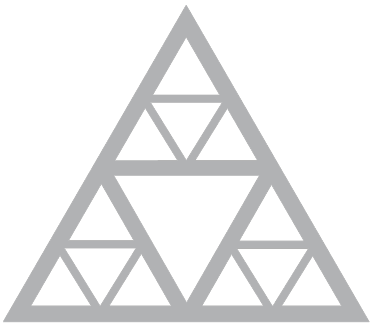

École des Ponts

ParisTech

Rapport sur le projet des réseaux de distribution d'eau

Optimisation et contrôle

Fatine Boujnouni
Mohammed Amine Kheldouni

4 janvier 2019

Table des matières

1 Introduction

Au cours de ce projet, on se propose d'étudier le problème de la résolution des équations caractérisant l'état d'équilibre d'un réseau de distribution d'eau potable. Ce problème est bien évidemment formalisé en un problème d'optimisation sous contraintes d'égalité. Ces contraintes, comme expliqué sur le support du projet, sont linéaires traduisant la première loi de Kirchhoff dans un dans un graphe orienté $\mathcal{G}$.

Rappelons que l'on utilisera les notations fournies dans le support pour la modélisation du problème. Nous allons, dans un premier temps, nous intéressé au problème primal d'optimisation sans contrainte en construisant un oracle pour tester la convergence de notre méthode avec un gradient à pas fixe. Ensuite, nous résoudrons le problème primal par différents algorithmes d'optimisation et testerons ces derniers. Enfin, on reviendra à la formulation de base sous contrainte du problème de l'équilibre du réseau et nous étudierons cette dernière par dualité, toujours avec les mêmes algorithmes d'optimisation.

Le projet sera programmé en SCILAB et utilisera les fichiers de la structure de base du modèle que l'on exécutera grâce au moniteur ($Moniteur\_Skel.sce$).

2 Le problème primal

2.1 Réponses aux questions théoriques

Montrons tout d'abord que les problèmes (13), (14) et (19) sont équivalents. Pour cela nous allons montré qu'une solution du problème 13 est solution du problème 14 qui elle sera solution du problème 19, puis qu'une solution du problème 19 est solution du problème 13.
On considère donc le problème 13. Soit $q^{\#}$ solution du problème. Alors $Aq^{\#} = f$ (contrainte satisfaite par la solution optimale $q^{\#}$). Par la partition de l'ensemble des noeuds du graphes on écrit A et f comme suit :

$$A = \begin{pmatrix} A_r \\ A_d \end{pmatrix} \qquad f = \begin{pmatrix} f_r \\ f_d \end{pmatrix}$$

Donc

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

Enfin, en faisant disparaître f_r de l'expression de la fonction objective à minimiser, on aboutit à la formulation (14).
D'autre part, en effectuant les manipulations décrites sur le support (partition de la matrice A_d puis inversion matricielle pour avoir une expression de q en fonction de q_c) on fait disparaître la dernière contrainte et on retrouve le problème (19).
Enfin, pour $q^{\#}$ solution de (19), on prend $q = q^{(0)} + Bq_c$. Avec $q^{(0)}$ et B définis comme sur l'énoncé du sujet. En remontant dans les calculs précédents, on vérifie aisément que la contrainte du problème (13) est vérifiée ($Aq = f$) et que les problèmes de minimisation sont équivalents.

La fonction $J : q \to \frac{1}{3} < q, r \bullet q \bullet |q| > + < p_r, A_r q >$ est clairement convexe.
En effet :
$\forall (q_1, q_2) \in \mathbb{R}^{n-m_d}, \ \forall \alpha \in [0,1]$,

$$\begin{aligned} J(\alpha q_1 + (1-\alpha)q_2) &= \tfrac{1}{3} < \alpha q_1 + (1-\alpha)q_2, r \bullet (\alpha q_1 + (1-\alpha)q_2) \bullet |\alpha q_1 + (1-\alpha)q_2| > + \\ &\quad < p_r, A_r(\alpha q_1 + (1-\alpha)q_2) > \\ &\leq \tfrac{1}{3} < \alpha q_1 + (1-\alpha)q_2, r \bullet (\alpha q_1 + (1-\alpha)q_2) \bullet (|\alpha q_1| + |(1-\alpha)q_2|) > + \\ &\quad < p_r, A_r(\alpha q_1 + (1-\alpha)q_2) > \quad \text{(par inégalité triangulaire)} \\ &\leq \alpha J(q_1) + (1-\alpha) J(q_2) \\ &\quad \text{(par bilinéarité du produit scalaire et en majorant } \alpha \text{ et } (1-\alpha) \text{ par 1)} \end{aligned}$$

