Compte rendu de TP de statistiques

Célestn Bigarré

30 mars 2020

L'objectif de ce TP est d'implémenter un algorithme d'optimisation non-régulière, l'algorithme “Forward-Backward” pour calculer différents estimateurs statistiques. Ces estimateurs sont solutions d'un problème de minimisation pénalisé par la norme 1 ce qui rend leur calcul impossible avec les méthodes classiques de descente de gradient, car $||\cdot||_1$ n'est pas différentiable.

1 Piecewise constant denoising

On coonsidère un signal constant par morceaux $\bar{x} \in \mathbb{R}^N$ auquel on rajoute un bruit blanc gaussien standard, $y_k = \bar{x}_k + \mathcal{N}(0,1)$. Les signaux utilisés sont présenté à la Figure 1.

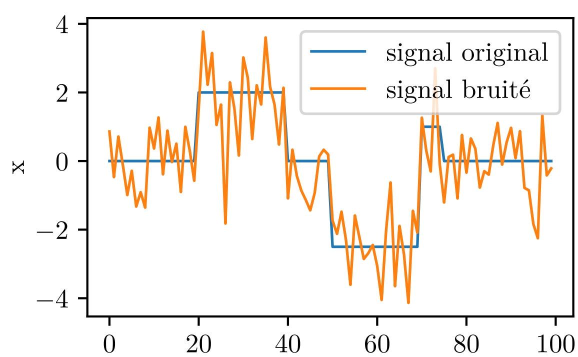


Figure 1 – Le signal $\bar{x}$ et sa version bruitée y

On cherche à calculer un estimateur de $\bar{x}$ à partr du signal bruité y. L'estimaeur choisi est solution du problème de minimisation suivant :

$$\hat{x}_\lambda = \arg\min_{x\in\mathbb{R}^N} \frac{1}{2}\left\|x-y\right\|_2^2 + \lambda\left\|Lx\right\|_1 \tag{$*$}$$

Avec $L : \mathbb{R}^N \to \mathbb{R}^{N-1}$, l'opérateur des différences finies.

1.1 Effets du parametre λ

L'estimateur $\hat{x}$ est parametré par λ qui vient pondérer la norme 1 de Lx dans le problème d'optimisation.

On remarque que pour $\lambda = 0$, l'estimateur est le signal bruité lui même. De plus, pour $\lambda = +\infty$ l'estimateur devient la fonction constante minimisant les moindres carrés. En effet, pour $\lambda = +\infty$,

$$\lambda\left\|Lx\right\|_1 = \begin{cases} 0 & \text{si } x \text{ est constant} \\ +\infty & \text{sinon} \end{cases}$$

En fait, le paramètre λ agit sur le niveau de “sparcité” de $L\hat{x}$, c'est-à-dire sur son nombre de coefficients nuls. Plus λ est grand, plus $L\hat{x}$ aura de coefficient nuls, c'est-à-dire plus x sera constant sur de grand intervalles.

Le paramètre λ joue donc sur le caractère constant par morceaux de $\hat{x}$.

1.2 Problème dual

On a déjà montré dans le DM préparatoire que le problème dual s'écrit,

$$\hat{u}_\lambda \in \arg\min_{u \in \mathbb{R}^{N-1}} \frac{1}{2} \|y - L^*\|_2^2 \qquad \text{soumis à } \|u\|_\infty \leq \lambda \tag{$**$}$$

On rappel de plus que la relation entre les solutions du problème primal et du problème dual est,

$$\hat{x}_\lambda = y - L^* \hat{u}_\lambda$$

1.3 Résolution par l'algorithme "Forward-Backward"

On cherche maintenant à résoudre ce problème avec l'algorithme "Forward-Backward" appliqué à la formulation duale $**$.

En posant :

$$\begin{aligned} g :& \mathbb{R}^N \to \mathbb{R} \\ u &\mapsto \frac{1}{2} \|y - L^* u\|_2^2 \end{aligned}$$

et,

$$\begin{aligned} f :& \mathbb{R}^{N-1} \to \bar{\mathbb{R}} \\ u &\mapsto i_{\left\{v \in \mathbb{R}^{N-1} \,/\, \|v\|_\infty \leq \lambda\right\}} \end{aligned}$$

On a bien que $g \in C^\infty$, en particulier $f \in C^1$ avec ∇g lipschitzienne et f fonction propre. Le problème dual consiste en la minimisation de $f+g$, on est donc bien placé dans le cadre de l'algorithme "Forward-Backward".

