Algorithmes pour les polynômes lacunaires

Louis Leroux

UNIVERSITÉ de CAEN BASSE NORMANDIE

U.F.R Sciences

ECOLE DOCTORALE SIMEM

Laboratoire de Mathématiques Nicolas Oresme

THÈSE

Présentée par

Louis LEROUX

et soutenue

le 24 mars 2011,

en vue de l'obtention du

DOCTORAT de l'UNIVERSITÉ de CAEN

Specialité : Mathématiques et leurs interactions

Arrété du 07 Août 2006

Algorithmes pour les polynômes lacunaires

MEMBRES du JURY

Francesco Amoroso	Professeur, Université de Caen (Directeur)
André Galligo	Professeur, Université de Nice (Rapporteur)
Grégoire Lecerf	Chargé de Recherches CNRS, École Polytechnique
Denis Simon	Professeur, Université de Caen
Christopher Smyth	Professeur, Université d'Edimbourg (Rapporteur)
Martín Sombra	Professeur, ICREA & Université de Barcelone

REMERCIEMENTS

Je suis très heureux d'adresser mes premiers remerciements à Francesco Amoroso qui a encadré mes deux mémoires de Master puis cette thèse. Francesco a vraiment été pour moi un excellent directeur, toujours très disponible. Il m'a toujours soutenu et encouragé tout au long de ces années. Je tiens à le remercier aussi pour les très (trop ?) nombreuses versions préliminaires qu'il a toujours relues avec une grande attention. Je le remercie enfin pour m'avoir permis de rencontrer Martín Sombra qui a co-dirigé ce travail de thèse.

Je suis donc également très heureux de remercier Martín Sombra qui a beaucoup fait beaucoup pour moi au cours de ces années de thèse. Malgré la distance qui nous séparait, il a aussi su se rendre très disponible que ce soit par mail où lors de mes nombreux séjours à Bordeaux et Barcelone. Martín s'est vraiment beaucoup investi dans mes recherches et mon travail a connu des avancées spectaculaires lors de chacune de nos rencontres !

Sur un plan plus pratique, je remercie mes deux directeurs de m'avoir intégré au PICS de Patrice Philippon, que je remercie aussi au passage, ce qui m'a permis de me déplacer très facilement et de faire de longs séjours à Barcelone.

Si j'avais le courage de refaire une thèse, je la ferais avec vous deux sans hésiter : Grazie mille ! & ¡ Muchas gracias !

J'adresse mes plus sincères remerciements à André Galligo, Grégoire Lecerf et Chris Smyth qui ont bien voulu être les rapporteurs de ma thèse. Ils ont fait ce travail avec beaucoup de sérieux et leur remarques ont véritablement amélioré la qualité de ce manuscrit.

Je remercie vivement Denis Simon d'avoir également accepté de faire partie de ce jury et je lui suis reconnaissant pour toutes les bonnes idées qu'il m'a données.

J'ai bénéficié pendant ces quatre années des excellentes conditions de travail qu'offre le Laboratoire de Mathématiques Nicolas Oresme dont je remercie tous les membres, enseignants-chercheurs et personnels administratifs. Je remercie plus particulièrement tous les doctorants et mes collègues de bureau pour ces bonnes

années passées à leur côté. S'il est si agréable de faire une thèse au LMNO, c'est aussi grâce à l'excellente ambiance qui règne au sein de l'équipe de doctorants.

Je remercie l'Université de Barcelone et l'équipe d'Algèbre et Géométrie qui m'a particulièrement bien accueilli lors de mes séjours à Barcelone malgré mes piètres performances en castillan et en catalan...

Je remercie du fond du cœur mes grands-parents, parents, beaux-parents, sœurs, frère, belles-sœurs, beaux-frères, nièces et neveux pour leur soutien inconditionnel et la joie infinie que j'ai de les côtoyer ! Je suis très heureux et fier de vous dédier ce travail. Même si pour la plupart vous n'y comprendrez pas grand chose, sachez que vous y êtes pour bien plus que vous ne le pensez. J'en profite également pour rendre hommage à ma petite mamie Rueil disparue pendant ma thèse, j'aurais tellement voulu qu'elle soit là aujourd'hui...

Mes derniers remerciements vont tout naturellement à ma petite Zazou qui m'a supporté pendant ces années tout en ne comprenant rien à ce que je faisais ! Je suis plus que jamais heureux de partager ta vie et très fier d'avoir prochainement un petit garçon avec toi...

à Zazou & Léon,

TABLE DES MATIÈRES

INTRODUCTION

Il n'existe pas de définition précise pour qualifier un polynôme de « lacunaire ». Moralement, c'est un polynôme qui a « peu » de termes non nuls par rapport à son degré. Un polynôme comme

$$10X^{10^{100}} - 3X^{50\,000} + 19X^{250} - 8X + 6$$

peut ainsi être qualifié de lacunaire.

Nous distinguerons deux façons de représenter les polynômes : la représentation dense et la représentation lacunaire. Commençons par les décrire dans le cas des polynômes à une variable. Soient A un anneau que l'on suppose unitaire et commutatif et $f(X) \in A[X]$. Ecrivons

$$f(X) = \sum_{i=0}^{d} a_i X^i,$$

où $a_d \neq 0$. On appelle *représentation dense* de f la liste de coefficients

$$[a_0, \ldots, a_d].$$

C'est la façon la plus courante de représenter un polynôme. Selon que A soit l'anneau $\mathbb{Z}$ des entiers, le corps $\mathbb{Q}$ des rationnels, un corps de nombre $\mathbb{Q}[\alpha]$, etc. il existe des façons différentes d'encoder les coefficients de f. Pour un anneau A donné et une façon d'encoder les éléments de A, appelons $\mathrm{h}_A(f)$ une borne pour le nombre de bits nécessaires pour encoder chaque coefficient de f. Notons également $\ell(f)$ le nombre de bits nécessaires pour encoder la représentation dense de f. On a alors

$$d + \mathrm{h}_A(f) \leq \ell(f) \leq (d+1)\,\mathrm{h}_A(f).$$

Lorsque la plupart des coefficients de f sont nuls (comme c'est le cas pour le polynôme donné plus haut en exemple), cette représentation n'est pas adaptée puisqu'elle nécessite de stocker beaucoup de 0.

Ecrivons maintenant

$$f(X) = \sum_{j=1}^{N} a_j X^{\alpha_j}$$

où, pour chaque $1 \leq j \leq N$, on a $a_j \neq 0$ et $d = \max_{1 \leq j \leq N} \alpha_j$. Alors la *représentation lacunaire* de f est la liste des paires

$$[(a_1, \alpha_1), \ldots, (a_N, \alpha_N)]$$

des coefficients non nuls affectés des exposants correspondants. Remarquons que l'entier N est alors le nombre de coefficients non nuls de f. Si $L(f)$ désigne le nombre de bits nécessaires pour encoder la représentation lacunaire de f, on a :

$$N + \mathrm{h}_A(f) + \log_2(d) \leq L(f) \leq N(\mathrm{h}_A(f) + \log_2(d) + 1).$$

En particulier, la taille de cet encodage est linéaire en le logarithme du degré de f. La représentation lacunaire peut donc être préférée si le nombre de termes non nuls de f et $\mathrm{h}_A(f)$ sont au plus de l'ordre du logarithme de son degré.

Lorsqu'on analyse la complexité d'un algorithme qui prend en entrée des polynômes, la représentation lacunaire permet de différencier le nombre de termes non nuls du degré du polynôme au contraire de la représentation dense, qui ne fait pas la distinction entre les deux.

Définissons maintenant ces deux représentations pour des polynômes à plusieurs variables. On fixe un entier $n \geq 1$ et un polynôme

$$f(X_1, \ldots, X_n) \in A[X_1, \ldots, X_n].$$

On note dans la suite $\boldsymbol{X}$ pour $(X_1, \ldots, X_n)$ et pour $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n) \in \mathbb{Z}^n$ on définit :

$$\boldsymbol{X}^{\boldsymbol{\alpha}} := X_1^{\alpha_1} \cdots X_n^{\alpha_n}.$$

On peut alors écrire f de la façon suivante :

$$f(\boldsymbol{X}) = \sum_{i=1}^{N} a_i \boldsymbol{X}^{\boldsymbol{\alpha}_i},$$

où pour chaque $1 \leq i \leq N$, on a $a_i \neq 0$. La représentation lacunaire de f est alors naturellement définie de façon analogue au cas univarié. C'est la liste de ses coefficients non nuls affectés des vecteurs d'exposants correspondants :

$$[(a_1, \boldsymbol{\alpha}_1), \ldots, (a_n, \boldsymbol{\alpha}_n)].$$

On appelle encore $L(f)$ le nombre de bits nécessaires pour encoder f. Si d désigne le maximum des degrés partiels de f, on a

$$N + \mathrm{h}_A(f) + n + \log_2 d \leq L(f) \leq N(\mathrm{h}_A(f) + n(\log_2 d + 1)).$$

La représentation dense se généralise beaucoup moins bien au cas multivarié et elle devient plus lourde d'utilisation. En effet, pour des polynômes à une variable, on a un ordre très naturel sur les monômes (le degré) et on ordonne les coefficients des polynômes selon cet ordre pour obtenir leur représentation dense. Comme il n'y a pas de façon canonique de généraliser cet ordre en dimension supérieure, il faut d'abord fixer un ordre total sur les monômes de $A[\boldsymbol{X}]$ et si on veut pouvoir définir une représentation dense, il faut même fixer un ordre ayant de bonnes propriétés. On fixe donc un ordre total sur les monômes que l'on note $\prec$ et on suppose que chaque monôme n'a qu'un nombre fini de monômes qui lui sont inférieurs. Relativement à cet ordre monomial, tous les monômes peuvent être ordonnés du plus petit au plus grand et c'est cette propriété qui est nécessaire pour pouvoir définir la représentation dense. On peut alors écrire

$$f(\boldsymbol{X}) = \sum_{i=1}^{r} a_i \boldsymbol{X}^{\boldsymbol{\alpha}_i},$$

où $\boldsymbol{\alpha}_1 \prec \cdots \prec \boldsymbol{\alpha}_r$ est la liste des r premiers monômes rangés dans l'ordre croissant et où $\boldsymbol{\alpha}_r$ est le plus grand monôme apparaissant dans f. La représentation dense de f est alors la liste

$$[a_1, \ldots, a_r]$$

de ses coefficients rangés dans l'ordre croissant relativement à l'ordre $\prec$. Le nombre de bits nécessaires pour encoder la représentation d'un polynôme en plusieurs variables dépend également de l'ordre que l'on a fixé. Comme nous n'utiliserons jamais cette représentation, on ne donne pas plus de détail sur celle-ci.

Selon la représentation des polynômes que l'on choisit, on utilise des algorithmes différents pour l'arithmétique de base. Pour les deux types de représentation, il existe des méthodes sophistiquées pour additionner ou multiplier deux polynômes, calculer leur pgcd, etc. On peut consulter [**GG03**] pour la description complète de ces algorithmes et de leur temps d'exécution dans le cas dense et les avancées récentes obtenues par J. Hoeven et G. Lecerf [**HL10**] dans le cas lacunaire.

Plusieurs résultats profonds de géométrie diophantienne et de géométrie algébrique ont des applications intéressantes à la factorisation des polynômes lacunaires. Un bon exemple d'une telle application est l'algorithme décrit par H.W. Lenstra dans [**Len99**] pour déterminer les facteurs de petits degré d'un polynôme lacunaire univarié à coefficients algébriques. Un des ingrédients essentiels de cet algorithme est le théorème de Dobrowolski qui permet de minorer la hauteur de

Weil d'un nombre algébrique. Cet algorithme a ensuite été généralisé indépendamment par M. Avendaño, T. Krick et M. Sombra dans [**AKS07**] et par E. Kaltofen et P. Koiran dans [**KK06**] pour des polynômes à plusieurs variables. Ces deux algorithmes reposent sur des généralisations du théorème de Dobrowolski dues à F. Amoroso et S. David [**AD00**] et à Amoroso et U. Zannier [**AZ00**] respectivement. Les deux algorithmes que nous présentons dans cette thèse ont été obtenus par une démarche analogue. Ce sont des généralisations d'algorithmes déjà connus pour des polynômes à une variable que nous étendons aux cas multivarié.

Dans la première partie, nous étudions un algorithme dû à M. Filaseta, A. Granville et A. Schinzel qui permet de déterminer une représentation des facteurs cyclotomiques d'un polynôme lacunaire à coefficients entiers [**FGS08**, Algorithme C]. Cet algorithme a une complexité quasi-linéaire en le logarithme du degré du polynôme donné en entrée. Leur résultat repose sur un théorème de J.H. Conway et A.J. Jones sur l'annulation de sommes de racines de l'unité. Nous présentons d'abord plusieurs améliorations de cet algorithme qui nous permettent de clarifier la dépendance de sa complexité en la hauteur (i.e. le logarithme du maximum en module des coefficients) et en le nombre de termes non nuls du polynôme donné en entrée. On appellera Algorithme A la procédure obtenue. Ces améliorations simplifient également l'implémentation pratique de cet algorithme mais surtout, nous permettent de le généraliser à la dimension supérieure pour répondre au problème suivant. Etant donnés $F_1, \dots, F_k \in \mathbb{Z}[\boldsymbol{X}]$, déterminer les points de torsion, c'est à dire les points dont toutes les coordonnées sont des racines de l'unité, qui sont des solutions du système d'équations :

$$F_1(\boldsymbol{X}) = \cdots = F_k(\boldsymbol{X}) = 0. \tag{0.0.1}$$

Cet ensemble de points de torsion peut a priori ne pas être fini, mais un théorème de M. Laurent [**Lau84**] assure que ces points peuvent être représentés de façon finie. Décrivons brièvement cette représentation. On dit qu'une sous-variété B de $\mathbb{G}_m^n := (\overline{\mathbb{Q}}^*)^n$ est de torsion si c'est le translaté d'un sous-groupe algébrique de $\mathbb{G}_m^n$ par un point de torsion. Soit $V = V(F_1, \dots, F_k)$ la sous-variété de $\mathbb{G}_m^n$ définie par les polynômes $F_1, \dots, F_k$ et V_{tors} l'ensemble des points de torsion de V. Le résultat de Laurent affirme qu'il existe un nombre fini de sous-variétés de torsion $B_1, \dots, B_t$ telles que

$$\bigcup_{1 \le i \le t} B_i = \overline{V_{\text{tors}}}.$$

Dans les sections 1.3 et 1.4, on généralise l'algorithme A pour déterminer une telle représentation de $\overline{V_{\text{tors}}}$. On appelle Algorithme C la procédure obtenue. Pour pouvoir énoncer ce résultat, on a besoin d'introduire certaines notations.

Pour $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{G}_m^n$ et $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_n) \in \mathbb{Z}^n$, on pose :

$$\boldsymbol{x}^{\boldsymbol{\lambda}} := x_1^{\lambda_1} \cdots x_n^{\lambda_n} \in \mathbb{G}_m.$$

Si Λ est un sous-groupe de $\mathbb{Z}^n$ de rang k, alors l'ensemble

$$H_\Lambda := \left\{ \boldsymbol{x} \in \mathbb{G}_m^n \mid \boldsymbol{x}^{\boldsymbol{\lambda}} = 1, \ \forall \boldsymbol{\lambda} \in \Lambda \right\}$$

est un sous-groupe algébrique de $\mathbb{G}_m^n$ de dimension $n - k$. De plus, l'application $\Lambda \mapsto H_\Lambda$ est une bijection entre l'ensemble des sous-groupes de $\mathbb{Z}^n$ et l'ensemble des sous-groupes algébriques de $\mathbb{G}_m^n$ [**S96**, Lemma 2]. Soit $M_{k,n}(\mathbb{Z})$ l'ensemble des matrices à k lignes et n colonnes et à coefficients entiers. Etant donnés une matrice $L = (\lambda_{i,j})_{\substack{1 \le i \le k \\ 1 \le j \le n}} \in M_{k,n}(\mathbb{Z})$, et $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{G}_m^n$ on définit :

$$\boldsymbol{x}^L := \left(\boldsymbol{x}^{\boldsymbol{\lambda}_1}, \ldots, \boldsymbol{x}^{\boldsymbol{\lambda}_k} \right) \in \mathbb{G}_m^k,$$

où $\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_k$ sont les lignes de la matrice L. Soient maintenant Λ un sous-groupe de $\mathbb{Z}^n$ de dimension k, $\{\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_k\}$ une $\mathbb{Z}-$base de Λ et L la matrice de taille $k \times n$ dont les lignes sont les vecteurs $\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_k$. Soit également $\boldsymbol{\omega} \in \mu_\infty^k$, alors l'ensemble :

$$B(L, \boldsymbol{\omega}) := \left\{ \boldsymbol{x} \in \mathbb{G}_m^n \mid \boldsymbol{x}^L = \boldsymbol{\omega} \right\} \tag{0.0.2}$$

est une sous-variété de torsion et en fait toutes les sous-variétés de torsion peuvent être décrites de cette façon. On pourra consulter la section 1.1 pour plus de détails. Pour $\ell \in \mathbb{N}$, on note $\mathsf{I}(\ell)$ la complexité de la multiplication de deux entiers de longueurs binaires majorées par ℓ. On peut maintenant citer le résultat principal de la première partie :

Théorème. — *(Algorithme C) Il existe un algorithme qui a la propriété suivante. Soient $F_1, \ldots, F_k \in \mathbb{Z}[X_1, \ldots, X_n]$ des polynômes donnés par leur représentation lacunaire et V la sous-variété de $\mathbb{G}_m^n$ qu'ils définissent. L'algorithme détermine une famille finie de sous-variétés de torsion $B_1, \ldots, B_t$ telle que :*

$$\bigcup_{1 \le i \le t} B_i = \overline{V_{\mathrm{tors}}},$$

où les B_i sont représentées comme dans (0.0.2)*. De plus, cet algorithme effectue au plus*

$$O\left(N^{nkN} (\mathsf{I}(\log d) \log\log d + h) \right)$$

opérations binaires, où les entiers N, d et h désignent respectivement des bornes pour le nombre de termes non nuls, le degré total et la hauteur de chacun des F_i.

La complexité de cet algorithme est encore quasi-linéaire en le degré des polynômes donnés en entrée mais exponentielle en leurs nombre de termes, en leur

hauteur et en leur nombre de variable. Ainsi, cet algorithme est particulièrement bien adapté lorsque les polynômes en entrée sont lacunaires.

