La méthode de support pour la résolution d'un programme fractionnaire linéaire (PFL) à variables non-négatives

Doctorant : Hakmi Mohammed Amin, `Hakmimoh32@gmail.com`.
Directeur de thèse : Mohand Bentobache, Université de Laghouat, `mbentobache@yahoo.com`.
Co-directeur de thèse : Mohand Ouamer Bibi, Université de Béjaia, `mobibi.dz@gmail.com`.
Doctoriales de Recherche Opérationnelle, le 12 et 13 Décembre 2018.

1 Introduction

Dans ce travail, nous présentons la méthode de support que nous avons proposée pour la résolution des programmes fractionnaires linéaires à variables non-négatives. L'algorithme suggéré utilise une direction afin de passer d'une solution réalisable à une autre solution améliorée. Nous avons prouvé que cette direction est une direction d'amélioration. De plus, nous avons énoncé et démontré le critère d'optimalité et de suboptimalité d'une solution réalisable de support pour un problème de programmation linéaire fractionnaire. Afin de comparer notre méthode avec la méthode du simplexe, nous avons implémenté les deux méthodes avec le langage de programmation C++, puis nous les avons comparées sur des problèmes générés aléatoirement.

2 Méthodologie

2.1 Position du problème et définitions

Le problème de programmation fractionnaire linéaire se présente sous la forme standard suivante :

$$\max F(x) = \frac{P(x)}{Q(x)} = \frac{\sum_{j=1}^{n} p_j x_j + p_0}{\sum_{j=1}^{n} q_j x_j + q_0}, \tag{1}$$

$$Ax = b, \quad x \geq 0, \tag{2}$$

où p, q et x sont des n-vecteurs ; b un m-vecteur ; A une matrice de dimension $(m \times n)$, avec $rangA = m < n$, p_0 et q_0 deux nombres réels. On suppose que $Q(x) > 0$, pour tout $x = (x_1, x_2, \ldots, x_n)^T$ vérifiant les contraintes (2). Définissons les ensembles d'indices suivants :

$$I = \{1, 2, \ldots, m\},\ J = \{1, 2, \ldots, n\},\ J = J_B \cup J_N,\ J_B \cap J_N = \emptyset,\ |J_B| = m.$$

On peut alors écrire et fractionner les vecteurs et la matrice A de la manière suivante :

• $x = x(J) = (x_j, j \in J)$, $x = \begin{pmatrix} x_B \\ x_N \end{pmatrix}$, $x_B = x(J_B) = (x_j, j \in J_B)$, $x_N = x(J_N) = (x_j, j \in J_N)$, $p = p(J) = (p_j, j \in J)$, $p = \begin{pmatrix} p_B \\ p_N \end{pmatrix}$, $p_B = p(J_B) = (p_j, j \in J_B)$, $p_N = p(J_N) = (p_j, j \in J_N)$, $q = q(J) = (q_j, j \in J)$, $q = \begin{pmatrix} q_B \\ q_N \end{pmatrix}$, $q_B = q(J_B) = (q_j, j \in J_B)$, $q_N = q(J_N) = (q_j, j \in J_N)$, $A = A(I,J) = (a_{ij}, i \in I, j \in J) = (a_j, j \in J)$, $a_j = \begin{pmatrix} a_{1j} \\ \vdots \\ a_{mj} \end{pmatrix}$, $A = (A_B, A_N)$, $A_B = A(I, J_B)$, $A_N = A(I, J_N)$.

• Un vecteur x vérifiant les contraintes (2) est appelé solution réalisable (SR) du problème (1)-(2). On suppose dans ce qui suit que l'ensemble des solutions réalisables S est régulier, i.e., non vide et borné.

• Définissons le nombre réel $\alpha > 0$ comme suit : $\alpha = \min_{x \in S} Q(x)$.

• Une solution réalisable x^0 est dite optimale si : $\forall x \in S$, $F(x^0) \geq F(x)$.

• D'autre part, une solution réalisable x^ε est appelée ε-optimale ou suboptimale si :

$$F(x^0) - F(x^\varepsilon) \leq \varepsilon,$$

où x^0 est une solution optimale du problème (1)-(2) et $\varepsilon \geq 0$.

