CAHIER DU LAMSADE

Laboratoire d'Analyse et Modélisation de Systèmes pour l'Aide à la Décision

(Université de Paris-Dauphine)

Unité Associée au C.N.R.S. n° 825

VALIDATION NUMERIQUE DE L'ALGORITHME DE KARMARKAR

CAHIER N° 76
avril 1987

P. TOLLA

NUMERICAL VALIDATION OF THE KARMARKAR's ALGORITHM

ABSTRACT

The Karmarkar's algorithm is an iterative method like the classical non linear methods. The implementation of this algorithm on a computer is generally stopped when the objective function value is less than a "small" fixed positive scalar β. If β is too small, we may perform many unprofitable iterations ; if β is too large, the computed solution may be far from the optimum. We suggest, in this paper, an optimal criterion which stops the implementation when we have obtained the best solution for the used computer. Another procedure gives the number of significant digits of this solution.

VALIDATION NUMERIQUE DE L'ALGORITHME DE KARMARKAR

RESUME

Comme la plupart des méthodes d'optimisation non linéaire, l'algorithme de Karmarkar relève des méthodes itératives. L'exécution sur ordinateur de cet algorithme est généralement arrêtée lorsque la valeur de la fonction économique est inférieure à un scalaire β "petit" défini à l'avance. Si β est trop petit, l'algorithme bouclera ou l'exécution comportera un grand nombre d'itérations inutiles. Si β est trop grand, la solution obtenue sera éloignée de l'optimum. Nous proposons ici un test d'optimalité qui arrête l'exécution quand la meilleure solution, qui peut être obtenue à l'aide de l'ordinateur utilisé, a été trouvée. Une autre procédure donne le nombre de chiffres significatifs exacts de chaque composante de la solution.

1. Introduction:

En fin d'année 1984 , N. Karmarkar a proposé un algorithme de résolution des programmes linéaires [9] . Contrairement à la méthode de Khatchyan [10] , satisfaisante au plan théorique ,mais qui s'est avérée trés inférieure à la méthode du Simplexe de Dantzig [5] , tant en ce qui concerne sa rapidité que sa capacité de traiter des problèmes de grande taille ; cette nouvelle méthode est , aux dires de son auteur , 50 à 100 fois plus rapide que la méthode du Simplexe et particulièrement efficace dans le cas de problèmes de grande taille à matrices de contraintes trés creuses . Cependant, N. Karmarkar est très avare d'informations sur sa méthode et s'est jusqu'ici refusé à en faire l'expérimentation sur les problèmes qui lui étaient proposés en particulier par le COAL , ce qui laisse planer un doute sérieux sur ses performances . Un certain nombre d'expérimentations ont été faites , en particulier par Nickels-Rödder-Xu-Zimmermann [14] Roos [15] ,Tomlin [18] ,Vanderbei-Meketon-Freedman [21] ,Iri-Imai [8] , Cavalier-Soyster [4] , Shanno [16] et un bon nombre de modifications ont été proposées par ces auteurs , et aussi Anstreicher [1][2] , Todd-Burrell [17] , Minoux [13] , Birge [3], Gay [6] , Gill-Murray [7] , Megiddo [12] , ... Celles-ci ont donné des résultats pas encore comparables avec ceux de la méthode du Simplexe mais pourtant assez prometteurs étant donné

la "jeunesse" de cet algorithme .

L'algorithme de Karmarkar relève des méthodes itératives et, comme on le verra par la suite , a de nombreux points communs avec les méthodes d'optimisation non linéaire . En conséquence , le test d'arrêt qu'il propose est un test à seuil . Nous avons montré que ces tests sont peu efficaces, car si le seuil est trop petit, on risque de faire des itérations inutiles qui peuvent entraîner une détérioration de la solution et sont très coûteuses en temps de calcul ou même constater un bouclage de l'algorithme; si le seuil est trop élevé , la solution obtenue peut être très éloignée de l'optimum . De plus, Karmarkar propose , quand la solution est imprécise,de procéder à des itérations supplémentaires pour obtenir plus de précision ; comment peut-on juger de la précision de la solution si on ne la connaît pas à l'avance ?

C'est pourquoi , nous proposons , ici , un nouveau critère d'arrêt optimal qui stoppe l'exécution de la méthode dès que le maximum de précision que l'on peut obtenir avec l'ordinateur utilisé est atteint ; de plus , le logiciel ainsi constitué donne le nombre de chiffres décimaux significatifs exacts de chaque composante de la solution optimale .

2. L'algorithme de Karmarkar:

2.1. Le problème traité et les hypothèses de travail:

Soit le programme linéaire :

$$\text{Minimiser } c^t.x$$

$$\text{avec} \quad A.x = 0$$

$$e^t.x = 1$$

$$x \geqslant 0$$

où c et x sont des vecteurs de R^n , b un vecteur de $(R^+)^m$, e le vecteur à n composantes toutes égales à 1 , A une matrice à coefficients réels à m lignes et n colonnes .