Dans la deuxième partie, nous étudions un algorithme également dû à Filaseta, Granville et Schinzel qui permet de déterminer, sous certaines hypothèses, le plus grand diviseur commun de deux polynômes lacunaires [**FGS08**, Algorithme B]. La complexité de cet algorithme est quasi-linéaire en les degrés des polynômes donnés en entrée. Cependant il est difficile d'évaluer la dépendance de cette complexité en les autres paramètres, c'est à dire la hauteur et le nombre de termes non nuls des polynômes. En effet, pour pouvoir préciser cette dépendance, il faudrait pouvoir quantifier un théorème de E. Bombieri et U. Zannier (Théorème 2.1.3) sur lequel s'appuie cet algorithme. On peut se référer à la section 2.1.1 pour plus de détails à ce propos.

De la même façon que dans la première partie, nous présentons des simplifications et des améliorations de cet algorithme (Algorithme D, section 2.1) qui nous permettent de le généraliser à la dimension 2 pour attaquer le problème suivant. Etant donnés $f_1, f_2, f_3 \in \mathbb{Q}[X,Y]$, peut-on représenter les points de $V(f_1,f_2,f_3)$ avec moins d'équations ? Un système de trois polynômes en deux variables est surdéterminé. En effet, on peut démontrer qu'il existe $g_1, g_2 \in \mathbb{C}[X,Y]$ tels que

$$V(f_1,f_2,f_3) = V(g_1,g_2).$$

Une extension naturelle serait d'obtenir un algorithme pour le calcul de g_1, g_2 avec une complexité quasi-linéaire en le degré des f_i. Dans cette direction et en supposant établie une version effective de la conjecture de Zilber, qui est généralisation naturelle du théorème de Bombieri et Zannier, on obtient le résultat un peu plus faible suivant.

Théorème. — *(Algorithme E) Il existe un algorithme qui a la propriété suivante : étant donnés $f_1, f_2, f_3 \in \mathbb{Z}[X,Y]$ donnés par leur représentation lacunaire, l'algorithme détermine un ensemble de polynômes $\{p_i\}_{i\in I}$ de $\mathbb{Z}[X,Y]$ et un ensemble fini de triplets de polynômes $\{(q_{j,1},q_{j,2},q_{j,3})\}_{j\in J}$ de $\mathbb{Z}[X,Y]$ donnés par leur représentation lacunaire tels que*

$$V(f_1,f_2,f_3) = \bigcup_{i\in I} V(p_i) \cup \bigcup_{j\in J} V(q_{j,1},q_{j,2}) \backslash V(q_{j,3}). \tag{0.0.3}$$

De plus cet algorithme nécessite au plus

$$O_{N,h}(\mathsf{l}(\log d)\log\log d)$$

opérations binaires, où N, d et h désignent respectivement une borne pour le nombre de termes non nuls, le degré total et la hauteur des f_i. De plus I et J possèdent $O_{N,h}(1)$ éléments, le nombre de termes non nuls et la hauteur des polynômes en sortie de l'algorithme sont majorés par $O_{N,h}(1)$ et ces polynômes sont de degré au plus $O_{N,h}(d)$.

L'algorithme E que l'on décrit dans la section 2.3 a bien la complexité annoncée mais la représentation des points de $V(f_1, f_2, f_3)$ qu'il renvoie est cependant moins bonne que voulue. Nous n'arrivons qu'à représenter les zéros de f_1, f_2, f_3 que comme une réunion finie de systèmes d'équations formant une intersection complète sur un ouvert de $\mathbb{A}^2$. Expliquons brièvement la démarche suivie pour mettre en évidence la raison de ce défaut de représentation. Soit $(\sigma, \tau) \in \mathbb{A}^2$ un zéro commun des polynômes f_1, f_2 et f_3. On étudie d'abord le cas où σ est nul, ce qui ramène simplement à des polynômes en une variable et donc peut être traité avec les algorithmes A et D selon que σ est racine de l'unité ou non. Le cas où τ est nul est bien entendu identique. Une fois ces cas traités, on peut supposer que σ et τ sont tous deux non nuls, ce qui nous permet de nous placer dans $\mathbb{G}_m^2$. Introduisons maintenant une notation supplémentaire. Pour $0 \leq i \leq 2$, on définit :

$$\mathcal{H}_i := \bigcup H,$$

où la réunion porte sur tous les sous-groupes algébriques de $\mathbb{G}_m^2$ de dimension au plus i. On a :

$$\mathcal{H}_0 \subsetneq \mathcal{H}_1 \subsetneq \mathcal{H}_2 = \mathbb{G}_m^2.$$

Chaque ensemble $\mathcal{H}_i$ est l'ensemble des points $(x_1, x_2) \in \mathbb{G}_m^2$ tels que le rang du sous-groupe de $\overline{\mathbb{Q}}^*$ engendré par x_1 et x_2 est inférieur ou égal à i. Ainsi, l'ensemble $\mathcal{H}_0$ est l'ensemble des points de torsion de $\mathbb{G}_m^2$, c'est à dire μ_∞^2 et

$$\mathcal{H}_1 = \left\{ (\zeta_1 u^{a_1}, \zeta_2 u^{a_2}) : a_1, a_2 \in \mathbb{Z}, u \in \overline{\mathbb{Q}}^*, \zeta_1, \zeta_2 \in \mu_\infty \right\}.$$

On va alors distinguer trois cas selon que (σ, τ) appartient à $\mathcal{H}_0$, à $(\mathcal{H}_1 \backslash \mathcal{H}_0)$ ou à $(\mathcal{H}_2 \backslash \mathcal{H}_1)$. Le cas où $(\sigma, \tau) \in \mathcal{H}_0$ peut être traité avec l'algorithme C de la première partie, on donne malgré tout un énoncé détaillé (Théorème 2.3.4) dans la section 2.3.2 (Algorithme E0). Le cas où $(\sigma, \tau) \in (\mathcal{H}_1 \backslash \mathcal{H}_0)$ est assez technique mais ne nécessite que le théorème de Bombieri et Zannier. Il est décrit dans la section 2.3.4 et on appelle Algorithme E1 (Théorème 2.3.8) la procédure correspondante. On note que ces deux cas sont inconditionnels et en particulier, ils ne dépendent pas de la conjecture de Zilber. De plus, dans les deux cas, les points isolés de $V(f_1, f_2, f_3) \cap \mathcal{H}_0$ et de $V(f_1, f_2, f_3) \cap (\mathcal{H}_1 \backslash \mathcal{H}_0)$ sont bien décrits comme les zéros communs de deux équations polynomiales. C'est pour traiter le cas où le point (σ, τ) appartient à $(\mathcal{H}_2 \backslash \mathcal{H}_1)$ qu'on utilise la conjecture de Zilber. La méthode

utilisée ramène à des sous-variétés de $\mathbb{G}_m^k$ (avec $k > 2$) définies par des équations de petit degré, mais éventuellement de dimension strictement positive. Or, une variété de dimension strictement positive n'est pas forcément intersection complète et c'est justement cette difficulté qui nous empêche d'obtenir la représentation des points isolés de $V(f_1, f_2, f_3) \cap (\mathcal{H}_2 \backslash \mathcal{H}_1)$ comme réunion d'intersections complètes sur $\mathbb{A}^2$. On appelle Algorithme E2 (Théorème 2.3.6) la procédure obtenue, elle est décrite dans la section 2.3.3.

L'algorithme E, tout comme l'algorithme D, a pour l'instant plus une valeur théorique puisqu'il est conditionné à la validité de la conjecture de Zilber. Or, si de nombreuses avancées ont été faites ces derniers temps dans la direction de cette conjecture, elle reste à ce jour ouverte dans le cas général. Plus de détails sur cette conjecture et sur les applications que l'on en fait sont donnés à la section 2.3.1.

Seuls les algorithmes A et C ont été implémentés en Maple et nous discutons dans l'annexe d'un exemple détaillé pour chacun de ces algorithmes. On discute également plus précisément du temps d'exécution de l'algorithme A ainsi que de la représentation des résultats renvoyés par cet algorithme. Même si les autres algorithmes n'ont pas complètement été implémentés puisqu'ils dépendent d'un résultat quantitatif conjectural, on peut quand même en donner des exemples d'applications intéressants, ce que nous faisons également en annexe.

CHAPITRE 1

SOUS-VARIÉTÉS DE TORSION

L'objet de cette première partie est de décrire un algorithme qui détermine une représentation des points de torsion inclus dans une variété définie par un système d'équations polynomiales. Comme sa complexité est quasi-linéaire en le logarithme du degré des polynômes définissant la variété et en leur hauteur, mais exponentielle en leur nombre de termes non nuls, il est particulièrement bien adapté pour des polynômes lacunaires.

1.1. Introduction et résultats principaux

Soient $F_1, \ldots, F_k \in \mathbb{Z}[X_1, \ldots, X_n]$ une famille de polynômes à plusieurs variables. On cherche à déterminer les points de torsion solutions du système d'équations :

$$F_1(X_1, \ldots, X_n) = \cdots = F_k(X_1, \ldots, X_n) = 0. \tag{1.1.1}$$

C'est à dire les points dont toutes les coordonnées sont des racines de l'unité. Ce problème apparait naturellement lors de la résolution d'équations trigonométriques et également dans la résolution de certains problèmes de nature géométrique, voir par exemple [**CJ76**], [**PR98**].

Pour $n = 1$, le problème est équivalent à la recherche des facteurs cyclotomiques communs à $F_1, \ldots, F_k$ puisque ces polynômes sont à coefficients entiers. Grâce à un résultat de Conway et Jones [**CJ76**] sur l'annulation de sommes de racines de l'unité, Filaseta, Granville et Schinzel ont récemment décrit un algorithme dans [**FGS08**], qui résout ce problème avec une complexité quasi-linéaire en le logarithme du degré des polynômes donnés en entrée.

Pour $n \geq 2$, le système (1.1.1) peut s'annuler en une infinité de points de torsion mais un théorème de structure de M. Laurent [**Lau84**] assure que cet ensemble peut en principe être décrit de façon finie (voir plus loin pour plus de détails). Dans cette partie, on va d'abord simplifier l'algorithme décrit dans [**FGS08**] et

préciser sa complexité (voir l'algorithme A dans la section 1.2). Ensuite, nous le généraliserons au cas multivarié (voir les algorithmes B et C dans les sections 1.3 et 1.4). La complexité de ces algorithmes est encore quasi-linéaire en le logarithme du degré des polynômes donnés en entrée et en leur hauteur, mais exponentielle en leur nombre de variables et en leur nombre de termes non nuls. Notre algorithme peut donc être vu comme une version effective du théorème de Laurent, particulièrement bien adaptée pour les polynômes lacunaires, c'est à dire pour les polynômes ayant peu de termes non nuls mais potentiellement un grand degré.

On fixe dans la suite $n \in \mathbb{N}^*$. On commence par rappeler certains résultats de géométrie sur les sous-variétés du tore algébrique $\mathbb{G}_m^n := (\overline{\mathbb{Q}}^*)^n$. Toutes les définitions et tous les résultats cités ici peuvent aussi être trouvés dans [**S96**] ou dans [**Z09**] et on renvoie à ces textes pour les démonstrations et pour plus de détails.

L'ensemble $\mathbb{G}_m^n$ est un groupe pour la multiplication coordonnée par coordonnée :

$$(x_1, \ldots, x_n) \cdot (y_1, \ldots, y_n) = (x_1 y_1, \ldots, x_n y_n).$$

Un *sous-groupe algébrique* $H \subset \mathbb{G}_m^n$ est un sous-groupe de $\mathbb{G}_m^n$ qui est une variété algébrique, c'est à dire un fermé pour la topologie de Zariski. Un point d'ordre fini dans le groupe $\mathbb{G}_m^n$ sera appelé un *point de torsion*. L'ensemble des points de torsion de $\mathbb{G}_m^n$ est l'ensemble μ_∞^n, où μ_∞ désigne l'ensemble des racines de l'unité de $\overline{\mathbb{Q}}$. Une *sous-variété de torsion* est un translaté d'un sous-groupe algébrique par un point de torsion. C'est donc un ensemble de la forme

$$\boldsymbol{\eta} H := \{\boldsymbol{\eta} \cdot \boldsymbol{h} \mid \boldsymbol{h} = (h_1, \ldots, h_n) \in H\},$$

où H est un sous-groupe algébrique et $\boldsymbol{\eta} = (\eta_1, \ldots, \eta_n) \in \mu_\infty^n$. Pour une sous-variété V de $\mathbb{G}_m^n$, on pose

$$V_{\text{tors}} := V \cap \mu_\infty^n,$$

pour désigner l'ensemble des points de torsion de V. Un théorème de Laurent [**Lau84**] affirme que la clôture de Zariski $\overline{V_{\text{tors}}}$ de V_{tors} est la réunion d'un nombre fini de sous-variétés de torsion :

$$\overline{V_{\text{tors}}} = B_1 \cup \cdots \cup B_t. \tag{1.1.2}$$

Ce résultat a d'abord été conjecturé par S. Lang [**Lan83**]. Il a été prouvé par Y. Ihara, J.P. Serre et J. Tate lorsque V est une courbe [**Lan83**] et par Laurent dans le cas général. On note également que le résultat analogue a également été démontré par M. Raynaud [**Ra83**] pour les variétés abéliennes et par M. Hindry [**Hi88**] pour les variétés semi-abéliennes.

On rappelle ici certaines notations déjà données en introduction, avant d'énoncer notre résultat principal. Pour $\boldsymbol{x} = (x_1, \dots, x_n) \in \mathbb{G}_m^n$ et $\boldsymbol{\lambda} = (\lambda_1, \dots, \lambda_n) \in \mathbb{Z}^n$, on pose :

$$\boldsymbol{x}^{\boldsymbol{\lambda}} := x_1^{\lambda_1} \cdots x_n^{\lambda_n} \in \mathbb{G}_m.$$

Si Λ est un sous-groupe de $\mathbb{Z}^n$ de rang k, alors l'ensemble

$$H_\Lambda := \left\{ \boldsymbol{x} \in \mathbb{G}_m^n \mid \ \boldsymbol{x}^{\boldsymbol{\lambda}} = 1, \ \forall\, \boldsymbol{\lambda} \in \Lambda \right\}$$

est un sous-groupe algébrique de dimension $n-k$. De plus, l'application $\Lambda \mapsto H_\Lambda$ est une bijection entre l'ensemble des sous-groupes de $\mathbb{Z}^n$ et l'ensemble des sous-groupes algébriques de $\mathbb{G}_m^n$ [**S96**, Lemma 2].

Soit $M_{k,n}(\mathbb{Z})$ l'ensemble des matrices à k lignes et n colonnes et à coefficients entiers. Pour $L = (\lambda_{i,j})_{\substack{1 \le i \le k \\ 1 \le j \le n}} \in M_{k,n}(\mathbb{Z})$, et $\boldsymbol{x} = (x_1, \dots, x_n) \in \mathbb{G}_m^n$ on définit :

$$\boldsymbol{x}^L := \left(\boldsymbol{x}^{\boldsymbol{\lambda}_1}, \dots, \boldsymbol{x}^{\boldsymbol{\lambda}_k} \right) \in \mathbb{G}_m^k,$$

où $\boldsymbol{\lambda}_1, \dots, \boldsymbol{\lambda}_k$ sont les lignes de la matrice L. Soient maintenant Λ un sous-groupe de $\mathbb{Z}^n$ de dimension k, $\{\boldsymbol{\lambda}_1, \dots, \boldsymbol{\lambda}_k\}$ une $\mathbb{Z}-$base de Λ et L la matrice de taille $k \times n$ dont les lignes sont les vecteurs $\boldsymbol{\lambda}_1, \dots, \boldsymbol{\lambda}_k$. Soit également $\boldsymbol{\omega} \in \mu_\infty^k$, alors l'ensemble :

$$B(L, \boldsymbol{\omega}) := \left\{ \boldsymbol{x} \in \mathbb{G}_m^n \mid \boldsymbol{x}^L = \boldsymbol{\omega} \right\} \tag{1.1.3}$$

est une sous-variété de torsion et en fait toutes les sous-variétés de torsion peuvent être décrites de cette façon. Notre algorithme prend en entrée un nombre fini de polynômes et renvoie un nombre fini de sous-variétés de torsion dont la réunion est exactement $\overline{V_{\text{tors}}}$, comme dans (1.1.2). De plus, chaque sous-variété de torsion de dimension $n-k$ sera représentée par une paire $(L, \boldsymbol{\omega}) \in M_{k,n}(\mathbb{Z}) \times \mu_\infty^k$ comme dans (1.1.3). On remarque qu'on peut facilement décrire les points de torsion appartenant à une telle sous-variété de torsion puisque

$$B(L, \boldsymbol{\omega})_{\text{tors}} = \left\{ \boldsymbol{\zeta} \in \mu_\infty^n \mid \boldsymbol{\zeta}^L = \boldsymbol{\omega} \right\}.$$

Décrivons maintenant l'entrée de notre algorithme. Soit

$$F(\boldsymbol{X}) = \sum_{i=1}^{N} a_i \boldsymbol{X}^{\boldsymbol{\alpha}_i}$$

un polynôme en n variables et à coefficients entiers. On rappelle que la représentation lacunaire de F est la liste

$$[(a_1, \boldsymbol{\alpha}_1), \dots, (a_N, \boldsymbol{\alpha}_N)]$$

de ses coefficients non nuls affectés des exposants correspondants et sa *hauteur* est définie par :

$$\mathrm{h}(F) := \max_{1 \le i \le N} \{\log |a_i|\}.$$

Si le polynôme F est de degré total d et de hauteur h, le nombre de bits nécessaires pour encoder sa représentation lacunaire est :

$$O\big(N(h + n\log d)\big).$$

Pour $\ell \in \mathbb{N}$, on pose comme c'est l'usage $\mathsf{I}(\ell)$ pour désigner la complexité de la multiplication de deux entiers de longueurs binaires majorées par ℓ. On peut maintenant citer le résultat principal de cette première partie :

Théorème 1.1.1. — *(Algorithme C) Il existe un algorithme qui a la propriété suivante. Soient $F_1, \ldots, F_k \in \mathbb{Z}[X_1, \ldots, X_n]$ des polynômes donnés par leur représentation lacunaire et V la sous-variété de $\mathbb{G}_m^n$ qu'ils définissent. L'algorithme détermine une famille finie de sous-variétés de torsion $B_1, \ldots, B_t$ telle que :*

$$\bigcup_{1\le i\le t} B_i = \overline{V_{\mathrm{tors}}},$$

où les B_i sont représentées comme dans (1.1.3)*. De plus, cet algorithme effectue au plus*

$$O\left(N^{nkN}(\mathsf{I}(\log d)\log\log d + h)\right)$$

opérations binaires, où les entiers N, d et h désignent respectivement des bornes pour le nombre de termes non nuls, le degré total et la hauteur de chacun des F_i.

