COURS 11

Algorithme rapide pour le pgcd

1. Algorithme d'Euclide rapide

Les méthodes à base d'approximants de Padé-Hermite ne peuvent remplacer l'algorithme d'Euclide que dans le cas du calcul de pgcd de polynômes. Historiquement, c'est d'abord un algorithme rapide pour le pgcd d'entiers (le « demi-pgcd ») qui est apparu, il a ensuite été étendu aux polynômes, et les algorithmes rapides pour les approximants de Padé-Hermite sont de conception plus récente. Nous décrivons maintenant l'algorithme de « demi-pgcd ». Pour simplifier, nous nous plaçons dans le cas du pgcd de polynômes, mais cette approche s'étend au cas des entiers.

Soient donc A et B dans $\mathbb{K}[X]$, avec $n = \deg A$ et $\deg B < n$ (ce qui n'est pas une forte restriction). Posons comme précédemment $R_0 = A$, $R_1 = B$. Soient ensuite R_i les restes successifs de l'algorithme d'Euclide et soit en particulier $R_N = \mathrm{pgcd}(A, B)$ le dernier non nul d'entre eux. On sait qu'il existe une matrice 2×2 $M_{A,B}$ de cofacteurs telle que :

$$M_{A,B} \begin{bmatrix} R_0 \\ R_1 \end{bmatrix} = \begin{bmatrix} U_N & V_N \\ U_{N+1} & V_{N+1} \end{bmatrix} \begin{bmatrix} R_0 \\ R_1 \end{bmatrix} = \begin{bmatrix} R_N \\ 0 \end{bmatrix}.$$

L'algorithme de pgcd rapide calcule d'abord cette matrice ; à partir de là, on en déduit aisément le pgcd, pour $O(\mathsf{M}(n))$ opérations supplémentaires.

À titre préparatoire, notons qu'une étape de l'algorithme d'Euclide classique peut elle aussi s'écrire sous une forme matricielle. En effet, si Q_{i+1} est le quotient de la division de R_{i-1} par R_i, on peut écrire :

$$\begin{bmatrix} 0 & 1 \\ 1 & -Q_{i+1} \end{bmatrix} \begin{bmatrix} R_{i-1} \\ R_i \end{bmatrix} = \begin{bmatrix} R_i \\ R_{i+1} \end{bmatrix}.$$

Ainsi, la matrice $M_{A,B}$ est un produit de matrices élémentaires de ce type. Elle est inversible.

Demi-pgcd : définition. Comme étape intermédiaire pour obtenir une division euclidienne rapide, on utilise l'algorithme dit du « demi-pgcd » (*half-gcd* en anglais, d'où la notation *HGCD*), qui permet de faire des « pas de géant » dans la liste des restes successifs.

Le demi-pgcd est défini comme suit. Les degrés des restes successifs R_i décroissent strictement, donc il existe un unique indice j tel que :

- $\deg R_j \geq \frac{n}{2}$;
- $\deg R_{j+1} < \frac{n}{2}$.

Euclide rapide via demi-pgcd

Entrée : $R_0 = A$ et $R_1 B$ dans $\mathbb{K}[X]$ avec $\deg B < \deg A = n$.
Sortie : La matrice $M_{A,B}$.

1. Soit $M_{\text{hgcd}} = \mathsf{HGCD}(A, B)$.
2. Calculer R_j et R_{j+1}.
3. Si $R_{j+1} = 0$, renvoyer M_{hgcd}.
4. Soit M la matrice de la division euclidienne de R_j par R_{j+1}, et R_{j+2} le reste correspondant.
5. Si $R_{j+2} = 0$, renvoyer MM_{hgcd}.
6. Calculer $M_{R_{j+1},R_{j+2}}$ par un appel récursif, et renvoyer $M_{R_{j+1},R_{j+2}} M M_{\text{hgcd}}$.

FIG. 1.

