ÉLÉMENT DE PORTFOLIO 04

Publication

1 DÉFINITION DE CET ÉLÉMENT

Titre de l'élément : Faster Change of Order Algorithm for Gröbner Bases under Shape and Stability Assumptions

URL de l'élément : `https://dl.acm.org/doi/10.1145/3476446.3535484`

2 MOTIVATIONS DU CHOIX DE CET ÉLÉMENT

Cet article [2], publié à la conférence ISSAC 2022 (International Symposium on Symbolic and Algebraic Computation), qui est la conférence de référence pour le calcul formel, propose un changement de paradigme pour le changement d'ordre monomial pour les bases de Gröbner, sous certaines hypothèses de *généricité*. La stratégie est d'utiliser la structure de module sur l'anneau des polynômes en une variable au lieu de la simple structure d'espace vectoriel sur le corps des coefficients. L'entrée est une matrice de taille $D \times D$ et de densité $t/D \leq 1$. Nous obtenons une complexité de l'ordre de $t^{\omega-1}D$, où ω est un exposant pour le produit de matrices. Ceci améliore la complexité de l'état de l'art d'un facteur qui est le minimum entre $(\frac{D}{t})^{\omega-1}$ et $\frac{D}{t}t^{3-\omega}$, tous deux ≥ 1 et exponentiels en le nombre de variables dans les situations génériques. Les gains de performance observés via une implémentation prototype sont significatifs sur des cas classiques.

3 PRÉSENTATION DE CET ÉLÉMENT

La résolution de systèmes polynomiaux zéro-dimensionnels (avec un nombre fini de solutions dans une clôture algébrique du corps engendré par les coefficients), par calcul de bases de Gröbner se décompose classiquement en deux étapes. Un premier calcul de base de Gröbner, pour un ordre du dégré, est tout d'abord effectué en utilisant l'algorithme de Buchberger [3] ou les algorithmes F_4 et F_5 de Faugère [4,5]. Ensuite, un algorithme de conversion, dit de changement d'ordre, est appliqué à la base de Gröbner donnée en entrée pour obtenir une base de Gröbner pour l'ordre lexicographique. Pour des systèmes génériques, cette seconde étape devient prépondérante sur la première lorsque la taille des problèmes grossit.

Or, à l'instar de l'élimination de Gauß pour les systèmes linéaires, les bases de Gröbner pour l'ordre lexicographique permettent aisément de calculer les coordonnées des solutions. Une telle base contient un polynôme non nul purement en la dernière variable, ce qui permet de calculer la dernière coordonnée de chaque solution. Puis, elle contient des polynômes en les deux dernières variables, ce qui permet de déterminer l'avant-dernière coordonnée en fonction de la dernière, et ainsi de suite. On peut citer pour cette étape, les algorithmes FGLM [6], de Neiger et Schost [12] ou SPARSE-FGLM de Faugère et Mou [7,8]. Dans le cas *générique*, l'entrée est une matrice de taille D et de densité t/D. Dans ce cas, d'une part, l'algorithme de [12] a une complexité quasi-linéaire en D^ω, où $2 \leq \omega \leq 3$ est un exposant pour la multiplication de matrice et d'autre part, celui de [8] a une complexité linéaire en tD^2. Ainsi, suivant le ratio t/D et la valeur prise pour ω, l'un ou l'autre algorithme est asymptotiquement plus rapide. Notre article [2] présente un nouvel algorithme pour cette seconde étape dont la complexité est quasi-linéaire en $t^{\omega-1}D$. Ainsi, quel que soient t/D et ω, il est asymptotiquement le plus rapide.

3.1 Idée générale

Cette étape de changement d'ordre s'appuie, classiquement, sur l'utilisation de matrices de multiplication. Il s'agit de matrices de taille $D \times D$ ayant deux types de lignes : des lignes denses et des lignes issues de la matrice identité (des 0 partout sauf un coefficient qui est 1). En particulier, dans la situation générique mentionnée précédemment, seule une matrice est nécessaire, celle dite de la dernière variable. Elle possède de plus exactement t lignes denses.

