COURS 27

Solutions rationnelles de systèmes linéaires à coefficients polynomiaux

Résumé

L'algorithmique des systèmes linéaires à coefficients des polynômes en une variable est très similaire à celle des fractions rationnelles. En outre, il est important de tirer parti du produit rapide de matrices.

On considère le système linéaire

$$A(x)Y(x) = B(x), \tag{1}$$

où A et B sont donnés et Y est inconnu. A est une matrice $n \times n$ de polynômes, régulière (de déterminant non nul) et B est un vecteur de polynômes. De manière équivalente, on peut voir A (ou B) comme un polynôme à coefficients des matrices (ou des vecteurs). Pour fixer les notations, on suppose $\deg A \leq d$ et $\deg B < d$.

On notera $\mathcal{M}_{m,k}(R)$ l'ensemble des matrices $m \times k$ (m lignes et k colonnes) dont les coefficients sont dans R. On notera $\mathbb{K}[x]_d$ l'ensemble des polynômes de degré au plus d à coefficients dans le corps $\mathbb{K}$. La fonction M est telle que la multiplication dans $\mathbb{K}[x]_d$ coûte au plus $M(d)$ opérations dans $\mathbb{K}$. L'exposant ω est tel que la multiplication de deux matrices $n \times n$ à coefficients dans k coûte au plus n^ω opérations dans k.

Nous aurons aussi besoin de produits de matrices de $\mathcal{M}_{n,n}(\mathbb{K}[x]_d)$. Dans le cas le plus général, ce produit est connu pour pouvoir être executé en $O(n^\omega M(d))$ opérations dans $\mathbb{K}$, borne que nous utiliserons dans les estimations de complexité. Lorsque le corps $\mathbb{K}$ contient suffisamment de points, par évaluation-interpolation, ce coût descend à $O(n^\omega d + n^2 M(d) \log d)$; dans les mêmes conditions, en choisissant des points en progression géométrique, cette complexité a été abaissée récemment à $O(n^\omega d + n^2 M(d))$, voir [1].

1. Des séries aux solutions rationnelles

Une première observation est que la structure de la solution cherchée est donnée par les formules de Cramer.

Lemme 1. *Le système possède une solution dont les coordonnées sont des fractions rationnelles. Ses numérateurs et dénominateurs ont degrés bornés par $nd-1$ et nd.*

Démonstration. Le système (1) se récrit

$$A_1 y_1 + \cdots + A_n y_n = B,$$

où y_i est la $i$ème coordonnée de Y et A_i la $i$ème colonne de A. La formule de Cramer

$$\det(A_1, \ldots, A_{i-1}, B, A_{i+1}, \ldots, A_n) = y_i \det(A),$$

est obtenue en remplaçant B par sa valeur dans le membre gauche et en développant le déterminant.

Il en découle que y_i est le quotient de déterminants de matrices dans $\mathcal{M}_{n,n}(\mathbb{K}[x]_d)$. Ces déterminants ont donc degré au plus $nd-1$ pour le numérateur et nd pour le dénominateur. □

L'algorithme de résolution suivant est justifié par la complexité quasi-optimale des approximants de Padé.

Résolution de $A(x)Y(x) = B(x)$ [Moenk-Carter 1979]
Entrée : $A \in \mathcal{M}_{n,n}(\mathbb{K}[x]_d)$, $B \in \mathcal{M}_{n,1}(\mathbb{K}[x]_{d-1})$
Sortie : le vecteur de fraction rationnelles Y tel que $AY = B$.

1. Calculer le développement en série de $A^{-1}B$ à précision $2nd$;
2. Reconstruire les coefficients de Y par approximant de Padé.

Dans tous les algorithmes qui vont suivre, c'est la recherche de série solution qui va dominer la complexité.

2. L'algorithme de Newton

La méthode de Newton que nous avons employée pour de nombreux calculs rapides sur les séries s'applique encore lorsque les séries ont des matrices pour coefficients.

LEMME 2. *Soit $A(x)$ une matrice $n \times n$ de séries formelles telle que $A(0)$ est inversible. Alors l'itération*

$$Y_{k+1} = Y_k + Y_k(I - AY_k) \bmod x^{2^{k+1}}$$

avec $Y_0 = A(0)^{-1}$ converge vers A^{-1} avec $A^{-1} - Y_k = O(x^{2^k})$.