On a vu plus haut (définition de l'algorithme étendu) qu'il existe $U_j, V_j, U_{j+1}, V_{j+1}$ tels que :

$$U_j R_0 + V_j R_1 = R_j \quad \text{et} \quad U_{j+1} R_0 + V_{j+1} R_1 = R_{j+1},$$

ces polynômes étant en outre uniques si on rajoute les conditions de degré voulue sur la relation de Bézout. Ceci peut se réécrire sous la forme matricielle suivante :

$$\begin{bmatrix} U_j & V_j \\ U_{j+1} & V_{j+1} \end{bmatrix} \begin{bmatrix} R_0 \\ R_1 \end{bmatrix} = \begin{bmatrix} R_j \\ R_{j+1} \end{bmatrix}.$$

Pour fixer les idées, en première approximation, on peut estimer que les polynômes $U_j, V_j, U_{j+1}, V_{j+1}$ ont des degrés de l'ordre de $\frac{n}{2}$ (attention, ce n'est qu'une estimation, pas une égalité).

Notons M_{hgcd} la matrice ci-dessus donnant les restes R_j et R_{j+1}. L'algorithme du demi-pgcd (noté HGCD ci-dessous) a pour vocation de calculer cette matrice. Avant d'en étudier le fonctionnement, voyons comment il permet d'obtenir le calcul du pgcd étendu. L'idée est qu'un appel à HGCD en degré n permet d'obtenir les restes de degré approximativement $\frac{n}{2}$; alors, un appel en degré $\frac{n}{2}$ permet d'obtenir les restes de degré approximativement $\frac{n}{4}$, et ainsi de suite jusqu'à trouver le pgcd. L'algorithme est détaillé en Figure 1.

À l'étape 4, R_j et R_{j+1} ont par définition des degrés qui encadrent $\frac{n}{2}$, mais on n'a pas de borne supérieure sur le degré de R_j, qui peut être proche de n. Aussi, pour obtenir deux polynômes de degré au plus $\frac{n}{2}$ pour l'appel récursif, il est nécessaire d'effectuer cette étape de division euclidienne.

Soit $\mathsf{H}(n)$ la complexité de l'algorithme de demi-pgcd sur des entrées de degré au plus n. Pour estimer la complexité de l'algorithme de pgcd rapide, on fait l'hypothèse que $\mathsf{H}(n+n') \geq \mathsf{H}(n) + \mathsf{H}(n')$ et que $\mathsf{M}(n)$ est négligeable devant $\mathsf{H}(n)$. Ces hypothèses sont vérifiées pour l'algorithme proposé plus loin.

PROPOSITION 1. *L'algorithme ci-dessus calcule $M_{A,B}$ en $O(\mathsf{H}(n))$ opérations.*

DÉMONSTRATION. La validité de l'algorithme est immédiate, puisque toutes les matrices calculées ne sont au fond que des matrices de passage, qui font passer de deux restes successifs à deux autres restes successifs. Multiplier ces matrices permet de composer ces opérations de transition.

Estimons la complexité. Le coût de l'appel à HGCD est $\mathsf{H}(n)$. Ensuite, toutes les opérations des étapes 2 à 5 ont une complexité en $O(\mathsf{M}(n))$, en utilisant une division euclidienne rapide pour l'étape 4. On effectue ensuite un appel récursif, puis de nouveau des opérations dont le coût est en $O(\mathsf{M}(n))$. Ainsi, la complexité $\mathsf{B}(n)$ de l'algorithme de pgcd rapide satisfait la récurrence :

$$\mathsf{B}(n) \leq \mathsf{B}(n/2) + \mathsf{H}(n) + C\mathsf{M}(n),$$

C étant une constante. Le lemme « diviser pour régner » permet de conclure. □

Autrement dit, le coût du pgcd est, à une constante près, le même que celui du demi-pgcd. Il devient donc légitime de se consacrer à ce dernier uniquement.

Demi-pgcd : algorithme. Pour mettre en place une stratégie « diviser pour régner » pour le demi-pgcd, on utilise un moyen de « couper les polynômes » en deux. L'idée qui fonctionne est de ne garder que les coefficients de poids forts des polynômes d'entrée et d'utiliser le fait que *le quotient de la division euclidienne de deux polynômes ne dépend que de leurs coefficients de poids fort*, d'une façon tout à fait quantifiée par la suite.

Remarquons qu'on accède aux coefficients de poids fort d'un polynôme en prenant son quotient par un polynôme de la forme X^k : par exemple, si A est

$$X^{10} + 2X^9 + 5X^8 + 3X^7 + 11X^6 + \cdots,$$

le quotient de A par X^6 est

$$X^4 + 2X^3 + 5X^2 + 3X + 11.$$

Voyons comment cette idée permet d'obtenir notre algorithme. Partant de deux polynômes A et B de degrés de l'ordre de n, on leur associe f et g de degré approximativement $\frac{n}{2}$, en nommant f et g les quotients respectifs de A et B par $X^{\frac{n}{2}}$: on a jeté les coefficients de petits degrés.

