Agrégation externe de mathématiques — Sujet Lyon, adapté de la session 2005

Épreuve de modélisation, option calcul scientifique : méthodes numériques et symboliques

RÉSOLUTION DE SYSTÈMES LINÉAIRES EN ENTIERS NATURELS

Résumé. On étudie les solutions à coefficients dans $\mathbb{N}$ d'un système d'équations linéaires.

Thème applicatif, mots-clefs : Équations linéaires, géométrie affine euclidienne.

Il est rappelé que le jury n'exige pas une compréhension exhaustive du texte. Vous êtes laissé(e) libre d'organiser votre discussion comme vous l'entendez. Des suggestions de développement, largement indépendantes les unes des autres, vous sont proposées en fin de texte. Vous n'êtes pas tenu(e) de les suivre. Il vous est conseillé de mettre en lumière vos connaissances à partir du fil conducteur constitué par le texte. Le jury appréciera que la discussion soit accompagnée d'exemples traités sur ordinateur.

1 Introduction

La résolution de systèmes linéaires est un problème bien connu lorsqu'on travaille dans un corps ou dans un anneau euclidien. On s'intéresse ici à la résolution de systèmes d'équations linéaires dans une structure beaucoup moins riche : celle de *monoïde*, $\mathbb{N}$ en l'occurrence. L'étude proposée se borne au seul cas des équations linéaires *homogènes*.

2 Notations

Un système linéaire homogène en entiers naturels à p équations et q inconnues s'écrit sous la forme

$$\begin{cases} a_{1,1}x_1 + a_{1,2}x_2 + \cdots + a_{1,q}x_q = 0 \\ \quad\vdots \\ a_{p,1}x_1 + a_{p,2}x_2 + \cdots + a_{p,q}x_q = 0 \end{cases}$$

où, pour $1 \leq i \leq p$ et $1 \leq j \leq q$, $a_{i,j} \in \mathbb{Z}$, et l'inconnue $x = (x_1, \ldots, x_q) \in \mathbb{N}^q$. On note A la matrice du système et a l'application linéaire de $\mathbb{R}^q$ dans $\mathbb{R}^p$ associée à A dans les bases canoniques. Pour $1 \leq j \leq q$,

$$a(e_j) = (a_{1,j}, \ldots, a_{p,j})$$

est appelé le j^{e} *vecteur défaut* du système.

3 Ensembles de solutions

L'ensemble $\mathcal{S}$ des solutions d'un tel système est stable par combinaisons linéaires à coefficients dans $\mathbb{N}$, ce qui lui confère une structure de *monoïde additif*.

Définition 3.1. Une partie $\mathcal{Y}$ de $\mathcal{S}$ est appelée une *famille génératrice* de $\mathcal{S}$ si toute solution de $\mathcal{S}$ s'écrit comme combinaison linéaire à coefficients dans $\mathbb{N}$ d'éléments de $\mathcal{Y}$.

On munit $\mathbb{N}^q$ de l'ordre partiel strict $\succ$, qui étend l'ordre naturel $>$ sur les entiers :

$$(x_1, \ldots, x_q) \succ (x'_1, \ldots, x'_q) \text{ si } \forall i \in [\![1, q]\!],\ x_i \geq x'_i \text{ et } \exists i \in [\![1, q]\!],\ x_i > x'_i.$$

On dit alors que x *domine strictement* x'.

Définition 3.2. Une solution $x = (x_1, \ldots, x_q)$ est dite *minimale* si ce n'est pas la solution triviale $(0, \ldots, 0)$ et si elle ne domine strictement aucune solution non nulle du système.

Définition 3.3. On appelle *famille génératrice minimale* toute famille génératrice de $\mathcal{S}$ constituée exclusivement de solutions minimales.

Exemple 3.4. Les systèmes suivants illustrent la variété des situations possibles :

(S_1) $x_1 + x_2 + 2x_3 - 3x_4 = 0$ — possède 6 solutions minimales ;

(S_2) $\begin{cases} x_1 + 3x_2 - 2x_3 - x_4 = 0 \\ x_1 + x_2 + 2x_3 - 3x_4 = 0 \end{cases}$ — possède 2 solutions minimales ;

(S_3) $\begin{cases} 4x_1 + 6x_2 + 2x_3 - 2x_4 = 0 \\ -3x_1 + x_2 + 5x_3 + 3x_4 = 0 \end{cases}$ — ne possède pas de solution minimale ;

(S_4) $\begin{cases} -7x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 = 0 \\ -5x_1 + x_2 + x_3 + x_4 + x_5 + x_6 = 0 \\ -3x_1 + x_2 + x_3 + x_4 = 0 \end{cases}$ — possède 11 942 solutions minimales.

