REVUE FRANÇAISE D'AUTOMATIQUE, D'INFORMATIQUE ET DE RECHERCHE OPÉRATIONNELLE. RECHERCHE OPÉRATIONNELLE

J. ABADIE

Un nouvel algorithme pour la programmation non linéaire

Revue française d'automatique, d'informatique et de recherche opérationnelle. Recherche opérationnelle, tome 12, n° 2 (1978), p. 233-238.

<http://www.numdam.org/item?id=RO_1978__12_2_233_0>

R.A.I.R.O. Recherche opérationnelle/Operations Research
(vol. 12, n° 2, mai 1978, p. 233 à 238).

UN NOUVEL ALGORITHME POUR LA PROGRAMMATION NON LINÉAIRE (*)

par J. ABADIE (1)

Résumé. — *L'article expose un nouvel algorithme pour la résolution des problèmes de programmation non-linéaire. Des expériences numériques préliminaires y sont présentées, qui montrent cette méthode pleine de promesses. Une généralisation termine l'exposé pour donner une nouvelle classe d'algorithmes.*

Notre objet est d'exposer succinctement une nouvelle méthode de résolution des problèmes de programmation non linéaire, que nous considérons sous la forme canonique

$$\min \varphi(X), \qquad f(X) = 0, \qquad a \leqq X \leqq b, \tag{1}$$

où X, a, b sont des vecteurs-colonne de $\mathbf{R}^n$; où $a \leqq X \leqq b$ signifie $a_j \leqq X_j \leqq b_j$, $\forall j \in \mathscr{J} = \{1, \ldots, n\}$; où $f(X) = 0$ signifie $f_i(X) = 0, \forall i \in \mathscr{I} = \{1, \ldots, m\}$, les fonctions φ, f_i étant deux fois continûment différentiables. Nous supposons tout d'abord $a_j = -\infty$, $b_j = +\infty$, $\forall j \in \mathscr{J}$.

$f(X)$ est considéré comme un vecteur-colonne de $\mathbf{R}^m$. Nous désignons les gradients de φ, f_i en X par $\varphi'(X)$, $f_i'(X)$, qui sont des vecteurs-ligne de $\mathbf{R}_n$. La notation $f'(X)$ désigne le jacobien en X des fonctions $f_i(X)$, $i \in \mathscr{I}$; l'élément en ligne i et colonne j est $\partial f_i / \partial X_j$. Nous supposerons toujours que le rang du jacobien est égal au nombre m de ses lignes. On peut alors en extraire une sous-matrice inversible d'ordre m composée de m colonnes $j \in \mathscr{B}$. Nous appelons $\mathscr{N}$ le complémentaire de $\mathscr{B}$ par rapport à $\mathscr{J}$. Le vecteur X se décompose de façon naturelle en $x = (X_j)_{j \in \mathscr{N}}$ et $y = (X_j)$, $j \in \mathscr{B}$, qui sont des vecteurs-colonne d'espaces $\mathbf{R}^{n-m}$ et $\mathbf{R}^m$. Le jacobien est partitionné en $\partial f / \partial x$ et $\partial f / \partial y$.

Donnons à X un accroissement z, qui se décompose lui-aussi en h, k, et posons

$$X^1 = X + z \qquad \text{c'est-à-dire} \quad x^1 = x + h, \quad y^1 = y + k. \tag{2}$$

Lions h, k par la relation

$$f(X) + \frac{\partial f}{\partial x} h + \frac{\partial f}{\partial y} k = 0, \tag{3}$$

qui définit k en fonction de h, et considérons l'expression lagrangienne

$$L(h) = \varphi(X+z) + u f(X+z), \tag{4}$$

où u est un vecteur-ligne de $\mathbf{R}_m$, et où $u f(X+z)$ est donc le produit scalaire de u et $f(X+z)$. Tenant compte de (3), le gradient $L'(0)$ (vecteur-ligne de

(*) Reçu novembre 1977.

(1) Électricité de France et Université Paris-IX — Dauphine.

