École des Ponts ParisTech
Mars 2018 - Avril 2018

Cours Optimisation et Contrôle

DistribEauPti - Projet sur les réseaux de distribution d'eau

Marc-Antoine Augé et Matthieu Roux

Table des matières

1 Séance 1 - Définition de l'oracle **2**
1.1 Calculs différentiels 2

2 Résultats théoriques entre les séances **3**
2.1 Équivalence des problèmes 13, 14 et 19 3
2.2 Unicité des solutions de 2 et de 3 4

3 Séances 2 et 3 - Implémentations d'algorithmes **5**

4 Séance 4 - Minimisation du problème contraint **5**

1 Séance 1 - Définition de l'oracle

1.1 Calculs différentiels

On pose

$$F(q) = \frac{1}{3} < q, r \circ q \circ |q| > + < p_r, A_r q >$$

et

$$q(q_c) = q^{(0)} + Bq_c$$

On cherche à calculer $\nabla F(q(q_c))$ et $HF(q(q_c))$ le Hessien.
Remarquons tout d'abord que les matrices sont à coefficients réels donc transposition et adjonction sont deux opérations identiques.
Commencons par $\nabla F(q)$ en écrivant le produit de Hadamard terme à terme et le produit scalaire sous forme de somme :

$$F(q) = \frac{1}{3} \sum_{i=1}^{n} q_i^2 . r_i . |q_i| + < p_r, A_r q >$$

On note alors ϵ_i le signe de q_i :

$$F(q) = \frac{1}{3} \sum_{i=1}^{n} \epsilon_i . r_i . q_i^3 + < p_r, A_r q >$$

D'où immédiatement, étant donné que le gradient du second terme est $A_r^T . p_r$:

$$\nabla F(q) = (\epsilon_i . r_i . q_i^2)_{1 \leq i \leq n} + A_r^T . p_r$$

On peut le réécrire sans la notation ϵ_i et en utilisant un produit de Hadamard :

$$\boxed{\nabla F(q) = (r_i . q_i . |q_i|)_{1 \leq i \leq n} + A_r^T . p_r = r \circ q \circ |q| + A_r^T . p_r}$$

Puis par composition, comme $q(q_c) = q^{(0)} + q_c$, on a, en notant par abus de notation : $F(q_c) = F(q(q_c))$:

$$\boxed{\nabla F(q_c) = B^T \nabla F(q) = B^T (r \circ q \circ |q| + A_r^T . p_r)}$$

Pour calculer le Hessien, on a tout d'abord, en notant $\text{diag}(X)$ la matrice diagonale possédant sur sa diagonale les coefficients de X :

$$HF(q) = 2.\text{diag}(r \circ |q|)$$

Par composition, étant donné que le $H(q(q_c)) = 0$:

$$\boxed{HF(q_c) = HF(q(q_c)) = 2.B^T.\text{diag}(r \circ |q|).B}$$

2 Résultats théoriques entre les séances

2.1 Équivalence des problèmes 13, 14 et 19

On pose les problèmes :

$$\left\{\begin{array}{ll} \text{Min} & \dfrac{1}{3} < q, r \circ q \circ |q| > + < p_r, f_r > \\ \text{s.c.} & q \in \mathbb{R}^n \\ & f_r \in \mathbb{R}^{m_r} \quad Aq - f = 0 \end{array}\right. \tag{1}$$

$$\left\{\begin{array}{ll} \text{Min} & \dfrac{1}{3} < q, r \circ q \circ |q| > + < p_r, A_r q > \\ \text{s.c.} & q \in \mathbb{R}^n \\ & A_d q - f_d = 0 \end{array}\right. \tag{2}$$

$$\left\{\begin{array}{ll} \text{Min} & \dfrac{1}{3} < q^{(0)} + Bq_c, r \circ (q^{(0)} + Bq_c) \circ |q^{(0)} + Bq_c| > + < p_r, A_r(q^{(0)} + Bq_c) > \\ \text{s.c.} & q_c \in \mathbb{R}^{n-m_d} \end{array}\right. \tag{3}$$

Théorème 1 *Les problèmes 1, 2 et 3 sont équivalents.*

Preuve.

Lemme 1 *Les problèmes 1 et 2 sont équivalents.*

Preuve.

On partitionne le graphe en deux : d'un côté les nœuds dont les pressions sont connues (p_r) et d'un autre côté celles non connues (p_d)

La contrainte $Aq - f = 0$ se réécrit avec cette décomposition :

$$A_r q = f_r \text{ et } A_d q = f_d$$

De fait, en injectant $A_r q = f_r$ dans la fonction objectif, les deux problèmes sont équivalents.