De plus, le cas d'égalité pour $\alpha \in]0,1[$ est retrouvé en remplaçant les inégalités par des égalités. Ainsi dans l'inégalité triangulaire, cela revient à dire que

$$|\alpha q_1 + (1-\alpha)q_2| = |\alpha q_1| + |(1-\alpha)q_2|$$

Donc q_1 et q_2 sont proportionnels, mais comme q_1 et q_2 sont régis par les mêmes vecteurs $q^{(0)}$ et B (A et f étant fixés), on aboutit à $q_1 = q_2$.
Par suite, l'inégalité de convexité de la fonction J est stricte pour $q_1 \neq q_2$ et $\alpha \in]0,1[$.

$$\Rightarrow J \text{ est strictement convexe.}$$

Finalement, les critères des deux problèmes (14) et (19) sont strictement convexes et admettent donc une solution unique les minimisant.

2.2 Implémentation de l'oracle

Dans cette section, nous avons programmé les oracles renvoyant les valeurs du critère et de son gradient pour q_c donné. Pour cela, nous avons effectué le calcul du gradient de la fonction objective.
Avant d'esquisser les grandes lignes de calcul du gradient, notons bien la fonction $\varphi : q_c \to q^{(0)} + Bq_c = q$. On cherche alors pour le problème (19) la minimisation de la fonction $J \circ \varphi$. On remarque que la fonction est linéaire et donc facilement différentiable. Le calcul du gradient pour l'oracle se fait donc comme suit :

$$\nabla(J \circ \varphi)(q_c) \quad = \quad \nabla J(\varphi(q_c))\nabla\varphi(q_c) \qquad \text{(par la règle de la chaîne)}$$

Or

$$\frac{d < q, r \bullet q \bullet |q| >}{dq} = 3 \ \ r \bullet q \bullet |q|$$

$$\Rightarrow \ \nabla(J \circ \varphi)(q_c) \quad = \quad B^T \times (r \bullet q \bullet |q| + A_r^T p_r)$$

Ainsi, on teste sur SCILAB l'oracle (OraclePG) avec la fonction du gradient à pas fixe (GradientF) et on obtient une convergence comme suit :

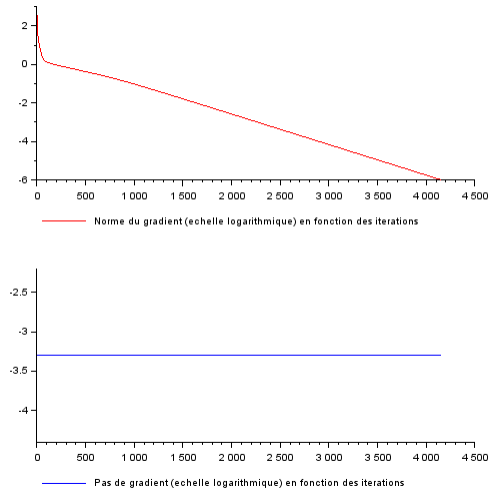

FIGURE 1 – Convergence du programme d'optimisation pour un gradient à pas fixe

La méthode converge donc vers une norme du gradient nulle pour un peu plus de 4000 itérations (cf. courbe rouge) et un pas fixe (courbe bleue constante).
En poussant plus loin les calculs, on trouve l'expression de la Hessienne de la fonction objective en redériviant le Jacobien $G = B^T \times (r \bullet q \bullet |q| + A_r^T p_r)$.