Le domaine réalisable est l'intersection de l'espace affine $E = \{x/A.x=0\}$ et du simplexe $S = \{x/e^t.x=1, x\geqslant 0\}$. Les hypothèses de départ sont les suivantes :

- le minimum de la fonction est 0 ;
- le problème a une solution finie et le point de départ de l'algorithme $x^0 = (1/n,\ldots,1/n)^t$ est réalisable .

2.2. L'algorithme:

Soit le point réalisable de la k-ième itération

$$x^{(k)} = (x^{(k)}_1,\ldots,x^{(k)}_n)$$

$D^{(k)}$ est la matrice diagonale de coefficients diagonaux $x^{(k)}_1,\ldots,x^{(k)}_n$.

Soit $T_{(k)}$ la transformation projective qui projette le point $x^{(k)}$ au centre a' du simplexe S , avec $a' = (1/n,\ldots,1/n)^t$. Elle est définie par :

$$T_{(k)}(x) = \frac{\{D^{(k)}\}^{-1}.x}{e^t\{D^{(k)}\}^{-1}.x}$$

On a $T_{(k)}(x^{(k)}) = (1/n,\ldots,1/n)^t$

Soit $B^{(k)} = \begin{bmatrix} A.D^{(k)} \\ \hline e^t \end{bmatrix}$

qui est la matrice $A.D^{(k)}$ augmentée d'une ligne de 1 .

L'algorithme de Karmarkar s'écrit :

Pas 1 : Constituer la matrice $B^{(k)}$.

Pas 2 : Calculer $c^{(k)}_p = [I-B^{(k)t}(B^{(k)}B^{(k)t})^{-1}B^{(k)}]D^{(k)}c$.

Pas 3 : Normer le vecteur $c^{(k)}_p$ en $c'^{(k)}_p = c^{(k)}_p / \text{Norme}(c^{(k)}_p)$.

Pas 4 : Calculer $b^{(k)}$ point d'optimisation dans le simplexe S par

$$b^{(k)} = a' - \beta r c'^{(k)}_p$$

où ß est un scalaire que Karmarkar conseille de prendre égal à 1/4 , et r est le rayon de la plus grande sphère inscrite dans le simplexe :

$$r = \frac{1}{\sqrt{n(n-1)}}$$

Pas 5 : Calculer le nouveau point courant dans l'espace de départ

$$x^{(k+1)} = T^{(k)}(b^{(k)}) = \frac{D^{(k)} b^{(k)}}{e^t D^{(k)} b^{(k)}}$$

Soit $L = \log(1+|D_{max}|) + \log(1+\mu)$,

où $D_{max} = \max\{\det(X)/X$ sous-matrice carrée extraite de A$\}$

et $\mu = \max\{|c_i|, |b_i|, i=1,..,n\}$

Si l'on effectue O(nL) itérations de l'algorithme , la solution trouvée est à $2^{-O(L)}$ de l'optimum .

2.3. Principe de l'algorithme:

Soit le point $x^{(k)}$ admissible: il est projeté à l'aide de la transformation projective $T^{(k)}$ au centre du simplexe S.Cependant, cette projection ne transforme pas la fonction économique linéaire $c^t.x$ en une fonction linéaire ; aussi Karmarkar utilise-t-il la fonction potentiel f définie par :

$$f(x) = \sum_{j=1}^{n} \ln(c^t.x/x_j)$$

qui a la propriété de conserver sa morphologie par projection . Karmarkar a montré [10,th.1] que toute réduction adéquate de f(x) implique une réduction de $c^t.x$. Soit g la transformée par la projection $T^{(k)}$ de f :

$$g(y) = \sum_{j=1}^{n} \ln(c^t D_a y / y_j) - \sum_{j=1}^{n} \ln(a_j)$$

Toute réduction de g entraîne une réduction de f ; on va donc résoudre le problème projeté :

$$\text{Minimiser } g(y)$$

$$\text{avec} \quad AD_a y = 0$$

$$e^t y = 1$$

$$y > 0$$

Ce problème étant non linéaire ,on le linéarise et on le résout en limitant le domaine des points réalisables à une sphère intérieure et concentrique à la plus grande sphère inscrite dans le simplexe S , d'où le nouveau programme :

$$\text{Minimiser } z = c^t D_a h$$

$$AD_a h = 0$$

$$e^t h = 0$$

$$h^t h < \beta^2 r^2 \quad \text{avec } 0 < \beta < 1$$

Les théorèmes 4 et 5 ,([10]) , permettent d'affirmer que l'on obtient par la résolution de ce problème une solution améliorée d'une quantité @ prédéfinie. La solution obtenue dans le simplexe est ensuite projetée dans le domaine initial à l'aide de la transformation projective inverse, $T_{(k)}^{-1}$, et donne ainsi un point amélioré .