Lemme 2 *Les problèmes 2 et 3 sont équivalents.*

Preuve.

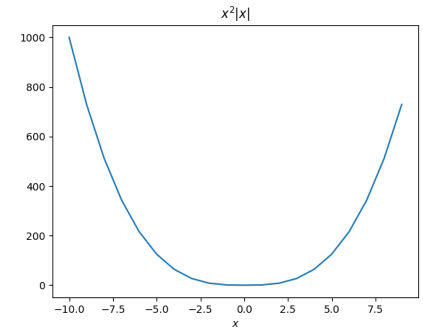


FIGURE 1 – $x^2.|x|$

Prouvons tout d'abord que 2 implique 3 :
On souhaite intégrer la contrainte $A_d q - f_d = 0$ à la fonction objectif. Pour faire cela, on aimerait dire "$q = A_d^{-1} f_d$" mais A_d n'est, a priori ni carrée, ni inversible.
On pose alors $A_d = (A_{d,T}, A_{d,C})$ de sorte à avoir $A_{d,T}$ inversible et de rang maximal ($rgA_d = rgA_{d,T}$).
On applique cette même partition à $q = (q_T, q_C)^T$ et la contrainte du problème 2 se réécrit :

$$A_{d,T} q_T + A_{d,C} q_C - f_d = 0$$

Ou, puis-ce que $A_{d,T}$ est inversible :

$$\boxed{q_T = A_{d,T}^{-1}(f_d - A_{d,C} q_C)}$$

On pose alors $B = (-A_{d,T}^{-1} A_{d,C}, I_{n-m_d})^T$ et $q^{(0)} = (A_{d,T}^{-1} f_d, 0)^T$ de sorte à avoir :

$$q = q^{(0)} + B q_c$$

Ce qui donne le résultat en injectant dans la fonction objectif de 2.

Enfin, 3 implique 2 en réécrivant B et q et en rajoutant des contraintes pour traduire $q_T = A_{d,T}^{-1}(f_d - A_{d,C} q_C)$.

D'après les deux lemmes précédents, les problèmes 1, 2 et 3 sont tous trois équivalents.

2.2 Unicité des solutions de 2 et de 3

Stricte convexité du critère du problème 2 : D'une part, $q \mapsto < p_r, A_r q >$ est strictement convexe car linéaire. D'autre part, on peut réécrire le premier produit scalaire comme suit :

$$< q, r \circ q \circ |q| >= \sum_i q_i.q_i|q_i|$$

Or, chaque terme de cette somme est fortement convexe (*cf.* figure 1) car les résistances sont positives donc le critère est convexe.

Stricte convexité du critère du problème 3 : Pour les mêmes arguments, puis-ce que $q_c \mapsto q^{(0)} + B q_c$ est une fonction convexe et par composition d'une fonction linéaire et d'une fonction convexe : le critère du problème 3 est convexe.

Unicité de la solution du problème 3 : Le critère est strictement convexe, s.c.i, cooercif puis-ce que $< p_r, q^{(0)} + Bq_c >= 0(< q^{(0)} + Bq_c, r \circ q^{(0)} + Bq_c \circ |q^{(0)} + Bq_c| >)$ et donc admet une unique solution, d'après le théorème **5.3** du cours.
On en déduit alors, par équivalence des problèmes 2 et 3 que le problème 2 admet au moins une solution qui est unique par la stricte convexité du critère.

3 Séances 2 et 3 - Implémentations d'algorithmes

Nous avons cherché à résoudre le problème d'optimisation avec différents algorithmes vus en cours :
- Mise en place d'une recherche linéaire vérifiant les conditions de Wolfe avec l'algorithme de Fletcher-Lemaréchal. Cette étape permet de trouver un pas optimal et conduit notamment à l'**algorithme de gradient à pas variable**.
- Mise en place de l'algorithme de **Polak-Ribière**, algorithme de gradient conjugué non-linéaire où la direction étudiée dépend de la direction de l'étape précédente.
- Mise en place de **l'algorithme de BFGS** (Broyden-Fletcher-Goldfarb-Shanno) qui est une méthode de quasi-Newton où on utilise une approximation de l'inverse du Hessien de la fonction à minimiser.
- Mise en place de **l'algorithme de Newton** qui n'approxime pas l'inverse du Hessien

Nous avons obtenu les résultats présentés sur la Figure 2 et synthétisés sur le Tableau 3. Nous remarquons notamment les points suivants :
- L'ajout de la recherche linéaire via l'algorithme de Fletcher-Lemaréchal est une très bonne amélioration qui demande un peu plus de calculs par itération mais en en faisant près de $10\times$ moins demeure deux fois plus rapide qu'un algorithme de gradient à pas fixe.
- L'algorithme de Newton est extrêmement rapide (6 itérations) mais demande une bonne connaissance du problème (le Hessien) : toutefois une approximation de ce Hessien, comme à travers l'algorithme BFGS donne également d'excellents résultats.
- L'initialisation dans l'algorithme de Fletcher-Lemaréchal est importante. Si on initialise avec le α précédent, comme sur la Figure 2c, alors la convergence est très mauvaise.
- On remarque que l'algorithme de Fletcher-Lemaréchal fait osciller le pas de gradient.