On obtient la matrice Hessienne en dérivant par une règle de la chaîne $q \to r \bullet q \bullet |q|$ et φ :

$$\mathcal{H} = B^T \begin{pmatrix} \ddots & & 0 \\ & 2r \bullet q & \\ 0 & & \ddots \end{pmatrix} B$$

Enfin, on obtient avec un second oracle incluant la Hessienne ($OraclePH$) le même schéma de convergence.

3 Résolution du programme d'optimisation

Dans cette section, on note $q^{(k)}$ le point courant lors de la k-ième itération de la résolution du problème d'optimisation. $G^{(k)}$ la valeur du gradient de $J \circ \varphi$ en le point courant $^{q(k)}$ et $d^{(k)}$ la direction de la descente de gradient au prochain tour d'itération.

3.1 Conditions de Wolfe

La règle de Wolfe consiste en une recherche linéaire qui à pour but de déterminer un pas $\alpha^{(k)}$ vérifiant les conditions de Wolfe. C'est ce pas $\alpha^{(k)}$ qu'on cherche à déterminer pour effectuer la résolution du problème d'optimisation avec diverses méthodes que l'on verra dans la prochaine section.
Rappelons juste les conditions de Wolfe :

— La fonction objective doit décroître de manière significative :

$$J \circ \varphi(q_c^{(k)} + \alpha^{(k)} d^{(k)}) \leq J \circ \varphi(q_c^{(k)}) + \omega_1 \alpha^{(k)} G^{(k)}$$

— Le pas $\alpha^{(k)}$ doit être suffisamment grand :

$$(\nabla J \circ \varphi(q_c^{(k)} + \alpha^{(k)} d^{(k)})^T d^{(k)} \geq \omega_2 (G^{(k)})^T d^{(k)}$$

Nous avons donc implémenté dans le fichier ($Wolfe\_Skel.sci$) la procédure itérative de Flechter & Lemaréchal permettant de déterminer le pas α satisfaisant les deux conditions précédemment énoncées.

3.2 Algorithme de gradient à pas variable

Pour tester notre recherche linéaire du pas α satisfaisant les conditions de Wolfe pour lequel on effectue notre descente de gradient pour minimiser la fonctionnelle de l'énergie du réseau, nous avons testé l'algorithme de Fletcher & Lemaréchal sur un gradient à pas variable. Nous avons obtenu le schéma de convergence suivant :

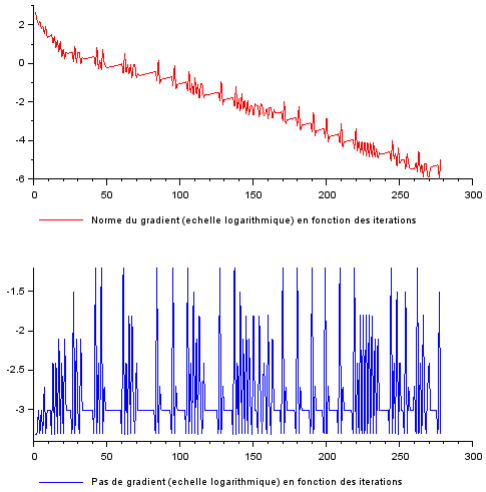

FIGURE 2 – Convergence de la méthode du gradient à pas variable avec une recherche linéaire du pas

En effet, il y a bien convergence car $\|\nabla J \circ \varphi\| \to 0$, le logarithme étant décroissant vers $-\infty$. Pour un pas du gradient variable et qui fluctue assez fortement, on obtient une convergence en à peu prêt 270 itérations. Ce qui est nettement plus performant que les résultats trouvés dans la première section sans recherche linéaire. Cependant on voit que le pas de gradient à tendance à rester constant sur quelques intervalles lorsqu'il vaut -3.

3.3 Algorithme de gradient à pas conjugué

Pour ce second algorithme de descente de gradient, nous avons adopté la formulation de Polak-Ribière en formalisant la direction prise lors de la descente comme suit :

$$d = \begin{cases} -G^{(1)} & \text{si } k = 1 \\ -G^{(k)} + \beta^{(k)} d^{(k-1)} & \text{sinon} \end{cases}$$

Où $\beta^{(k)} = \frac{G^{(k)T}(G^{(k)} - G^{(k-1)})}{\|G^{(k-1)}\|^2}$.