L'exécution de l'algorithme est arrêtée lorsque :

- soit $c^t x^{(k+1)} = 0$

- soit $c^t x^{(k+1)} / c^t x^{(k)} < 2^{-O(L)}$

2.4. Mise du problème sous une forme utilisable par l'algorithme:

Tel qu'il se présente , l'algorithme de Karmarkar traite un programme linéaire de la forme suivante :

$$\text{Minimiser } z = c^t.x$$

$$\text{avec} \qquad A.x = 0$$

$$e^t x = 1$$

$$x \geqslant 0$$

Il faut donc transformer de façon adéquate les programmes linéaires classiques :

$$\text{Minimiser } z = c^t.x$$

$$A.x = b$$

$$x \geqslant 0$$

Pour créer la contrainte $e^t x = 1$, dans le cas où la solution est bornée , on peut "théoriquement" trouver M tel que :

$$e^t x < M .$$

En ajoutant une variable d'écart x_{n+1} , on aura :

$$e^t X = M \quad \text{où } X = (x_1,\ldots,x_n,x_{n+1})^t$$

Posons $x'_i = X_i/M$ pour $i = 1,\ldots,n+1$; on aura ainsi $e^t x' = 1$.
Considérons d'autre part la matrice A' obtenue à partir de la matrice A en ajoutant à droite de celle-ci un vecteur colonne nul ; nous avons alors :

$$A'X = b$$

ou

$$MA'x' = b = b(e^t x') .$$

On en tire :

$$[MA'-be^t]x' = 0 .$$

Si l'on pose $A'' = MA'-be^t$

et $c' = (c_1,\ldots,c_n,0)^t$

on aura le nouveau programme linéaire :

$$\text{Minimiser } z' = c'^t x'$$

$$\text{avec } A''x' = 0$$

$$e^t x' = 1$$

$$x' \geqslant 0$$

3. Un nouveau test d'arrêt optimal:

3.1. Inconvénients du test d'arrêt classique:

Nous avons vu qu'à l'optimum la valeur optimale de la fonction économique du problème transformé est théoriquement nulle . Comme l'algorithme de Karmarkar relève des méthodes itératives , il est très rare que l'on puisse atteindre cette valeur , l'ordinateur ne travaillant pas en précision infinie en raison de la taille limitée de ses registres . Aussi les logiciels itératifs d'optimisation utilisent des tests d'arrêt à seuil ß . Soit f la fonction économique dont la valeur doit être théoriquement nulle à l'optimum et $x^{(k)}$ le point obtenu à la k-ième itération :

si $f(x^{(k)}) < ß$, on stoppe l'algorithme et le point x est considéré comme la solution optimale .

Ce procédé a des inconvénients très sérieux :

- Si ß est choisi trop grand , l'exécution de l'algorithme est arrêtée trop tôt : la solution obtenue peut être éloignée de l'optimum effectif . On en a un cas très significatif pour l'exemple de Wood

en programmation non linéaire sans contrainte , où le point courant passe très près d'un point selle ; la seule quatrième décimale de chaque composante de la solution évolue bien que l'on se trouve très loin de la solution optimale et si ß est trop grand l'exécution se terminera en ce point .

- Si ß est choisi trop petit , la valeur calculée de la fonction utilisée pour le test d'arrêt aura une décroissance très faible , ou dépassera toujours ß ; on assistera donc soit à un bouclage de l'algorithme , soit à l'exécution d'un grand nombre d'itérations inutiles avec, dans beaucoup de cas, une détérioration de la précision de la solution . Ceci est extrêmement préjudiciable à la mise en oeuvre de la méthode de Karmarkar dont la caractéristique essentielle est un faible nombre d'itérations par rapport à la méthode du Simplexe , chacune d'elles se révélant , actuellement , trés coûteuse .

3.2. Le test d'arrêt optimal de J. Vignes [22]:

Ce test a l'avantage d'être exactement adapté à la taille des registres de travail de l'ordinateur utilisé . Le test va porter en optimisation non-linéaire sur la nullité des composantes du gradient de la fonction économique , Grad(f(x)) , à l'optimum mais pourra être adapté très simplement au cas d'une fonction économique dont on sait qu'elle est théoriquement nulle à l'optimum .

Le principe du test d'arrêt optimal de Vignes est le

suivant:

A chaque itération , on calcule les nombres de chiffres décimaux significatifs exacts , C_j , des composantes de Grad(f(x)) :

- Si pour tout j=1,...,n , $C_j < 1$, on a atteint une solution informatiquement satisfaisante et le processus itératif doit être arrêté .
- Si l'un au moins des C_j est supérieur ou égal à 1, le processus itératif doit être poursuivi .

3.3. Calcul du nombre de chiffres significatifs exacts par la méthode de permutation-perturbation de Vignes [23] :

3.3.1. Méthode de permutation-perturbation :

En informatique , les règles de l'algèbre ne sont pas vérifiées ; dans une expression arithmétique , toute permutation des opérateurs permutables conduira à un résultat numérique qui peut être différent ; le nombre de résultats différents que l'on peut obtenir correspond au nombre total de ces permutations . On notera C_{op} le nombre total de combinaisons possibles correspondant aux diverses permutations .