$\mathbf{R}_{n-m}$) satisfait à

$$L'(0)\,h = \left[\frac{\partial \varphi}{\partial x} - \frac{\partial \varphi}{\partial y}\left(\frac{\partial f}{\partial y}\right)^{-1}\frac{\partial f}{\partial x}\right]h - \left[u + \frac{\partial \varphi}{\partial y}\left(\frac{\partial f}{\partial y}\right)^{-1}\right]f(x).$$

Simplifions cette expression en déterminant u par

$$\frac{\partial \varphi}{\partial y} + u\frac{\partial f}{\partial y} = 0 \tag{5}$$

et en posant

$$\frac{\partial \varphi}{\partial x} + u\frac{\partial f}{\partial x} = g. \tag{6}$$

Le gradient $L'(0)$ s'écrit alors :

$$L'(0) = g. \tag{7}$$

Les conditions du 1er ordre de Lagrange s'écrivent $f(X) = 0$, $g = 0$: nous supposons qu'elles ne sont pas réalisées.

Si l'on avait la matrice des dérivées secondes $L''(0)$, supposée inversible, la méthode de Newton pour minimiser $L(h)$ donnerait

$$h = -L''(0)^{-1}\,g^T.$$

Nous employons en fait une méthode quasi newtonienne, qui donne à chaque itération une approximation H de $L''(0)^{-1}$, déduite de l'approximation de H à l'itération précédente et des accroissements Δx, Δg de x, g entre ces deux itérations. h se calcule alors par

$$h = -H\,g^T. \tag{8}$$

La formule choisie pour modifier H est celle de Broyden-Fletcher-Shanno (BFS), que le lecteur pourra trouver, par exemple, dans l'excellent livre d'Avriel ([3], p. 333).

Nous arrivons ainsi, à un premier stade de l'algorithme dont l'itération courante est, en désignant par X son point initial, et par ε_f, ε_g deux constantes positives :

pas 0 : *on connaît un point initial* X;

pas 1 : *calculer* $\varphi'(X), f'(X)$;

pas 2 : *extraire de* $f'(X)$ *la matrice inversible* $\partial f/\partial y$, *ce qui donne la base* $\mathscr{B}$;

pas 3 : *calculer* u *et* g *par* (5) *et* (6); *si* $\|f\| \leqq \varepsilon_f$ *et* $\|g\| \leqq \varepsilon_g$, FIN;

pas 4 : *si la base* $\mathscr{B}$ *est identique à la base de l'itération précédente, modifier* H *par la formule* BFS; *sinon, poser* $H = I$;

pas 5 : *calculer* h *par* (8), *puis* k *par* (3);

pas 6 : X^1, *calculé par* (2), *est le point initial de l'itération suivante :*

Ce premier algorithme se réduit à une méthode classique à métrique variable lorsque $m = 0$, et à une méthode quasi newtonienne pour résoudre un système d'équations non linéaires de n équations à n inconnues lorsque $m = n$: cette dernière réduction montre qu'il ne peut y avoir convergence que si le point de départ est assez proche de la solution souhaitée. Nous modifions donc cet algorithme comme suit. Les pas 1 à 5 ne changent pas. Posons

$$X^\theta = X + \theta z \tag{9}$$

et introduisons la fonction $l(\theta)$ déduite de $L(h)$:

$$l(\theta) = \varphi(X^\theta) + uf(X^\theta),$$

dont nous connaissons la dérivée $l'(0) = gh < 0$:

pas 6.1 : *poser* $\theta_1 = 1$; *si* $l(\theta_1) < l(0)$, *poser* $\theta^* = \theta_1$, *et aller en* 6.3;

pas 6.2 : *calculer, par une suite d'interpolations quadratiques,* θ^* *tel que* $0 < \theta^* < \theta_1$, $l(\theta^*) < l(0)$;

pas 6.3 : X^{θ^*} *est le point initial de l'itération suivante :*

En fait, pour l'instant, ce n'est pas ce que nous avons programmé dans notre code. Nous avons plutôt utilisé une idée récente de M. J. D. Powell, incluse dans une autre méthode [8] : cette idée consiste à introduire la fonction de pénalité

$$P(\theta) = \varphi(X^\theta) + \sum_{i \in \mathcal{I}} p_i \left| f_i(X^\theta) \right|, \tag{10}$$