En formalisant ces calculs dans le programme SCILAB (cf *Gradient_C.sci*) on obtient une convergence encore plus performante avec un pas de gradient qui fluctue moins que l'algorithme de descente de gradient à pas variable.

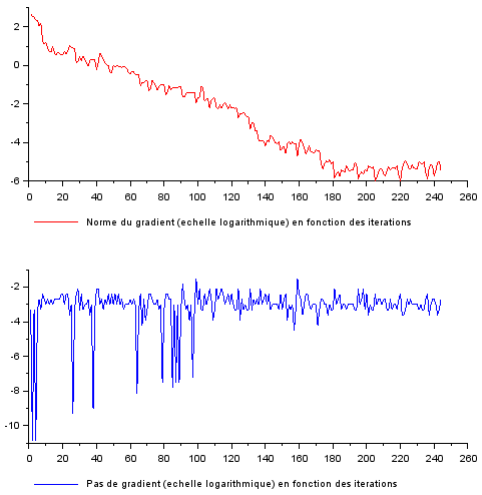

FIGURE 3 – Convergence de la méthode du gradient à pas conjugué

En effet, la courbe rouge (norme du gradient) décroit très rapidement vers une valeur négative logarithmiquement et prouve donc la convergence de la simulation au bout d'environ 200 itérations. D'autre part, la courbe bleue (le pas de gradient) fluctue énormément au début pour se stabiliser en moyenne autour de la même valeur que précédemment ($\alpha = -3$) avec beaucoup moins de sauts que dans l'algorithme précédent.

3.4 Algorithme de quasi-Newton

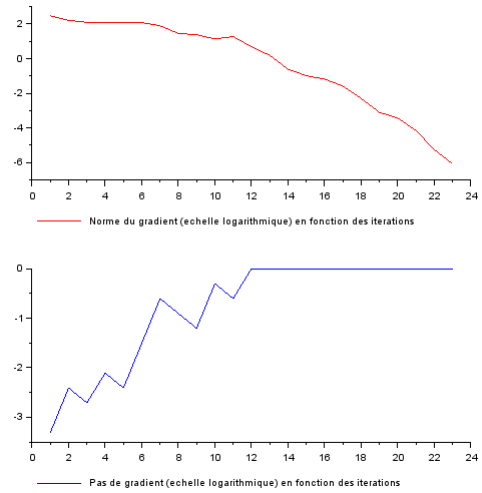

FIGURE 4 – Convergence de a méthode par l'algorithme de quasi-Newton

Pour la méthode BFGS, nous avons une convergence encore plus rapide (en 24 itérations).

On remarque néanmoins que le pas du gradient à changer d'échelle (logarithmiquement) et se stabilise après 12 itérations pour une valeur de $\alpha = 1$ (de logarithme nul).

3.5 Algorithme de Newton

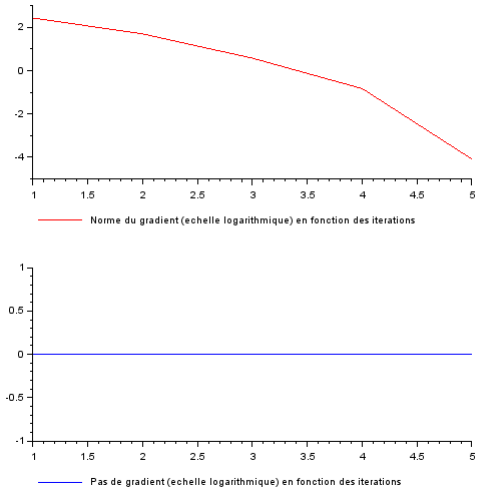

FIGURE 5 – Convergence de a méthode par l'algorithme de Newton pour l'oracle hessien

Pour la méthode de Newton, l'algorithme converge au bout de 5 itérations, et le pas de gradient reste fixe pour cette méthode. Certes, l'algorithme de Newton repose sur le calcul du hessien de la fonction (donc est plus coûteux en terme de calcul par itération), mais permet une convergence très rapide.