Comme au niveau de chaque opération arithmétique , les opérateurs virgule flottante engendrent une erreur d'arrondi ou de troncature , cette opération a deux résultats , l'un par défaut , l'autre par excès , chacun représentant aussi légitimement le résultat exact . Donc , si l'algorithme demande l'exécution de k opérations arithmétiques le nombre de résultats informatiques représentant le résultat exact est 2^k .

Le cardinal de l'ensemble des résultats informatiques

images du résultat exact après application de la méthode de permutation-perturbation sera donc : $2^k.C_{op}$.

3.3.2. Evaluation statistique du nombre de chiffres décimaux significatifs exacts du résultat :

Soit R_i un élément de la population des résultats qui sont obtenus par permutation-perturbation d'un algorithme numérique ; . le nombre C_i de chiffres décimaux significatifs du résultat peut être calculé par :

$$C_i = \mathrm{Log}_{10} \frac{|R_i|}{\varepsilon_i}$$

avec $\varepsilon_i = \sqrt{(R_i - \bar{R})^2 + \delta^2}$

où ε_i est l'évaluation statistique de l'erreur commise sur R_i , $\bar{R}$ la moyenne des éléments de la population des résultats informatiques et δ^2 sa variance .

Maillé [11] a démontré que le meilleur estimateur de R_i était sa moyenne empirique et que, dans la pratique, il suffit de considérer une population de 3 éléments R_i pour estimer R . Pour une telle population l'intervalle de confiance à 95% est :

$$I = [R - 2.48\delta , R + 2.48\delta] \text{ (Table de Student)}$$

et le nombre de chiffres décimaux significatifs :

$$C = \mathrm{Log}_{10} \frac{|\bar{R}|}{\delta} - 0.69 .$$

3.3.3. Pratique de la méthode de permutation-perturbation :

Dans la pratique et plus particulièrement dans le cas

d'une expression arithmétique ,on crée deux éléments supplémentaires de la façon suivante :

- on permute d'abord aléatoirement les monomes constituant l'expression ,
- on perturbe le résultat de chaque monome en ajoutant aléatoirement 0 ou 1 au dernier bit de la mantisse de chacun des composants calculés de chaque monome .

3.4. Un test d'arrêt optimal adapté à l'algorithme de Karmarkar :

3.4.1. Remarque préliminaire :

Le test d'arrêt optimal de Vignes s'applique parfaitement à l'algorithme de Karmarkar que l'on peut classer parmi les méthodes itératives telles que les méthodes d'optimisation non linéaire ; cependant , le problème d'optimisation à résoudre est linéaire ; aussi pouvons nous lui appliquer une procédure plus simple et moins coûteuse dérivée de la méthode des résidus normés due encore à Vignes [22] . La description de la méthode de permutation-perturbation que nous avons décrite dans le paragraphe précédent nous sera , malgré cela,très utile par la suite car nous utiliserons une variante de cette méthode pour donner une évaluation du nombre de chiffres décimaux significatifs de la solution optimale .

3.4.2. La méthode des résidus normés :

Lorsqu'on a résolu un système linéaire , il est fort utile de savoir si la solution obtenue par une méthode classique (méthode de Gauss , de Gauss-Jordan ,...) est la meilleure que l'on puisse obtenir avec l'ordinateur considéré , si elle peut

être améliorée ou si l'algorithme utilisé ne peut fournir de solution convenable et s'il faut mettre en oeuvre un autre type de méthode pour obtenir une solution satisfaisante .

Soit l'équation linéaire :

$$a_1x_1 + a_2x_2+\ldots+a_nx_n = b$$

Vignes définit le vecteur résidu théorique $\hat{\rho}$ par :

- Dans le cas d'une arithmétique avec troncature:

$$\hat{\rho} = 2^{-p}\, N \sqrt{(a_1x_1)^2+\ldots+(a_nx_n)^2+ b^2}$$

- Dans le cas d'une arithmétique avec arrondi :

$$\hat{\rho} = 2^{-p} \sqrt{N \left[(a_1x_1)^2+\ldots+(a_nx_n)^2+ b^2 \right]}$$

où p est le nombre de bits de mantisse pour l'ordinateur utilisé et N le nombre de coefficients a_i non nuls de l'équation .

$\hat{\rho}$ fournit un ordre de grandeur , auquel on doit s'attendre, du résidu de l'équation linéaire pour une solution x calculée .

Considérons maintenant le système linéaire :

$$a_{i1}x_1+\ldots+ a_{in}x_n = b_i \quad , \; i=1,\ldots,n$$

Soit ρ_i les composantes calculées du vecteur résidu pour la solution x calculée du système linéaire et $\hat{}_i$ les composantes du vecteur résidu théorique $\hat{}$ de chaque équation du système linéaire .