où les p_i sont des scalaires positifs. C'est sur cette fonction $P(\theta)$, et non $l(\theta)$, que nous appliquons les pas 6.1 à 6.3, en prenant $p_i = |u_i|$ si $u_i \neq 0$, une autre valeur positive si $u_i = 0$. On s'assure facilement que la dérivée pour $\theta = 0$, dans notre méthode, existe :

$$P'(0) = gh + \sum_{i \in \mathcal{I}} (u_i - \varepsilon_i p_i) f_i(X) < 0, \tag{11}$$

$$\varepsilon_i = +1 \quad \text{si} \quad f_i(X) \geqq 0, \qquad \varepsilon_i = -1 \quad \text{si} \quad f_i(X) < 0, \qquad \forall i \in \mathcal{I}. \tag{12}$$

Remarquons par ailleurs que le « pas 2 » doit être soumis à deux impératifs, contradictoires parfois : assurer une « bonne inversion » de $\partial f/\partial y$, et ne pas changer trop souvent de base (voir pas 4). On pourra opérer par une suite d'éliminations gaussiennes en cherchant, pour le premier impératif, le pivot de plus grand module π dans chaque ligne, et pour le second en acceptant le pivot ayant le même indice de colonne qu'à l'itération précédente si son module n'excède pas $\alpha\pi$ (nous prenons $\alpha = 1/2$).

Passons maintenant au cas où, dans (1), certains b_j ou a_j ne sont pas $\pm \infty$. Les modifications à apporter à l'algorithme concerne les points suivants :

pas 2 : la base $\mathscr{B}$ ne doit, si cela est possible, contenir aucun indice j tel que $X_j = a_j$ ou $X_j = b_j$; si cela n'est pas possible, effectuer des pivotages jusqu'à ce que $k_j \geqq 0$ si $X_j = a_j$, et $k_j \leqq 0$ si $X_j = b_j$: cela peut être assuré par une méthode de programmation linéaire.

pas 3 : définissons la *face* $\mathscr{F}$ par

$$\mathscr{F} \subset \mathscr{N} \qquad et \qquad \mathscr{F} = \{j \,|\, (X_j = a_j \text{ et } g_j \geqq 0) \text{ ou } (X_j = b_j \text{ et } g_j \leqq 0\};$$

on remplace g par sa projection en imposant $g_j = 0$, $\forall j \in \mathscr{F}$, sans modifier les autres composantes de g.

pas 4 : on modifie la matrice H pour qu'elle approxime la restriction de $L''(0)^{-1}$ au sous-espace $\mathscr{N} - \mathscr{F}$.

pas 6 : au lieu de considérer d'emblée $\theta = 1$, on limite θ par la condition $a \leqq X^\theta \leqq b$, qui donne $\theta \leqslant \theta_1$; c'est cette valeur de θ_1 que l'on prend en considération, si $\theta_1 < 1$, au

pas 6.1 : avec la modification suivante : si θ_1 est trop petit, ce que l'on teste par $l(\theta_1) < l(0)$ et $[l(\theta_1) - l(0)]/[\theta_1 l'(0)] > 1 - \varepsilon$ (nous prenons $\varepsilon = 1/10$), on procède à des doublements, voire à des triplements de θ, suivis de projections de X^θ sur le parallélotope $a \leqq X \leqq b$, tant que la fonction $l(\theta)$ diminue et que le test précédent est satisfait [rappelons qu'en fait c'est $P(\theta)$ que nous utilisons, jusqu'à présent, ou lieu de $l(\theta)$].

EXPÉRIENCES NUMÉRIQUES

Des expériences préliminaires ont été menées en utilisant 4 problèmes-test : les problèmes 2 et 3 de Colville, et les problèmes 1 et 3 d'Himmelblau (ce sont, dans chaque série, les deux premiers ayant des contraintes non linéaires; les problèmes 2 et 3 de Colville figurent aussi sous les numéros 18 et 11 dans la série Himmelblau; le lecteur trouvera les énoncés dans l'excellent livre [5] de cet auteur). A titre de comparaison, nous avons utilisé une variante de la méthode GRG [1], [2], qui passe pour la plus rapide des méthodes existantes (*voir* les expériences numériques comparatives de Colville [4], Staha [7], Sandgren [6], où la méthode GRG se classe chaque fois première). La variante utilisée est bien plus rapide et précise que celles qui ont figuré dans les comparaisons de ces auteurs. La machine est une IBM 370/168 de la Direction des Études et Recherches d'Électricité de France, avec OPT = 2.

