ci manquant la variable ***xtrue***). Les valeurs des erreurs relatives en fonction de la valeur de sont reprises dans les tableaux ci-dessous (cf : *Tableau 1 - Example0*).  
  
Pour « Example0 », on remarque que la valeur de qui convient le mieux sont les valeurs égales ou plus petites que 0 alors que la valeur qui convient le moins est 1

|  |  |
| --- | --- |
| Valeurs de lambda | Erreur relative de reconstruction |
| 1 | 82.5734 % |
| 0 | 1.23e-12 % |
| 1e-1 | 3.81e-12 % |
| 1e-2 | 9.75e-12 % |
| 1e-3 | 2.24e-12 % |
| 1e-4 | 1.31e-12 % |
| 1e-5 | 1.23e-12 % |
| 1e-6 | 1.23e-12 % |
| 1e-7 | 1.23e-12 % |
| 1e-8 | 1.23e-12 % |
| 1e-9 | 1.23e-12 % |
| 1e-10 | 1.23e-12 % |

Tableau 1 - Example0

Pour « Example1 », je ne peux malheureusement pas le mesurer, celui-ci ne possédant pas de xtrue, il m’est donc impossible de faire un tableau de lambda dessus.

Une image contenant texte

Description générée automatiquement