On appelle résidu normé ρ^*_i la quantité définie par :

$$\rho^*_i = |\rho_i| \; / \; \hat{\rho}_i$$

On peut trouver trois situations :

(i) Les n résidus normés sont de l'ordre de grandeur de l'unité : alors, la solution satisfait numériquement le système .

(ii) L'un au moins des résidus normés est nettement supérieur à l'unité et nettement inférieur à 2^p :

$$1 \ll \rho^*_i \ll 2^p$$

alors , la solution ne satisfait pas numériquement le système mais elle peut, en général, être améliorée par affinage itératif .

(iii) L'un au moins des résidus normés à un ordre de grandeur comparable à 2^p : la solution est mauvaise et ne peut généralement pas être améliorée , ce qui signifie que la méthode de résolution n'est pas adaptée au système proposé .

3.4.3. Adaptation de la méthode des résidus normés à l'algorithme de Karmarkar :

Dans cette méthode , notre critère d'arrêt consiste à vérifier que la fonction économique a atteint un zéro théorique :

$$c_1 x^*_1 + \ldots + c_n x^*_n = 0$$

Nous sommes en présence d'une équation linéaire à second membre nul ; aussi , nous allons lui appliquer la méthode des résidus normés modifiée de la façon suivante :

- Dans le cas d'une arithmétique avec troncature , le vecteur résidu théorique s'écrira :

$$\hat{\rho} = 2^{-p} N \sqrt{(c_1 x_1)^2 + \ldots + (c_n x_n)^2}$$

- Dans le cas d'une arithmétique avec arrondi on aura :

$$\hat{\rho} = 2^{-p} \sqrt{N \left[(c_1 x_1)^2 + \ldots + (c_n x_n)^2\right]}$$

Le test d'arrêt optimal devient alors :

(i) Si le résidu normé est de l'ordre de grandeur de l'unité , la fonction économique a atteint un zéro

théorique aussi on arrête l'exécution de l'algorithme .

(ii) Dans le cas contraire la fonction économique n'est pas nulle et il faut poursuivre l'exécution de l'algorithme .

Comme on peut le constater,ce test est três peu coûteux et d'une mise en oeuvre extrêmement simple . Il permet d'autre part de stopper le déroulement de l'algorithme lorsque des itérations supplémentaires n'apporteront aucune amélioration à la fonction .

3.5. Calcul de la précision de la solution dans les méthodes itératives :

Vignes [23] propose la procédure suivante :

Aprés avoir calculé une première solution optimale $x^{*(1)}$ en arrêtant la méthode itérative à l'aide du critère d'arrêt optimal, on resoud deux fois le problème initial en partant chaque fois du point initial perturbé . A partir de ces trois éléments de la population des vecteurs solutions , on calcule le nombre C_i de chiffres décimaux significatifs exacts de chaque composante de la solution en utilisant la relation :

$$C_i = \mathrm{Log}_{10} \frac{\varepsilon_i}{|x_i^{*(1)}|} \quad , \; i=1,\ldots,n$$

avec $$\varepsilon_i = \sqrt{(x^{*(1)}{}_i - x_i)^2 + \sigma^2{}_i} \quad , \; i=1,\ldots n$$

où x_i et σ_i sont respectivement la moyenne et la variance empiriques des $x^{*(j)}{}_i$.

Cette méthode a prouvé son efficacité mais elle multiplie par trois le nombre d'itérations et donc le temps de calcul .

3.6. Une variante de la méthode de perturbation :

La méthode décrite dans le paragraphe précédent a l'inconvénient d'être assez coûteuse . D'autre part , dans l'algorithme de Karmarkar , l'une des hypothèses est que le point de départ doit être réalisable ; si cette hypothèse n'est pas vérifiée , l'algorithme peut diverger ou conduire à une solution optimale différente en cas de dégénérescence duale et le calcul du nombre de chiffres décimaux exacts sera erroné . C'est pourquoi nous avons proposé dans [20] de partir , non du point initial perturbé , mais de l'antépénultième solution perturbée . Comme cette solution est réalisable et intérieure au domaine des points admissibles (suivant le principe de l'algorithme) , elle reste réalisable après perturbation ; d'autre part, chacune des deux nouvelles solutions optimales sera obtenue après au plus deux itérations supplémentaires , ce qui réduit de façon importante le coût de mise en oeuvre de l'algorithme .

4. Expérimentation numérique :

4.1. Le code utilisé :

Nous disposions du logiciel très robuste mis aimablement à notre disposition par C. Roos [15] . Il a l'avantage, par une modification simple du programme linéaire, de partir d'une solution admissible évidente , ce qui n'est pas le cas en général lorsque le programme linéaire comporte des contraintes d'inégalité inférieure supérieure et d'égalité .

Soit le programme linéaire :

$$\text{Maximiser } c^t.x$$

$$\text{avec} \quad A.x \leqslant b$$

$$x \geqslant 0$$

Son dual s'écrit :

$$\text{Minimiser } u.b$$

$$\text{avec} \quad u.A \geqslant c$$

$$u \geqslant 0$$

où u est un vecteur ligne à m composantes .