• Soit un sous-ensemble d'indices $J_B \subset J$ tel que $|J_B| = |I| = m$. L'ensemble J_B est alors appelé support si

$$\det A_B = \det A(I, J_B) \neq 0.$$

• Le couple $\{x, J_B\}$ formé de la solution réalisable x et du support J_B est appelé solution réalisable de support (SRS).

• Une SRS est dite non-dégénérée si : $x_j > 0$, $\forall j \in J_B$.

• Définissons les vecteurs des multiplicateurs : $\pi_P^T = p_B^T A_B^{-1}$ et $\pi_Q^T = q_B^T A_B^{-1}$, ainsi que les vecteurs des coûts réduits : $(\Delta')^T = \pi_P^T A - p^T$, $(\Delta'')^T = \pi_Q^T A - q^T$ et $\Delta(x) = \Delta' - F(x)\Delta''$.

• Pour $\Delta(x) \geq 0$, le nombre

$$\beta(x, J_B) = \frac{\sum_{j \in J_N} \Delta_j(x) x_j}{\alpha} = \frac{\Delta_N^T(x) x_N}{\alpha} = \frac{\Delta^T(x) x}{\alpha}$$

est appelé estimation de suboptimalité.

• Définissons les deux sous-ensembles J_{N+} et J_{N0} de J_N comme suit :

$$J_{N+} = \{j \in J_N / x_j > 0\} \text{ et } J_{N0} = \{j \in J_N / x_j = 0\}.$$

On a alors démontré les théorèmes suivants :

Théorème 1. *(Critère d'optimalité).*
Soit $\{x, J_B\}$ une SRS du problème (1)-(2). Alors les relations

$$\begin{cases} \Delta_j(x) \geq 0, & si\ j \in J_{N0}, \\ \Delta_j(x) = 0, & si\ j \in J_{N+}, \end{cases} \tag{3}$$

sont suffisantes pour l'optimalité de la solution réalisable x. Ces mêmes relations sont aussi nécessaires dans le cas où la SRS $\{x, J_B\}$ est non-dégénérée.

Théorème 2. *(Condition suffisante de suboptimalité).*
Soit $\{x, J_B\}$ une SRS du problème (1)-(2) et ε un nombre positif ou nul arbitraire. Si

$$\Delta(x) \geq 0 \ \ et\ \ \beta(x, J_B) \leq \varepsilon, \tag{4}$$

alors la solution réalisable x est ε-optimale.

2.2 Une itération de l'algorithme

Étant donné un nombre réel positif ou nul quelconque ε et une SRS initiale $\{x, J_B\}$, le but de l'algorithme est alors de construire une solution réalisable ε-optimale x^ε ou carrément une solution optimale x^0. L'itération de l'algorithme consiste donc à faire le passage de $\{x, J_B\}$ à $\{\bar{x}, \bar{J}_B\}$ tel que $F(\bar{x}) \geq F(x)$. Pour ce faire, on construit la nouvelle solution réalisable $\bar{x}$ de la manière suivante :

$$\bar{x} = x + \theta d,\ \theta \geq 0,$$

où d est un n-vecteur appelé direction d'amélioration ; θ est le pas le long de cette direction. Dans cet algorithme, on choisira la métrique du simplexe et on ne fera donc varier qu'une seule composante x_j parmi toutes celles qui ne vérifient pas les relations d'optimalité (3). Pour que l'accroissement soit maximal, il faut alors prendre θ aussi grand que possible et choisir l'indice j_0 tel que

$$|\Delta_{j_0}(x)| = \max_{j \in J_{NNO}} |\Delta_j(x)|, \tag{5}$$

où l'ensemble J_{NNO} représente le sous-ensemble de J_N des indices non-optimaux :

$$J_{NNO} = \{j \in J_N : [\Delta_j(x) < 0 \text{ et } x_j = 0] \text{ ou } [\Delta_j(x) \neq 0 \text{ et } x_j > 0]\}.$$

On posera donc

$$\begin{aligned} d_{j_0} &= -\mathrm{sign}\Delta_{j_0}(x), \\ d_j &= 0,\ j \neq j_0,\ j \in J_N, \\ d(J_B) &= -A_B^{-1} A_N d(J_N) = A_B^{-1} a_{j_0} \mathrm{sign}\Delta_{j_0}(x). \end{aligned} \tag{6}$$