3.6 Comparaison des résultats

	Gradient fixe	Pas variable	Gradient conjugué	BFGS	Newton
Iteration	4161	293	209	24	7
Temps CPU	0.36	0.125	0.11	0	0
Critère optimal	-3.73	-3.73	-3.73	-3.73	-3.73
Norme du gradient	10^{-6}	9×10^{-7}	8×10^{-7}	9×10^{-7}	2.5×10^{-13}

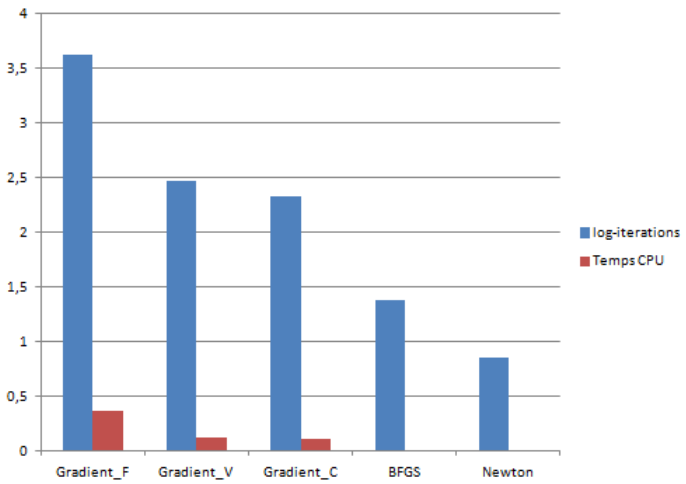

FIGURE 6 – Histogramme les itérations (en log) et du temps CPU pour le problème primal

L'histogramme compare les itérations en échelle logarithmique des différentes méthodes, ainsi que le temps passé par le CPU pour fournir le résultat optimal. On remarque donc que les algorithmes sont de plus en plus performants de la gauche vers la droite. En effet, le gradient à pas fixe effectue un grand nombre d'itération et passe beaucoup de temps à faire le calcul tandis que la méthode de Newton converge très rapidement pour un temps de calcul négligeable.

4 Dualité du problème de l'équilibre du réseau

4.1 Théorie sur le problème dual

Dans cette section, on revient vers la formulation sous contraintes (14) du problème de de l'équilibre du réseau. On va traiter le problème d'optimisation par dualité.
En notant λ le multiplicateur de lagrange associé à la contrainte $(14b)$, le lagrangien du problème s'écrit comme suit :

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

On écrit de suite ses conditions d'optimalité par dérivation :

$$\frac{\partial \mathcal{L}}{\partial q}(q^\#, \lambda) = 0 \qquad \frac{\partial \mathcal{L}}{\partial \lambda}(q^\#, \lambda) = 0$$

Or la dérivation par rapport au multiplicateur satisfait l'unique contrainte du problème, à savoir $A_d q^\# = f_d$ et la dérivation par rapport au débit q fournit :

$$r \bullet q^\# \bullet |q^\#| + A_r^T p_r + A_d^T \lambda = 0$$

Et pour $\lambda = p_d$, en regroupant les termes en A_d et A_r on aboutit à l'équation (8).
Posons maintenant $Z = A_r^T p_r + A_d^T \lambda$
Il vient alors que

$$r \bullet q^{\#} \bullet |q^{\#}| = -Z$$

Il en découle donc que le débit optimal s'écrit sous la forme suivante :

$$q^{\#} = sg(-Z)\sqrt{\frac{|Z|}{r}}$$

Où la division entre $|Z|$ et r est la division d'Hadamard entre deux vecteurs.
Par suite, la fonction duale ϕ s'écrit sous la forme :