Proposition 3.5. *Tout système linéaire homogène en entiers naturels admet une unique famille génératrice minimale (éventuellement vide si l'unique solution est la solution nulle). De plus, cette famille génératrice est de cardinal fini.*

Remarque 3.6. La finitude de la famille génératrice minimale repose sur le *lemme de Dickson* : toute suite infinie $(x^{(1)}, x^{(2)}, \ldots)$ d'éléments de $\mathbb{N}^q$ contient deux indices $i < j$ tels que $x^{(i)} \preceq x^{(j)}$. En particulier, toute antichaîne de $(\mathbb{N}^q, \preceq)$ — c'est-à-dire tout ensemble de vecteurs deux à deux incomparables pour $\preceq$ — est finie. Ce résultat, qui se prouve par récurrence sur q en utilisant le bon ordre de $\mathbb{N}$, est également l'outil central dans la preuve de terminaison des algorithmes présentés aux sections suivantes.

4 Généralités sur les algorithmes de résolution

On décrit ici quelques algorithmes permettant de déterminer la famille génératrice minimale $\mathcal{M}$ de l'ensemble des solutions d'un système linéaire homogène en entiers naturels.

Le principe de base consiste à obtenir les solutions minimales par construction de *séquences* de q-uplets d'entiers naturels dans lesquelles une composante est incrémentée à chaque étape. Par exemple, la solution minimale $(0, 1, 2, 1)$ peut être obtenue par la séquence

$$(0, 0, 1, 0) \to (0, 0, 1, 1) \to (0, 0, 2, 1) \to (0, 1, 2, 1)$$

(parmi plusieurs autres possibles). La solution triviale étant exclue, les calculs sont initialisés par un vecteur de la base canonique. De plus, tous les calculs de séquences possibles sont menés en parallèle, afin d'obtenir toutes les solutions minimales en temps fini et de stopper une séquence dès que le dernier q-uplet calculé domine strictement une solution minimale déjà obtenue.

Ce faisant, on est amené à engendrer tous les q-uplets strictement dominés par une solution minimale. Il est cependant possible de limiter l'exploration de cet espace de recherche en n'autorisant l'incrémentation des composantes que lorsqu'une certaine contrainte géométrique (C) est satisfaite. Remarquons toutefois que le même q-uplet peut participer à plusieurs séquences différentes.

L'algorithme générique $\mathcal{A}$ (dépendant de la contrainte (C)), formalisé ci-après (cf. Figure 1), capture plus précisément ces idées.

La recherche d'une contrainte (C) judicieuse est capitale. On va procéder progressivement, du cas particulier d'une seule équation au cas d'un système général.

$L \leftarrow \{e_1, \ldots, e_q\}$
$M \leftarrow \emptyset$
tant que $L \neq \emptyset$ **faire**
    $L' \leftarrow \emptyset$
    pour chaque $x \in L$ **faire**
        si il n'existe pas $v \in M \cup (L \setminus \{x\})$ tel que $v \preceq x$ ou $x \preceq v$
        alors
            si $a(x) = 0$
            alors $M \leftarrow M \cup \{x\}$
            sinon $L' \leftarrow L' \cup \{x + e_j \mid 1 \leq j \leq q,\ (x, e_j)$ vérifie $(C)\}$
        fin si
    fin pour
    $L \leftarrow L'$
fin tant que
renvoyer M

FIGURE 1 – Algorithme générique $\mathcal{A}$ (dépendant de la contrainte (C)).

Exemple 4.1 (Contrainte de boule)**.** Fixons un rayon $R \in \mathbb{N}$ et munissons $\mathbb{N}^q$ de la norme ℓ^1 : $\|x\|_1 = x_1 + \cdots + x_q$. On définit la contrainte (C_R) par :

$$(x, e_j) \text{ vérifie } (C_R) \quad \Longleftrightarrow \quad \|x + e_j\|_1 \leq R,$$

c'est-à-dire que l'on n'incrémente x dans la direction e_j que si le vecteur résultant reste dans la boule $\mathcal{B}(0, R) = \{y \in \mathbb{N}^q \mid \|y\|_1 \leq R\}$.

Question. Pourquoi est-on certain que l'algorithme $\mathcal{A}$ muni de la contrainte (C_R) se termine ? Cette contrainte garantit-elle les propriétés de correction et de complétude ?

5 Cas particulier d'un système réduit à une seule équation

Considérons le cas d'une équation unique :

$$a_{1,1}x_1 + a_{1,2}x_2 + \cdots + a_{1,q}x_q = 0.$$

L'entier relatif $a_{1,1}x_1 + \cdots + a_{1,q}x_q$ représente le *défaut* du vecteur $x = (x_1, \ldots, x_q)$:

— si ce défaut est nul, on détient une solution ;
— s'il est négatif, on essaie de l'augmenter ;
— s'il est positif, on essaie de le diminuer.