Calcul de ∇g

$$\begin{aligned} \nabla_u g &= \nabla_u \left(\frac{1}{2} \langle y - L^* u,\, y - L^* u \rangle \right) \\ &= \frac{1}{2} \nabla_u \left(\langle y,\, y \rangle - 2 \langle y,\, L^* u \rangle + \langle L^* u,\, L^* u \rangle \right) \\ &= -\nabla_u \langle u,\, Ly \rangle + \frac{1}{2} \langle u,\, LL^* u \rangle \\ &= \frac{1}{2} 2LL^* u - Ly \\ &= L \left(L^* u - y \right) \end{aligned}$$

Comme ∇g est une fonction affine, sa constante de lipschitz ν est donnée par :

$$\nu = \|LL^*\|^{-1}$$

Calcul de $P_{\|\cdot\|_\infty \leq \gamma}$

Le calcul de $\operatorname{prox}_{\gamma f} = P_{\|\cdot\|_\infty \leq \lambda}$ est présenté dans le DM, pour rappel on a :

$$\left(\operatorname{prox}_{\gamma f}(u) \right)_k = \begin{cases} \lambda & \text{si } x_k \geq \lambda \\ x_k & \text{si } -\lambda \leq x_k \leq \lambda \\ -\lambda & \text{si } x_k \leq -\lambda \end{cases}$$

Algorithme de "Forward-Backward"

L'algorithme s'écrit,

$$\begin{cases} u_0 \in R^{N-1} \\ v_n = u_n - \gamma \nabla g(u_n) \\ u_{n+1} = u_n + \lambda_n (\operatorname{prox}_{\gamma f} v_n - u_n) \end{cases}$$

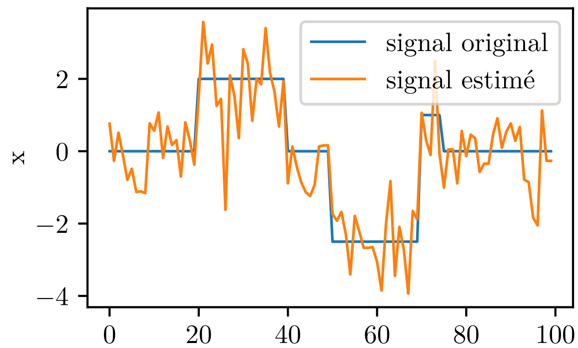
Avec,

$$\gamma \in]0, \frac{2}{\nu}[\qquad \delta = \min\{1, 1/(\nu\gamma)\}$$

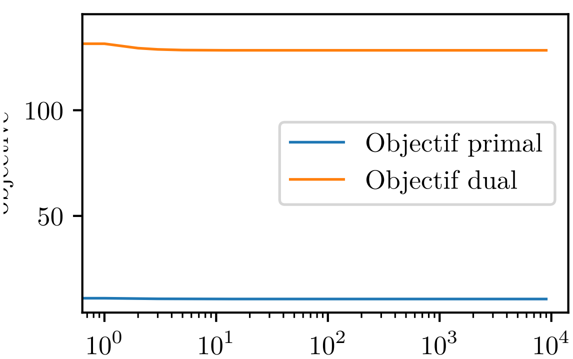
$$(\lambda_n)_{n\in\mathbb{N}} \in [0, \delta[^{\mathbb{N}} \qquad \sum_{n\in\mathbb{N}} \lambda_n(\delta - \lambda_n) = +\infty$$

1.4 Simulations

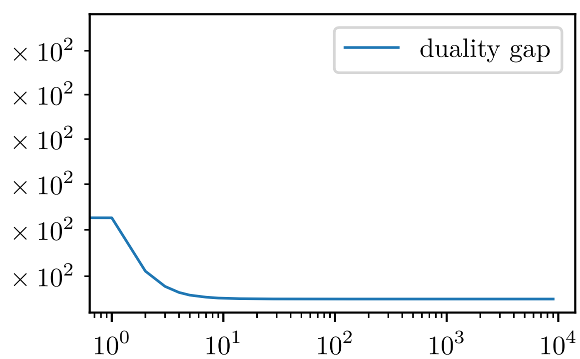
Les figures 2 à 8 présentent les estimateurs pour différentes valeurs de paramètres λ et γ