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

On en déduit le gradient du dual :

$$\nabla\phi(\lambda) = A_d q^{\#} - f_d$$

En redérivant une nouvelle fois le Lagrangien on retrouve l'expression de la Hessienne comme suit (par la régle de la chaîne) :

$$\nabla^2\phi(\lambda) = -A_d \begin{pmatrix} \ddots & & 0 \\ & \frac{1}{\sqrt{4r \bullet Z}} & \\ 0 & & \ddots \end{pmatrix} A_d^T$$

Pour maximiser la fonction duale ϕ, il suffit donc de considèrer le problème de minimisation de $-\phi$.
Nous avons ensuite implémenté les oracles (cf *Oracle_DG.sci* et *Oracle_DH.sci*) pour effectuer la résolution numérique du problème dual.
Les résultats pour les algorithmes de convergence précédents sont fournis ci-dessous.

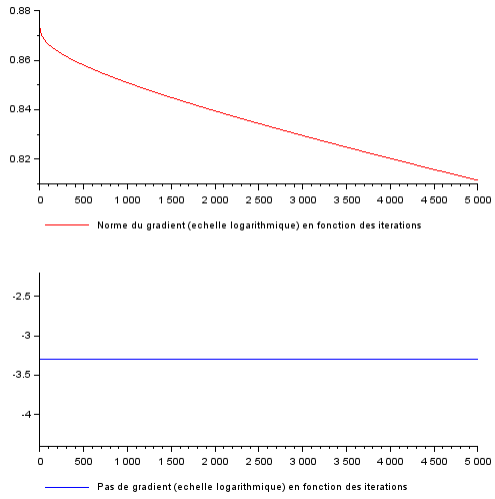

FIGURE 7 – Convergence du gradient à pas fixe pour le problème dual

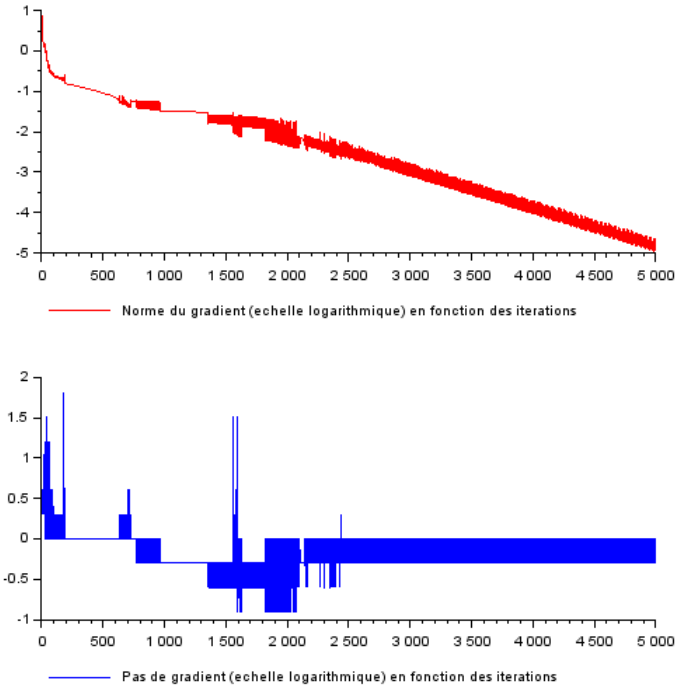

FIGURE 8 – Convergence du gradient à pas variable pour le problème dual

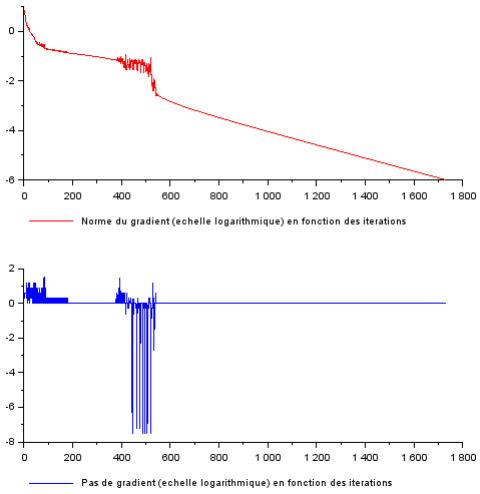

FIGURE 9 – Convergence du gradient conjugué pour le problème dual

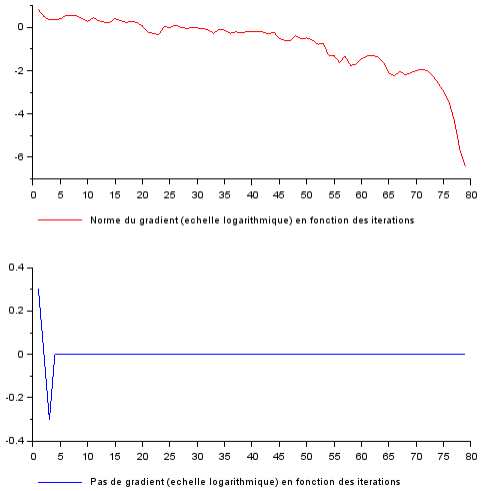

