Université Montpellier
L3 maths 2021-2022

HAX606X: Optimisation Convexe
Joseph Salmon, Tanguy Lefort

Travaux pratiques n° 4 : Descente de gradient projeté

Préambule : Pour ce TP, on pourra utiliser les fichiers suivants[1]. Les questions notées **Pour aller plus loin**, peuvent être omises en première lecture. Le code du TP précédent pourra être réutilisé.

Dans ce TP, nous allons utiliser la méthode de descente de gradient pour trouver un minimiseur d'une fonction $f : \mathbb{R}^d \to \mathbb{R}$ pour $d \geqslant 2$ en posant des contraintes sur f ou sur l'ensemble $\mathcal{C}$ de contrainte du problème.

1 Algorithme de la descente de gradient projeté

Dans cette partie on suppose f, et on va introduire la méthode de descente de gradient projeté (🇬🇧 : *projected gradient method*) pour une telle fonction, avec l'ensemble de contrainte convexe fermé $\mathcal{C}$. On note $\Pi_{\mathcal{C}}$ la projection convexe, *i.e.,* la[2] fonction (convexe) qui envoie un élément dans l'ensemble convexe fermé $\mathcal{C}$.

Algorithme 1 : Algorithme de descente de gradient projeté (à pas fixe)

input : $\nabla f, \Pi_{\mathcal{C}}, x_0, \gamma, n_{\text{iter}}$
param : ϵ
init : $x \leftarrow x_0$
for $i = 1, \ldots, n_{\text{iter}}$ **do**
  $g \leftarrow \nabla f(x)$
  if $\|g\|^2 \leqslant \epsilon^2$ **then** `// Sortie de boucle si le gradient est petit`
    Break
  $x \leftarrow \Pi_{\mathcal{C}}(x - \gamma g)$ `// projection sur l'ensemble des contraintes de la mise à jour`
return x

On définit f_{test} la fonction de test pour la suite de cette partie :

$$f_{test} : \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \mapsto (x_1 - 1)^2 + 3 \cdot (x_2 + 1)^2 \ .$$

Question 1. (Minimum global vs minimum contraint) Quel est le minimum global de f_{test} (utiliser la descente de gradient du TP précédent) ? Utiliser la méthode `minimize` de la librairie `scipy` pour trouver un minimum sur $\mathbb{R}^2_+$. Pour cela, utiliser l'argument `bounds` et une méthode qui supporte cette contrainte, par exemple `method=TNC`, voir la documentation si besoin.

Question 2. (Descente de gradient projeté) Coder la méthode de descente du gradient projeté (à pas fixe) en prenant comme argument une fonction de projection $\Pi_{\mathcal{C}} : \mathbb{R}^d \to \mathbb{R}^d$ quelconque. Instanciez ensuite ces projections aux cas suivants :

1. disponibles ici : http://josephsalmon.eu/HAX606X.html
2. https://fr.wikipedia.org/wiki/Théorème_de_projection_sur_un_convexe_fermé

- Résoudre numériquement :

$$\min_{x\in\mathbb{R}^2} f_{test}(x) \quad \text{s.c.} \begin{cases} x_1 \geqslant 0 \\ x_2 \geqslant 0 \end{cases}$$

  Remarque : cette formulation induit la positivité des paramètres. Elle est notamment utilisée en physique quand il y a une interprétation directe comme la vitesse ou la masse d'un objet.

- Résoudre numériquement :

$$\min_{x\in\mathbb{R}^2} f_{test}(x) \quad \text{s.c. } \|x\|_2^2 \leqslant 1$$

Comparer vos résultats avec la première question.

Question 3. (Autres types de contraintes) **Pour aller plus loin** : En s'aidant de l'exemple dans la documentation de la fonction `minimize` de `scipy`, résoudre le problème suivant :

$$\min_{x\in\mathbb{R}^2} f_{test}(x) \text{ s.c. } \begin{cases} x_1 - 3x_2 \geqslant -2 \\ x_1 + x_2 \geqslant 0 \end{cases} .$$

Ce problème est avec des contraintes d'inégalité linéaire. Pour le résoudre à la main, il serait possible d'utiliser la méthode des multiplicateurs de Lagrange.