L'idée principale est de remplacer cette matrice par une matrice de taille $t \times t$ mais dont les coefficients sont des polynômes en une variable, de degrés moyens D/t. Cette matrice peut être vue comme une compression de la

matrice originelle et nécessite le même espace mémoire pour la stocker. Nous illustrons ci-dessous la matrice scalaire originelle M et la matrice polynomiale compressée P dans le cas de trois variables x, y et z :

$$M = \left(\begin{array}{cccc|cc|cc} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 7 & 26 & 26 & 3 & 6 & 0 & 14 \\ \hline 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 12 & 0 & 26 & 0 & 14 & 1 & 10 & 24 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 26 & 20 & 10 & 11 & 0 & 2 & 27 & 5 \end{array}\right), \quad P = \begin{pmatrix} z^4 - 26z^3 - 26z^2 - 7z & -6z - 3 & -14z \\ -26z^2 - 12 & z^2 - z - 14 & -24z - 10 \\ -11z^3 - 10z^2 - 20z - 26 & -2z & z^2 - 5z - 27 \end{pmatrix}.$$

En s'appuyant sur les algorithmes rapides sur les matrices polynomiales [11], nous pouvons calculer la forme normale de Hermite de cette matrice en temps quasi-linéaire en $t^{\omega-1}D$. Sous les hypothèses de généricité données plus haut, cette forme nous permet de lire la base de Gröbner lexicographique du système donné en entrée et même une paramétrisation des solutions.

$$H = \begin{pmatrix} z^8 + 26z^7 + 8z^6 + 17z^5 + 19z^4 + z^3 + 28z^2 + 20z + 18 & 0 & 0 \\ 28z^7 + 23z^6 + 17z^5 + 25z^4 + 24z^3 + 17z^2 + 14z + 4 & 1 & 0 \\ 6z^7 + 13z^6 + 22z^5 + 12z^4 + 28z^3 + 24z^2 + 26z + 14 & 0 & 1 \end{pmatrix}.$$

Ainsi, on peut lire que les solutions (x_i, y_i, z_i) sont exactement telles que z_i annule le premier polynôme en z, de degré 8. De plus, la seconde ligne nous permet de paramétriser y_i en fonction de z_i, via le polynôme de la première colonne. De manière similaire, la dernière ligne nous informe que x_i se paramétrise en fonction de z_i, encore une fois via le polynôme de la première colonne.

3.2 Expérimentations

Nous avons étudié le comportement de cet algorithme en combinant l'implantation optimisée du calcul de forme normale de Hermite disponible dans POLYNOMIAL MATRIX LIBRARY (PML) [10] et des routines de `msolve` et comparé à l'implémentation actuelle de SPARSE-FGLM dans `msolve` [1].

Dans la Table 1, nous donnons les temps pour la première étape du calcul de bases de Gröbner par F_4 (et son *tracer*, voir le portfolio concernant `msolve`) avec les temps des différents algorithmes pour le changement d'ordre. Pour cette seconde étape, les points de comparaisons sont l'algorithme SPARSE-FGLM dans sa version originelle [8] (Wied.) ou par blocs [9, 13] (bl-Wied.) et notre nouvel algorithme (HNF).

TABLE 1 – Temps (en s) pour un système carré en n variables et degré d sur un corps fini premier avec un premier de 30 bits.

			Step 1 : P		Step 2 : H		
			`msolve`		`msolve`	NTL	PML
n, d	D	t	F_4	F_4-tr	Wied.	bl-Wied.	HNF
$11, 2$	2048	462	11.6	1.1	1.2	1.7	0.8
$12, 2$	4096	924	115.9	8.3	6.5	14.5	5.3
$13, 2$	8192	1716	970	62	103.6	110	34.8
$14, 2$	16384	3432	7921	460	1011	880	240
$15, 2$	32768	6435	61381	3193	7844	6691	1665
$16, 2$	65536	12870	482515	24523	58744	52709	11359
$8, 3$	6561	1107	122.6	12.8	23.6	44.7	15.1
$9, 3$	19683	3139	3552.7	361	1302	1163	314
$10, 3$	59049	8953	95052	8664	34844	29974	6709
$6, 4$	4096	580	9.9	2.2	4	8.8	3.5
$7, 4$	16384	2128	876	128	575	545	157
$8, 4$	65536	8092	57237	6977	36454	33452	7231