(a) Estimateur $\hat{x}$ et signal originel x

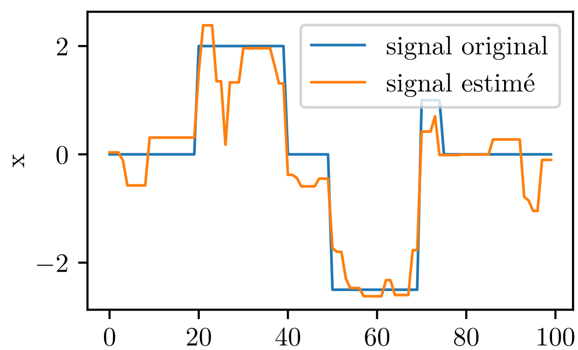


(b) Haut : fonctions objectifs en fonction du nombre d'itérations, Bas : Duality gap

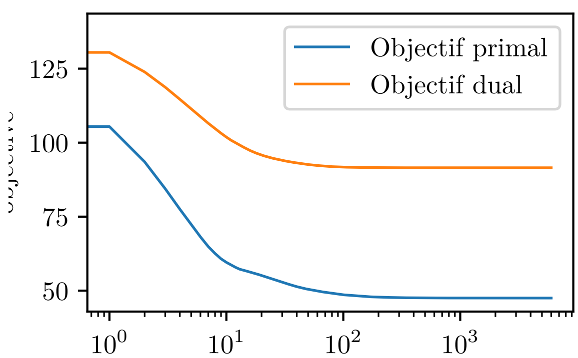


(c) Haut : fonctions objectifs en fonction du nombre d'itérations, Bas : Duality gap

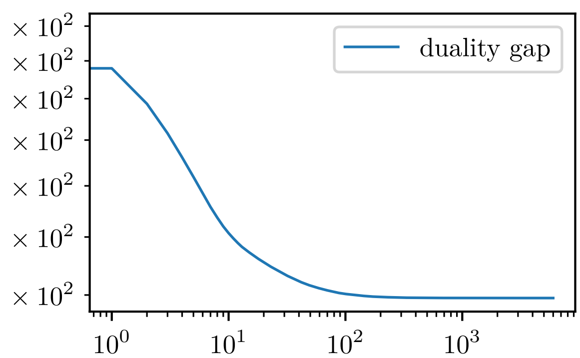
FIGURE 2 – Algorithme "Forward-Backward" pour $\lambda = 0.1$, $gamma = \frac{1}{\nu}$



(a) Estimateur $\hat{x}$ et signal originel x

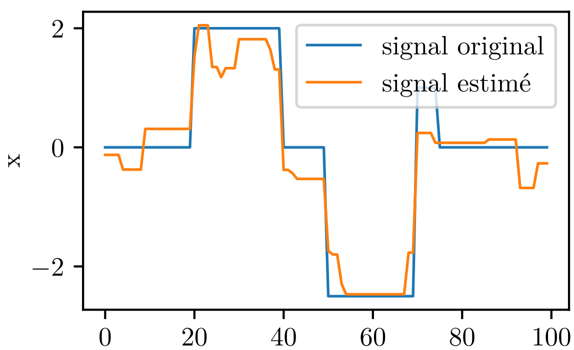


(b) Haut : fonctions objectifs en fonction du nombre d'itérations, Bas : Duality gap

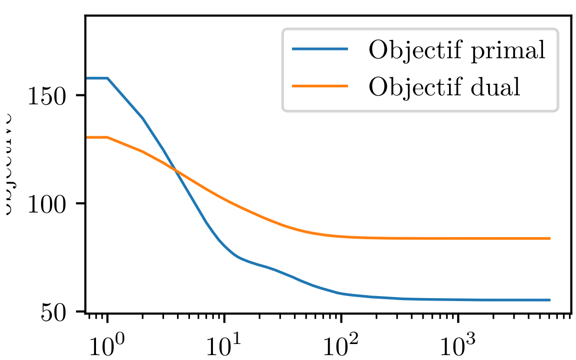


(c) Haut : fonctions objectifs en fonction du nombre d'itérations, Bas : Duality gap

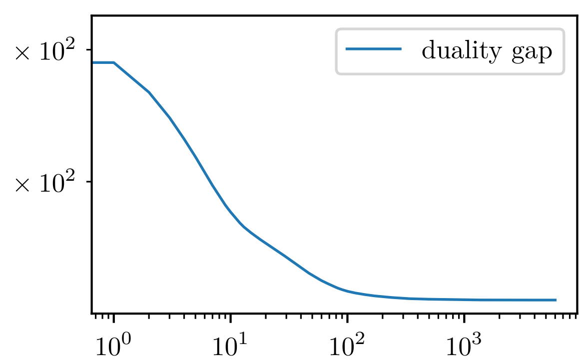
FIGURE 3 – Algorithme "Forward-Backward" pour $\lambda = 1$, $gamma = \frac{1}{\nu}$