4 Séance 4 - Minimisation du problème contraint

Q.1 Le lagrangien du problème **14** s'écrit :

$$\mathcal{L}(\lambda, q) = \frac{1}{3} < q, r \circ q \circ |q| > + < p_r, A_r q > + < \lambda, A_d q - f_d >$$

Les conditions nécessaires d'optimalité sont, étant donné que $\mathcal{L}$ est différentiable :

$$\nabla \mathcal{L}_q = 0 \text{ et } \nabla_\lambda \min_q \mathcal{L}(q, \lambda) = 0$$

La première condition se réécrit :

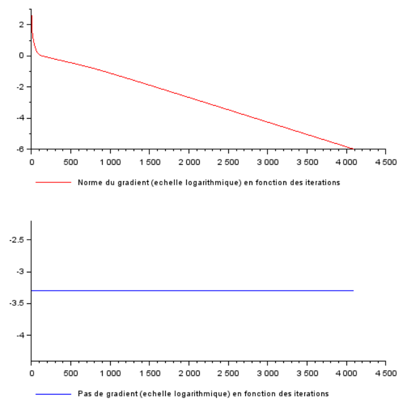
$$0 = \nabla \mathcal{L}_q = r \circ q \circ |q| + A_r^T p_r + A_d^T \lambda$$

Ce qui se réécrit, correspondant à **l'expression 8** :

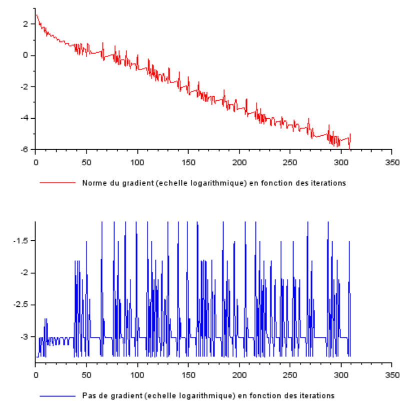
$$0 = r \circ q \circ |q| + A^T p$$

où l'on note :

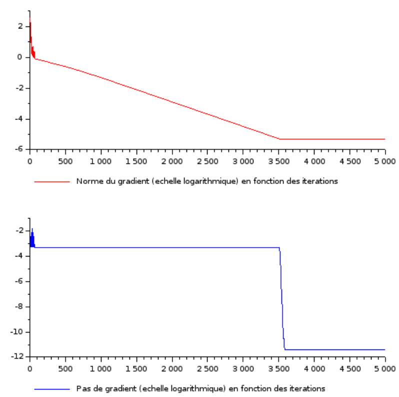
$$A = (A_r, A_d)^T \text{ et } p = (p_r, \lambda)^T$$



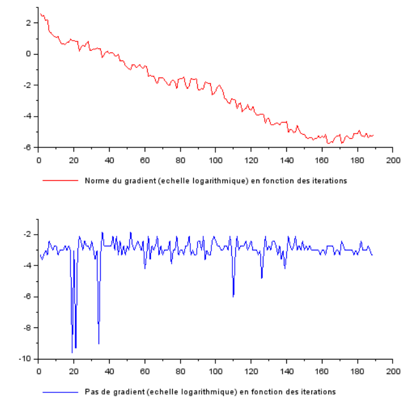
(a) Gradient à pas fixe



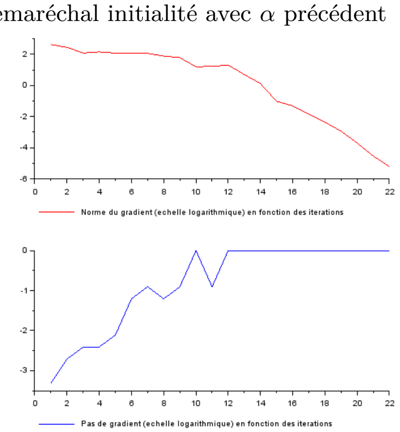
(b) Gradient à pas variable (conditions de Wolfe)