Remarques : Pour $f, g : \mathbb{N} \to \mathbb{R}_+$, on écrit :

$$f(n) = \widetilde{O}(g(n))$$

s'il existe un constante c strictement positive telle que :

$$f(n) = O(g(n)\log(1 + g(n))^c).$$

Avec cette notation, on a :

$$\mathsf{I}(\ell) = O\left(\ell\log(\ell)\log\log(\ell)\right) = \widetilde{O}(\ell)$$

grâce à l'algorithme de A. Schönnhage et V. Strassen [**GG03**, Theorem 8.24]. Cette complexité a récemment été améliorée par M. Fürer [**F07**] qui a obtenu

$$\mathsf{I}(\ell) = \ell\log(\ell)2^{O(\log^*(\ell))},$$

où

$$\log^*(\ell) = \min\{i \ge 0 : \log^{(i)}(\ell) \le 1\},$$

avec $\log^{(0)}(\ell) = \ell$ et $\log^{(i+1)}(\ell) = \log(\log^{(i)}(\ell))$. Ainsi, la complexité de l'algorithme C relatif au théorème 1.1.1 est

$$N^{nkN}\left(\widetilde{O}(\log d) + O(h)\right).$$

D'autres problèmes proches de celui-ci ont été étudiés dans le cas de polynômes à une variable. Soient $F \in \mathbb{Z}[X]$ donné par sa représentation lacunaire et m un entier positif. Le problème de tester si F s'annule en une racine de l'unité d'ordre m s'appelle le *cyclotomic test* (CT). Dans [**CTV09**], Q. Cheng, S.P. Tarasov et M.N. Vyalyi ont décrit deux algorithmes qui résolvent ce problème en temps polynomial. Il montrent ainsi que CT∈P.

Le *generalized cyclotomic test* (GCT) consiste à déterminer si F s'annule en une quelconque racine de l'unité, c'est à dire, s'il existe $m \in \mathbb{N}$ tel que (F, m) soit une instance positive de CT. Le fait que CT∈P implique clairement que GCT∈NP. D'un autre côté, D. Plaisted a montré précédemment que ce problème est NP-difficile [**P84**, Thm 5.1] et ainsi, GCT est NP-complet. Filaseta et Schinzel ont proposé un algorithme sous-exponentiel qui résout GCT [**FS04**] et, en collaboration avec Granville, ils ont généralisé cet algorithme pour déterminer tous les facteurs cyclotomiques de F [**FGS08**].

Nous allons ici simplifier ce dernier algorithme et clarifier la dépendance en le nombre de termes non nuls N de F qui est en fait exponentielle. Nous allons également améliorer l'implémentation pratique de cet algorithme et son temps d'exécution. La dépendance exponentielle en N semble inévitable et en fait, nous pensons que la taille de la sortie peut être exponentielle en la taille de l'entrée. Dans la sous-section 1.2.4, nous décrirons une famille de polynômes qui accrédite cette conjecture.

Pour $n \geq 2$, les seules méthodes constructives existantes pour déterminer les points de torsion d'une variété sont décrites dans [**Ru93**], [**BS02**], [**Ro07**], [**AS08**]. Leurs complexités ne sont pas explicitées dans ces références mais certainement, elles sont au moins de type $d^{2^n}h$, où n désigne le nombre de variables des polynômes en entrée, d une borne pour leur degré, et h une borne pour leur hauteur, à cause de l'utilisation systématique de résultants itérés. Ainsi, ces méthodes sont plus adaptées pour des polynômes denses, de petit degré et faisant intervenir un petit nombre de variables. Au contraire, notre algorithme est particulièrement bien adapté pour des polynômes ayant peu de termes non nuls mais ayant potentiellement un très grand degré (10^{100} par exemple), voir les exemples de la sous-section 1.2.4 et de l'annexe.

De l'analyse de l'algorithme relatif au théorème 1.1.1, on déduit la borne suivante pour le nombre de sous-variétés de torsion incluses dans une sous-variété du tore $\mathbb{G}_m^n$:

Corollaire 1.1.2. — *Soient $F_1, \dots, F_k \in \mathbb{Z}[X_1, \dots, X_n]$ des polynômes ayant chacun au plus N termes non nuls et soit V la sous-variété de $\mathbb{G}_m^n$ qu'ils définissent. Alors $\overline{V_{\text{tors}}}$ peut être décrite comme la réunion d'au plus*

$$(N!)^k \exp\left(3(n+1)\sqrt{kN \log kN}\right)$$

sous-variétés de torsion.

Historiquement, la première borne effective pour le nombre de sous-variétés de torsion incluses dans une sous-variété de $\mathbb{G}_m^n$ a été obtenue par Bombieri et Zannier [**BZ95**]. Peu de temps après, W. Schmidt détermina une borne ne dépendant que du degré et du nombre de variables des polynômes définissant la sous-variété [**S96**]. L'intérêt principal de notre borne dans le corollaire 1.1.2 est qu'elle est indépendante du degré des polynômes définissant V. Remarquons tout de même qu'une borne analogue peut être obtenue indépendamment avec les méthodes décrites dans [**S96**].

Il est à noter que le nombre de sous-variétés connexes (ou irréductibles) de torsion dépend nécessairement du degré des polynômes définissant V. Actuellement, la meilleure borne est due à F. Beukers et C.J. Smyth [**BS02**] lorsque $n = 2$ et à F. Amoroso et E. Viada dans le cas général [**AV09**]. Le résultat d'Amoroso et Viada affirme que le nombre de sous-variétés irréductibles de torsion incluses dans une variété définie par des polynômes de degré au plus d peut être majoré par :

$$d^n \left(200n^5 \log(n^2 d)\right)^{n^2(n-1)^2}.$$

Cette majoration est très proche de la borne optimale que l'on conjecture, à savoir

$$c(n)d^n,$$

pour une certaine fonction $c(n)$ ne dépendant que de n. D'autres résultats dans cette direction ont été obtenus par G. Rémond [**Re02**], Amoroso et S. David [**AD06**] et David et P. Philippon [**DP07**].

La suite de cette partie suivra la progression suivante. Dans un premier temps, nous présenterons notre simplification de l'algorithme de Filaseta Granville et Schinzel dans la section 1.2. Nous le généraliserons ensuite au cas des hypersurfaces en dimension supérieure dans la section 1.3 avant de traiter le cas des sous-variétés quelconques de $\mathbb{G}_m^n$ dans la section 1.4. Finalement, nous donnerons dans la section 1.5 la démonstration du corollaire 1.1.2

1.2. Facteurs cyclotomiques d'un polynôme à une variable

Dans cette section, on décrit une procédure, que nous appelons Algorithme A, qui permet de déterminer les facteurs cyclotomiques d'un polynôme F à coefficients entiers. Cet algorithme est particulièrement bien adapté aux polynômes lacunaires en raison de sa complexité quasi-linéaire en le logarithme du degré et en la hauteur du polynôme F mais exponentielle en son nombre de termes non nuls.

Soit $F(X) \in \mathbb{Z}[X]$ un polynôme de degré d. Déterminer l'ensemble des $\zeta \in \mu_\infty$ tels que $F(\zeta) = 0$ est équivalent à déterminer l'ensemble des entiers positifs m pour lesquels

$$\Phi_m(X) | F(X),$$

où Φ_m désigne le m-ième polynôme cyclotomique. On souhaite déterminer cet ensemble en temps polynomial en $\log d$. Cependant, le nombre total de facteurs cyclotomiques de F n'est pas polynomial en $\log d$ dans le pire des cas, comme le montre l'exemple suivant. Soit x un entier plus grand que 17 et soit

$$F(X) = X^d - 1,$$

où d est le produit $p_1 \cdots p_r$ de tous les nombres premiers plus petits que x. Une version effective du théorème de Tchebychev **[RS62]** donne les inégalités suivantes :

$$r \geq \frac{x}{\log x} \quad \text{et} \quad \log d \leq 1.02\, x \qquad \text{pour } x \geq 17,$$

De plus

$$F(X) = \prod_{m|d} \Phi_m(X) \quad \text{et} \quad \#\{m \in \mathbb{N} : m|d\} = 2^r \geq 2^{\frac{x}{\log x}} \geq 2^{\frac{\log d}{1.02 \log\log d}},$$

donc le nombre de facteurs cyclotomiques de F n'est pas polynomial en le logarithme de son degré.

Même dans le cas où le nombre de facteurs est plus petit, les déterminer peut être aussi difficile que de factoriser des nombres entiers. Pour illustrer ce problème, considérons le polynôme suivant :

$$F(X) = X^{pq} - 1,$$

où p et q sont deux nombres premiers distincts. On a alors

$$F(X) = \Phi_1(X)\Phi_p(X)\Phi_q(X)\Phi_{pq}(X)$$

et déterminer la factorisation de ce polynôme est aussi difficile que de factoriser l'entier pq. Mais ce problème est difficile et actuellement, aucun algorithme ne peut factoriser pq en temps polynomial en $\log(pq)$.

On voit que ces deux difficultés qu'il nous faut contourner (le grand nombre de facteurs et la factorisation d'entiers), nous empêchent d'obtenir la représentation que l'on souhaitait des facteurs cyclotomiques de F, c'est à dire la liste complète des entiers m pour lesquels $\Phi_m(X)|F(X)$. Pour pouvoir obtenir une complexité polynomiale en $\log d$, on s'autorise donc une représentation un peu plus faible des résultats. Ainsi, la sortie de l'algorithme A sera un ensemble S_F de paires d'entiers (m, e) telles que

$$V(F)_{\text{tors}} = \bigcup_{(m,e)\in S_F} V(\Phi_m(X^e)),$$

où $V(F) := \{x \in \overline{\mathbb{Q}} : F(x) = 0\}$. Donner une telle représentation réduit le problème de déterminer la liste complète des facteurs cyclotomiques de F à la factorisation d'entiers grâce au lemme suivant :

Lemme 1.2.1. — *Soient* $m, e \in \mathbb{N}^*$. *On pose* $e_1 := \prod_{p|m} p^{\text{ord}_p(e)}$ *et* $e_2 := e/e_1$. *Alors*

$$\Phi_m(X^e) = \Phi_{me_1}(X^{e_2}) = \prod_{d|e_2} \Phi_{me_1 d}(X).$$

Démonstration : Pour prouver la première égalité, il suffit de remarquer que les racines de $\Phi_{me_1}(X^{e_2})$ sont aussi racines de $\Phi_m(X^e)$. Puisque ces deux polynômes sont sans facteur carré et sont tous deux unitaires, ils sont égaux. Un argument similaire montre que $\Phi_m(X^e)$ divise le polynôme $\prod_{d|e_2} \Phi_{me_1 d}(X)$. De plus, ces polynômes sont unitaires et on peut montrer qu'ils ont même degré en utilisant la formule :

$$\sum_{d|e_2} \varphi(d) = e_2,$$

où φ désigne la fonction indicatrice d'Euler. Finalement, ces trois polynômes sont égaux, ce que l'on voulait démontrer. □

Précisons une nouvelle fois que l'algorithme A se contente de renvoyer des paires d'entiers (m, e) telles que $\Phi_m(X^e) \mid F(X)$ sans chercher à lister les facteurs cyclotomiques correspondant, et ceci pour conserver une complexité polynomiale en le logarithme du degré de F.

Cette représentation des facteurs cyclotomiques, bien que moins coûteuse, a malgré tout des inconvénients. Le premier est qu'il peut y avoir des répétitions dans la sortie de l'algorithme, i.e. un même facteur peut apparaître dans deux paires distinctes d'entiers. Supposons par exemple que pour un polynôme donné,

l'algorithme renvoie les paires $(2, 30)$ et $(3, 42)$. D'après le lemme 1.2.1, ces deux paires correspondent respectivement aux ensembles de facteurs cyclotomiques :

$$\{\Phi_2, \Phi_6, \Phi_{10}, \Phi_{30}\} \quad \{\Phi_3, \Phi_6, \Phi_{21}, \Phi_{42}\}.$$

On constate donc que le facteur Φ_6 est répété puisqu'il appartient aux deux ensembles de facteurs.

Une fois qu'on a déterminé des facteurs d'un polynôme, un problème qui se pose naturellement est de chercher leur multiplicité. Le deuxième inconvénient de la représentation des facteurs cyclotomiques renvoyée par l'algorithme A est qu'elle ne donne pas de renseignement sur leur multiplicité. Cependant, un résultat de G. Hajos [**Ha53**] affirme que si un polynôme $F \in \mathbb{C}[X]$ possède N termes non nuls, alors la multiplicité de chacun de ses facteurs est au plus N.

Cette majoration de la multiplicité suggère la stratégie suivante pour déterminer la multiplicité des facteurs cyclotomiques d'un polynôme F à coefficients entiers : appliquer l'algorithme A au polynôme F et à ses $N - 1$ premières dérivées, puis lister dans chacun des cas tous les facteurs cyclotomiques à partir de leur représentation donnée par l'algorithme A. Il ne reste enfin qu'à comparer ces différentes listes de facteurs. Cependant, comme on l'a déjà vu, lister tous les facteurs cyclotomiques de F à partir de la sortie de l'algorithme A ne peut pas être réalisé avec la complexité que l'on souhaite, c'est à dire polynomiale en le logarithme du degré du polynôme F.

Comme nous nous sommes intéressés à la généralisation de l'algorithme de Filaseta Granville et Schinzel au cas multivarié, nous avons dans un premier temps négligé cette question de multiplicité. Cependant, il serait intéressant de comprendre si on peut améliorer encore la représentation des résultats en une variable pour obtenir des renseignements sur la multiplicité des facteurs, tout en gardant la même complexité puis de voir ce qu'il est possible de faire dans le cas multivarié pour avoir des informations sur la multiplicité des points de torsion dans une sous-variété donnée.

Dans le reste de cette section, on établit le résultat suivant :

Théorème 1.2.2. — *(Algorithme A) Il existe un algorithme déterministe qui a la propriété suivante. Pour $F \in \mathbb{Z}[X]$ donné par sa représentation lacunaire, l'algorithme détermine un ensemble*

$$S_F := \{(m_1, e_1), \ldots, (m_t, e_t)\}$$

de couples d'entiers tels que

$$V(F)_{\mathrm{tors}} = \bigcup_{i=1}^{t} V\left(\Phi_{m_i}\left(X^{e_i}\right)\right).$$

De plus, cet algorithme nécessite

$$O\left(N^N(\mathsf{I}(\log d)\log\log d + h)\right)$$

opérations binaires, où d désigne le degré de F, h sa hauteur et N le nombre de termes non nuls de F.

L'algorithme A relatif à ce théorème est décrit dans la section 1.2.2 et un exemple d'application de cet algorithme est donné dans l'exemple A de l'annexe. Ce résultat est essentiellement le même que le théorème [**FGS08**, Thm C] mais nous en donnons une démonstration beaucoup plus simple qui nous permet d'expliciter la dépendance en le nombre N de termes et d'améliorer son implémentation et son temps d'exécution. Un aspect plus important de cette nouvelle preuve est qu'elle se généralise au cas multivarié, comme nous le verrons à la section 1.3.

On donne maintenant un certain nombre de résultats préliminaires avant de démontrer le théorème 1.2.2. Nous construisons enfin une famille de polynômes qui ont beaucoup de facteurs cyclotomiques « séparés » dans la sous-section 1.2.4.

1.2.1. Résultats préliminaires. — On rappelle tout d'abord le coût de l'arithmétique de base dans $\mathbb{Z}$:

Le produit de deux entiers de longueur binaire majorée par ℓ peut être calculé en

$$\mathsf{I}(\ell) = O(\ell\log(\ell)\log\log(\ell)) = O_\varepsilon(\ell^{1+\varepsilon}) \quad \forall\,\varepsilon > 0$$

opérations binaires grâce à l'algorithme de Schönhage-Strassen [**SS71**].

Le pgcd de deux entiers de longueur binaire majorée par ℓ peut être calculé en

$$O(\ell(\log(\ell))^2\log\log(\ell))) = O_\varepsilon(\ell^{1+\varepsilon}) \quad \forall\,\varepsilon > 0$$

opérations binaires grâce à l'algorithme de Knuth-Schönhage [**Knu70**]. On renvoie à [**GG03**] pour la description et l'analyse de la complexité de ces algorithmes.

Si $\boldsymbol{\zeta} \in \mu_\infty^n$, on désigne par $\mathrm{ord}(\boldsymbol{\zeta})$ l'ordre de $\boldsymbol{\zeta}$ dans le groupe μ_∞^n, qui est le plus petit commun multiple de l'ordre de ses coordonnées. Pour $m \in \mathbb{N}^*$, on définit aussi :

$$\Psi(m) := 2 + \sum_{p|m}(p-2),$$

où p parcourt l'ensemble de tous les nombres premiers divisant m. L'ingrédient essentiel à la démonstration du théorème 1.2.2 est le résultat suivant dû à Conway et Jones [**CJ76**] :

Théorème 1.2.3. — *(Conway-Jones 1976) Soit ζ_m une racine de l'unité d'ordre m. Soient également $a_1, \ldots, a_N$, $\alpha_1, \ldots, \alpha_N$ des entiers et $S := a_1\zeta_m^{\alpha_1} + \cdots + a_N\zeta_m^{\alpha_N}$. Si*

1. *$S = 0$,*
2. *aucune somme extraite de S ne s'annule,*
3. *$\mathrm{pgcd}(\alpha_2 - \alpha_1, \ldots, \alpha_N - \alpha_1, m) = 1$,*

alors m est sans facteur carré et $\Psi(m) \leq N$.