Résoudre le programme linéaire initial est équivalent à trouver une solution du système d'inéquations linéaires :

$$A.x \leqslant b$$

$$u.A \geqslant c$$

$$u.b - c.x = 0$$

$$x \geqslant 0$$

$$u \geqslant 0$$

En utilisant les variables d'écart non-négatives y et v , on obtient le système suivant :

$$A.x + y = b$$

$$u.A - v = c$$

$$u.b - c.x = 0$$

$$x \geqslant 0$$

$$y \geqslant 0$$

$$u \geqslant 0$$

$$v \geqslant 0$$

Soit les vecteurs x° , y° , u° et v° respectivement de même dimension que x, y, u et v ; alors, le système précédent a une solution si et seulement si le programme linéaire suivant a

pour valeur optimale de sa fonction économique μ zéro :

$$\begin{aligned} &\text{Minimiser} && \mu \\ &\text{avec} && A.x + y + \mu(b - A.x^{\circ} - y^{\circ}) = b \\ & && u.A - v + \mu(c - u^{\circ}.A + v^{\circ}) = c \\ & && u.b - c.x - \mu(u^{\circ}.b - c.x^{\circ}) = 0 \\ & && x \geqslant 0 \ , \ y \geqslant 0 \ , \ u \geqslant 0 \ , \ v \geqslant 0 \ , \ \mu \geqslant 0 \end{aligned}$$

De plus, ce programme linéaire équivalent au programme linéaire initial admet comme solution réalisable :

$$x = x^{\circ} \ , \ y = y^{\circ} \ , \ u = u^{\circ} \ , \ v = v^{\circ} \ , \ \mu = 1 \ .$$

L'inconvénient de cette transformation est que le nouveau programme linéaire à résoudre est de beaucoup plus grande taille que le programme linéaire initial ce qui peut être gênant lorsque celui-ci est lui même très grand .

Il faut de plus remarquer que, dans son logiciel , C. Roos n'a pas utilisé la méthode de projection suggérée par Karmarkar dans ses différents articles mais l'orthogonalisation de Graham-Schmidt . D'autre part , contrairement à Karmarkar qui fait un pas $\beta = 1/4$ dans la direction de la projection du gradient , C. Roos propose une recherche monodimensionnelle simplifiée ; cette idée a été testée parallèlement par Lisser et Maculan et a permis de réduire sensiblement le nombre d'itérations nécessaires à la recherche de la solution optimale .

4.2. Les exemples traités :

Le logiciel transformé par nos soins a été testé sur les exemples suivants déjà utilisés par un certain nombre d'auteurs :

4.2.1. HITAC , MPSII Manual :

Minimiser $c^t.x$

avec $A.x \geqslant b$

$x \geqslant 0$

m = 11 , n = 17

c = (25,30,350,150,40,20,100,40,60,100,17,20,20,12,75,900,20)

b = (2300,75,38,660,1300,10,1900,1.2,1.2,63,400)

Matrice A

351,270,260,451,156,59,721,58,130,118,77,51,40,24,38,311,40

6.2,8,17.5,12,12.7,2.9,0.6,6,17.5,20,1.9,1.3,1.2,1.6,3,34.2,0.8

0.8,1.5,20.5,44.3,11.2,33,816,35,6,35,0.1,0.2,0.2,0.2,0.4,0.7,0.3

6,11,6,9,65,100,10,120,80,12,5,35,40,45,98,470,14

2,480,90,90,90,36,780,5,100,90,12,57,10,15,25,600,4

0.4,1,1.3,1.2,2.6,0.1,0.1,1.4,3,0.7,0.5,0.5,0.5,0.4,3.3,2.3,0.2

0,0,25,0,800,120,2400,0,60,40,0,1300,6,33,2600,10000,40

0.09,0.1,0.04,0.4,0.1,0.04,0.01,0.02,0.02,0.15,0.1,0.06,0.03,0.08

,0.12,0.21,0.09

0.03,0.03,0.11,0.1,0.4,0.15,0.03,0.02,0.15,0.2,0.03,0.04,0.02,0.05

,0.3,1,0.02

0,0,0,0,0,2,0,0,1,1,15,7,10,50,100,20,50

0,0,0,0,10,0,0,0,530,0,0,0,0,0,0,0,0

Solution optimale : $z^* = 354.03042$

4.2.2. Exemples de Klee et Minty :

$$\text{Maximiser} \sum_{j=1}^{n} \beta^{n-j} x_j$$

$$\text{avec} \quad 2 \sum_{j=1}^{i-1} \beta^{i-j} x_j + x_i \leqslant 1 \ , \ i=1,\ldots,n$$

$$x_j \geqslant 0 \ , \ j=1,\ldots,n$$

$\beta = 0.4$

4.2.3. Problèmes tirés au hasard :

Maximiser $e^t.x$

avec $A.x \leqslant 10^4$

$x \geqslant 0$

$e = (1,\ldots,1)^t$

A est une matrice à m lignes et n colonnes avec $m < n$ et composée de nombres au hasard tirés entre 1 et 1000 .