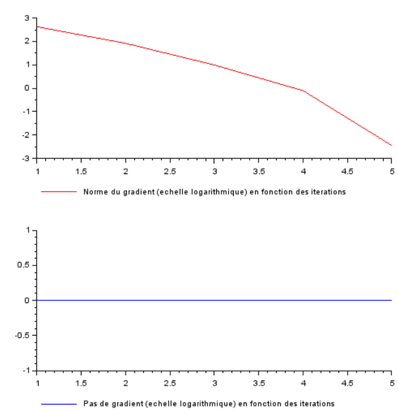
(c) Gradient à pas variable où Fletcher-Lemaréchal initialité avec α précédent



(d) Polak-Ribière



(e) BFGS



(f) Newton

FIGURE 2 – Comparaison des différents algorithmes implémentés

Méthode	Scilab	Pas fixe	Pas variable	Polak-Ribiere	BFGS	Newton
Itérations	-	4094	310	190	23	6
Temps CPU (s)	0	0.34375	0.15625	0.109375	0.03125	0
Critère optimal	-3.734007	-3.734007	-3.734007	-3.734007	-3.734007	-3.734007
Norme du gradient	10^-7	10^-6	9.10^-7	7.10^-7	4.10^-7	$7.85.10^-8$

Figure 3 – Comparaison des différents algorithmes

La deuxième condition d'optimalité s'écrit, où $\hat{q}$ est un minimiseur de $\mathcal{L}$:

$$A_d\hat{q}(\lambda) - f_d = 0$$

Puis, comme $f_r = A_r\hat{q}$, on retrouve **l'expression 6** :

$$A\hat{q} = f$$

Q.2. D'après l'expression vérifiée par $\hat{q}$, on peut l'obtenir de manière explicite. On introduit pour cela le vecteur ϵ des signes :

$$\epsilon_i = -\frac{1}{r_i}(A^T.p)_i$$

Et on obtient :

$$\boxed{q_i = \epsilon_i\sqrt{\left|\frac{1}{r_i}\left(A^T.p\right)_i\right|}}$$

La fonction duale $\Phi(\hat{q}) = \min_q \mathcal{L}(q, \lambda)$ s'écrit alors :

$$\boxed{\Phi = \frac{1}{3} < \hat{q}(\lambda), r \circ \hat{q}(\lambda) \circ |\hat{q}(\lambda)| > + < p_r, A_r\hat{q}(\lambda) > + < \lambda, A_d\hat{q}(\lambda) - f_d >}$$

Le Hessien se calcule comme suit :

$$\Phi(\lambda) = \mathcal{L}(q^\#, \lambda)$$

$$\nabla_\lambda\Phi(\lambda) = A_d q^\# - f_d$$

$$H_\lambda\Phi(\lambda) = \nabla_q(A_d q^\# - f_d)\nabla_\lambda q^\#$$

Or, on peut exprimer la jacobienne de $q^\#$ par rapport à λ en remarquant que :

$$\frac{\partial}{\partial\lambda_j}(r_i \circ q_i^\# \circ |q_i^\#|) = -(A_d^T)_{ij}$$

$$\frac{\partial q_i^\#}{\partial\lambda_j} = \frac{-1}{2r_i \circ |q_i^\#|}(A_d^T)_{ij}$$

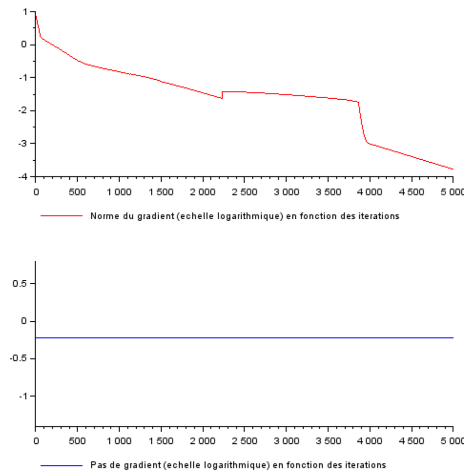
$$\nabla_\lambda q^\# = \frac{-1}{2}diag(\frac{1}{r_i \circ |q_i^\#|})A_d^T$$

Et alors on obtient :

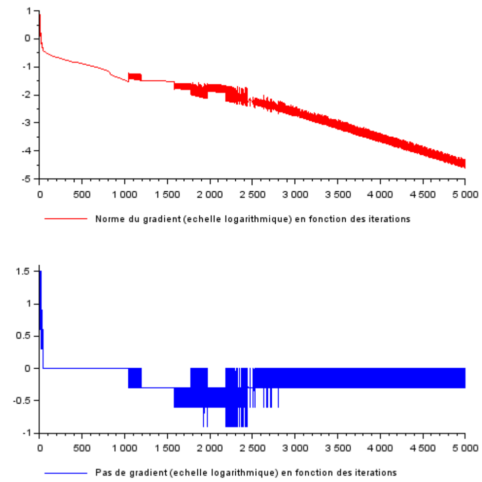
$$\boxed{H_\lambda\Phi(\lambda) = \frac{-1}{2}A_d diag(\frac{1}{r_i \circ |q_i^\#|})A_d^T}$$