On dira qu'une somme de racines de l'unité qui s'annule est *minimale* si la condition 2 est satisfaite. Notons que la condition 3 est équivalente à

$$\mathrm{ord}(\zeta_m^{\alpha_2-\alpha_1}, \ldots, \zeta_m^{\alpha_N-\alpha_1}) = m.$$

Nous utiliserons également le lemme suivant :

Lemme 1.2.4. — *Soient $m_1, \ldots, m_s \in \mathbb{N}^*$, alors*

$$\Psi(m_1 \cdots m_s) \leq \sum_{i=1}^{s} \Psi(m_i) - 2(s-1).$$

Démonstration : On a

$$\begin{aligned}\Psi(m_1 \cdots m_s) &= 2 + \sum_{p \mid m_1 \cdots m_s} (p-2) \\ &\leq 2 + \sum_{i=1}^{s} \sum_{p \mid m_i} (p-2) \leq \sum_{i=1}^{s} \Psi(m_i) - 2(s-1),\end{aligned}$$

ce qui est le résultat voulu. □

On aura également besoin de déterminer le cardinal de certains ensembles pour calculer la complexité de l'algorithme A. Les deux lemmes suivants sont donnés dans ce but.

Lemme 1.2.5. — *Soient $N \in \mathbb{N}^*$ et E_N l'ensemble des partitions de l'ensemble d'entiers $\{1, \ldots, N\}$ qui ne contiennent que des ensembles d'au moins deux éléments. Alors :*

$$\#E_N \leq N!$$

Démonstration : Soient σ un élément du groupe symétrique $\mathfrak{S}_N$, le groupe des permutations de l'ensemble $\{1,\dots,N\}$ et $\sigma = c_1 \circ \cdots \circ c_s$ sa décomposition en cycles à supports disjoints. On associe à la permutation σ la partition $\{J_1,\dots,J_s\}$ de l'ensemble $\{1,\dots,N\}$ telle que J_j est le support de c_j pour chaque $1 \leq j \leq s$. Puisque cette application est une surjection de l'ensemble $\mathfrak{S}_N$ sur l'ensemble des partitions de $\{1,\dots,N\}$ et puisque E_N est strictement inclus dans cet ensemble, on a $\#E_N \leq N! = \#\mathfrak{S}_N$. □

Remarque : On peut améliorer cette estimation en $\#E_N \leq \frac{N!}{e}$ en considérant le sous-ensemble de $\mathfrak{S}_N$ formé des permutations sans point fixe, dont l'image contient encore E_N.

Les premières valeurs de $\#E_N$ sont (à partir de $N = 2$) :

$$1,\ 1,\ 4,\ 11,\ 41,\ 162,\ 715,\ 3\,425,\ 17\,722,\ 98\,253,\ 580\,317,\dots$$

Cette suite est recensée comme la suite A000296 dans la *On-Line Encyclopedia of Integer Sequences* consultable à l'adresse suivante : `http://www.research.att.com/~njas/sequences/`. On pourra consulter cette référence pour plus d'informations sur cette suite d'entiers.

Lemme 1.2.6. — *Soient $N, n \in \mathbb{N}$. On pose*

$$Q_N := \{m \in \mathbb{N} : \Psi(m) \leq N \text{ et } m \text{ est sans facteur carré}\}$$

et

$$Q_{n,N} := \{\boldsymbol{\omega} \in \mu_\infty^n : \mathrm{ord}(\boldsymbol{\omega}) \in Q_N\}.$$

Alors

$$\#Q_N \leq \exp\left(3\sqrt{N \log N}\right)$$

et

$$\#Q_{n,N} \leq \exp\left(3(n+1)\sqrt{N \log N}\right).$$

Démonstration : Montrons d'abord la première inégalité. Soient $m \in Q_N$ que l'on écrit $m = \prod_{i=1}^r p_i$ comme produit de ses facteurs premiers. Puisque $\Psi(m) \leq N$, on a

$$\sum_{i=1}^{r} p_i \leq N + 2(r-1).$$

De plus,

$$N - 2 \geq \sum_{i=1}^{r}(p_i - 2) \geq \sum_{i=2}^{r}(p_i - 2) \geq \sum_{i=1}^{r-1}(2i-1) = (r-1)^2.$$

Ainsi, $r \leq \sqrt{N-2}+1$ et

$$\sum_{i=1}^{r} p_i \leq N + 2\sqrt{N-2}. \tag{1.2.1}$$

Essayons maintenant d'obtenir une meilleure borne pour r. Par un calcul direct, on peut vérifier que :

$$r \leq 3\sqrt{N/\log N} \qquad \text{pour } N \leq 10\,000.$$

On suppose maintenant que $N \geq 10\,000$. Une version effective du théorème de Tchebychev (voir **[RS62]**) donne les inégalités :

$$\pi(x) \leq 1.26\,\frac{x}{\log x} \quad \text{et} \quad \sum_{p \leq x} p \geq \frac{x^2}{2\log x} \qquad \text{pour } x \geq 347. \tag{1.2.2}$$

Alors, pour $x := 1.15\sqrt{N\log N}$, on a $x \geq 347$ et ainsi :

$$\sum_{p \leq x} p \geq \frac{1.15^2 N \log N}{2\log(1.15\sqrt{N\log N})} \geq N + 2\sqrt{N-2}. \tag{1.2.3}$$

La dernière inégalité provient d'une simple étude de fonction. En associant les deux inégalités (1.2.1) et (1.2.3), on obtient $\sum_{p \leq x} p \geq \sum_{i=1}^{r} p_i$ et alors :

$$r \leq \pi(1.15\sqrt{N\log N}) \leq 1.26\,\frac{1.15\sqrt{N\log N}}{\log(1.15\sqrt{N\log N})} \quad \text{d'après (1.2.2).}$$

On a finalement

$$r \leq 1.26\,\frac{1.15\sqrt{N\log N}}{\log(\sqrt{N})} \leq 3\sqrt{N/\log N}.$$

Ainsi, chaque nombre entier dans Q_N possède au plus $3\sqrt{N/\log N}$ facteurs premiers distincts, tous inférieurs à N. Il s'en suit que

$$\#Q_N \leq N^{3\sqrt{N/\log N}} = \exp\left(3\sqrt{N\log N}\right).$$

Démontrons maintenant la seconde inégalité. Chaque $\boldsymbol{\omega} \in Q_{n,N}$ d'ordre $m \in Q_N$ peut être représenté par un $(n+1)-$uplet :

$$(d_1, \ldots, d_n, m),$$

où les entiers d_i sont entre 0 et $m-1$ et tels que

$$\boldsymbol{\omega} = \left(\zeta_m^{d_1}, \ldots, \zeta_m^{d_n}\right),$$

où $\zeta_m = \exp\left(\frac{2i\pi}{m}\right)$. Comme on a déjà établit que $m \leq \exp\left(3\sqrt{N\log N}\right)$, il y a au plus

$$\exp\left(3(n+1)\sqrt{N\log N}\right)$$

tels $(n+1)-$uplets et le résultat s'en déduit. □

1.2.2. Recherche des facteurs cyclotomiques - Algorithme A. — Le but de cette section est de démontrer le théorème 1.2.2. On va d'abord déterminer des conditions suffisantes pour qu'un polynôme à coefficients entiers s'annule en une racine de l'unité donnée et nous en déduirons une procédure (Algorithme A) qui satisfait les conditions du théorème 1.2.2. On déterminera enfin son temps d'exécution.

On pose dans la suite :

$$F(X) = \sum_{i=1}^{N} a_i X^{\alpha_i} \in \mathbb{Z}[X]$$

et on fixe ζ_m une racine de l'unité d'ordre m telle que

$$F(\zeta_m) = 0. \tag{1.2.4}$$

On va déterminer des conditions équivalentes à la condition (1.2.4) que l'on pourra tester de façon algorithmique. On considère $\{J_1, \ldots, J_s\}$ une partition de l'ensemble $\{1, \ldots, N\}$ telle que

$$\sum_{i \in J_j} a_i \zeta_m^{\alpha_i} = 0 \qquad \forall\ 1 \leq j \leq s$$

et telle que chacune de ces sommes soit minimale. Pour $1 \leq j \leq s$, on définit :

$$F_j(X) := \sum_{i \in J_j} a_i X^{\alpha_i}$$

tels que $F(X) = \sum_{j=1}^{s} F_j(X)$.

On écrit alors pour $1 \leq j \leq s$:

$$F_j(X) = X^{b_j} G_j(X^{e_j}), \tag{1.2.5}$$

où l'entier b_j est tel que $G_j(0) \neq 0$ et où l'entier e_j est tel que les exposants qui apparaissent dans G_j soient premiers entre eux. On pose alors :

$$m_j := \frac{m}{\mathrm{pgcd}(m, e_j)}.$$

Si ζ_{m_j} désigne une racine de l'unité d'ordre m_j, l'équation (1.2.4) est alors équivalente à :

$$G_j(\zeta_{m_j}) = 0 \text{ et } m_j = \frac{m}{\mathrm{pgcd}(m, e_j)} \qquad \forall\ 1 \leq j \leq s. \tag{1.2.6}$$

De plus, par construction, les sommes $G_j(\zeta_{m_j})$ satisfont les conditions du théorème 1.2.3 et on a donc

$$\Psi(m_j) \leq N_j \qquad \forall\ 1 \leq j \leq s,$$

où $N_j := \#J_j$ et m_j est sans facteur carré.
On pose $e := \mathrm{pgcd}(e_1, \ldots, e_s)$ et, pour $1 \leq j \leq s$, on pose aussi $e'_j := e_j/e$. Finalement, on définit

$$m' := \frac{m}{\mathrm{pgcd}(m, e)}$$

et on observe que

$$\frac{m}{\mathrm{pgcd}(m, e_j)} = \frac{m'}{\mathrm{pgcd}(m', e'_j)} \qquad \forall\ 1 \leq j \leq s.$$

La condition (1.2.6) est alors équivalente à :

$$G_j(\zeta_{m_j}) = 0 \text{ et } m_j = \frac{m'}{\mathrm{pgcd}(m', e'_j)} \qquad \forall\ 1 \leq j \leq s. \tag{1.2.7}$$

De plus, puisque $m' | \mathrm{ppcm}(m_1, \ldots, m_s)$, on a :

$$\begin{aligned}
\Psi(m') &\leq \Psi(\mathrm{ppcm}(m_1, \ldots, m_s)) \\
&\leq \Psi(m_1 \cdots m_s) \\
&\leq \sum_{j=1}^{s} \Psi(m_j) - 2(s-1) \qquad \text{d'après le lemme 1.2.4} \\
&\leq \sum_{j=1}^{s} N_j - 2(s-1) \\
&\leq N - 2(s-1)
\end{aligned}$$

et m' est sans facteur carré puisqu'il divise le nombre $\mathrm{ppcm}(m_1, \ldots, m_s)$ qui est lui-même sans facteur carré. De plus, on remarque que lorsqu'une partition de l'ensemble $\{1, \ldots, N\}$ et un entier m' sont fixés, la condition (1.2.7) peut être testée puisque les polynômes G_j et les entiers m_j, e'_j ne dépendent que de la partition et de l'entier m'. L'algorithme va donc consister à tester la condition (1.2.7) pour tous les choix possibles de partition et pour tous les choix possibles d'entier m'.

Remarquons que l'on peut se restreindre aux partitions contenant des ensembles d'au moins deux éléments pour la simple raison qu'un multiple entier non nul d'une racine de l'unité ne peut être nul. Cette remarque justifie la restriction aux partitions qui ne contiennent que des ensembles ayant au moins deux éléments dans le lemme 1.2.5.

On obtient la procédure suivante :

Algorithme A

Entrée : Un polynôme $F(X) = \sum_{i=1}^{N} a_i X^{\alpha_i}$ donné par sa représentation lacunaire.
Sortie : Un ensemble S_F satisfaisant les conclusions du théorème 1.2.2.

1. $S_F \leftarrow \emptyset$.
2. Pour chaque partition $\{J_1, \ldots, J_s\}$ de l'ensemble $\{1, \ldots, N\}$ telle que $\#J_j \geq 2$, pour chaque $1 \leq j \leq s$, faire :
   (a) Pour chaque $1 \leq j \leq s$, déterminer les polynômes G_j et les entiers e_j associés à l'ensemble J_j, comme dans (1.2.5).
   (b) Faire $e \leftarrow \mathrm{pgcd}(e_1, \ldots, e_s)$ et pour $1 \leq j \leq s$, faire $e'_j \leftarrow e_j/e$.
   (c) Pour chaque entier sans facteur carré m' satisfaisant $\Psi(m') \leq N - 2(s-1)$ faire :
      (i) Pour $1 \leq j \leq s$, faire $m_j \leftarrow \dfrac{m'}{\mathrm{pgcd}(m', e'_j)}$.
      (ii) Si $\Phi_{m_j}(X) | G_j(X)$ pour chaque $1 \leq j \leq s$, alors faire $S_F \leftarrow S_F \cup \{(m', e)\}$.
3. Renvoyer S_F et terminer.

D'après la discussion précédente, cet algorithme renvoie bien l'ensemble S_F du théorème 1.2.2. Expliquons maintenant comment calculer l'ensemble d'entiers Q_N considéré à l'étape (2.c). On rappelle que cet ensemble était défini dans le lemme 1.2.6 par

$$Q_N = \{m \in \mathbb{N} \,|\, m \text{ est sans facteur carré et } \Psi(m) \leq N\}.$$

On commence par construire l'ensemble $P_N = \{p_1, \ldots, p_t\}$ des nombres premiers inférieurs ou égaux à N, ce qui peut être fait à l'aide du crible d'Ératosthène en au plus

$$O(N \log^3(N) \log\log(N))$$

opérations binaires, en utilisant [**GG03**, Theorem 18.10]. On détermine ensuite les sous-ensembles $\{q_1, \ldots, q_r\}$ de P_N pour lesquels

$$2 + \sum_{i=1}^{r} (q_i - 2) \leq N. \tag{1.2.8}$$

Vérifier l'inégalité (1.2.8) nécessite d'additionner au plus N entiers de longueur binaire majorée par $\log N$, ce qui peut être fait en au plus $O(N \log N)$ opérations binaires pour un sous-ensemble donné $\{q_1, \ldots, q_r\}$ de P_N. De plus, $\#P_N < N$ donc cet ensemble possède au plus 2^N sous-ensembles et vérifier l'inégalité (1.2.8) pour chacun d'eux nécessite au plus $O(2^N N \log N)$ opérations binaires. Finalement, pour chaque sous-ensemble $\{q_1, \ldots, q_r\}$ de P_N satisfaisant l'inégalité (1.2.8), on calcule le produit $\prod_{i=1}^{r} q_i$. L'ensemble de tous ces produits donne finalement Q_N.

Calculer chacun de ces produits nécessite d'effectuer au plus N produits d'entiers de longueur binaire majorée par $\log N$ et puisque ce produit n'excède jamais

$$\exp(3\sqrt{N \log N})$$

d'après le lemme 1.2.6, ceci peut être fait en au plus $O(N^2 \log N)$ opérations binaires. Effectuer cette opération pour chacun des $\{q_1, \ldots, q_s\}$ peut être fait en au plus $O(2^N N^2 \log N)$ opérations binaires et ceci est aussi le nombre d'opérations binaires nécessaires pour déterminer Q_N.

Expliquons maintenant comment on peut tester la divisibilité $\Phi_{m_j}(X)|G_j(X)$ à l'étape (2.c.ii). Pour faire cela, on utilise un algorithme simple décrit par Filaseta et Schinzel [**FS04**, Theorem 3]. Celui-ci nécessite de factoriser l'entier m_j et est basé sur l'équivalence suivante :

$$\Phi_{m_j}(X)|G_j(X) \Longleftrightarrow X^{m_j} - 1 \mid G_j(X) \times \prod_{p|m_j} \left(X^{m_j/p} - 1\right).$$

On renvoie à **[FS04]** pour la description complète de cet algorithme et pour son temps d'exécution. Dans notre cas, puisqu'on connait déjà la factorisation de l'entier m_j, certaines étapes de leur algorithme peuvent être omises et la relation de divisibilité $\Phi_{m_j}(X)|G_j(X)$ peut être vérifiée en

$$O\left(N^3 (\log N)^2 2^{3\sqrt{N \log N}} (h + \mathsf{I}(\log d))\right)$$

opérations binaires.

On peut maintenant estimer la complexité totale de l'algorithme A. Dans ce calcul, on considère comme négligeable le coût de l'énumération des partitions et des éléments de Q_N. On rappelle qu'à l'étape (2.a), les polynômes G_j sont définis par :

$$F_j(X) = X^{b_j} G_j(X^{e_j}).$$

Pour obtenir ces polynômes, on doit effectuer N soustractions d'entiers de longueur binaire majorée par $\log d$, on doit ensuite effectuer au plus N calculs de pgcd d'entiers de longueur binaire majorée par $\log d$, et finalement effectuer N divisions euclidiennes sur des entiers de longueur binaire majorée par $\log d$. Tous ces calculs nécessitent au plus $O(N\mathsf{I}(\log d) \log\log d)$ opérations binaires.

A l'étape (2b), on doit calculer au plus $s-1$ pgcd d'entiers de longueur binaire majorée par $\log d$ pour calculer e. On doit ensuite effectuer au plus s divisions euclidiennes sur des entiers de longueur binaire majorée par $\log d$ pour obtenir les entiers e'_j. En tout, ceci nécessite donc $O(N\mathsf{I}(\log d) \log\log d)$ opérations binaires.

A l'étape (2c), pour obtenir les entiers m_j, on effectue s divisions et s calculs de pgcd d'entiers de longueur binaire majorée par $\log d$ et ceci nécessite encore $O(N\mathsf{I}(\log d)\log\log d)$ opérations binaires. Pour vérifier les relations de divisibilité, on applique s fois l'algorithme de Filaseta et Schinzel [**FS04**, Theorem 3] et ceci peut être fait en

$$O\left(N^4(\log N)^2 2^{3\sqrt{N\log N}}(h+\mathsf{I}(\log d))\right)$$

opérations binaires.