L'itération de l'étape k utilise deux produits de matrices $n \times n$ de polynômes de degré au plus 2^{k+1} et a donc complexité $O(n^\omega M(2^{k+1}))$. Le lemme de complexité du diviser-pour-régner et les hypothèses habituelles sur la fonction M donnent alors la proposition suivante.

PROPOSITION 1. *Avec les hypothèses du lemme prédédent, on peut calculer N termes de la série A^{-1} en $O(n^\omega M(N))$ opérations dans $\mathbb{K}$.*

DÉMONSTRATION. [du lemme] Pour $k = 0$, le lemme découle du choix de Y_0. Si la propriété est acquise pour k, alors $I - AY_k = O(x^{2^k})$. Ensuite,

$$\begin{aligned} I - AY_{k+1} &= I - AY_k - AY_k(I - AY_k) \bmod x^{2^{k+1}} \\ &= (I - AY_k)^2 \bmod x^{2^{k+1}} \\ &= O(x^{2^{k+1}}), \end{aligned}$$

d'où la propriété voulue pour $k+1$ en multipliant à gauche par A^{-1}. □

Avec cet algorithme, le calcul de la fraction rationnelle solution de (1) demande $O(n^\omega M(nd))$ opérations dans $\mathbb{K}$, résultat que nous allons améliorer dans les sections qui suivent.

3. Développement comme une fraction rationnelle

La méthode de Newton fonctionne pour toute matrice inversible de séries. Il est possible d'améliorer l'efficacité lorsque la matrice est une matrice de polynômes. La base de cette amélioration est contenue dans le lemme suivant.

LEMME 3. *Soit $A(x) \in \mathcal{M}_{n,n}(\mathbb{K}[x]_d)$ et $B(x) \in \mathcal{M}_{n,m}(\mathbb{K}[x]_{d-1})$, avec A inversible. Pour tout k, il existe une matrice $B_k \in \mathcal{M}_{n,m}(\mathbb{K}[x]_{d-1})$ telle que*

$$A^{-1}B = a_0 + a_1 x + \cdots + a_{k-1}x^{k-1} + x^k A^{-1} B_k, \tag{2}$$

où $a_i \in \mathcal{M}_{n,m}(\mathbb{K})$.

DÉMONSTRATION. Si les a_i sont les coefficients du développement en série de $A^{-1}B$, alors

$$B - A(a_0 + \cdots + a_{k-1}x^{k-1})$$

est une matrice de $\mathcal{M}_{n,m}(\mathbb{K}[x]_{d+k-1})$ qui, par construction, est divisible par x^k, d'où le lemme. □

Ce résultat se traduit algorithmiquement comme suit.

Développement de $A^{-1}B$
Entrée : A, B, k et $S = A^{-1} \bmod x^k$;
Sortie : $a_0, \ldots, a_{k-1}$ et B_k définis par l'équation (2).

1. Calculer $SB =: a_0 + \cdots + a_{k-1}x^{k-1} \bmod x^k$;
2. Calculer $B_k = (B - A(a_0 + \cdots + a_{k-1}x^{k-1}))x^{-k}$.
3. Renvoyer $a_0, \ldots, a_{k-1}, B_k$.

PROPOSITION 2. *Soit $A \in \mathcal{M}_{n,n}(\mathbb{K}[x]_d)$ avec $A(0)$ inversible, alors on peut calculer les $N \geq d$ premiers coefficients de A^{-1} en $O(n^\omega NM(d)/d)$ opérations dans $\mathbb{K}$. Pour $B \in \mathcal{M}_{n,1}(\mathbb{K}[x]_{d-1})$, on peut calculer les $N \geq d$ premiers coefficients de $A^{-1}B$ en $O(n^2 NM(d)/d)$ opérations dans $\mathbb{K}$.*

DÉMONSTRATION. L'algorithme calcule d'abord $S = A^{-1} \bmod x^d$ en $O(n^\omega M(d))$ opérations dans $\mathbb{K}$ par l'algorithme de Newton de la première section.

Ensuite, on applique N/d fois l'algorithme ci-dessus avec $k = d$ pour calculer à chaque itération d coefficients et un nouveau $B_{(i+1)d}$. Si on part de $B_0 = I$, le résultat fournit les N premiers coefficients de A^{-1} ; si $B_0 = B$, alors on obtient les coefficients de $A^{-1}B$.