3.3 Impact scientifique

Cet algorithme ouvre la voie à un changement de paradigme dans le calcul de bases de Gröbner. Jusqu'à présent, le calcul de bases de Gröbner était appréhendé sous le prisme de l'algèbre linéaire creuse à coefficients scalaires. Nous remplaçons ici cette algèbre linéaire "scalaire" par de l'algèbre linéaire sur des polynômes à une variable. Ce

nouvel algorithme rend le changement d'ordre plus rapide que le calcul de la première base de Gröbner via F_4 [4], voir Table 1. Puisque ce premier calcul est dorénavant le facteur limitant, une prochaine étape sera d'étudier et modifier F_4 afin d'utiliser de l'algèbre linéaire polynomiale de sorte à en accélérer le calcul.

4 RÉFÉRENCES BIBLIOGRAPHIQUES

[1] J. Berthomieu, Ch. Eder, and M. Safey El Din. msolve : A Library for Solving Polynomial Systems. In *2021 International Symposium on Symbolic and Algebraic Computation*, pages 51–58, Saint Petersburg, Russia, July 2021.

[2] J. Berthomieu, V. Neiger, and M. Safey El Din. Faster Change of Order Algorithm for Gröbner Bases under Shape and Stability Assumptions. In *Proceedings of the 2022 International Symposium on Symbolic and Algebraic Computation*, ISSAC '22, pages 409–418, New York, NY, USA, 2022. Association for Computing Machinery.

[3] B. Buchberger. A theoretical basis for the reduction of polynomials to canonical forms. *SIGSAM Bull.*, 10(3) :19–29, 1976.

[4] J.-Ch. Faugère. A New Efficient Algorithm for Computing Gröbner bases (F4). *Journal of Pure and Applied Algebra*, 139(1) :61–88, 1999.

[5] J.-Ch. Faugère. A New Efficient Algorithm for Computing Gröbner Bases Without Reduction to Zero (F5). In *Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation*, ISSAC '02, pages 75–83, New York, NY, USA, 2002. ACM.

[6] J.-Ch. Faugère, P. Gianni, D. Lazard, and T. Mora. Efficient Computation of Zero-dimensional Gröbner Bases by Change of Ordering. *J. Symbolic Comput.*, 16(4) :329–344, 1993.

[7] J.-Ch. Faugère and Ch. Mou. Fast algorithm for change of ordering of zero-dimensional gröbner bases with sparse multiplication matrices. In *Proceedings of the 36th International Symposium on Symbolic and Algebraic Computation*, ISSAC '11, pages 115–122, New York, NY, USA, 2011. ACM.

[8] J.-Ch. Faugère and Ch. Mou. Sparse FGLM algorithms. *Journal of Symbolic Computation*, 80(3) :538–569, 2017.

[9] S. G. Hyun, V. Neiger, H. Rahkooy, and É. Schost. Block-Krylov techniques in the context of sparse-FGLM algorithms. *J. Symb. Comput.*, 98 :163–191, 2020. Special Issue on Symbolic and Algebraic Computation : ISSAC 2017.

[10] S. G. Hyun, V. Neiger, and É. Schost. Implementations of efficient univariate polynomial matrix algorithms and application to bivariate resultants. In *Proceedings ISSAC 2019*, pages 235–242. ACM, 2019. `https://github.com/vneiger/pml`.

[11] G. Labahn, V. Neiger, and W. Zhou. Fast, deterministic computation of the Hermite normal form and determinant of a polynomial matrix. *J. Symb. Comput.*, 42 :44–71, 2017.

[12] V. Neiger and É. Schost. Computing syzygies in finite dimension using fast linear algebra. *Journal of Complexity*, 60 :101502, 2020.

[13] A. Steel. Direct Solution of the (11,9,8)-MinRank Problem by the Block Wiedemann Algorithm in Magma with a Tesla GPU. In *Proceedings PASCO 2015*, page 2–6. ACM, 2015.