2 Problème des moindres carrés linéaires

Dans cette partie, le but est de trouver la fonction linéaire qui approche le mieux un nuage de points (au sens de la norme euclidienne). C'est-à-dire pour un ensemble de points $(x_i, y_i)_{i=1}^n$, nous cherchons à résoudre le problème (convexe) :

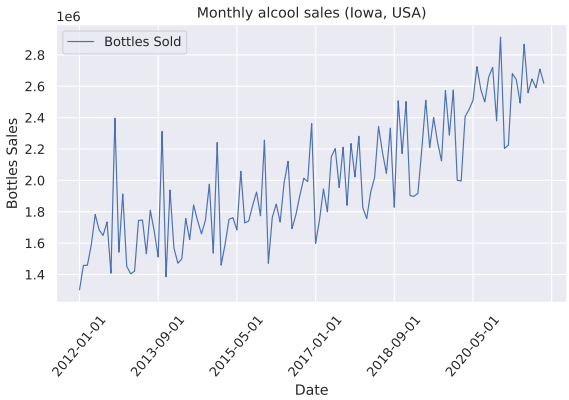
$$\widehat{w} = \begin{pmatrix} \widehat{w}_0 \\ \widehat{w}_1 \end{pmatrix} \in \underset{(w_0,w_1)\in\mathbb{R}^2}{\arg\min} \underbrace{\frac{1}{2n}\sum_{i=1}^{n} \big(y_i - (w_0 + w_1 x_i)\big)^2}_{f(w_0,w_1)} . \tag{1}$$

Question 4. (Moindres carrés linéaires) Pour résoudre le problème des moindres carrés linéaires, utilisons la descente de gradient afin de trouver la solution $\hat{w}$. Pour cela on notera que l'on peut écrire $f(w) = \frac{1}{2n}\|y - Xw\|_2^2$ avec

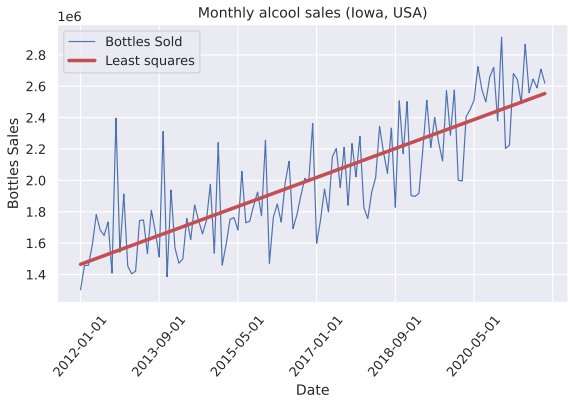
$$y = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \in \mathbb{R}^n, \quad X = \begin{pmatrix} 1 & x_1 \\ \vdots & \vdots \\ 1 & x_n \end{pmatrix} \in \mathbb{R}^{n\times 2} .$$

Il faut donc calculer (mathématiquement) le gradient $\nabla f(w)$ puis implémenter par exemple la descente de gradient pour résoudre le problème. On codera une fonction `gradient` qui prend en argument y, X et w et qui renvoie $\nabla f(w)$ (pour pouvoir généraliser plus tard notre algorithme).

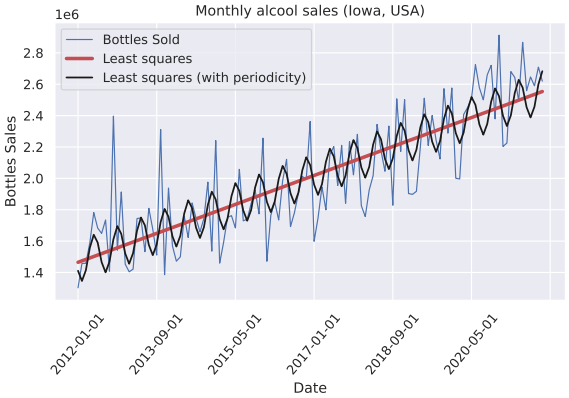
En lançant le scripte `script_season.py` associé au TP, visualisez les données `Iowa_Liquor_tp.csv` en traçant le nuage de points (x_i, y_i) (avec $x_i = i - 1$, pour $i = 1, \ldots, n$).



(a) Données brutes



(b) Données et moindres carrés



(c) Données brutes, moindres carrés et traitement de la saisonnalité

FIGURE 1 – Consommation d'alcool en IOWA (USA) : données et modèles prédictifs.

On observe qu'il y a une périodicité annuelle dans les données. On implémentera alors la même méthode pour en tenir compte, mais cette fois en résolvant le problème :

$$\begin{pmatrix} \hat{w}_0 \\ \hat{w}_1 \\ \hat{w}_2 \\ \hat{w}_3 \end{pmatrix} \in \underset{(a,b)\in\mathbb{R}^2}{\arg\min} \underbrace{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \left(w_0 + w_1 \cdot x_i + w_2 \cdot \cos(\tfrac{2\pi x_i}{6}) + w_3 \cdot \sin(\tfrac{2\pi x_i}{6})\right)\right)^2}_{f(w_0,w_1,w_2,w_3)} .$$

Vérifiez votre solution avec le solveur de moindres carrés proposés dans le script `script_season.py`.

Remarque : La méthode des moindres carrés utilisée dans ce code est celle de la librairie `scikit-learn`. Cette librairie est très utilisée en apprentissage statistique ! Il est possible de récupérer la solution $\hat{w}$ (voir le code ci-dessous), afin de pouvoir comparer les solutions.

```
what = lm.coef_
w0hat = lm.intercept  # ordonnée à l'origine : w_0
print(what, w0hat)
```