Les deux étapes de l'algorithme ci-dessus (avec $k = d$) coûtent $O(n^\omega M(d))$ opérations dans $\mathbb{K}$ si B a n colonnes, $O(n^2 M(d))$ s'il n'en a qu'une. □

Avec cet algorithme, le calcul de la fraction rationnelle solution de (1) demande $O(n^3 M(d))$ opérations dans $\mathbb{K}$.

4. L'algorithme de Storjohann

L'algorithme de Storjohann permet de calculer les N premiers coefficients du développement en série de $A^{-1}B$ en $O(n^{\omega-1} N \log N M(d))$ opérations dans $\mathbb{K}$. Lorsque $\omega < 3$, cette quantité croît avec n moins vite que la taille de l'inverse A^{-1}, qui n'est donc pas calculée en entier. Ainsi, la résolution du système linéaire (1) a un coût en $O(n^\omega M(d) \log(nd))$ opérations dans $\mathbb{K}$.

L'algorithme repose sur le développement de $A^{-1}B$ de la section précédente, joint d'une part à une technique de diviser-pour-régner, d'autre part au regroupement des calculs intermédiaires pour remplacer des groupes de n produits matrice-vecteur par des produits matrice-matrice.

Pour commencer, on peut modifier l'algorithme de développement de $A^{-1}B$ de la section précédente pour calculer B_k sans calculer tous les coefficients $a_0, \ldots, a_{k-1}$. L'entrée nécessaire est moins grosse, et la complexité plus faible lorsque k est grand devant d. Pour une série $V = v_0 + v_1 x + \cdots$, on note

$$[V]_a^b := v_a x^a + \cdots + v_{a+b} x^{a+b},$$

avec la convention que les coefficients d'indice négatif de V sont nuls.

Développement de $A^{-1}B$ tronqué
Entrée : A, B, k et $S = [A^{-1}]_{k-2d+1}^{2d-2}$;
Sortie : B_k défini par l'équation (2).

1. Calculer $U := [SB]_{k-d}^{d-1}$;
2. Calculer $B_k := [B - AU]_k^{d-1} x^{-k}$.
3. Renvoyer B_k.

DÉMONSTRATION. Pour prouver la correction de cet algorithme, il faut s'assurer que les troncatures de séries sont suffisantes pour calculer le même polynôme B_k que précédemment.

Pour la première étape de l'algorithme, il suffit d'observer que B ayant degré au plus $d-1$, le coefficient de x^i dans $A^{-1}B$ ne dépend que de $[A^{-1}]_{i-d-1}^{d+1}$, pour tout i. Donc les coefficients calculés par cette première étape sont les mêmes que les a_i que précédemment, pour $i = k-d, \ldots, k-1$.

Ensuite, il faut calculer $[x^k B_k]_k^{d-1}$. L'extraction des coefficients de l'équation (2) donne

$$\begin{aligned}[x^k B_k]_k^{d-1} &= [B - A(a_0 + \cdots + a_{k-1}x^{k-1})]_k^{d-1} \\ &= [B - A(a_{k-d}x^{k-d} + \cdots + a_{k-1}x^{k-1})]_k^{d-1},\end{aligned}$$

ce qui conclut la preuve. □

Une observation importante concernant cet algorithme est que c'est le même polynôme S qui peut servir pour calculer B_{i+k} pour tout i. L'idée est alors de grouper plusieurs vecteurs B_i et de calculer les B_{i+k} correspondants par des produits matrice-matrice. Ainsi, si l'on connaît $B_0, B_{2m}, \ldots, B_{2sm}$ et $[A^{-1}]_{m-2d}^{2d}$, alors le calcul de $B_m, B_{3m}, \ldots, B_{(2s+1)m}$ ne requiert que $O(n^{\omega-1} s M(d))$ opérations dans $\mathbb{K}$.

Itérant cette idée, partant de B_0 et supposant connus $[A^{-1}]_{2^k-2d+1}^{2d-2}$ pour $k = 0, \ldots, \lceil \log(N/d) \rceil =: k_{\max}$, on obtient d'abord $B_0, B_{2^{k_{\max}}}$, puis à l'étape suivante $B_0, B_{2^{k_{\max}-1}}, B_{2^{k_{\max}}}, B_{3.2^{k_{\max}-1}}$, et ainsi de suite jusqu'à $B_0, B_d, B_{2d}, \ldots, B_{d\lceil N/d \rceil}$ en $O(k_{\max} n^{\omega-1} N M(d)/d)$ opérations dans k. En multipliant alors ces vecteurs par $[A^{-1}]_0^d$ on obtient finalement les N premiers coefficients de $A^{-1}B$ pour le même coût.