D'autre part, le pas θ doit vérifier les relations suivantes :

$$-\theta d_j \leq x_j,\ j \in J_B; \tag{7}$$

$$\theta\, \mathrm{sign}\Delta_{j_0}(x) \leq x_{j_0}. \tag{8}$$

En calculant les différentes valeurs maximales que peut prendre le pas θ dans les relations (7) et (8), on aura

$$\theta_{j_1} = \min_{j \in J_B} \theta_j, \tag{9}$$

où

$$\theta_j = \begin{cases} -x_j / d_j, & si\ d_j < 0; \\ \infty, & si\ d_j \geq 0. \end{cases} \tag{10}$$

D'autre part,

$$\theta_{j_0} = \begin{cases} x_{j_0}, & si\ \Delta_{j_0}(x) > 0; \\ \infty, & si\ \Delta_{j_0}(x) < 0. \end{cases} \tag{11}$$

Par conséquent, le pas maximal le long de la direction d pour que $\bar{x}$ soit une solution réalisable est égal à :

$$\theta^0 = \min\{\theta_{j_1}, \theta_{j_0}\}. \tag{12}$$

En tenant compte des relations (6) et (12), la nouvelle solution réalisable $\bar{x}$ s'écrit :

$$\bar{x} = x + \theta^0 d \text{ et } F(\bar{x}) = F(x) + \frac{\theta^0 |\Delta_{j_0}(x)|}{Q(\bar{x})}.$$

Les composantes $\bar{x}_j$, $j \in J_N$ vérifient les relations suivantes :

$$\bar{x}_j = x_j, \text{ pour } j \neq j_0;$$

$$\bar{x}_{j_0} = \begin{cases} x_{j_0} - \theta^0, & si\ \Delta_{j_0}(x) > 0; \\ x_{j_0} + \theta^0, & si\ \Delta_{j_0}(x) < 0. \end{cases}$$

Nous calculons le vecteur des coûts réduits correspondant à la SRS $\{\bar{x}, J_B\}$:

$$\Delta_N(\bar{x}) = \Delta_N' - F(\bar{x})\Delta_N''.$$

Si $\Delta_N(\bar{x}) \geq 0$, alors nous calculons l'estimation de suboptimalité de la nouvelle SRS $\{\bar{x}, J_B\}$:

$$\beta(\bar{x}, J_B) = \frac{\Delta_N^T(\bar{x})\bar{x}_N}{\alpha}. \tag{13}$$

Si $\beta(\bar{x}, J_B) \leq \varepsilon$, alors la solution réalisable $\bar{x}$ est ε-optimale et on peut arrêter l'algorithme.
Si $\Delta_N(\bar{x}) \geq 0$ et $\beta(\bar{x}, J_B) > \varepsilon$ ou $\Delta_N(\bar{x}) \ngeq 0$, on remplacera le support J_B par un nouveau support $\bar{J}_B$ de la manière suivante :

(i) Si $\theta^0 = \theta_{j_0}$, alors il est inutile de changer de support. On écrira donc

$$\bar{x} = x + \theta^0 d,\ \bar{J}_B = J_B.$$

(ii) Si $\theta^0 = \theta_{j_1}$, alors la composante d_{j_1} est forcément non nulle :

$$d_{j_1} = e_{i_1}^T A_B^{-1} a_{j_0} \mathrm{sign}\Delta_{j_0}(x) = x_{i_1 j_0} \mathrm{sign}\Delta_{j_0}(x) \neq 0,$$

où $A_B^{-1} a_{j_0} = X(I) = (x_{ij_0}, i \in I)$, i_1 étant la position de l'indice j_1 dans J_B et e_{i_1} est un vecteur unitaire dont la composante non nulle se trouve à la $i_1^{\text{ème}}$ place. Il s'ensuit que $x_{i_1 j_0} \neq 0$, et d'après la règle algébrique utilisée dans la méthode du simplexe, on aura

$$\det \bar{A}_B \neq 0,\ \bar{A}_B = A(I, \bar{J}_B),\ \bar{J}_B = (J_B \backslash \{j_1\}) \cup \{j_0\}.$$

La nouvelle SRS $\{\bar{x}, \bar{J}_B\}$ s'écrit :

$$\bar{x} = x + \theta^0 d,\ \bar{J}_B = (J_B \backslash \{j_1\}) \cup \{j_0\}.$$