La contrainte géométrique appropriée est alors la suivante.

Définition 5.1 (Contrainte (C_1))**.** Étant donnés un vecteur $x \in \mathbb{N}^q$ et un vecteur e_j de la base canonique, le couple (x, e_j) vérifie la contrainte (C_1) si

$$a(x) \cdot a(e_j) < 0,$$

où $\cdot$ désigne le produit usuel sur $\mathbb{R}$.

Remarque 5.2. Même si (x, e_j) vérifie la contrainte (C_1), il n'est pas toujours vrai que le défaut de $x + e_j$ est plus petit en valeur absolue que celui de x. En d'autres termes, (C_1) oriente l'exploration dans la bonne direction *en signe*, mais ne garantit pas une décroissance du défaut à chaque pas.

Exemple. Considérons l'équation à $q = 2$ inconnues $x_1 - 10x_2 = 0$, et posons $x = (1, 0)$. Alors :

— $a(x) = 1$ et $a(e_2) = -10$, donc $a(x) \cdot a(e_2) = -10 < 0$: la contrainte (C_1) est bien vérifiée ;
— pourtant, $a(x + e_2) = a(1, 1) = 1 - 10 = -9$, soit $|a(x + e_2)| = 9 > 1 = |a(x)|$: le défaut a augmenté en valeur absolue.

Proposition 5.3 (Borne sur le défaut)**.** *Posons $M = \max_{1 \le j \le q} |a_j|$. Pour tout vecteur x engendré par l'algorithme $\mathcal{A}$ muni de la contrainte (C_1), on a*

$$|a(x)| \le M.$$

Esquisse de preuve. On raisonne par induction sur la construction des vecteurs dans L.

1. *Initialisation.* Les vecteurs de départ sont les e_j, et $|a(e_j)| = |a_j| \le M$.
2. *Hérédité.* Supposons $|a(x)| \le M$ et que (x, e_j) vérifie (C_1), c'est-à-dire $a(x) \cdot a(e_j) < 0$: les deux quantités sont de signes opposés. Pour deux réels u et v de signes opposés, on a

$$|u + v| = ||u| - |v|| \le \max(|u|, |v|).$$

Donc $|a(x + e_j)| = |a(x) + a(e_j)| \le \max(|a(x)|, |a(e_j)|) \le M$.

□

Corollaire 5.4. *Dans le cas d'une seule équation, les composantes de toute solution minimale sont bornées : il n'existe qu'un nombre fini de solutions minimales possibles. Combiné avec le lemme de Dickson (tout ensemble de q-uplets d'entiers naturels deux à deux incomparables est fini) et la condition d'élagage de l'algorithme, on en déduit que $\mathcal{A}$ termine.*

Remarque 5.5. Le lemme précédent ne borne pas directement les composantes x_j, car un défaut faible est compatible avec des composantes arbitrairement grandes (par exemple $x_1 - x_2 = 0$ admet (k, k) comme solution pour tout k). La borne effective sur les composantes résulte d'une analyse plus fine combinant la borne sur le défaut et la condition de minimalité ; elle est en général exponentielle en la taille des coefficients.

6 Algorithme de résolution équation par équation

Disposant d'un algorithme pour une seule équation, on peut en déduire un algorithme général par substitutions successives.

Notons $a^{(1)}(x) = 0, \ldots, a^{(p)}(x) = 0$ les p équations du système $a(x) = 0$, et notons $s_1, \ldots, s_\ell$ les solutions minimales de la première équation $a^{(1)}(x) = 0$ (obtenues par l'algorithme du § 5). Toute solution minimale du système complet est en particulier solution (peut-être non minimale) de cette première équation. Or, par définition d'une famille génératrice (Définition 3.1 appliquée à la première équation), elle s'écrit comme combinaison linéaire à coefficients dans $\mathbb{N}$ des s_k : elle est donc de la forme $\sum_k y_k s_k$ avec $y_k \in \mathbb{N}$. La résolution du système d'origine se ramène donc à celle du nouveau système à p équations et ℓ inconnues :

$$\sum_{k=1}^{\ell} y_k \, a(s_k) = 0.$$

Mais la première équation de ce nouveau système est triviale $(0 = 0)$; il s'agit donc en fait d'un système à $p - 1$ équations et ℓ inconnues. Il suffit alors de réitérer le processus.