De plus, comme il y a au plus $N!$ partitions à considérer à l'étape (2) d'après le lemme 1.2.5 et puisqu'il y a au plus

$$\exp\left(3\sqrt{N\log N}\right)$$

entiers à considérer à l'étape (2.c) d'après le lemme 1.2.6, le nombre total d'opérations effectuées par l'algorithme A peut être majoré par :

$$O\left(N^N(\mathsf{I}(\log d)\log\log d+h)\right).$$

Ceci termine la démonstration du théorème 1.2.2. □

1.2.3. Une conséquence de l'algorithme A. — Dans leur article [**FGS08**], les auteurs discutent également du problème de la recherche de la multiplicité des facteurs cyclotomiques d'un polynôme F à coefficients entiers. Ils souhaitent en particulier décrire le produit de tous les facteurs cyclotomiques du polynôme F. Ils considèrent par exemple le polynôme

$$X^{(a-1)^2}+X^a-X-1.$$

On peut montrer que le produit de tous les facteurs cyclotomiques de ce polynôme est

$$\Phi(X)=\frac{(X^a-1)(X^{a-1}-1)}{X-1}=\prod_{m|a}\Phi_m(X)\times\prod_{\substack{m|(a-1)\\ m\neq 1}}\Phi_m(X).$$

Lorsqu'on développe cette expression, on constate que le polynôme Φ possède $2a-2$ termes non nuls ; il est donc impossible de l'expliciter avec une complexité polynomiale en le logarithme du degré de F, i.e. polynomiale en $\log a$. Remarquons que le polynôme $(X^a-1)(X^{a-1}-1)=X^{2a-1}-X^a-X^{a-1}+1$, qui ne possède que quatre termes non nuls, possède les mêmes facteurs cyclotomiques que le polynôme $X^{(a-1)^2}+X^a-X-1$ mais avec des multiplicités qui peuvent être différentes. Ainsi, en acceptant d'avoir des multiplicités différentes, on a obtenu un polynôme ayant beaucoup moins de termes non nuls et possédant les mêmes facteurs cyclotomiques que $X^{(a-1)^2}+X^a-X-1$. Ce phénomène est général, comme le montre le corollaire suivant.

Proposition 1.2.7. — *Soit $F \in \mathbb{Z}[X]$ possédant N termes non nuls. Alors il existe un polynôme $\Phi \in \mathbb{Z}[X]$ vérifiant :*

1. *Φ est produit de polynômes cyclotomiques,*
2. *Φ possède les mêmes facteurs cyclotomiques que F,*
3. *Φ possède au plus*

$$\exp\left(3N!\sqrt{N\log N}\exp\left(3\sqrt{N\log N}\right)\right)$$

termes non nuls.

Démonstration : Appelons $\{(m_1, e_1), \ldots, (m_t, e_t)\}$ l'ensemble de paires d'entiers renvoyé par l'algorithme A appliqué au polynôme F et posons :

$$\Phi := \prod_{i=1}^{t} \Phi_{m_i}\left(X^{e_i}\right). \tag{1.2.9}$$

Le polynôme Φ vérifie clairement les deux premières conditions de la proposition. Essayons maintenant de majorer le nombre de termes non nuls de Φ. A chaque boucle effectuée par l'algorithme A, on renvoie au plus une paire d'entiers, ainsi l'entier t peut être majoré par le nombre de boucles effectuées par l'algorithme. Or, à chaque boucle correspond une partition du polynôme F et un entier vérifiant les conditions du théorème 1.2.3. Comme il y a au plus $N!$ telles partitions et au plus $\exp\left(3\sqrt{N\log N}\right)$ tels entiers, on a

$$t \leq N!\exp\left(3\sqrt{N\log N}\right).$$

De plus, pour chaque $1 \leq i \leq t$, on a aussi $m_i \leq \exp\left(3\sqrt{N\log N}\right)$. Donc, pour i compris entre 1 et t, le polynôme $\Phi_{m_i}\left(X^{e_i}\right)$ possède au plus $\exp\left(3\sqrt{N\log N}\right)$ termes non nuls. Finalement, en développant le produit (1.2.9), on obtient au plus

$$\left(\exp\left(3\sqrt{N\log N}\right)\right)^t \leq \exp\left(3N!\sqrt{N\log N}\exp\left(3\sqrt{N\log N}\right)\right)$$

termes non nuls. On a ainsi le résultat voulu. □

On a vu que la partie cyclotomique de $F(X) = X^{(a-1)^2} + X^a - X - 1$ possède exactement $2a-2$ termes non nuls. L'algorithme A appliqué à ce polynôme renvoie l'ensemble de paires d'entiers $\{(1, a), (1, a-1)\}$ donc le polynôme

$$\Phi := \Phi_1\left(X^a\right)\Phi_1\left(X^{a-1}\right) = X^{2a-1} - X^a - X^{a-1} + 1$$

qui n'a que 4 termes non nuls, possède les mêmes facteurs cyclotomiques que le polynôme $X^{(a-1)^2} + X^a - X - 1$. Dans ce cas, le polynôme $\Phi_1(X)$ est un facteur de F avec multiplicité 1 et est un facteur de Φ avec multiplicité 2 tandis que les autres facteurs cyclotomiques de F et Φ ont pour multiplicité 1.

1.2.4. Une famille de polynômes ayant beaucoup de facteurs cyclotomiques « séparés ». — Nous pensons que la taille de la sortie de l'algorithme A peut être exponentielle en la taille de l'entrée. Lorsque $N = 2n$ est un entier pair, on construit une famille de polynômes pour laquelle la sortie de l'algorithme ne peut être écrite avec moins de $n!$ paires d'entiers. Cependant, cette famille de polynôme possède un très grand degré (doublement exponentiel en N) et ne permet donc pas de démontrer notre affirmation sur la taille de la sortie.

Proposition 1.2.8. — *Soient $n \in \mathbb{N}$ et $p_1, \ldots, p_{n!}$ des nombres premiers distincts plus grand que $2n$. Alors il existe un polynôme $F \in \mathbb{Z}[X]$ ayant $2n$ termes non nuls et satisfaisant les deux conditions suivantes :*

1. $X^{p_r} - 1 | F(X)$ *pour* $1 \leq r \leq n!$.
2. $X^{p_r p_{r'}} - 1 \nmid F(X)$ *pour* $1 \leq r < r' \leq n!$.

Remarque : Notons qu'il serait très facile d'obtenir un analogue de cette proposition pour des polynômes ayant un nombre impair de termes non nuls en adaptant les arguments qui suivent.

Démonstration : Soit $F(X) = \sum_{i=1}^{n} X^{\alpha_i} - \sum_{i=n+1}^{2n} X^{\alpha_i}$. On va déterminer des valeurs de α_i pour lesquelles le polynôme F satisfait la proposition 1.2.8. On considère les partitions $\{J_1, \ldots, J_n\}$ de l'ensemble $\{1, \ldots, 2n\}$ constituées de paires $J_1, \ldots, J_n$ telles que chaque paire J_j contienne exactement un élément de l'ensemble $\{1, \ldots, n\}$ et un élément de l'ensemble $\{n+1, \ldots, 2n\}$. Par exemple,

$$\{\{1, n+1\}, \{2, n+2\}, \ldots, \{n, 2n\}\}$$

est l'une de ces partitions. Remarquons qu'il y a exactement $n!$ telles partitions que l'on énumère de la façon suivante :

$$\pi_r := \{J_{r,1}, \ldots, J_{r,n}\} \qquad \text{pour } 1 \leq r \leq n!.$$

A chacune des partitions $\pi_r = \{J_{r,1}, \ldots, J_{r,n}\}$, on associe de façon injective un nombre premier p_r strictement plus grand que $2n$. D'après le théorème des restes chinois, on peut choisir α_i tel que :

$$i \in J_{r,k} \Longleftrightarrow \alpha_i \equiv k - 1 \mod p_r \qquad \text{pour } 1 \leq r \leq n!.$$

Par construction, tous les α_i sont distincts puisqu'ils satisfont tous des systèmes de congruences différents. On a alors :

$$X^{p_r} - 1 \big| F(X) \qquad \text{pour } 1 \leq r \leq n!.$$

Montrons maintenant que le polynôme que nous venons de construire satisfait aussi la deuxième condition de la proposition 1.2.8. Raisonnons par l'absurde en

supposant par exemple que $X^{p_1p_2}-1$ divise $F(X)$. On a alors $\Phi_{p_1p_2}(X)|F(X)$. Ainsi $f(\zeta)=0$, où ζ désigne une racine de l'unité d'ordre p_1p_2. Soit également

$$f(\zeta)=\sum_{i=1}^{n}\zeta^{\alpha_i}-\sum_{i=n+1}^{2n}\zeta^{\alpha_i}=:S_1+\cdots+S_t,$$

une décomposition de $f(\zeta)$ en sommes minimales. Soit m_j l'ordre de S_j, pour j entre 1 et t. Comme $m_j|p_1p_2$, on a $m_j\in\{1,p_1,p_2,p_1p_2\}$ et m_j vérifie

$$\Psi(m_j)\leq \#S_j\leq 2n$$

d'après le théorème 1.2.3. Or on a

$$\begin{aligned}\Psi(p_1)&=p_1>2n,\\ \Psi(p_2)&=p_2>2n,\\ \Psi(p_1p_2)&=p_1+p_2-2>2n,\end{aligned}$$

et donc $m_j=1$. Ainsi, chaque S_j est un multiple d'une somme minimale de la forme $\pm1\cdots\pm1=0$. Comme S_j est minimale, elle doit posséder 2 termes non nuls et on a donc également $t=n$. Ainsi pour chaque $1\leq j\leq n$, on peut récrire la somme S_j de la façon suivante :

$$S_j=\zeta^{\alpha_{a_j}}-\zeta^{\alpha_{b_j}}$$

avec

$$1\leq a_j\leq n<b_j\leq 2n \quad\text{et}\quad \alpha_{a_j}\equiv\alpha_{b_j}\mod p_1p_2.$$

Alors

$$\alpha_{j,1}\equiv\alpha_{j,2}\mod p_1 \quad\text{et}\quad \alpha_{j,1}\equiv\alpha_{j,2}\mod p_2 \quad\text{pour } 1\leq j\leq n.$$

On a donc nécessairement $\pi_r=\pi_1$ et $\pi_r=\pi_2$. Ceci est une contradiction puisque les partitions π_1 et π_2 sont distinctes. On peut alors en conclure que $X^{p_1p_2}-1$ ne divise pas $F(X)$, ce qu'on voulait démontrer. □

Soient $N=2n$ et F le polynôme construit dans la preuve de la proposition 1.2.8. Il est facile de voir que lorsqu'on applique l'algorithme A à ce polynôme, on obtient au moins $n!$ paires d'entiers.

Montrons maintenant que le degré de F est doublement exponentiel en N. Ainsi la taille de l'entrée sera également exponentielle en N. On remarque tout d'abord que le degré de F peut être majoré par : $p_1p_2\cdots p_{n!}$ d'après le théorème des restes chinois. On peut donc majorer le degré du polynôme F par une double exponentielle en n, si on choisit $p_1,p_2,\ldots,p_{n!}$ les plus petits possibles. Déterminons maintenant une minoration de $\deg(F)$. On a $\deg(F)\geq|\alpha_{n+1}-\alpha_1|$, donc il suffit de minorer $|\alpha_{n+1}-\alpha_1|$. On considère pour cela toutes les partitions pour lesquelles 1

et $n+1$ sont regroupés dans la même paire. On vérifie facilement qu'il y a $(n-1)!$ telles partitions. On les énumère de la façon suivante : $\pi'_1, \ldots, \pi'_{(n-1)!}$ et on désigne par $p'_1, \ldots, p'_{(n-1)!}$ les nombres premiers correspondant à chacune de ces partitions. Comme α_1 et α_{n+1} sont égaux modulo chacun de ces nombres premiers, on a

$$\alpha_1 - \alpha_{n+1} \equiv 0 \mod p'_1 p'_2 \cdots p'_{(n-1)!}.$$

De plus, $\alpha_1 - \alpha_{n+1} \neq 0$ (car α_1 et α_{n+1} ne sont pas égaux modulo les autres nombres premiers) et on a donc

$$|\alpha_1 - \alpha_{n+1}| \geq p'_1 p'_2 \cdots p'_{(n-1)!}.$$

Ainsi $|\alpha_1 - \alpha_{n+1}|$ est plus grand que le produit des $(n-1)!$ premiers nombres premiers et on a :

$$|\alpha_1 - \alpha_{n+1}| \geq \exp\left((n-1)! \log\left((n-1)!\right)\right).$$

Finalement, le degré du polynôme F est bien doublement exponentiel en n donc cet exemple ne suffit pas à démontrer que la taille de la sortie peut ne pas être polynomiale en la taille de l'entrée.

Pour faciliter la compréhension de la proposition 1.2.8, nous détaillons maintenant la construction du polynôme obtenu dans la preuve de cette proposition dans les cas $N = 6$ et $N = 8$.

Exemple 1.2.9. — Soient $n = 3$, $N = 6$ et

$$F(X) = X^{\alpha_1} + X^{\alpha_2} + X^{\alpha_3} - X^{\alpha_4} - X^{\alpha_5} - X^{\alpha_6}.$$

pour des exposants α_i à déterminer. On fait les choix suivants de nombres premiers pour les $n! = 6$ partitions :

$$\begin{aligned}
\{\{1,4\},\{2,5\},\{3,6\}\} &\longmapsto 7 \\
\{\{1,4\},\{2,6\},\{3,5\}\} &\longmapsto 11 \\
\{\{1,5\},\{2,4\},\{3,6\}\} &\longmapsto 13 \\
\{\{1,5\},\{2,6\},\{3,4\}\} &\longmapsto 17 \\
\{\{1,6\},\{2,4\},\{3,5\}\} &\longmapsto 19 \\
\{\{1,6\},\{2,5\},\{3,4\}\} &\longmapsto 23
\end{aligned}$$

et on considère le système de congruences suivant :

	mod 7	mod 11	mod 13	mod 17	mod 19	mod 23
α_1	0	0	0	0	0	0
α_2	1	1	1	1	1	1
α_3	2	2	2	2	2	2
α_4	0	0	1	2	1	2
α_5	1	2	0	0	2	1
α_6	2	1	2	1	0	0

Les plus petites solutions positives de ce systèmes sont :

$$\begin{aligned} \alpha_1 &= 0 \qquad \alpha_4 = 6\,282\,199 \\ \alpha_2 &= 1 \qquad \alpha_5 = 2\,501\,941 \\ \alpha_3 &= 2 \qquad \alpha_6 = 6\,088\,721 \end{aligned}$$

qui donnent le polynôme suivant :

$$F(X) = 1 + X + X^2 - X^{2501941} - X^{6088721} - X^{6282199}$$

pour lequel l'algorithme A renvoie les paires :

$$\{(2,2),\,(1,7),\,(1,11),\,(1,13),\,(1,17),\,(1,19),\,(1,23)\}.$$

et on ne peut pas mieux regrouper ces paires d'après la proposition 1.2.8.

Exemple 1.2.10. — Pour $n = 4$ et $N = 8$, on construit le polynôme

$$\begin{aligned} F(X) =& 1 + X + X^2 + X^3 - X^{10649315971896428139150081202897150286932} \\ &- X^{10417696267221214855704118228748809801239} \\ &- X^{27477501331112879055248604558804562320 62} \\ &- X^{62691493819963495180758597285521842 6387} \end{aligned}$$

de la même façon que précédemment en considérant les 24 nombres premiers compris entre 11 et 107. Pour ce polynôme, l'algorithme A renvoie les paires d'entiers suivantes :

$$\begin{aligned} \{&(1,13),\,(1,17),\,(1,19),\,(1,22),\,(1,23),\,(1,31),\,(1,37),\,(1,41),\,(1,43), \\ &(1,47),\,(1,53),\,(1,58),\,(1,59),\,(1,61),\,(1,71),\,(1,79),\,(1,83),\,(1,89), \\ &(1,97),\,(1,101),\,(1,103),\,(1,107),\,(1,134),\,(1,146)\} \end{aligned}$$

qui correspondent à 29 facteurs cyclotomiques distincts et, d'après la proposition 1.2.8, on ne peut pas les regrouper en moins de 24 paires, comme ci-dessus.

1.3. Sous-variétés de torsion d'une hypersurface

Dans cette section, nous étendons les résultats précédents au cas des hypersurfaces, pour obtenir une représentation des points de torsion appartenant à une hypersurface. Plus précisément, on prouve le théorème 1.1.1 dans le cas où V est une hypersurface et on décrit l'algorithme B qui en découle dans la sous-section 1.3.3.

1.3.1. Polynôme cyclotomique généralisé. — Avant de traiter le cas des hypersurfaces, on fait quelques rappels sur les polynômes cyclotomiques généralisés. On définit tout d'abord une notation introduite par Schinzel. Soit

$$F(\boldsymbol{X}) = \sum_{i=1}^{N} a_i \boldsymbol{X}^{\boldsymbol{\alpha}_i} \in \mathbb{C}[X_1^{\pm 1}, \dots, X_n^{\pm 1}]$$

un polynôme de Laurent, on définit alors

$$JF(\boldsymbol{X}) = X_1^{a_1} \cdots X_n^{a_n} F(\boldsymbol{X}),$$

où les entiers $a_1, \dots, a_n$ sont aussi petits que possibles et de telle sorte que JF soit un polynôme. Détaillons cette construction sur un exemple. Si

$$F = -7X_1^4 X_2 X_4 - X_1^2 X_3^{-3} X_4 - 12X_4,$$

alors

$$JF = -7X_1^4 X_2 X_3^3 - X_1^2 - 12X_3^3.$$

En particulier, remarquons que la variable X_4 qui intervenait dans F a disparu dans JF.

On peut maintenant définir ce qu'est un *polynôme cyclotomique généralisé* : c'est un polynôme de la forme

$$J\Phi_m\left(\boldsymbol{X}^{\boldsymbol{\alpha}}\right),$$

où $m \in \mathbb{N}^*$, $\boldsymbol{\alpha} \in \mathbb{Z}^n$ et où Φ_m désigne comme d'habitude le m-ième polynôme cyclotomique. On peut montrer qu'un tel polynôme est irréductible dans $\mathbb{Q}[\boldsymbol{X}]$ si et seulement si $\boldsymbol{\alpha}$ est un vecteur *primitif*, c'est à dire si ses coordonnées sont premières entre elles dans leur ensemble. Cette notion généralise la notion de polynôme cyclotomique en plusieurs variables. Par exemple, le polynôme

$$J\Phi_6\left(X_1^5 X_2^{-3}\right) = X_1^{10} - X_1^5 X_2^3 + X_2^6$$

est cyclotomique généralisé.