(a) Estimateur $\hat{x}$ et signal originel x

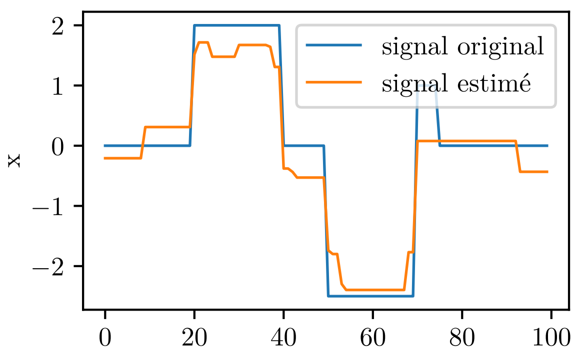


(b) Haut : fonctions objectifs en fonction du nombre d'itérations, Bas : Duality gap

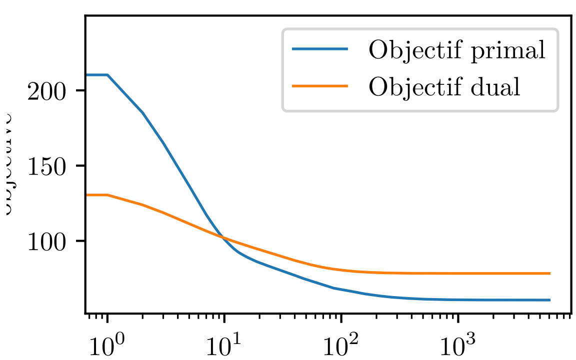


(c) Haut : fonctions objectifs en fonction du nombre d'itérations, Bas : Duality gap

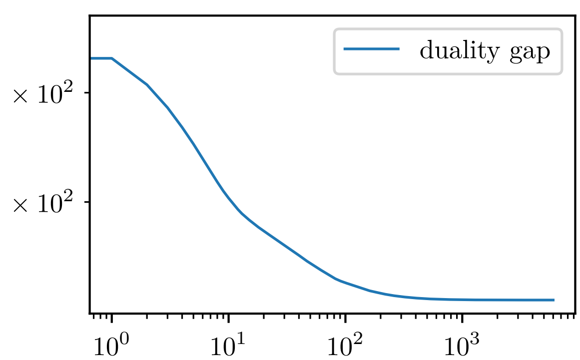
FIGURE 4 – Algorithme “Forward-Backward” pour $\lambda = 1.5$, $gamma = \frac{1}{\nu}$



(a) Estimateur $\hat{x}$ et signal originel x

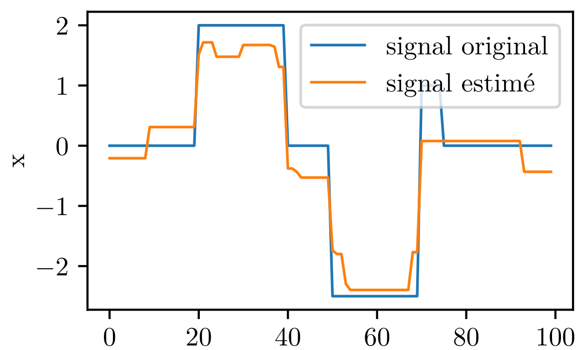


(b) Haut : fonctions objectifs en fonction du nombre d'itérations, Bas : Duality gap

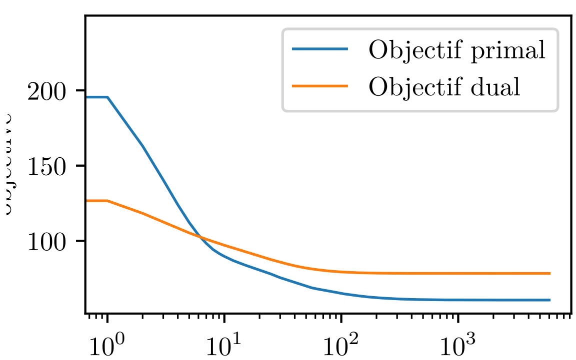


(c) Haut : fonctions objectifs en fonction du nombre d'itérations, Bas : Duality gap

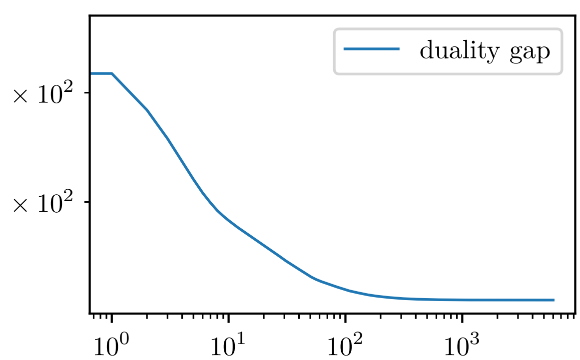
FIGURE 5 – Algorithme “Forward-Backward” pour $\lambda = 2$, $gamma = \frac{1}{\nu}$