Il reste à voir comment calculer les $[A^{-1}]_{2^k-2d-1}^{2d-2}$. Là encore, le point de départ est l'identité (2), avec $B = I$, $k = m$ et $k = p$:

$$\begin{aligned} A^{-1} &= a_0 + \cdots + a_{m-1}x^{m-1} + x^m A^{-1} B_m \\ &= a_0 + \cdots + a_{m-1}x^{m-1} + x^m(a_0 + \cdots + a_{p-1}x^{p-1} + x^p A^{-1} B_p)B_m. \end{aligned}$$

La seconde ligne est obtenue par substitution de la valeur de A^{-1} donnée par la première ligne avec $m = p$. Ces équations entraînent pour tout $\ell \geq 0$ tel que $m + p - \ell \geq d$

$$\begin{cases} B_{m+p} & = [-A[A^{-1}]_{p-2d+1}^{2d-2} B_m]_p^{d-1} x^{-p}, \\ [A^{-1}]_{m+p-\ell}^{\ell-1} & = [[A^{-1}]_{p-\ell-d+1}^{\ell+d-2} B_m]_{m+p-\ell}^{\ell-1}. \end{cases}$$

L'algorithme suivant s'en déduit en utilisant cette identité avec $m = 2^k - d$, $p = 2^k$ et $\ell = d - 1$.

Développement de A^{-1} — indices puissances de 2
Entrée : $S = [A^{-1}]_0^{d-1}$,$T = [A^{-1}]_{2^k-2d+1}^{2d-2}$ et B_{2^k-d} défini par (2) avec $B = I$;
Sortie : $B_{2^{k+1}-d}$ et $[A^{-1}]_{2^{k+1}-2d+1}^{2d-2}$.

1. Calculer $[A^{-1}]_{2^{k+1}-2d+1}^{d-2} = [TB_{2^k-d}]_{2^{k+1}-2d+1}^{d-2}$;
2. Calculer $B_{2^{k+1}-d} := [-ATB_{2^k-d}]_{2^k}^{d-1}/x^{2^k}$;
3. Calculer $[A^{-1}]_{2^{k+1}-d}^{d-1} = [x^{2^{k+1}-d} B_{2^{k+1}-d} S]_{2^{k+1}-d}^{d-1}$;
4. Renvoyer $B_{2^{k+1}-d}$ et $[A^{-1}]_{2^{k+1}-2d+1}^{2d-2}$.

Pour résumer, ces algorithmes mènent au résultat suivant.

Théorème 1 (Storjohann 2002). *Soit A une matrice $n \times n$ polynomiale de degré d avec $A(0)$ inversible et B un vecteur polynomial de degré au plus $d - 1$, alors on peut calculer le numérateur et le dénominateur de $A^{-1}B$ en $O(n^\omega M(d))$ opérations dans $\mathbb{K}$.*

Notes

Dans tout ce texte, nous avons supposé que $A(0)$ est inversible. En fait, ces résultats s'étendent au cas où A est inversible sans que $A(0)$ le soit. Il suffit de tirer aléatoirement un point et d'effectuer les développements en série au voisinage de ce point. L'algorithme devient probabiliste. Si la caractéristique est positive, il se peut que A soit inversible sans que sa valeur en aucun point du corps ne le soit. On peut alors construire une extension du corps assez grande pour trouver un point ou effectuer ces calculs. Ces considérations sont prises en compte dans [3].

Bibliographie

[1] Bostan (Alin) and Schost (Éric). – Polynomial evaluation and interpolation on special sets of points. *Journal of Complexity*, vol. 21, n° 4, 2005, pp. 420–446. – Festschrift for Arnold Schönhage.

[2] Moenck (Robert T.) and Carter (John H.). – Approximate algorithms to derive exact solutions to systems of linear equations. In *EUROSAM '79 : Proceedings of the International Symposiumon on Symbolic and Algebraic Computation. Lecture Notes in Computer Science*, vol. 72, pp. 65–73. – Springer-Verlag, London, UK, 1979.

[3] Storjohann (Arne). – High-order lifting. In Mora (Teo) (editor), *ISSAC'2002*. pp. 246–254. – ACM Press, July 2002. Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation, July 07–10, 2002, Université de Lille, France.