Dans les articles, [**AKS07**] et [**KK06**], les auteurs décrivent des algorithmes pour déterminer les facteurs de petits degré d'un polynôme lacunaire $F \in \mathbb{Z}[\boldsymbol{X}]$ à plusieurs variables. Dans ces deux algorithmes, on doit déterminer de deux façons différentes les facteurs cyclotomiques généralisés et les facteurs non cyclotomiques

généralisés de F. Il est donc important de pouvoir déterminer les facteurs cyclotomiques généralisés d'un polynôme à plusieurs variables. Notre algorithme B permet lui, de déterminer les facteurs cyclotomiques généralisés de degré quelconque du polynôme F. Pour obtenir ces facteurs à partir de la sortie de l'algorithme B que nous allons décrire dans la section suivante, il suffit de ne conserver que les paires (L, ω) pour lesquelles L est de rang 1. En effet, ces paires correspondent à des translatés de sous-groupes de codimension 1 et donc à des facteurs cyclotomiques généralisés de F. On pourra consulter l'exemple d'application de l'algorithme B décrit dans l'annexe pour plus de détails.

1.3.2. Résultats préliminaires. — Tout d'abord, nous aurons besoin de la généralisation suivante du théorème de Conway et Jones 1.2.3 au cas multivarié.

Corollaire 1.3.1. — *Soient $\boldsymbol{\zeta} \in \mu_\infty^n$ un point d'ordre m et $F(\boldsymbol{X}) = \sum_{i=1}^N a_i \boldsymbol{X}^{\boldsymbol{\alpha}_i} \in \mathbb{Z}[X_1^{\pm 1}, \dots, X_n^{\pm 1}]$ un polynôme de Laurent. Si*

1. $F(\boldsymbol{\zeta}) = 0$,
2. *aucune somme extraite de $F(\boldsymbol{\zeta})$ ne s'annule,*
3. $\displaystyle\sum_{1 \leq i,j \leq N} \mathbb{Z}(\boldsymbol{\alpha}_i - \boldsymbol{\alpha}_j) = \mathbb{Z}^n$,

alors m est sans facteur carré et $\Psi(m) \leq N$.

Démonstration : On a juste à montrer que $S := \sum_{i=1}^N a_i \boldsymbol{\zeta}^{\boldsymbol{\alpha}_i - \boldsymbol{\alpha}_1}$ satisfait les trois conditions du théorème 1.2.3. Les deux premières sont clairement satisfaites puisque $F(\boldsymbol{\zeta})$ est une somme minimale par hypothèse. Soit $M := \operatorname*{ppcm}_{1 \leq i \leq N} \left(\operatorname{ord}(\boldsymbol{\zeta}^{\boldsymbol{\alpha}_i - \boldsymbol{\alpha}_1})\right)$. Il reste à vérifier la dernière condition, c'est à dire à vérifier que $M = m$. Comme la relation de divisibilité $M | m$ est claire, il nous reste juste à montrer la relation de divisibilité inverse. Dans la suite, on pose $\boldsymbol{\zeta} = (\zeta_m^{d_1}, \dots, \zeta_m^{d_n})$, où ζ_m est une racine de l'unité d'ordre m. Remarquons d'abord que pour tout $b_1, \dots, b_N \in \mathbb{Z}$, on a :

$$\operatorname{ord}\left(\boldsymbol{\zeta}^{\sum_{i=1}^N b_i(\boldsymbol{\alpha}_i - \boldsymbol{\alpha}_1)}\right) \Big| M.$$

De plus, comme $\displaystyle\sum_{i=1}^N \mathbb{Z}(\boldsymbol{\alpha}_i - \boldsymbol{\alpha}_1) = \mathbb{Z}^n$, il existe $b_1, \dots, b_N \in \mathbb{Z}$ tels que

$$\sum_{i=1}^N b_i(\boldsymbol{\alpha}_i - \boldsymbol{\alpha}_1) = (1, 0, \dots, 0).$$

Ainsi,

$$\operatorname{ord}\left(\zeta_m^{d_1}\right) = \operatorname{ord}\left(\boldsymbol{\zeta}^{\sum_{i=1}^N b_i(\boldsymbol{\alpha}_i - \boldsymbol{\alpha}_1)}\right) \Big| M$$

et de la même façon

$$\operatorname{ord}\left(\zeta_m^{d_i}\right) \mid M \qquad \text{pour } 2 \leq i \leq N.$$

Finalement, on a

$$m = \operatorname*{ppcm}_{1\leq i\leq N}\left(\operatorname{ord}\left(\zeta_m^{d_i}\right)\right) \mid M.$$

et la dernière condition du théorème 1.2.3 est vérifiée. □

Nous aurons également besoin de savoir calculer une base d'un sous-groupe de $\mathbb{Z}^n$ à partir d'un système de générateurs. Soient $\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_r \in \mathbb{Z}^n$ des vecteurs qui engendrent un sous-groupe R de $\mathbb{Z}^n$ de rang $k \leq r$ et M la matrice de taille $r \times n$ dont les lignes sont les vecteurs $\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_r$. A partir de M, on peut déterminer une matrice H de taille $k \times n$, qui est triangulaire supérieure, et dont les lignes forment une base de R. La matrice H est uniquement déterminée et s'appelle la *forme normale d'Hermite* de M et peut être calculée à l'aide du lemme suivant :

Lemme 1.3.2. — *Il existe un algorithme qui détermine la forme normale d'Hermite d'une matrice M de taille $k \times n$ à coefficients entiers de longueur binaire majorée par* $\log d$ *en au plus*

$$O\left(k^4 n^2 \mathsf{I}(\log d) \log\log d\right)$$

opérations binaires.

L'algorithme relatif au lemme 1.3.2 est décrit dans [**LS96**]. Les auteurs de cet article donnent une preuve de ce résultat avec une meilleure complexité, mais celle-ci sera suffisante pour l'utilisation que nous en ferons.

On renvoie à cet article ou à [**GG03**, Algorithm 16.26] pour une définition plus précise et pour plus de détails sur la forme normale d'Hermite.

1.3.3. Recherche des sous-variétés de torsion - Algorithme B. — Considérons le polynôme $F(\boldsymbol{X}) = \sum_{i=1}^{N} a_i \boldsymbol{X}^{\boldsymbol{\alpha}_i} \in \mathbb{Z}[X_1, \ldots, X_n]$ et $\boldsymbol{\zeta} \in \mu_\infty^n$ tels que

$$F(\boldsymbol{\zeta}) = 0. \tag{1.3.1}$$

Comme on l'a déjà fait dans la preuve du théorème 1.2.2, on va déterminer des conditions équivalentes à la condition (1.3.1) que l'on pourra tester de façon algorithmique.

Soit $\{J_1, \ldots, J_s\}$ une partition de $\{1, \ldots, N\}$ telle que :

$$\sum_{i\in J_j} a_i \boldsymbol{\zeta}^{\boldsymbol{\alpha}_i} = 0 \qquad \forall\ 1 \leq j \leq s$$

et telle que chacune de ces sommes soit minimale. On pose $N_j := \#J_j$ et, pour j compris entre 1 et s,

$$F_j(\boldsymbol{X}) := \sum_{i \in J_j} a_i \boldsymbol{X}^{\boldsymbol{\alpha}_i}.$$

On a ainsi $F(\boldsymbol{X}) = \sum_{j=1}^{s} F_j(\boldsymbol{X})$. On renumérote alors les coefficients et les exposants de F_j de la façon suivante :

$$F_j(\boldsymbol{X}) := \sum_{i=1}^{N_j} a_{j,i} \boldsymbol{X}^{\boldsymbol{\alpha}_{j,i}} \ .$$

Soit alors, pour $1 \leq j \leq s$:

$$R_j := \sum_{i=1}^{N_j} \mathbb{Z}(\boldsymbol{\alpha}_{j,1} - \boldsymbol{\alpha}_{j,i})$$

le sous-groupe de $\mathbb{Z}^n$ engendré par les différences entre les exposants de F_j. On désigne par k_j le rang de ce sous-groupe et par $\boldsymbol{\lambda}_{j,1}, \ldots, \boldsymbol{\lambda}_{j,k_j}$ une base de R_j. On peut alors déterminer un polynôme de Laurent G_j en k_j variables tel que :

$$F_j(\boldsymbol{X}) = \boldsymbol{X}^{\boldsymbol{\alpha}_{j,1}} G_j(\boldsymbol{X}^{\boldsymbol{\lambda}_{j,1}}, \ldots, \boldsymbol{X}^{\boldsymbol{\lambda}_{j,k_j}}). \tag{1.3.2}$$

On pose pour $1 \leq l \leq k_j$,

$$\omega_{j,l} := \boldsymbol{\zeta}^{\boldsymbol{\lambda}_{j,l}}.$$

On a alors

$$G_j(\omega_{j,1}, \ldots, \omega_{j,k_j}) = 0, \qquad \text{pour } 1 \leq j \leq s. \tag{1.3.3}$$

Le polynôme de Laurent G_j et le point $(\omega_{j,1}, \ldots, \omega_{j,k_j})$ de $\mu_\infty^{k_j}$ satisfont les conditions du corollaire 1.3.1. Si m_j désigne l'ordre de $(\omega_{j,1}, \ldots, \omega_{j,k_j})$ dans $\mu_\infty^{k_j}$, on peut conclure que m_j est sans facteur carré et que $\Psi(m_j) \leq N_j$. Soient maintenant

$$R := \sum_{j=1}^{s} R_j$$

et $\{\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_k\}$ une base de R. Pour $1 \leq t \leq k$, on pose $\omega_t := \boldsymbol{\zeta}^{\boldsymbol{\lambda}_t}$ et on note m' l'ordre de $(\omega_1, \ldots, \omega_k)$. Par construction, on a :

$$m' | \operatorname{ppcm}(m_1, \ldots, m_s).$$

Comme dans la preuve du théorème 1.2.2, l'entier m' est sans facteur carré et satisfait $\Psi(m') \leq N - 2(s-1)$.
Si les égalités (1.3.3) sont satisfaites pour $1 \leq j \leq s$, on a $F(\boldsymbol{\zeta}) = 0$ pour tout $\boldsymbol{\zeta}$ satisfaisant :

$$\omega_t = \boldsymbol{\zeta}^{\boldsymbol{\lambda}_t} \qquad \text{pour } 1 \leq t \leq k.$$

On a ainsi :

$$F(\boldsymbol{\zeta}) = 0 \qquad \forall\ \boldsymbol{\zeta} \in B\left(L, (\omega_1, \ldots, \omega_k)\right),$$

où L désigne la matrice de taille $k \times n$ dont les lignes sont les vecteurs $\boldsymbol{\lambda}_1, \dots, \boldsymbol{\lambda}_k$. On rappelle que $B(L, (\omega_1, \dots, \omega_k))$ a été défini dans (1.1.3) de la façon suivante :

$$B(L, (\omega_1, \dots, \omega_k)) = \left\{ \boldsymbol{x} \in \mathbb{G}_m^n \mid \boldsymbol{x}^L = (\omega_1, \dots, \omega_k) \right\}.$$

Résumons ce que nous venons de prouver : si $F(\boldsymbol{\zeta}) = 0$ pour un certain $\boldsymbol{\zeta} \in \mu_\infty^n$, alors il existe $L \in M_{n,k}(\mathbb{Z})$ et $\boldsymbol{\omega} \in \mu_\infty^k$, où $k \leq n$, tels que

$$\boldsymbol{\zeta} \in B(L, \boldsymbol{\omega}) \quad \text{et} \quad B(L, \boldsymbol{\omega}) \subset V(F).$$

De plus, la matrice L correspond à une partition du support de F et peut être calculée à partir de celle-ci. D'un autre côté

$$\boldsymbol{\omega} \in Q_{k,N} = \left\{ \boldsymbol{\zeta} \in \mu_\infty^k \, : \, \Psi(\mathrm{ord}(\boldsymbol{\zeta})) \leq N \right\},$$

dont le cardinal est au plus $\exp(3(k+1)\sqrt{N \log N})$ d'après le lemme 1.2.6. Ainsi, l'algorithme va consister à tester les égalités (1.3.3) pour tous les choix possibles de partitions du support de F et pour tous les choix possibles de points dans $Q_{n,N}$. On peut faire cela de la façon suivante :

Algorithme B

Entrée : Un polynôme $F(\boldsymbol{X}) = \sum_{i=1}^N a_i \boldsymbol{X}^{\boldsymbol{\alpha}_i} \in \mathbb{Z}[X_1, \dots, X_n]$ donné par sa représentation lacunaire.
Sortie : Une liste S_F de sous-variétés de torsion satisfaisant le théorème 1.1.1.

1. $S_F \leftarrow \emptyset$.
2. Pour chaque partition $\{J_1, \dots, J_s\}$ de $\{1, \dots, N\}$ telle que $\#J_j \geq 2$ pour tout $1 \leq j \leq s$, faire :
   (a) Calculer pour $1 \leq j \leq s$, des polynômes de Laurent G_j et des vecteurs $\boldsymbol{\lambda}_{j,1}, \dots, \boldsymbol{\lambda}_{j,k_j}$ comme dans (1.3.2).
   (b) Déterminer une base $\boldsymbol{\lambda}_1, \dots, \boldsymbol{\lambda}_k$ de R et calculer, pour $1 \leq j \leq s$ et pour $1 \leq h \leq k_j$ les entiers $\delta_{j,h,t}$ tels que $\boldsymbol{\lambda}_{j,h} = \sum_{t=1}^k \delta_{j,h,t} \boldsymbol{\lambda}_t$.
   (c) Pour chaque entier m' sans facteur carré tel que $\Psi(m') \leq N - 2(s-1)$ et pour chaque $(\omega_1, \dots, \omega_k) \in \mu_\infty^k$ d'ordre m', faire :
      (i) Pour $1 \leq j \leq s$ et pour $1 \leq h \leq k_j$, faire $\omega_{j,h} \leftarrow \prod_{t=1}^k \omega_t^{\delta_{j,h,t}}$ et $m_j \leftarrow \mathrm{ord}(\omega_{j,1}, \dots, \omega_{j,k_j})$.
      (ii) Si pour chaque $1 \leq j \leq s$, on a $G_j(\omega_{j,1}, \dots, \omega_{j,k_j}) = 0$, alors faire $S_F \leftarrow S_F \cup \{(L, (\omega_1, \dots, \omega_k))\}$, où L est la matrice de taille $k \times n$ dont les lignes sont les vecteurs $\boldsymbol{\lambda}_1, \dots, \boldsymbol{\lambda}_k$.
3. Renvoyer S_F et terminer.

La discussion précédant cet algorithme garantit sa correction et il ne nous reste donc qu'à évaluer son temps d'exécution.

A l'étape (2.a) il nous faut d'abord calculer, pour j compris entre 1 et s, les différences $\boldsymbol{\alpha}_{j,i} - \boldsymbol{\alpha}_{j,1}$. Ceci nécessite $O(Nn \log d)$ opérations binaires. On a ensuite à déterminer une base $(\boldsymbol{\lambda}_{j,1}, \ldots, \boldsymbol{\lambda}_{j,k_j})$ du sous-groupe R_j de $\mathbb{Z}^n$ et on peut l'obtenir en calculant la forme normale d'Hermite de la matrice dont les lignes sont constituées des vecteurs $\boldsymbol{\alpha}_{j,i} - \boldsymbol{\alpha}_{j,1}$. Ceci nécessite $O(N^4 n^2 \mathsf{I}(\log d) \log\log d)$ opérations binaires si on utilise l'algorithme relatif au lemme 1.3.2. Pour déterminer les polynômes G_j, il nous faut calculer les coordonnées des vecteurs $\boldsymbol{\alpha}_{j,i} - \boldsymbol{\alpha}_{j,1}$ dans cette base $(\boldsymbol{\lambda}_{j,1}, \ldots, \boldsymbol{\lambda}_{j,k_j})$. Pour cela, il faut inverser un système linéaire de taille $n \times n$ dont les coefficients sont majorés par $2d$. Ceci peut être fait en $O(n^3 \mathsf{I}(\log d) \log\log d)$ opérations binaires en utilisant la [**MS04**, Proposition 31]. Finalement, on peut majorer le nombre d'opérations binaires effectuées à cette étape par

$$O\left(N^5 n^3 \mathsf{I}(\log d) \log\log d\right).$$

A l'étape (2.b), on calcule la forme normale d'Hermite d'une matrice dont les lignes sont constituées des vecteurs $\boldsymbol{\lambda}_{j,h}$, où $1 \leq j \leq s$ et $1 \leq h \leq k_j$. On obtient alors une base $(\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_k)$ de R. Finalement, on calcule les coordonnées des vecteurs $\boldsymbol{\lambda}_{j,h}$ dans cette base. Cette étape nécessite encore

$$O\left(N^5 n^3 \mathsf{I}(\log d) \log\log d\right)$$

opérations binaires.

A l'étape (2.c), chaque nombre ω_t est représenté par la paire d'entiers (ρ_t, m') telle que

$$\omega_t = \exp\left(\frac{2i\rho_t \pi}{m'}\right).$$

A l'étape (2.c.i), chaque nombre $\omega_{j,h}$ est représenté par la paire d'entiers

$$(d_{j,h}, m') := \left(\sum_{t=1}^{k} \delta_{j,h,t}\rho_t \mod m', m'\right)$$

telle que

$$\omega_{j,h} = \exp\left(\frac{2id_{j,h}\pi}{m'}\right).$$

Ce calcul nécessite

$$O\left(n\mathsf{I}(\log d) M\left(\exp\left(3\sqrt{N \log N}\right)\right)\right) = O\left(nN^2 \mathsf{I}(\log d)\right)$$

opérations binaires, puisque $m' \leq \exp\left(3\sqrt{N \log N}\right)$ d'après le lemme 1.2.6. On peut alors calculer, pour $1 \leq j \leq s$, l'entier

$$m_j := \operatorname*{ppcm}_{1 \leq h \leq k_j} \left(\frac{m'}{\operatorname{pgcd}(m', d_{j,h})} \right).$$

Ces calculs nécessitent $O(nN^2)$ opérations binaires et finalement, cette étape requiert en tout

$$O(nN^2 \mathsf{I}(\log d))$$

opérations binaires.

A l'étape (2.c.ii), on doit tester, pour $1 \leq j \leq s$, l'égalité

$$G_j(\omega_{j,1}, \ldots, \omega_{j,k_j}) = 0.$$

Ceci est équivalent à tester, pour $1 \leq j \leq s$ la relation de divisibilité

$$\Phi_{m_j}(X) \mid G_j(X^{d_{j,1}}, \ldots, X^{d_{j,k_j}})$$

ce qui peut être fait grâce à la méthode due à Filaseta et Schinzel, déjà utilisée à l'étape (2.c.ii) de l'algorithme A en

$$O\left(N^4 (\log N)^2 2^{3\sqrt{N \log N}} (h + \mathsf{I}(\log d))\right)$$

opérations binaires.