(a) Estimateur $\hat{x}$ et signal originel x

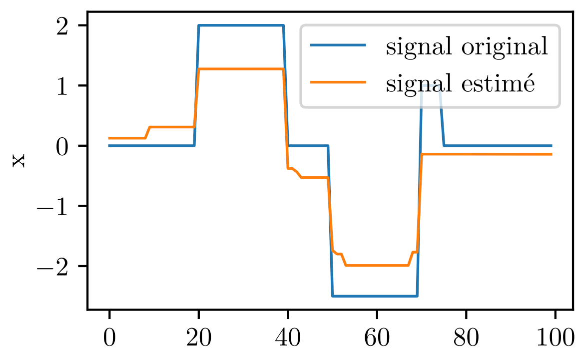


(b) Haut : fonctions objectifs en fonction du nombre d'itérations, Bas : Duality gap

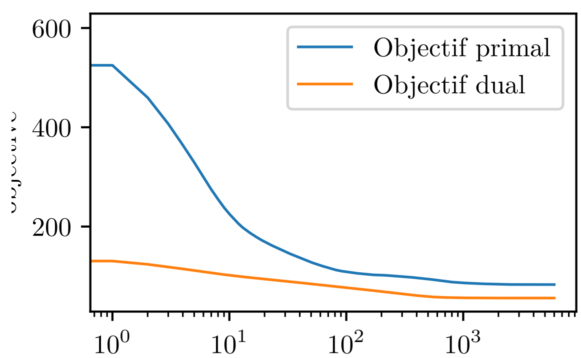


(c) Haut : fonctions objectifs en fonction du nombre d'itérations, Bas : Duality gap

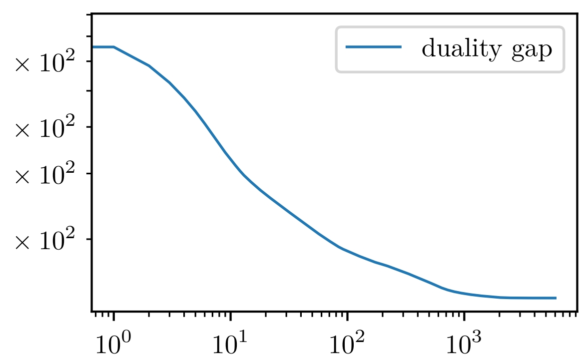
FIGURE 6 – Algorithme "Forward-Backward" pour $\lambda = 2$, $gamma = \frac{1.5}{\nu}$



(a) Estimateur $\hat{x}$ et signal originel x

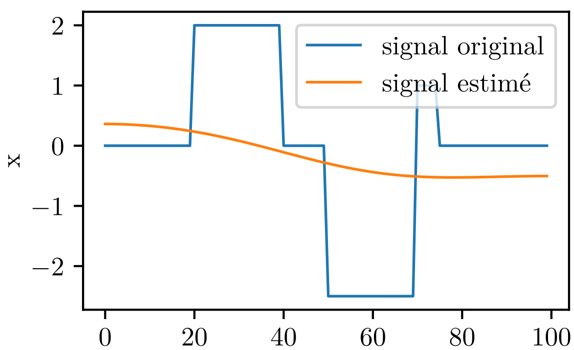


(b) Haut : fonctions objectifs en fonction du nombre d'itérations, Bas : Duality gap

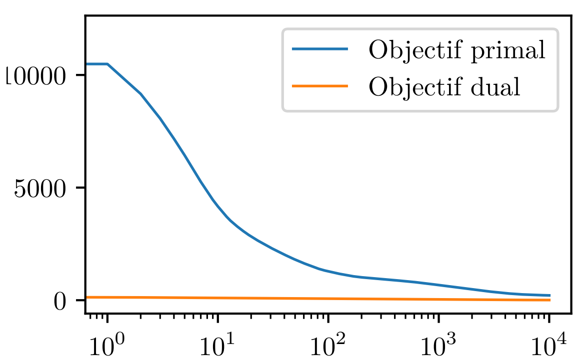


(c) Haut : fonctions objectifs en fonction du nombre d'itérations, Bas : Duality gap