FIGURE 10 – Convergence de la méthode de quasi-Newton pour le problème dual

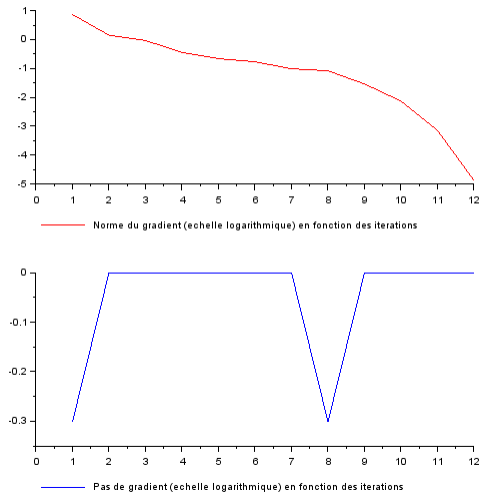

FIGURE 11 – Convergence de la méthode de Newton pour le problème dual

On remarque que les 4 algorithmes convergent globalement de la même manière que pour le problème primal. En effet, la hiérarchie de performance entre les algorithmes se conserve comme on le verra dans le tableau ci-dessous.

4.2 Comparaison des résultats

	Pas variable	Gradient conjugué	BFGS	Newton
Iteration	5000	1223	80	12
Temps CPU	1.125	0.28	0.015	0
Critère optimal	3.73	3.73	3.73	3.73
Norme du gradient	2×10^{-5}	10^{-6}	3×10^{-7}	7×10^{-7}

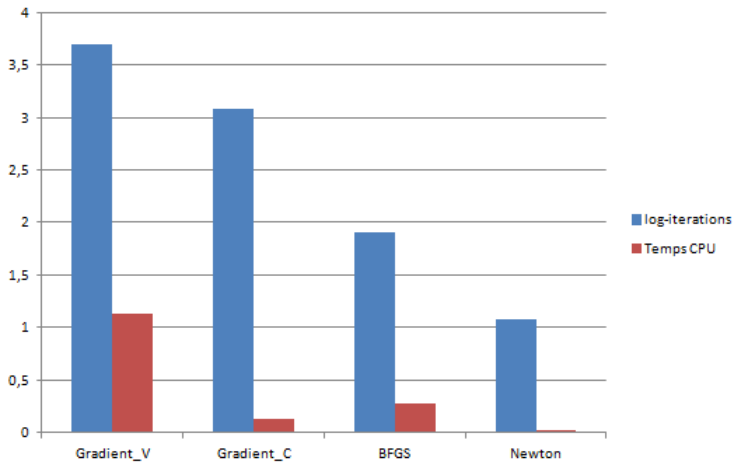

FIGURE 12 – Histogramme les itérations (en log) et du temps CPU pour le problème dual

Les résultats énoncés lors du commentaire du premier histogramme sur le problème primal se concervent. En effet, la méthode de Newton reste la plus efficace avec très peu d'itérations avant d'atteindre le point optimal. Néanmoins, il a fallu implémenter le calcul de la hessienne pour cette dernière méthode.