En ce qui concerne les deux problèmes Colville, deux points de départ différents sont proposés par cet auteur, l'un faisable et l'autre non. Ils sont

désignés dans la table ci-après par *f* et *nf*. Les points de départ donnés par Himmelblau pour ses problèmes 1 et 3 sont non-faisables.

Problèmes	Temps (en secondes)	
	GRG	nouvelle méthode
Colville 2*f*.....	0,343	0,110
Colville 2 *nf*.....	0,260	0,136
Colville 3*f*.....	0,083	0,017
Colville 3 *nf*.....	0,070	0,016
Himmelblau 1.....	0,045	0,014
Himmelblau 3.....	0,112	0,007

La précision obtenue est supérieure pour la nouvelle méthode (contraintes avec une erreur relative inférieure à 10^{-12}, environ 8 chiffres exacts pour les valeurs des x_j).

En conclusion, la nouvelle méthode semble prometteuse, et mérite d'être améliorée et approfondie.

GÉNÉRALISATION

Posons $A = f'(X)$, et désignons par A^+ un inverse à droite de A, c'est-à-dire une matrice (n, m) vérifiant $AA^+ = I$. Voici deux exemples :

Exemple 1 : $A^+ = \begin{pmatrix} O_{(n-m)\times m} \\ (\partial f/\partial y)^{-1} \end{pmatrix}$;

Exemple 2 : $A^+ = A^T (AA^T)^{-1}$.

La généralisation proposée consiste à estimer les vecteurs-ligne $u \in \mathbf{R}_m$, $\gamma \in \mathbf{R}_n$ par

$$u = -\varphi'(X) A^+, \gamma = \varphi'(X) + uf'(X).$$

La solution générale de (3), c'est-à-dire $f(X) + Az = 0$, est

$$z = -A^+ f(X) + (I - AA^+) d,$$

où $d \in \mathbf{R}^n$ est arbitraire. On prendra $d = -\tilde{H}\gamma^T$, où $\tilde{H}$ est une matrice $n \times n$ à transformer par la formule BFS, et où γ joue le rôle de la dérivée du lagrangien. Le lecteur retrouvera notre nouvelle méthode en prenant A^+ définie dans notre exemple 1 et en posant $\gamma = (g, O_m)$.

BIBLIOGRAPHIE

1. J. ABADIE et J. CARPENTIER, *Généralisation de la méthode du gradient réduit de Wolfe au cas de contraintes non-linéaires*, Note HR 6678, Électricité de France, Paris, Octobre 1965.
2. J. ABADIE et J. CARPENTIER, *Generalization of the Wolfe Reduced Gradient Method to the Case of Nonlinear Constraints*, p. 37-47 in *Optimization*, R. FLETCHER, éd., Academic Press, London and New York, 1969, SBN 12-260650-7.
3. M. AVRIEL, *Nonlinear Programming*, Prentice-Hall, Englewood Cliffs, New Jersey, U.S.A., 1976, ISBN 0-13-623603-0.
4. A. R. COLVILLE, A Comparative Study on Nonlinear Programming Codes, p. 487-501 in *Proceedings of the Princeton Symposium on Mathematical Programming*, H. W. KUHN ed., Princeton University Press, Princeton, New Jersey, U.S.A., 1970, ISBN 0-691-08088-7.
5. D. M. HIMMELBLAU, *Applied Nonlinear Programming*, McGraw-Hill, New York, U.S.A., 1972, ISBN 07-028921-2.
6. E. SANDGREN, *The Utility of Nonlinear Programming Algorithms*, Ph. D. Thesis, Purdue University, décembre 1977.
7. R. L. STAHA, *Constrained Optimization via Moving Exterior Truncations*, Ph. D. Thesis, The University of Texas at Austin, mai 1973.
8. M. J. D. POWELL, *A Fast Algorithm for Nonlinearly Constrained Optimization Calculations*, presented at the 1977 Dundee Conference on Numerical Analysis.