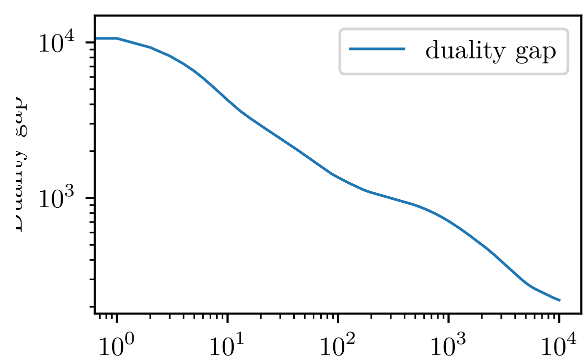
FIGURE 7 – Algorithme "Forward-Backward" pour $\lambda = 5$, $gamma = \frac{1}{\nu}$



(a) Estimateur $\hat{x}$ et signal originel x



(b) Haut : fonctions objectifs en fonction du nombre d'itérations, Bas : Duality gap



(c) Haut : fonctions objectifs en fonction du nombre d'itérations, Bas : Duality gap

FIGURE 8 – Algorithme "Forward-Backward" pour $\lambda = 100$, $gamma = \frac{1}{\nu}$

La variation du paramètre λ joue bien sur la taille des intervalles sur lesquels $\hat{x}$ reste constant. Sur la figure 2 on observe que pour une valeur de λ proche de 0, l'estimateur $\hat{x}$ n'est pas vraiment constant par morceaux. Pour $\lambda = 100$ sur la Figure 8 par contre, l'estimateur est problement constant si l'on prend en compte les erreurs numériques liées au passage de $\hat{u}$ à $\hat{x}$.

Pour des valeurs croissantes de λ comprises entre 1 et 5 (Figures 3, 4, 5 et 7) on remarque bien que le niveau de détail de $\hat{x}$ diminue lorsque λ augmente. Graphiquement, l'approximation de $\bar{x}$ par $\hat{x}$ ne semble pas monotone en fonction de λ.

Le trou de dualité, qui permet de mesurer la convergence de l'algorithme, est strictement décroissante en fonction du nombre d'itérations. Plus λ est faible, plus la convergence de l'algorithme est rapide (pour $\lambda = 100$ on note que la convergence n'est pas atteinte avec les 10000 itérations utilisées, expliquant pourquoi $\hat{x}$ n'est pas constant).

1.4.1 Erreur quadratique moyenne

On présente sur la figure 9 l'évolution de l'érreur quadratique moyenne entre $\hat{x}$ et $\bar{x}$ en fonction de λ. On peut vérifier la conjécture émise au paragraphe précédent, la valeur optimale de λ d'un point de vu de l'érreur quardratique en prédiction semble se situer autour de 2.

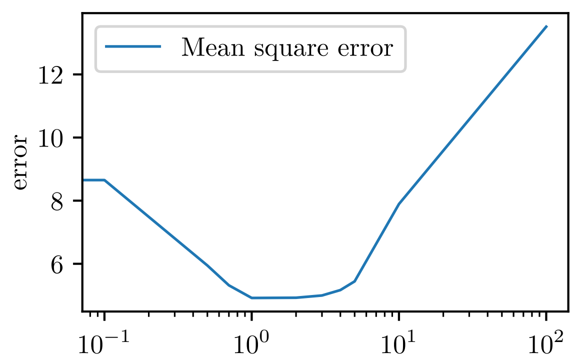


FIGURE 9 – Erreur quadratique moyenne en fonction du paramètre de régularisation (échelle log). $\lambda_n = 1$, $\gamma = \frac{1}{\nu}$, 10000 itérations pour chaque valeur de λ

2 Sparse logistic regression

On regarde maintenant un problème de classification bianire. Sur une base de données de N patients, on décrit l'état de santé (sain ou malade) par le vecteur $b \in \{0,1\}^N$. Pour chaque patient, on dipose de K variables prédictives regroupées dans la matrice $y \in \mathbb{R}^{N\times K}$.

L'éstimateur proposé est :

$$\hat{x}_\lambda = \arg\min_{x\in\mathbb{R}^K} \sum_{i=1}^{N} log\left(1+\exp(-b_i x \cdot y_i)\right) + \lambda \left\|x\right\|_1$$

2.1 Influence de λ

Comme dans le premier exercice, le paramètre de régularisation λ joue sur le nombre de coefficients nuls de l'estimateur. Dans le cas présent, plus λ sera élevé, plus le nombre de variables utilisées pour classer les patient sera petit.