Q4. On observe sur la figure 4 les résultats suivants, synthétisés sur le tableau 5

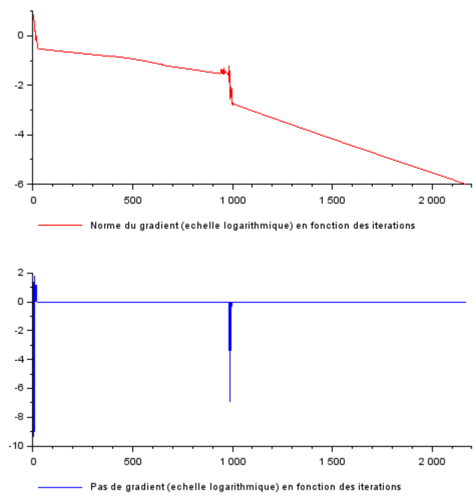
- Le nombre d'itérations pour résoudre le problème est bien plus important pour le problème dual que pour le problème primal. Par exemple, lorsque le primal avec le méthode à pas-variable met 310 itérations pour converger, le primal ne converge pas en 5000 itérations : il y a presque un facteur 10 entre la résolution du primal et celle du dual.
- Les temps CPU sont plus importants pour la résolution du problème dual que pour le problème et cela n'est pas uniquement dû au nombre d'itérations
- Pour la méthode à pas-fixe, il faut recalibrer le pas pour avoir une convergence relativement correcte : on a choisi 0.6.
- Les allures générales des courbes ne sont pas les mêmes pour la résolution du primal et celle du dual.
- La résolution du problème primal de 3 et du dual de 2 conduisent bien au même résultat (au signe - près, puis-ce que nous avons préféré résoudre un problème de maximisation plutôt que de minimisation).



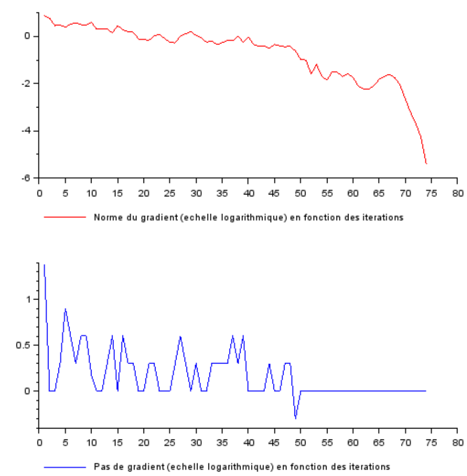
(a) Gradient à pas fixe (pas $\alpha = 0.6$)



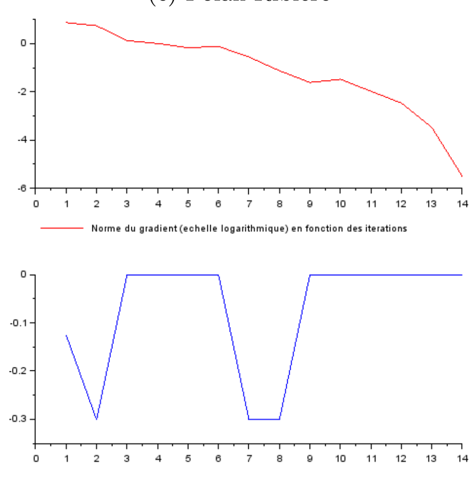
(b) Gradient à pas variable (conditions de Wolfe)



(c) Polak-Ribière



(d) BFGS



(e) Newton

Figure 4 – Comparaison des différents algorithmes implémentés sur le problème dual

Méthode	Scilab	Pas fixe	Pas variable	Polak-Ribiere	BFGS	Newton
Itérations	-	5000	5000	2092	78	17
Temps CPU (s)	0	0.4375	0.92187	0.46875	0.04687	0
Critère optimal	-	3.73401	3.734007	3.734007	3.734007	3.734007
Norme du gradient	$10^{-}7$	$10^{-}4$	$9.10^{-}7$	$7.10^{-}7$	$4.10^{-}7$	$7.85.10^{-}8$

Figure 5 – Comparaison des différents algorithmes pour l'algorithme dual