5 Comparaison des problèmes primal et dual

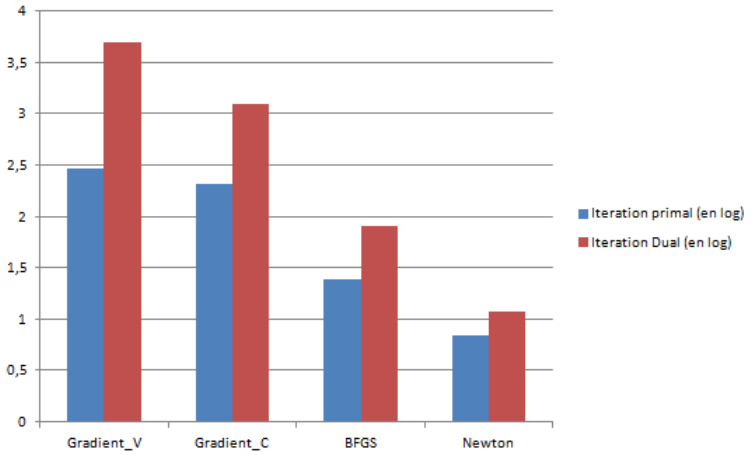

FIGURE 13 – Comparaison des convergences (itérations) des algorithmes entre problème primal et dual

Comme nous l'avons précisé tout à l'heure, l'ordre de grandeur en échelle logarithmique des itérations des algorithmes est le même pour les deux problèmes. Cependant, on remarque que le problème dual nécessite plus d'itérations que le problème primal pour converger.

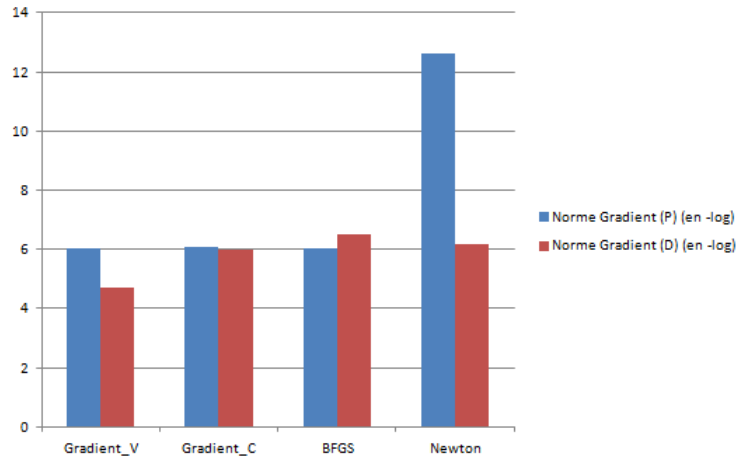

FIGURE 14 – Comparaison de la norme du gradient finale des algorithmes entre problème primal et dual

Sur cet histogramme nous avons tracé l'opposé de le logarithme de la norme du gradient pour confronter la précision de la convergence des deux problèmes. On retrouve une très bonne précision de convergence chez l'algorithme de Newton et un ordre de grandeur assez semblable pour les autres algorithmes.

6 Conclusion

En somme, nous avons réussi à modéliser le problème de distribution d'eaux en un problème d'optimisation mathématique minimisant la fonctionnelle de l'énergie du réseau. Cette modélisation nous a permis de prendre en compte les contraintes physiques à respecter, telles que la loi de Kirchhoff et la loi d'Ohm pour les réseaux.

Pour résoudre ce problème d'optimisation, nous avons implémenté en SCILAB différentes méthodes de convergence se basant sur des calculs de gradient et de hessien. Le résultat obtenu pour chacun des algorithmes confirme l'efficacité de ces derniers. Néanmoins, certains algorithmes tel que l'algorithme de Newton permettent une vitesse de convergence plus forte que d'autres.

Enfin, nous avons appris de ce projet l'importance de la modélisation mathématique des problèmes et l'envergure du domaine de l'optimisation dans les sciences de la décision. En effet, nous avons retrouvé un résultat riche en application partant d'un problème très concret ralliant entre pratique et théorie mathématique puis simulation numérique.