2.2 Bases de données

La base de donnée d'entrainement utilisée contient 100 patients, Avec

— 86 patients sains

— 14 patients malades

La base de données de test contient 257 patients, avec

— 181 patients sains

— 76 patients malades

2.3 Implémentation numérique par l'algorithme "Forward-Backward"

On utilise l'algorithme "Forward-Backward" pour calculer l'estimateur $\hat{x}_\lambda$. En effet, l'estimateur $\hat{x}_\lambda$ peut se réécrire,

$$\hat{x}_\lambda = \arg\min_{x\in\mathbb{R}^k} f(x) + g(x)$$

Avec $f \in C^1$ à gradient lipschitzien et g fonction propre, en posant :

$$\begin{cases} f(x) = \sum_{i=1}^{N} \log\left(1 + \exp(-b_i x \cdot y_i)\right) \\ g(x) = \lambda \left\|x\right\|_1 \end{cases}$$

2.3.1 Calcul de ∇f

On a $f : \mathbb{R}^K \to \mathbb{R}$, donc $\nabla f : \mathbb{R}^K \to \mathbb{R}^K$.

$$\nabla f(x) = (\nabla_k f)_{1\leq k\leq K}(x)$$

et

$$\begin{aligned} \nabla_k f(x) &= \partial_k f(x) \\ &= \sum_{i=1}^{N} \partial_k \log\left(1 + \exp(-b_i x \cdot y_i)\right) \\ &= \sum_{i=1}^{N} \frac{\partial_k \left(1 + \exp(-b_i x \cdot y_i)\right)}{1 + \exp(-b_i x \cdot y_i)} \\ &= \sum_{i=1}^{N} \frac{\exp(-b_i x \cdot y_i)\partial_k \left(-b_i x \cdot y_i\right)}{1 + \exp(-b_i x \cdot y_i)} \\ &= \sum_{i=1}^{N} \frac{-b_i y_{i,k} \exp(-b_i x \cdot y_i)}{1 + \exp(-b_i x \cdot y_i)} \\ &= \sum_{i=1}^{N} -b_i y_{i,k} \frac{1}{1 + \exp(b_i x \cdot y_i)} \end{aligned}$$

2.3.2 Estimation de ν

Comme f est en fait C^2, la consatante de Lipschitz ν de ∇f est majorée par $\sup_{\mathbb{R}^k} \left\|J\nabla f(x)\right\|_2$ où la norme 2 pour les matrice est la norme de Frobenius. On a donc

$$\nu \leq \sup_{\mathbb{R}^k} \left\|J\nabla f(x)\right\|$$

avec,

$$J\nabla f(x) = \left(\frac{\partial \nabla_i f}{\partial x_j}\right)_{1\leq i,j\leq K}$$

et

$$\begin{aligned}
\frac{\partial \nabla_i f}{\partial x_j} &= \sum_{k=1}^{K} -b_k y_{k,i} \partial_j \left(\frac{1}{1+\exp(b_i x \cdot y_i)}\right) \\
&= \sum_{k=1}^{K} -b_k^2 y_{k,i} y_{k,j} \frac{1}{(1+\exp(b_i x \cdot y_i))^2} \\
&\leq \sum_{k=1}^{K} |y_{k,i} y_{k,j}| && \text{car } b_i^2 = 1
\end{aligned}$$

On prend donc $\nu = \left\| \left(\sum_{k=1}^{K} |y_{k,i} y_{k,j}|\right)_{1\leq i,j\leq K} \right\|_2 \geq \sup_{\mathbb{R}^k} \|J\nabla f(x)\|_2$ qui convient et $\gamma = \frac{}{\nu}$. Pour la base de donnée d'entrainement utilisée cela correspond à une valeur de γ de l'ordre de 10^{-7}.

2.3.3 Résultats numériques

Le tableau 1 présente les performences de l'estimateur pour différentes valeurs de λ. On remarque que les performences sont assez stables (et très bonnes !) en fonction de λ. On remarqu que comme attendu, plus λ augmente, plus l'estimateur est creux. La meilleur performence sur la base de test est obtenue pour $\lambda = 2$ avec taux de sparcité de 0.52. Cela semble indiquer que la moitié des variables prédictives fournies sont utiles pour classer 96.8% des patients. Les bonnes performences de l'estimaeur $\hat{x}_1 0$ nous montrent cependant que l'on peut classer correctement 94% des patient en utilisant seulement 2% des variables à notre disposition.