On calcule la matrice M du demi-pgcd de f et g (moralement, les entrées de cette matrice ont degré $\frac{n}{4}$). La remarque ci-dessus permet de montrer qu'on obtient ainsi une matrice de passage pour les restes de A et B eux-mêmes

On applique donc cette matrice aux polynômes initiaux ; on obtient des restes H et I dont les degrés sont de l'ordre de $\frac{3n}{4}$. On effectue alors une seconde fois le même type d'opération, avec un appel récursif en degré $\frac{n}{2}$, pour gagner à nouveau $\frac{n}{4}$, et finalement arriver en degré $\frac{n}{2}$.

Les choix exacts des degrés de troncature sont subtils, et on admettra que les valeurs données en Figure 2 permettent d'assurer la correction de l'algorithme.

PROPOSITION 2. *L'algorithme ci-dessus calcule la matrice du demi-pgcd de A et B en $O(\mathsf{M}(n)\log(n))$ opérations dans $\mathbb{K}$.*

DÉMONSTRATION. Comme indiqué plus haut, on admet que les choix des degrés de troncature permettent d'assurer la validité de l'algorithme. Il est plus facile d'en effectuer l'analyse de complexité. Le coût $\mathsf{H}(n)$ peut être majoré par deux fois le coût en degré au plus $\frac{n}{2}$ (les deux appels ont lieu aux lignes 3 et 7), plus un certain nombre de multiplications de polynômes et une division euclidienne. En remarquant que tous les produits se font en degré au pire n, on en déduit la récurrence

$$\mathsf{H}(n) \leq 2\mathsf{H}(\frac{n}{2}) + C\mathsf{M}(n),$$

HGCD

Entrée : A et B dans $\mathbb{K}[X]$, avec $\deg B < \deg A = n$.
Sortie : La matrice M_{hgcd} de A et B.

1. Soit $m = \lceil \frac{n}{2} \rceil$. Si $\deg B < m$, renvoyer la matrice identité.
   C'est terminé.
2. Soient f et g les quotients respectifs de A et B par X^m.
   f et g ont des degrés approximativement $n/2$.
3. Soit $M = \mathsf{HGCD}(f, g)$, et A', B' définis par
$$M \begin{bmatrix} A \\ B \end{bmatrix} = \begin{bmatrix} A' \\ B' \end{bmatrix}.$$
4. Si $\deg B' < m$, renvoyer M.
   C'est terminé.
5. Soient $H = B', I = A' \bmod B'$, et M' la matrice de la division euclidienne correspondante.
   Moralement, B' a un degré de l'ordre au pire de $\frac{3n}{4}$. Par contre, A' peut avoir un degré plus grand. L'étape de division euclidienne permet de se ramener à des polynômes de degrés de l'ordre de $\frac{3n}{4}$.
6. Soit $\ell = 2m - \deg H$, h et i les quotients respectifs de H et I par X^ℓ.
   ℓ est de l'ordre de $\frac{n}{4}$, de sorte de h et i on des degrés de l'ordre $\frac{n}{2}$.
7. Soit $M'' = \mathsf{HGCD}(h, i)$. Renvoyer la matrice $M''M'M$.

FIG. 2.

où C est une constante. La conclusion découle comme d'habitude du lemme « diviser pour régner ». □

Complément : calcul d'un reste choisi. On peut en tirer un raffinement fort utile de l'algorithme de demi-pgcd : le calcul d'un reste particulier (sélectionné par une condition de degré), ainsi que des cofacteurs associés.

PROPOSITION 3. *Soient $R_0 = A$ et $R_1 = B$, avec $\deg A = n$ et $\deg B < \deg A$, et soient R_i les restes successifs de l'algorithme d'Euclide appliqué à A et B.*

Soit $\ell < n$, et R_j le premier reste de degré inférieur à ℓ. On peut calculer R_j et les cofacteurs associés U_j et V_j en $O(\mathsf{M}(n)\log(n))$ opérations de $\mathbb{K}$.