Si $\Delta_N(\bar{x}) \geq 0$ et $\beta(\bar{x}, \bar{J}_B) = 0$, alors $\{\bar{x}, \bar{J}_B\}$ est une SRS optimale ; on arrête l'algorithme.
Si $\Delta_N(\bar{x}) \geq 0$ et $\beta(\bar{x}, \bar{J}_B) \leq \varepsilon$, alors $\{\bar{x}, \bar{J}_B\}$ est une SRS ε-optimale et on peut arrêter l'algorithme.
Si $[\Delta_N(\bar{x}) \geq 0$ et $\beta(\bar{x}, \bar{J}_B) > \varepsilon]$ ou $\Delta_N(\bar{x}) \ngeq 0$, on recommencera alors une nouvelle itération avec $\{\bar{x}, \bar{J}_B\}$.

3 Résultats Numériques

Nous avons réalisé une implémentation de la méthode du simplexe et celle de la méthode du support sous le langage de programmation C++. Nous avons généré aléatoirement des programmes fractionnaires linéaires à variables non-négatives de la forme :

$$\max F(x) = \frac{P(x)}{Q(x)} = \frac{p^T x + p_0}{q^T x + q_0}, \tag{14}$$

$$Ax \leq b,\ x \geq 0, \tag{15}$$

avec A une matrice de dimension $n \times n$, $n \in \{20, 40, 60, 80, 100\}$. Nous avons comparé les algorithmes suivants :
Solver1. Simplexe : la méthode primale du simplexe commençant par la solution de base réalisable initiale (SBR) $(0, b)$.
Solver2. Support-SBR : la méthode primale de support commençant par la solution de base réalisable initiale (SBR) $(0, b)$.
Solver3. Support-SRS : la méthode de support commençant par une solution réalisable de support (SRS) initiale constituée d'un point intérieur du domaine réalisable généré aléatoirement et du support $J_B = \{n+1, n+2, \ldots, 2n\}$.

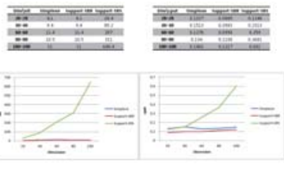

FIGURE 1. Graphe du nombre moyen d'iterations (nit) et du temps moyen d'exécution (cput) pour les différentes méthodes.

Nous constatons que les performances de la méthode du simplexe et celles de la méthode de Support-SBR sont presque identiques. En effet, la méthode de support est identique à la méthode du simplexe lorsque celle-ci commence par un point extrême. Cependant, on remarque bien que la méthode du simplexe et Support-SBR sont nettement plus rapides que la méthode Support-SRS. Cela est du au fait que le point intérieur initial généré aléatoirement pour initialiser la méthode de support est probablement très loin de la solution optimale. Par conséquent, il serait intéressant d'initialiser la méthode de support avec un point intérieur proche du point extrême optimal pour qu'elle puisse être compétitive avec la méthode du simplexe.

4 Conclusion

Dans cet article, nous avons généralisé la méthode de support, développée par R. Gabasov et F. M. Kirillova, pour la résolution des problèmes d'optimisation fractionnaire linéaire à variables non-négatives. Dans le futur, nous allons exploiter les résultats existants sur la dualité en programmation fractionnaire linéaire pour généraliser la méthode duale de support pour la résolution des problèmes de programmation fractionnaire linéaire. De plus, nous allons concevoir un programme C++ plus robuste pour pouvoir résoudre les PFL de grande dimension.

Références

[1] E.B. Bajalinov, *Linear-Fractional Programming : Theory, Methods, Applications and Software*, Applied optimization, Springer, University of Florida, USA., 2003.

[2] M.S. Bazaraa, H.D. Sherali and C.M. Shetty, *Nonlinear Programming. Theory and Algorithms*, Third Edition, Willey Sons, 2006.

[3] M. Bentobache, *Nouvelle méthode pour la résolution des problèmes de programmation linéaire sous forme canonique et à variables bornées*, Mémoire de magister, Université de Béjaia, 2005.

[4] M. Bentobache et M.O. Bibi. *Méthodes Numériques de la Programmation Linéaire et Quadratique : Théorie et Algorithmes*, Presses Académiques Francophones, Allemagne, 2016.

[5] R. Gabasov and F.M. Kirillova. *Methods of linear programming*, Vol. 1, 2 and 3, Edition of the Minsk University, 1977, 1978 and 1980 (in Russian).