(Pour mémoire car ces exemples n'ont pas encore été traités).

4.2.4. Problèmes d'affectation :

$$\text{Maximiser} \sum_{i=1}^{k} \sum_{j=1}^{k} c_{ij} x_{ij}$$

$$\sum_{i=1}^{k} x_{ij} \leqslant 1 \ , \ j=1,\ldots,k$$

$$\sum_{j=1}^{k} x_{ij} \leqslant 1 \ , \ j=1,\ldots k$$

Les c_{ij} ont été tirés au hasard entre 1 et 100 . Les différents problèmes ont été aussi traités en transformant les contraintes d'inégalité en égalités .

4.2.5. Programmes de transport simples :

$$\text{Minimiser} \sum_{i=1}^{m} c_{ij} x_{ij}$$

avec $\sum_{i=1}^{m} x_{ij} = a_j \quad , \; j=1,\ldots,n$

$$\sum_{j=1}^{n} x_{ij} = b_i \quad , \; i=1,\ldots,m$$

c_{ij} , a_j , b_i sont des entiers tirés au hasard de façon que :

$$\sum_{j=1}^{n} a_j = \sum_{i=1}^{m} b_i$$

La dernière contrainte qui est redondante a été supprimée .

4.2.6. Problèmes dérivés des systèmes de Hilbert :

Ces problème ont été particulièrement étudiés en raison de leur mauvaise stabilité numérique ; il est à peu prés impossible de résoudre de tels systèmes ayant plus de 12 équations et variables à l'aide de méthodes de pivot , le déterminant de telles matrices étant de l'ordre de 10^{-120} !

Minimiser $c^t.x$

avec $A.x \leqslant b$

$x \geqslant 0$

$$a_{ij} = \frac{1}{i+j} \quad , \; i = 1,\ldots,m \; , \; j = 1,\ldots m$$

$$b_i = \sum_{j=1}^{n} \frac{1}{i+j} \quad , \; i = 1,\ldots,m$$

$$c_i = \frac{2}{i+1} + \sum_{j=2}^{n} \frac{1}{i+j} \quad , \; i = 1,\ldots,m$$

Solution optimale unique : $x^* = (1,\ldots,1)^t$.

4.3. Les résultats :

Nous avons d'abord testé le logiciel de C. ROOS sur un micro-ordinateur RAINBOW 100B (256k RAM) après l'avoir transposé en PASCAL ; nous avons pu constater que ce logiciel fonctionnait très correctement. Cependant, le temps de calcul pour résoudre un programme linéaire dépassant 25 contraintes était assez important c'est-à-dire de l'ordre de 3/4 d'heure pour l'exemple dérivé du problème de Hilbert ! D'autre part, en particulier pour cet exemple très difficile en raison des problèmes de précision qui sont sa caractéristique essentielle,nous désirions tester les différents exemples en précision étendue,ce que nous ne pouvions bien sûr faire en PASCAL sur RAINBOW 100B . Aussi la majeure partie de l'expérimentation a été effectuée sur le MULTICS du CICRP à partir du programme écrit en FORTRAN standard .

Nous avons tout d'abord traité l'exemple HITAC ; la valeur de la fonction économique à l'optimum est en, double précision, exactement la valeur donnée ; par contre , en simple précision la valeur trouvée est à 0.05% de l'optimum . Il faut de l'ordre de 27 itérations pour trouver une solution optimale en double précision ; en simple précision , on obtient une solution "optimale" en une quinzaine d'itérations .

Les problèmes dérivés des matrices de Hilbert ont été résolus jusqu'à l'ordre 25 avec 3 à 4 chiffres significatifs exacts au maximum pour chaque composante de la solution ; au-dessus de cette taille, nous avons obtenu des solutions sans aucun chiffre décimal exact mais, cependant, pas trop éloignées de la solution: les valeurs des variables étaient de l'ordre de 0.7 au lieu de 1.

Les programmes de transport et d'affectation avec des don-

nées tirées au hasard et comportant au plus 15 sources et 15 puits ont été résolus avec une grande précision : 5 à 7 chiffres significatifs exacts ; par contre, en cas de dégénerescence , certaines variables n'étaient pas entières car on se trouvait probablement sur une facette . Il faut constater , dans ce cas , que , un bon nombre de variables étant théoriquement nulles , leur valeur calculée était non nulle,ce qui est un phénomène très fréquent dans les méthodes itératives ; ceci indiquait qu'elles étaient effectivement nulles . On peut concevoir , à ce niveau , une amélioration supplémentaire consistant à annuler ces variables , ce qui augmenterait la précision des résultats . Il se pose,pour ce type d' exemples résolus en nombres entiers, le problème de la recherche d'une solution réalisable de base optimale dont les composantes sont obligatoirement entières à partir d'une solution réalisable optimale qui n'est pas de base : ce problème a été résolu par Y. Wang qui a trouvé un algorithme prenant moins de n itérations d'une procédure proche de la méthode du simplexe .