DÉMONSTRATION (SUCCINTE). Si ℓ est plus petit que $\frac{n}{2}$, on fait un appel à HGCD pour se ramener en degré $\frac{n}{2}$, et on effectue un appel récursif. Si ℓ est plus grand que $\frac{n}{2}$, on fait un appel à HGCD sur les polynômes divisés par $X^{n-2\ell}$. □

Algorithmes sur les entiers. Les mêmes idées algorithmiques s'étendent au calcul sur les entiers, mais il est beaucoup plus difficile d'écrire un algorithme correct dans ce cas. On se contentera d'énoncer les résultats de complexité suivants :

THÉORÈME 1. *Soient $R_0 = A \in \mathbb{N}$, $R_1 = B \in \mathbb{N}$, avec $B \leq A$ et R_i les restes de l'algorithme d'Euclide appliqué à A et B. On peut calculer :*

– *le pgcd de A et B,*

– *le premier reste inférieur à un entier $\ell < A$,*

ainsi que les cofacteurs associés en $O(\mathsf{M}_{\mathbb{Z}}(\log A)\log(\log(A)))$ opérations binaires.

L'analogue de la reconstruction rationnelle est également calculable dans la même complexité. Si n est un entier positif, et A un entier compris entre 0 et $n-1$, et étant donné $m \leq n$, il s'agit de calculer des entiers U et V, avec $|U| < m$ et $0 \leq V \leq \frac{n}{m}$ tels que

$$\text{(1)} \qquad \operatorname{pgcd}(n, V) = 1 \quad \text{et} \quad A = \frac{U}{V} \mod n.$$

Notes

Il est très difficile de trouver une référence complète, lisible et sans erreur sur les algorithmes de pgcd rapide. Le chapitre 11 de [3] fournit une bonne partie des résultats techniques implicitement utilisés dans la section 1, mais l'algorithme qu'il propose est erroné. Les notes du cours [4] nous ont beaucoup inspiré, mais elles contiennent des erreurs pour le pgcd entier. On pourra également consulter des articles plus récents [1, 2], mais rien ne garantit qu'ils soient libres d'erreurs.

Ces algorithmes de pgcd rapide sont aussi assez délicats à implanter de manière efficace. Par exemple, pour le pgcd de polynômes, une bonne implantation doit prendre en compte les algorithmes de multiplication utilisés. Si on utilise la FFT, il est possible d'économiser des transformées de Fourier directes et inverses ; il faut alors régler le nombre de points d'évaluation en fonction des degrés du résultat final, et pas des résultats intermédiaires, etc. Si l'on travaille sur un corps fini « raisonnable » (les coefficients faisant de quelques bits jusqu'à quelques dizaines voire centaines de bits), on peut estimer que le seuil au-delà duquel utiliser l'algorithme rapide se situe autour des degrés 200 ou 300. Il faut noter que très peu de systèmes disposent de telles implantations (c'est le cas pour Magma et la bibliothèque NTL). En ce qui concerne le pgcd des entiers, la situation est sensiblement plus délicate, toujours à cause du problème des retenues (une bonne partie des algorithmes présentés dans les ouvrages de référence sont incorrects, à cause de ce problème). Pour donner une idée, dans les systèmes Magma, Mathematica, ou des bibliothèques telles que GMP (code toujours en développement et utilisé entre autres par les entiers de Maple), le seuil se situe autour de nombres de 30 000 bits (10 000 chiffres décimaux).

Bibliographie

[1] Pan (V. Y.) and Wang (X.). – Acceleration of Euclidean algorithm and extensions. In Mora (Teo) (editor), *ISSAC'2002*. pp. 207–213. – ACM, New York, 2002. Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation, July 07–10, 2002, Université de Lille, France.

[2] Stehlé (D.) and Zimmermann (P.). – A binary recursive Gcd algorithm. In *ANTS-VI. Lecture Notes in Computer Science*, vol. 3076, pp. 411–425. – Springer, 2004.

[3] von zur Gathen (Joachim) and Gerhard (Jürgen). – *Modern computer algebra*. – Cambridge University Press, New York, 2003, 2nd edition, xiv+785p.

[4] Yap (Chee). – *Fundamental Problems in Algorithmic Algebra*. – Oxford University Press, 2000.