λ	précision entrainement	précision test	sparcité
0.1	1.0	0.9649805447470817	0.02
1	1.0	0.9649805447470817	0.335
2	1.0	0.9688715953307393	0.52
10	0.96	0.9416342412451362	0.915

Table 1 – Résultats des estimateurs calculés par l'algorithme "Foward-Backward" pour différentes valeurs de λ. La précision est calculée comme la proportion de prédiction correctes sur la base testée, la sparcité est définie comme la proportion de coefficients nuls dans l'estimateur $\hat{x}_\lambda$
Pour toutes les simulations, $\gamma \approx 8.710 \times 10^{-7}$ et $\lambda_n = 1.25$. 10000 itérations.

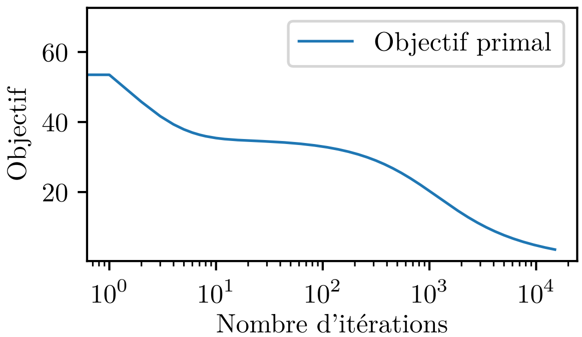
La figure 10 Présente l'évolution de la foonction objectif en fonction du nombre d'itérations. On remarque que les courbes sont très similaires pour toutes les valeurs de λ ce qui est justifié par le fait qu'un tout petit nombre de variables sont utiles pou prédire correctement l'état des patients.

Conclusion

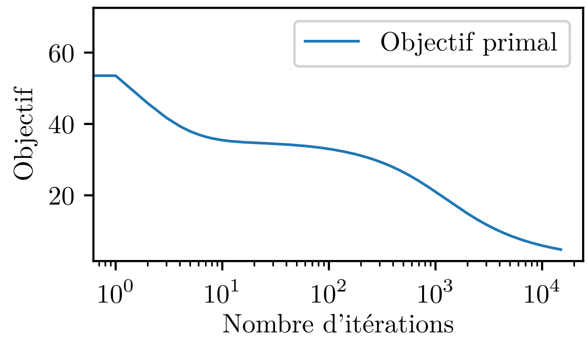
On a exploré dans ce TP deux implémentations de l'algorithme "Forward-Backward" pour calculer des estimateurs statistiques solution de problèmes de minimisations non réguliers.

On a pu montrer l'efficacité de cet algorithme dans les problèmes d'optimisation faisant intervenir la norme L^1 et explorer les impact des différents paramètres de l'algorithme ainsi que les effets du paramètre de régularisation λ.

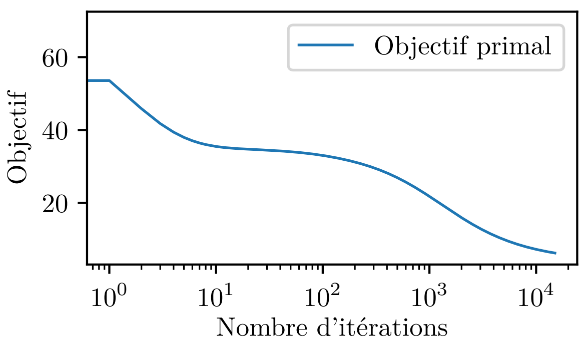
Le code source utilisé pour les calculs est écrit en python avec les bibliothèques *numpy* et *scipy*. Le code source est fournit avec ce rapport ou disponible à l'adresse : `github.com/celbig/TP_high_dim_stats`



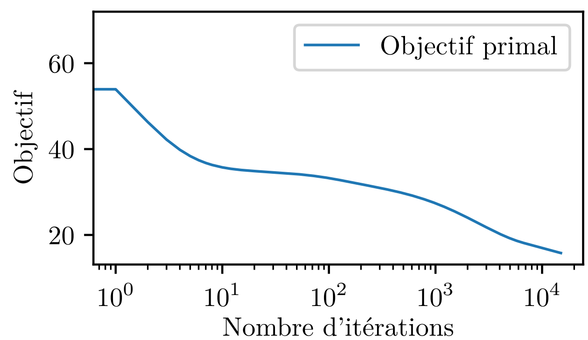
(a) $\lambda = 0.1$



(b) $\lambda = 1$



(c) $\lambda = 2$



(d) $\lambda = 10$

Figure 10 – Évolution de la fonction objectif pour l'algorithme "Forward-Backward" appliqué à la regression logistique sparse. Chaque figure présente la fonction d'objectif du problème de minimisation en fonction du nombre d'itérations.
$\gamma \approx 8.710 \times 10^{-7}$ et $\lambda_n = 1.25$. 10000 itérations.