Exemple 6.1. Considérons le système à 2 équations et 3 inconnues :

$$\begin{cases} x_1 - 2x_2 + x_3 = 0 \\ x_1 + x_2 - x_3 = 0. \end{cases}$$

Étape 1 : on vérifie que les solutions minimales de la première équation $x_1 - 2x_2 + x_3 = 0$ sont :

$$s_1 = (2, 1, 0), \quad s_2 = (0, 1, 2), \quad s_3 = (1, 1, 1).$$

Étape 2 : on pose $x = y_1 s_1 + y_2 s_2 + y_3 s_3$, soit :

$$x_1 = 2y_1 + y_3, \quad x_2 = y_1 + y_2 + y_3, \quad x_3 = 2y_2 + y_3.$$

En substituant dans la seconde équation $x_1 + x_2 - x_3 = 0$:

$$(2y_1 + y_3) + (y_1 + y_2 + y_3) - (2y_2 + y_3) = 3y_1 - y_2 + y_3 = 0.$$

On est ainsi ramené au système à une seule équation $3y_1 - y_2 + y_3 = 0$ dans $\mathbb{N}^3$, dont on peut calculer une famille génératrice minimale par les méthodes vues ci-dessus.

7 Algorithme global de résolution

L'idée consiste à généraliser directement l'algorithme présenté au § 5. Le principe est exactement le même, à ceci près que la contrainte unidirectionnelle (C_1) est remplacée par la contrainte (C_p) suivante, qui tient compte du fait que l'espace de recherche est cette fois de dimension p :

Définition 7.1 (Contrainte (C_p))**.** Étant donnés un vecteur $x \in \mathbb{N}^q$ et un vecteur e_j de la base canonique, le couple (x, e_j) vérifie la contrainte (C_p) si

$$\langle a(x),\, a(e_j) \rangle < 0,$$

où $\langle \cdot, \cdot \rangle$ désigne le produit scalaire usuel dans $\mathbb{R}^p$.

8 Propriétés des algorithmes

Les trois algorithmes décrits précédemment satisfont les propriétés suivantes :

— **Correction :** ils ne calculent que des solutions minimales ;

— **Complétude :** ils calculent *toutes* les solutions minimales ;

— **Terminaison :** ils s'arrêtent au bout d'un nombre fini d'étapes.

Remarque 8.1. La preuve de terminaison de l'algorithme du § 7 est délicate.

Remarque 8.2. Le problème que l'on cherche à résoudre est difficile en général. On pourrait se demander s'il est possible d'obtenir des bornes sur le nombre de solutions minimales ; l'exemple du système (S_4) (§ 3) n'est guère encourageant. De même, on pourrait essayer de borner la taille des solutions. De telles bornes ont été obtenues, mais elles sont toutes exponentielles. Par expérience, l'algorithme $\mathcal{A}$, dont la complexité reste inconnue à ce jour, est en général beaucoup plus efficace que l'algorithme naïf consistant à explorer exhaustivement un domaine calculé *a priori*.

Suggestions pour le développement

Il s'agit d'un menu à la carte : vous pouvez choisir d'étudier certains points, pas tous, pas nécessairement dans l'ordre, et de façon plus ou moins fouillée. Vous pouvez également vous poser d'autres questions que celles indiquées plus bas. Il est très vivement souhaité que vos investigations comportent une partie traitée sur ordinateur et, si possible, des représentations graphiques de vos résultats.

— Faire tourner l'algorithme $\mathcal{A}$ sur les systèmes (S_1) et (S_2) donnés au § 3 ; une exécution graphique par additions successives des vecteurs $a(e_j)$ sera appréciée.

— Programmer l'algorithme $\mathcal{A}$ dans sa version la plus générale (§ 7) et programmer l'algorithme de résolution équation par équation (§ 6).

— Proposer une interprétation géométrique des contraintes (C_1), puis (C_p), et justifier que ces contraintes n'empêchent pas d'assurer les propriétés de complétude et de correction.

— Dans le cas d'une seule équation (§ 5), prouver la terminaison de l'algorithme $\mathcal{A}$ et proposer une borne sur la taille des solutions.

— Caractériser géométriquement, à l'aide des vecteurs $a(e_j)$, les systèmes qui n'admettent pas de solution minimale.

— Commenter l'algorithme de résolution équation par équation (§ 6), en expliquant notamment pourquoi il est complet et à terminaison finie, et en décrire les inconvénients prévisibles.

— Proposer une extension de l'algorithme $\mathcal{A}$ pour résoudre un système linéaire *non homogène* en entiers naturels (second membre $(b_1, \ldots, b_p) \in \mathbb{Z}^p$). On commencera par préciser la structure de l'ensemble des solutions.

— Proposer une amélioration de l'algorithme $\mathcal{A}$ afin d'éviter d'engendrer le même q-uplet de plusieurs façons différentes.

— Prouver tout ou partie de la proposition énoncée au § 3.

— Montrer que la Proposition 5.3, combinée aux propriétés de terminaison et de complétude de l'algorithme $\mathcal{A}$, fournit une preuve constructive de la Proposition 3.5 : puisque l'algorithme termine en produisant un ensemble fini M de solutions minimales engendrant toutes les solutions, la finitude de la famille génératrice minimale en est un corollaire direct, sans argument algébrique indépendant.