Comme il y a au plus $N!$ partitions à considérer à l'étape (2) d'après le lemme 1.2.5 et puisque, d'après le lemme 1.2.6, il y a au plus

$$\exp\left(3(n+1)\sqrt{N \log N}\right)$$

points de torsion à considérer à l'étape (2.c), le nombre total d'opérations exécutées par l'algorithme B peut être majoré par :

$$O\left(N^{nN} (\mathsf{I}(\log d) \log\log d + h)\right).$$

Ceci termine la preuve du théorème 1.1.1 dans le cas d'une hypersurface.

1.4. Le cas général - Algorithme C

On démontre dans cette section le théorème 1.1.1 dans le cas général. Dans tout ce qui suit, on fixe $F_1, \ldots, F_k \in \mathbb{Z}[X_1, \ldots, X_n]$ des polynômes que l'on suppose lacunaires et on veut déterminer une représentation des sous-variétés de torsion incluses dans :

$$V = \{\boldsymbol{x} \in \mathbb{G}_m^n : F_1(\boldsymbol{x}) = \cdots = F_k(\boldsymbol{x}) = 0\}.$$

On va en fait expliquer comment on peut modifier l'algorithme B pour traiter simultanément le cas de plusieurs polynômes. Nous appelons Algorithme C la procédure qui en résulte, c'est l'algorithme relatif au théorème 1.1.1. Remarquons

au passage que les mêmes modifications peuvent être apportées à l'algorithme A pour déterminer les facteurs cyclotomiques communs à plusieurs polynômes.

Soient $\boldsymbol{\zeta} \in V \cap \mu_\infty^n$, N_i le nombre de termes non nuls de F_i, pour $1 \leq i \leq k$, et $N := \max(N_1, \ldots, N_k)$. Pour chaque entier i entre 1 et k, on peut réduire l'égalité $F_i(\boldsymbol{\zeta}) = 0$ à des sommes minimales comme on l'a déjà fait dans les sections précédentes, en considérant une partition de l'ensemble $\{1, \ldots, N_i\}$. Une fois cette réduction faite pour chaque entier i, on peut se ramener à l'étape (2) de l'algorithme B et toutes les autres étapes de l'algorithme B peuvent être conservées.

Pour le calcul de la complexité, seuls le nombre de partitions et le nombre de points de torsion à considérer sont à modifier. Le nombre de partitions peut être majoré par

$$N_1! \times \cdots \times N_k! \leq (N!)^k$$

et le nombre de points de torsion apparaissant à l'étape (2.c) est maintenant majoré par

$$\exp\left(3(n+1)\sqrt{kN \log kN}\right).$$

Finalement, le nombre total d'opérations binaires exécutées par cet algorithme peut être majoré par :

$$O\left(N^{nkN}\left(\mathsf{I}(\log d) \log\log d + h\right)\right).$$

et ceci termine la démonstration du théorème 1.1.1. □

Cet algorithme sera utilisé dans la deuxième partie dans le cas particulier d'une sous-variété de $\mathbb{G}_m^2$ définie par trois polynômes.

1.5. Une borne pour le nombre de sous-variétés de torsion

On donne ici la démonstration du corollaire 1.1.2. Soit V une variété définie par k polynômes ayant chacun au plus N termes non nuls. L'algorithme C donne une représentation de $\overline{V_{\text{tors}}}$ comme réunion de sous-variétés de torsion. On remarque, qu'à chaque boucle effectuée par l'algorithme C, celui-ci renvoie au plus une sous-variété de torsion. Ainsi, le nombre t de sous-variétés de torsion dans cette réunion peut être borné par le nombre de boucles effectuées par l'algorithme. Chaque boucle correspond à un choix de partition des polynômes définissant V et à un choix d'un point de μ_∞^n d'ordre m sans facteur carré et vérifiant de plus $\Psi(m) \leq N$. Comme nous l'avons déjà remarqué précédemment, il y a au plus $(N!)^k$ telles partitions et le nombre de points de μ_∞^n à considérer peut être majoré par $\exp\left(3(n+1)\sqrt{kN \log kN}\right)$, d'après le lemme 1.2.6. Ainsi,

$$t \leq (N!)^k \exp\left(3(n+1)\sqrt{kN \log kN}\right),$$

ce qui est bien la borne que l'on souhaitait obtenir. □

CHAPITRE 2

SYSTÈMES SURDÉTERMINÉS D'ÉQUATIONS POLYNOMIALES

La résolution de systèmes d'équations polynomiales est au cœur de nombreuses branches des mathématiques telles que l'algèbre, la topologie, la géométrie et l'analyse numérique. Les avancées récentes dans ce domaine ont permis de nombreuses applications à d'autres domaines tels que la robotique, le traitement du signal, la biologie moléculaire, l'optimisation... C'est donc encore aujourd'hui un sujet de recherche important qui fait l'objet de nombreuses publications. Le problème est le suivant : étant donnés K un sous-corps de $\mathbb{C}$ et $F_1, \ldots, F_r \in K[X_1, \ldots, X_n]$, on cherche une représentation des solutions du système d'équations :

$$F_1(X_1, \ldots, X_n) = \cdots = F_r(X_1, \ldots, X_n) = 0. \tag{2.0.1}$$

Dans toute la suite, on se contentera du cas où K est un sous-corps de $\overline{\mathbb{Q}}$.

On peut dans certains contextes, et notamment dans des problèmes physiques d'optimisation ou en analyse numérique, se contenter de déterminer des solutions approchées du système (2.0.1). Dans ce cas, on peut obtenir des approximations des solutions avec n'importe quelle précision. Cependant, une telle résolution n'est pas suffisante si on souhaite par exemple paramétrer les solutions, compter leur multiplicité...

De nombreuses méthodes ont été introduites pour résoudre ce problème de façon formelle telles que la théorie de l'élimination et la théorie des bases de Gröbner qui fournissent des algorithmes de résolution exacte du système (2.0.1). En contrepartie, ces algorithmes sont beaucoup plus couteux et rendent les calculs impraticables si les polynômes $F_1, \ldots, F_r$ ont un degré élevé ou dépendent de beaucoup de variables. On pourra par exemple se référer à **[CLO97]** et à la partie V de **[GG03]** pour la description et l'analyse de la complexité de ces algorithmes.

Nous nous intéressons dans cette partie à un type particulier de systèmes d'équations polynômiales : les *systèmes surdéterminés*, c'est à dire les systèmes contenant « trop » d'équations, i.e. plus d'équations que la codimension de leur ensemble

de zéros communs. Les systèmes que nous considérons sont définis par des polynômes lacunaires, c'est à dire des polynômes ayant peu de termes non nuls mais potentiellement un grand degré.

Par exemple, le calcul du pgcd de deux polynômes à une variable permet d'obtenir un polynôme dont les zéros sont exactement les zéros communs des deux polynômes de départ. Dans la section 2.1, on décrit un algorithme (Algorithme D) qui, étant donnés deux polynômes à une variable définissant un ensemble de zéros communs Z, détermine un polynôme dont l'ensemble des zéros Z' coïncide avec Z en dehors de μ_∞, c'est à dire :

$$Z \backslash \mu_\infty \subseteq Z' \subseteq Z.$$

La complexité de cet algorithme est linéaire en le logarithme du degré des polynômes donnés en entrée. Cet algorithme est une variante de [**FGS08**, Algorithme B]. L'intérêt principal de notre version est que les méthodes utilisées s'étendent pour traiter un problème analogue en deux variables comme nous l'expliquons maintenant.

Une généralisation naturelle du calcul du pgcd de polynômes à une variable est la suivante : étant donnés trois polynômes $F_1, F_2, F_3 \in \mathbb{Q}[X, Y]$, peut-on représenter les zéros communs de ces polynômes avec moins d'équations ? La réponse est oui. En effet, si D désigne le pgcd de ces polynômes, alors la variété

$$V\left(F_1/D, F_2/D, F_3/D\right)$$

est de dimension 0. Or une variété de dimension 0 est toujours une intersection complète, donc il existe $Q_1, Q_2 \in \mathbb{Q}[X, Y]$ tels que

$$V\left(F_1/D, F_2/D, F_3/D\right) = V(Q_1, Q_2).$$

Comme on a

$$V(F_1, F_2, F_3) = V(D) \cup V\left(F_1/D, F_2/D, F_3/D\right),$$

on obtient

$$\begin{aligned} V(F_1, F_2, F_3) &= V(D) \cup V(Q_1, Q_2) \\ &= V(DQ_1, DQ_2). \end{aligned}$$

Ces polynômes peuvent se calculer à l'aide de méthodes d'élimination et/ou avec des bases de Gröbner, voir par exemple l'algorithme E2b. Or, une telle démarche est trop coûteuse si F_1, F_2 et F_3 ont un grand degré. L'objet principal de cette partie est d'obtenir le même type de résultat pour une variété définie par des polynômes lacunaires. Pour cela, on étend les méthodes utilisées pour l'algorithme D sans avoir directement recours aux bases de Gröbner. On obtient l'algorithme E

qui donne une représentation des zéros communs des polynômes F_1, F_2, F_3 mais un peu moins bonne que celle espérée. En effet, il permet de décrire les zéros communs des trois polynômes comme réunion d'intersections complètes en dehors de certains ouverts. On peut se référer à la section 2.3 pour un énoncé plus précis.

2.1. Le cas univarié

On rappelle que si P est un polynôme à coefficients entiers, on appelle hauteur de ce polynôme le maximum en module de ses coefficients.

Le but de cette section est de démontrer le théorème suivant :

Théorème 2.1.1. — *(Algorithme D) Il existe un algorithme qui a la propriété suivante. Soient $f_1, f_2 \in \mathbb{Z}[X]$ donnés par leur représentation lacunaire, de degré au plus d et de hauteur au plus h et possédant chacun au plus N termes non nuls. L'algorithme détermine la représentation lacunaire d'un polynôme $p \in \mathbb{Z}[X]$ vérifiant :*

1. $p | \operatorname{pgcd}(f_1, f_2)$ *dans* $\mathbb{Z}[X]$.
2. $\operatorname{pgcd}(f_1, f_2)/p$ *est produit de facteurs cyclotomiques.*

Cet algorithme nécessite au plus

$$O_{N,h}(\log d)$$

opérations binaires. De plus, le nombre de termes non nuls de p et sa hauteur sont majorés par $O_{N,h}(1)$.

En particulier, si f_1 et f_2 n'ont pas de facteur cyclotomique en commun, le polynôme p est leur pgcd. En analysant l'algorithme D relatif à ce théorème, on peut obtenir comme corollaire que si f_1 et f_2 n'ont pas de facteur cyclotomique en commun, le nombre de termes non nuls de leur pgcd peut être borné uniquement en fonction du nombre de termes non nuls et de la hauteur de f_1 et f_2. Cette remarque apparaît déjà dans **[FGS08]**. La même conclusion n'est plus valable si f_1 et f_2 ont un facteur cyclotomique en commun, comme le montre l'exemple suivant dû à Schinzel **[Schi02]** : si a et b sont deux entiers positifs premiers entre eux, alors

$$\operatorname{pgcd}\left(X^{ab}-1, (X^a-1)(X^b-1)\right) = \frac{(X^a-1)(X^b-1)}{X-1}$$

et ce polynôme possède $2\min(a,b)$ termes non nuls. Ainsi, le nombre de termes non nuls du pgcd de deux polynômes dépend en général de leur degré. Dans ce cas particulier, on remarque que le polynôme $(X^a-1)(X^b-1)$ a les même zéros que le polynôme $\operatorname{pgcd}\left(X^{ab}-1, (X^a-1)(X^b-1)\right)$ et un nombre de termes non

nuls (quatre) indépendant de a et b. Ce phénomène est général, comme le montre le résultat suivant :

Proposition 2.1.2. — *Soient $f_1, f_2 \in \mathbb{Z}[X]$ de hauteur majorée par h et ayant chacun au plus N termes non nuls. Alors il existe un polynôme $q \in \mathbb{Z}[X]$ qui a les mêmes zéros que le polynôme* $\mathrm{pgcd}(f_1, f_2)$ *et qui possède $O_{N,h}(1)$ termes non nuls.*

Démonstration : On a vu à la section 1.4 que l'algorithme A peut être modifié de façon à déterminer une représentation des facteurs cyclotomiques communs de deux polynômes. On peut donc, en suivant la même démarche que dans la preuve de la proposition 1.2.9, construire un polynôme Φ qui est produit de polynômes cyclotomiques, qui a les mêmes facteurs cyclotomiques que $\mathrm{pgcd}(f_1, f_2)$ et qui possède $O_N(1)$ termes non nuls. Appelons maintenant p le polynôme dont l'existence est assurée par le théorème 2.1.1 appliqué aux polynômes f_1 et f_2. Ce polynôme a $O_{N,h}(1)$ termes non nuls et est un diviseur de $\mathrm{pgcd}(f_1, f_2)$ dont les facteurs non cyclotomiques coïncident avec ceux de $\mathrm{pgcd}(f_1, f_2)$. Posons finalement

$$q := p \cdot \Phi.$$

Il est maintenant clair que q vérifie les conditions de la proposition. □

L'algorithme D sera décrit dans la section 2.1.2. C'est une version améliorée et simplifiée de l'algorithme B de l'article **[FGS08]**. Les simplifications que nous apportons permettent de généraliser cet algorithme pour des polynômes à deux variables comme nous le verrons dans les sections suivantes.

2.1.1. Résultats préliminaires. — Un des ingrédients essentiels à la démonstration du théorème 2.1.1 est un résultat dû à Bombieri et Zannier, déjà utilisé dans **[FGS08]**, que nous détaillons après avoir introduit quelques notations.

On rappelle que $\mathbb{G}_m^k$ désigne le groupe multiplicatif $(\overline{\mathbb{Q}}^*)^k$. Pour $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{Z}^k$, on note $\langle \boldsymbol{a}, \boldsymbol{b} \rangle$ le produit scalaire usuel de $\boldsymbol{a}$ et $\boldsymbol{b}$ et $||\boldsymbol{a}||$ le maximum du module des coordonnées de $\boldsymbol{a}$. Pour $\sigma \in \overline{\mathbb{Q}}^*$ et $\boldsymbol{a} \in \mathbb{Z}^k$, on note

$$\sigma^{\boldsymbol{a}} := (\sigma^{a_1}, \ldots, \sigma^{a_k}) \in \mathbb{G}_m^k.$$

Pour $0 \leq i \leq k$, on peut étendre les définitions données dans l'introduction et noter :

$$\mathcal{H}_i := \bigcup H,$$

où la réunion porte sur tous les sous-groupes algébriques de $\mathbb{G}_m^k$ de dimension au plus i. On a alors :

$$\mathcal{H}_0 \subsetneq \mathcal{H}_1 \subsetneq \cdots \subsetneq \mathcal{H}_k = \mathbb{G}_m^k,$$

et on remarque que si $0 \leq i \leq k$, l'ensemble $\mathcal{H}_i$ est l'ensemble des points $\boldsymbol{x} \in \mathbb{G}_m^k$ tels que le rang du sous-groupe de $\overline{\mathbb{Q}}^*$ engendré par les coordonnées de $\boldsymbol{x}$ est inférieur ou égal à i. Par exemple, l'ensemble $\mathcal{H}_0$ est l'ensemble des points de torsion de $\mathbb{G}_m^k$, c'est à dire μ_∞^k et avec les notations ci-dessus :

$$\mathcal{H}_2 = \left\{ \boldsymbol{\zeta} u^{\boldsymbol{a}} v^{\boldsymbol{b}} : \boldsymbol{\zeta} \in \mu_\infty^k,\ u, v \in \overline{\mathbb{Q}}^*,\ \boldsymbol{a}, \boldsymbol{b} \in \mathbb{Z}^k \right\}.$$

Le résultat suivant était conjecturé par Schinzel qui l'a prouvé dans le cas particulier d'une variété de $\mathbb{G}_m^3$ définie par deux polynômes. La preuve du cas général est due à Bombieri et Zannier (manuscrit non publié [**BZ98**] puis publié dans [**Z00**]). L'énoncé a ensuite été précisé et la preuve simplifiée dans [**BMZ07**]. C'est ce dernier énoncé que l'on donne ici pour une variété non nécessairement irréductible.

Théorème 2.1.3. — *Soient $k \geq 2$ et χ une variété de $\mathbb{G}_m^k$ ayant au moins une composante de codimension au moins égale à 2. Alors il existe $B_1(\chi)$ ne dépendant que de χ vérifiant la propriété suivante. Soient $\tau \in \overline{\mathbb{Q}}^*$ et $\boldsymbol{a} \in \mathbb{Z}^k$ tels que le point $\boldsymbol{x} = \tau^{\boldsymbol{a}}$ appartient à une composante irréductible de χ de codimension au moins 2. Alors il existe $\boldsymbol{b} \in \mathbb{Z}^k \backslash \{0\}$ tel que*

$$\boldsymbol{x}^{\boldsymbol{b}} = 1 \qquad \text{et} \qquad ||\boldsymbol{b}|| \leq B_1(\chi).$$

En particulier, si τ n'est pas une racine de l'unité, on a $\langle \boldsymbol{a}, \boldsymbol{b} \rangle = 0$.