5. Conclusion :

Les méthodes proposées ici permettent de stopper l'exécution de l'algorithme de Karmarkar dès que la meilleure solution que l'on peut obtenir à l'aide de l'ordinateur utilisé est atteinte et de donner le nombre de chiffres significatifs exacts de la solution . Elle peut constituer une procédure de référence pour comparer de façon réaliste les logiciels qui utilisent cette nouvelle procédure de résolution des programmes linéaires .

BIBLIOGRAPHIE :

[1] K.M. ANSTREICHER , Analysis of a modified Karmarkar algorithm for linear programming , Yale school of Organization and Management , New Haven , 1985 .

[2] K.M. ANSTREICHER , A monotonic projective algorithm for fractionnal linear programming ,id. , 1986 .

[3] J.R. BIRGE , Solving Stochastic linear problems via a variant of Karmarkar's algorithm , University of Michigan , College of Engineering , Ann Arbor , Michigan , 1985 .

[4] T.M. CAVALIER , A.L. SOYSTER , Some computational experience and a modification of the Karmarkar algorithm , Department of Industrial and Mangement Systems Engineering , 207 Hammond Building , The Pennsylvania State University , University Park , PA 16802 , 1985 .

[5] G.B. DANTZIG , Linear programming and extensions , Princeton University Press , Princeton , New Jersey , 1963 .

[6] D.M. GAY , A variant of Karmarkar's linear programming algorithm for problems in standard form, AT&T Bell Laboratories, Murray Hill , New Jersey 07974 , 1985 .

[7] Ph.E. GILL , W. MURRAY , On projected Newton Barrier methods for linear programming and an equivalence to Karmarkar's projective method , Department of Operations Research , Stanford University , 1985 .

[8] M. IRI , H. IMAI , A multiplicative penalty function method for linear programming - Another "New and Fast" algorithm , Faculty of Engineering , University of Tokyo , Hongo 7-3-1 , Bunkyo-ku , Tokyo , Japan , 1985 .

[9] N. KARMARKAR , A new polynomial-time algorithm for linear programming , Combinatorica 4 (4) , p. 373-395 , 1984 .

[10] L.G. KHACHIAN ,A polynomial algorithm in linear programming, Doklady Akad. NAVK , USSR , Tome 244 ,1979 .

[11] M. MAILLE , Some methods to estimate accuracy of measures of numerical computations , Proceedings of Mathematics for Computer Science , Colloque International AFCET , Paris , 1982 .

[12] N. MEGIDDO ,A variation on Karmarkar's algorithm (preliminary report) , IBM Research Laboratory , San Jose , CA 95193 , 1985 .

[13] M. MINOUX , New suggested implementation of Karmarkar's algorithm , à paraître dans les Cahiers du LAMSADE , Université Paris-Dauphine , 1986 .

[14] W. NICKELS , W. RODDER , L. XU , H.J. ZIMMERMANN , Intelligent gradient search in linear programming , European Journal of Operational Research , 22 , p. 293-303 , North-Holland , 1985 .

[15] C. ROOS , On Karmarkar's projective method for linear programming , Department of Mathematics and Informatics , Report n° 85-23 ,Delft University of Technology , 1985 .

[16] D.F. SHANNO , Computing Karmarkar projection quickly ,Graduate School of Administration , University of California ,Davis , California , 1985 .

[17] M.J. TODD , B.P. BURRELL , An extension of Karmarkar's algorithm for linear programming using dual variables , School of Operations Research and Industrial Engineering ,

College of Engineering , Cornell University , Ithaca , New-York 14850 , 1985 .

[18] J.A. TOMLIN , An experimental approach to Karmarkar's projective method for linear programming , Ketron Inc. , Mountain View , CA 94040 , 1985 .

[19] P. TOLLA , Optimal termination criterion and accuracy tests in mathematical programming , à paraître dans Mathematics and Computers in Simulation , 1986 .

[20] P. TOLLA , Contribution à l'amélioration des logiciels de programmation mathematique en variables réelles , Thèse de doctorat d'état , Paris , 1983 .

[21] R.J. VANDERBEI , M.S. MEKETON , B.A. FREEDMAN , A modification of Karmarkar's linear programming algorithm , AT & T Bell Laboratories , Holmdel , New Jersey 07733 , 1985 .

[22] J. VIGNES , Algorithmes numériques Analyse et mise en oeuvre Tome 1 : arithmétique des ordinateurs , systèmes linéaires : 1974 , Tome 2 : équations et systèmes non linéaires , 1980 , TECHNIP , Paris .

[23] J. VIGNES , Implémentation des méthodes d'optimisation : test d'arret optimal , controle et précision de la solution, RAIRO-Recherche Opérationnelle , vol 18 , n°1 , 1984 .