Démonstration : Ce résultat est le théorème 4.1 de [**BMZ07**] dans le cas où χ est irréductible et de codimension au moins 2. Dans le cas général, si χ est pure de codimension 1, le résultat est vide et n'importe quelle valeur de $B_1(\chi)$ convient. On suppose donc que χ possède au moins une composante irréductible de codimension au moins 2. On écrit alors $\chi = \chi_1 \cup \chi_2$ où $\mathrm{codim}(\chi_1)=1$ et $\mathrm{codim}(\chi_2) \geq 2$. Puis on écrit χ_2 comme réunion de composantes irréductibles :

$$\chi_2 = \chi_{2,1} \cup \cdots \cup \chi_{2,r}.$$

Pour $1 \leq i \leq r$, on peut appliquer le théorème 4.1 de [**BMZ07**] à $\chi_{2,i}$ et trouver une borne $B_1(\chi_{2,i})$ vérifiant les conclusions du théorème 2.1.3 appliqué à $\chi_{2,i}$. Comme $\boldsymbol{x}$ appartient à l'un des $\chi_{2,i}$, il existe un vecteur $\boldsymbol{b} \in \mathbb{Z}^k \backslash \{0\}$ tel que

$$\boldsymbol{x}^{\boldsymbol{b}} = 1 \qquad \text{et} \qquad ||\boldsymbol{b}|| \leq B_1(\chi) := \max_{1 \leq i \leq r} B_1(\chi_{2,i}).$$

□

Remarque. L'algorithme D qui s'appuie sur ce théorème sera d'autant plus efficace que $B_1(\chi)$ sera petit. Cette borne est calculable, comme l'a fait remarquer Zannier aux auteurs de [**FGS08**] (voir la remarque qui suit le théorème 2 dans op. cit.). Mais malheureusement ce calcul n'a encore jamais été explicité. En effet, la démonstration du théorème 4.1 de [**BMZ07**] utilise plusieurs résultats sophistiqués non quantifiés comme des minorations de hauteurs et un théorème de structure (le théorème [**BMZ07**, Théorème 1.4]). Il faudrait donc au préalable quantifier ces résultats, ce qui demanderait probablement un travail considérable.

Obtenir une borne calculable nous permettrait de rendre notre algorithme D effectif. De plus, on pourrait alors également exhiber la dépendance de la complexité de l'algorithme D en la hauteur et en le nombre de termes non nuls des polynômes en entrée. Nous nous intéressons donc uniquement à la dépendance de l'algorithme D en le degré des polynômes en entrée.

Nous décrivons maintenant deux procédures (les algorithmes D1 et D2) nécessaires à l'exécution de l'algorithme D.

Lemme 2.1.4. — *(Algorithme D1) Il existe un algorithme qui a la propriété suivante. Etant donnés deux entiers k et r avec $k > r \geq 1$ et $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_r \in \mathbb{Z}^k$ des vecteurs linéairement indépendants, l'algorithme détermine un ensemble de $k-r$ vecteurs $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{k-r}\}$ qui est une base LLL-réduite du sous-groupe de $\mathbb{Z}^k$ formé des vecteurs orthogonaux aux vecteurs $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_r$. De plus, si $\max_{1\leq i\leq r} ||\boldsymbol{u}_i|| \leq m$, on a $\max_{1\leq i\leq k-r} ||\boldsymbol{v}_i|| = O_k(m)$.*

Démonstration : Voir [**C93**, Algorithm 2.7.2, p. 98]. □

La complexité de cet algorithme est polynomiale en la taille de l'entrée. Nous ne l'évaluons pas plus précisément car nous n'appliquons cet algorithme qu'à des vecteurs dont la taille est indépendante du degré des polynômes en entrée de l'algorithme D.

Pour $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{Z}^k$, on définit

$$X^{\boldsymbol{a}} := (X^{a_1}, \ldots, X^{a_k}),$$
$$X^{\boldsymbol{a}}Y^{\boldsymbol{b}} := (X^{a_1}Y^{b_1}, \ldots, X^{a_k}Y^{b_k}).$$

Proposition 2.1.5. — *(Algorithme D2) Il existe un algorithme qui a la propriété suivante. Soient $k \geq 2$, $F_1, F_2 \in \mathbb{Z}[X_1, \ldots, X_k]$ de degré au plus δ et de hauteur au plus h et $\boldsymbol{a} \in \mathbb{Z}^k$ satisfaisant $||\boldsymbol{a}|| \leq d$. S'il existe $\tau \in \overline{\mathbb{Q}}^* \backslash \mu_\infty$ tel que $\tau^{\boldsymbol{a}}$ appartient à une composante irréductible de $V(F_1, F_2) \subseteq \mathbb{G}_m^k$ de codimension 2, l'algorithme*

renvoie deux polynômes $F_1', F_2' \in \mathbb{Z}[X_1, \ldots, X_{k-1}]$ *et un vecteur* $\boldsymbol{a}' \in \mathbb{Z}^{k-1}$ *tels que*

$$\begin{aligned} F_1(X^{\boldsymbol{a}}) &= F_1'(X^{\boldsymbol{a}'}) \\ F_2(X^{\boldsymbol{a}}) &= F_2'(X^{\boldsymbol{a}'}) \end{aligned} \tag{2.1.1}$$

en au plus

$$O_{k,h,\delta}(\log d)$$

opérations binaires. De plus, les polynômes F_1' *et* F_2' *sont de degré et de hauteur majorés par* $O_{k,h,\delta}(1)$ *et* $||\boldsymbol{a}'|| = O_{k,h,\delta}(d)$.

Décrivons tout d'abord la procédure :

Algorithme D2

Entrée : Un triplet $(F_1, F_2, \boldsymbol{a})$, où $F_1, F_2 \in \mathbb{Z}[X_1, \ldots, X_k]$ et $\boldsymbol{a} \in \mathbb{Z}^k$.
Sortie : Un triplet $(F_1', F_2', \boldsymbol{a}')$, où $F_1', F_2' \in \mathbb{Z}[X_1, \ldots, X_{k-1}]$ et $\boldsymbol{a}' \in \mathbb{Z}^{k-1}$ satisfaisant la condition (2.1.1) de la proposition précédente ou « RIEN ».

1. Tester parmi tous les vecteurs de $\mathbb{Z}^k \backslash \{0\}$ de norme au plus $B_1(V(F_1, F_2))$ (où B_1 est défini au théorème 2.1.3) s'il en existe un qui est orthogonal à $\boldsymbol{a}$. Si oui, appeler $\boldsymbol{b}$ un tel vecteur, sinon renvoyer « RIEN » et terminer.
2. Déterminer une base $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{k-1}\}$ du sous-groupe de $\mathbb{Z}^k$ formé des vecteurs orthogonaux à $\boldsymbol{b}$ à l'aide de l'algorithme D1.
3. Déterminer les coordonnées $\boldsymbol{a}'$ de $\boldsymbol{a}$ dans cette base.
4. Poser $F_1'(X_1, \ldots, X_{k-1}) := F_1\left(X_1^{b_{1,1}} \cdots X_{k-1}^{b_{k-1,1}}, \ldots, X_1^{b_{1,k}} \cdots X_{k-1}^{b_{k-1,k}}\right)$,
$F_2'(X_1, \ldots, X_{k-1}) := F_2\left(X_1^{b_{1,1}} \cdots X_{k-1}^{b_{k-1,1}}, \ldots, X_1^{b_{1,k}} \cdots X_{k-1}^{b_{k-1,k}}\right)$.
5. Renvoyer $(F_1', F_2', \boldsymbol{a}')$ et terminer.

Démonstration de la proposition 2.1.5 : Montrons que la procédure D2 satisfait bien les conditions de cette proposition. Supposons qu'il existe $\tau \in \overline{\mathbb{Q}}^* \backslash \mu_\infty$ tel que $\tau^{\boldsymbol{a}}$ appartient à une composante irréductible de $V(F_1, F_2)$ de codimension 2. D'après le théorème 2.1.3, il existe $\boldsymbol{b} \in \mathbb{Z}^k \backslash \{0\}$ orthogonal à $\boldsymbol{a}$ vérifiant

$$||\boldsymbol{b}|| \leq B_1(V(F_1, F_2)).$$

Supposons qu'on choisisse ce vecteur à l'étape 1. Comme $\boldsymbol{a}$ et $\boldsymbol{b}$ sont orthogonaux, le vecteur $\boldsymbol{a}$ est bien une combinaison entière des vecteurs $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{k-1}$ calculés à l'étape 2. Pour voir que les polynômes F_1' et F_2' définis à l'étape 4 satisfont la proposition 2.1.5, il suffit de faire la substitution $X_i := X^{a_i'}$, pour $1 \leq i \leq k-1$.

Evaluons maintenant le nombre d'opérations binaires effectuées par l'algorithme ainsi que la taille de la sortie. La première étape nécessite de tester, pour $O_{k,h,\delta}(1)$ vecteurs de norme $O_{k,h,\delta}(1)$, s'ils sont orthogonaux à $\boldsymbol{a}$ et nécessite $O_{k,h,\delta}(\log d)$ opérations binaires. Remarquons que si l'algorithme ne s'arrête pas à l'étape 1, on a $||\boldsymbol{b}|| \leq B_1(V(F_1, F_2)) = O_{k,h,\delta}(1)$. A l'étape 2, on a également $||\boldsymbol{b}_i|| = O_{k,h,\delta}(1)$ d'après le lemme 2.1.4 et l'application de l'algorithme D1 au vecteur $\boldsymbol{b}$ nécessite encore $O_{k,h,\delta}(1)$ opérations binaires. Pour calculer les coordonnées de $\boldsymbol{a}$ à l'étape 3, il suffit d'inverser la matrice des $\boldsymbol{b}_i$, ce qui nécessite $O_{k,h,\delta}(1)$ opérations binaires, puis de multiplier le résultat par $\boldsymbol{a}$. Le coût de cette dernière multiplication est majoré par $O_{k,h,\delta}(\log d)$ qui est également une borne pour le nombre d'opérations effectuées à cette étape. On obtient ainsi également la borne $||\boldsymbol{a}'|| = O_{k,h,\delta}(d)$. L'étape 4 de l'algorithme n'est qu'une substitution qui nécessite $O_{k,h,\delta}(1)$ opérations binaires. On observe également que les polynômes F_1' et F_2' sont de hauteur et de degré majorés par $O_{k,h,\delta}(1)$ comme annoncé dans la proposition 2.1.5. □

Remarque. A l'étape 1, parcourir l'ensemble de tous les vecteurs de norme majorée par $B_1(V(F_1, F_2))$ n'est pas réalisable en pratique. Pour éviter ce problème, on propose la modification suivante. On commence par déterminer une base LLL-réduite $\{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_{k-1}\}$ du sous-groupe de $\mathbb{Z}^k$ formé des vecteurs orthogonaux au vecteur $\boldsymbol{a}$ à l'aide de l'algorithme D1. D'après le théorème 2.1.3, s'il existe $\tau \in \overline{\mathbb{Q}}^* \backslash \mu_\infty$ tel que $\tau^{\boldsymbol{a}}$ appartient à une composante irréductible de la variété $V(F_1, F_2)$ de codimension 2, alors il existe $\boldsymbol{b} \in \mathbb{Z}^k \backslash \{0\}$ orthogonal à $\boldsymbol{a}$ tel que

$$||\boldsymbol{b}|| \leq B_1(V(F_1, F_2)).$$

Par [**GG03**, Theorem 16.9, p. 466], on a alors

$$\min_{1 \leq i \leq k-1} ||\boldsymbol{a}_i|| \leq 2^{\frac{k-1}{2}} B_1(V(F_1, F_2)).$$

En posant $\boldsymbol{b} := \boldsymbol{a}_{i_0}$, où $||\boldsymbol{a}_{i_0}|| = \min_{1 \leq i \leq k-1} ||\boldsymbol{a}_i||$, on obtient un vecteur non nul orthogonal au vecteur $\boldsymbol{a}$ et de norme contrôlée sans avoir à parcourir l'ensemble de tous les vecteurs de norme au plus $B_1(V(F_1, F_2))$.

Cependant, la complexité théorique de l'algorithme qui découle de cette modification n'est plus linéaire en $\log d$ mais seulement *polynomiale* en $\log d$ à cause de l'application de l'algorithme LLL, c'est pourquoi on a préféré ne pas utiliser cette version ici.

Un exemple d'application de l'algorithme D2 est donné en annexe.

2.1.2. Pgcd de deux polynômes lacunaires - Algorithme D. — On est maintenant en mesure de démontrer le théorème 2.1.1 et de décrire l'algorithme D correspondant.

Algorithme D

Entrée : Deux polynômes $f_1, f_2 \in \mathbb{Z}[X]$ donnés par leur représentation lacunaire :

$$f_1(X) = \sum_{i=1}^{n} f_{1,i} X^{a_i} \qquad f_2(X) = \sum_{i=1}^{n} f_{2,i} X^{a_i}.$$

On autorise ici certains coefficients de f_1 et f_2 à être nuls de façon à ce qu'ils aient le même support. Si f_1 et f_2 ont chacun au plus N termes non nuls, on a $n \leq 2N$.
Sortie : $p \in \mathbb{Z}[X]$ donné par sa représentation lacunaire tel que $p |\, \mathrm{pgcd}(f_1, f_2)$ et $\mathrm{pgcd}(f_1, f_2)/p$ est produit de facteurs cyclotomiques.

1. Faire
$$\begin{aligned} F_1(X_1, \ldots, X_n) &\longleftarrow \sum_{i=1}^{n} f_{1,i} X_i \\ F_2(X_1, \ldots, X_n) &\longleftarrow \sum_{i=1}^{n} f_{2,i} X_i \\ \boldsymbol{a} &\longleftarrow (a_1, \ldots, a_n) \\ k &\longleftarrow n. \end{aligned}$$

2. Si $k = 1$ aller à l'étape 3, sinon appliquer l'algorithme D2 au triplet $(F_1, F_2, \boldsymbol{a})$. S'il renvoie « RIEN », aller à l'étape 3. Sinon, appeler $(F_1', F_2', \boldsymbol{a}')$ le triplet obtenu en sortie, faire $(F_1, F_2, \boldsymbol{a}) \longleftarrow (F_1', F_2', \boldsymbol{a}')$, $k \longleftarrow k - 1$ et aller à l'étape 2.

3. Calculer $D := \mathrm{pgcd}(F_1, F_2)$, faire $p(X) \longleftarrow D(X^{a_1}, \ldots, X^{a_k})$, renvoyer $p(X)$ et terminer.

Démonstration du théorème 2.1.1 : Montrons que l'algorithme D que nous venons de décrire satisfait bien les conditions du théorème 2.1.1.

Comme le nombre de variables diminue de 1 à chaque application de l'étape 2, il est clair que l'algorithme termine.

D'après la proposition 2.1.5, on a après chaque application de l'étape 2 :

$$\begin{aligned} f_1(X) &= F_1(X^{\boldsymbol{a}}) \\ f_2(X) &= F_2(X^{\boldsymbol{a}}). \end{aligned}$$

Il s'en suit que le polynôme p renvoyé par l'algorithme à l'étape 3 est bien un diviseur de $\text{pgcd}(f_1, f_2)$. De plus, si l'algorithme D2 renvoie « RIEN » à l'étape 2, c'est que tous les points de la variété $V(F_1, F_2) \cap (\mathcal{H}_1 \backslash \mathcal{H}_0)$ sont dans une composante de $V(F_1, F_2)$ de codimension 1, d'après le théorème 2.1.3. Donc ces points annulent le pgcd de F_1 et F_2. Finalement, le polynôme $\text{pgcd}(f_1, f_2)/p$ ne peut s'annuler qu'en des racines de l'unité et l'algorithme est donc correct.

Il ne nous reste donc qu'à évaluer son temps d'exécution. Supposons que les polynômes f_1 et f_2 sont de degré au plus d et de hauteur au plus h. Chaque exécution de l'étape 2 nécessite un appel à l'algorithme D2. Comme les polynômes F_1 et F_2 sont à chaque fois de hauteur et de degré $O_{N,h}(1)$ et que $||\boldsymbol{a}|| = O_{N,h}(d)$, ceci nécessite $O_{N,h}(\log d)$ opérations binaires. De plus, comme le nombre de variables des polynômes diminue de 1 à chaque application de l'étape 2, celle-ci est exécutée au plus $n-1$ fois. A l'étape 4, on doit déterminer le pgcd de 2 polynômes de degré et de hauteur $O_{N,h}(1)$. Ceci nécessite donc $O_{N,h}(1)$ opérations binaires et la dernière évaluation pour obtenir le polynôme p nécessite $O_{N,h}(\log d)$ opérations binaires. Au total, l'exécution de cet algorithme nécessite bien $O_{N,h}(\log d)$ opérations binaires. □

Remarque : La différence essentielle entre notre algorithme D et l'algorithme B décrit dans **[FGS08]** est la suivante. Dans **[FGS08]**, les auteurs appliquent également un analogue de notre algorithme D2 pour diminuer le nombre de variables (ce que nous faisons à l'étape 2). Cependant, pour effectuer ce changement de variables, ils doivent s'assurer que les polynômes auxquels ils l'appliquent sont premiers entre eux et doivent donc les diviser par leur pgcd. Avoir énoncé le théorème 2.1.3 sans l'hypothèse d'irréductibilité sur la variété χ nous permet de nous passer de cette hypothèse et donc d'éviter un calcul de pgcd à chaque étape. Cette modification est essentielle à la généralisation de cette procédure pour des polynômes à deux variables comme nous le verrons dans les prochaines sections.

Nous détaillons un exemple d'application de l'algorithme D dans l'annexe.

2.2. Le cas dense

Les variétés que nous considérons dans cette section sont des sous-variétés de l'espace affine $\mathbb{A}^k := \mathbb{A}^k(\overline{\mathbb{Q}})$. Soient $F_1, \ldots, F_r \in \mathbb{Z}[X_1, \ldots, X_k]$ des polynômes définissant une variété $V \subset \mathbb{A}^k$ de codimension 2, que l'on décompose en

$$V = V_2 \cup V_{\geq 3},$$

où V_2 (resp. $V_{\geq 3}$) désigne la réunion des composantes irréductibles de V de codimension 2 (resp. au moins 3). On cherche ici à résoudre le problème suivant : peut-on représenter les points de V_2 par seulement 2 équations ? La réponse est non puisqu'une variété de codimension 2 n'est pas toujours intersection complète. Cependant, on peut quand même obtenir trois polynômes $P_1, P_2, P_3 \in \mathbb{Z}[X_1, \ldots, X_k]$ tels que

$$V(P_1, P_2) \backslash V(P_3) = V \backslash V(P_3), \tag{2.2.1}$$

et tels que $V(P_3)$ ne contient aucune composante de V de codimension 2. Autrement dit, la variété V peut être définie comme une intersection complète en dehors d'une hypersurface qui ne contient aucune composante de V de codimension 2.

L'objectif de cette section est de décrire une procédure, l'algorithme E2b, qui permet de calculer des polynômes P_1, P_2, P_3 satisfaisant (2.2.1).

L'algorithme E2b sera utilisé comme procédure auxiliaire par l'algorithme E2 de la section 2.3.3. Comme l'algorithme E2 dépend de la validité de la conjecture de Zilber (voir section 2.3.1), nous ne pourrons pas expliciter la dépendance de sa complexité ni en la hauteur ni en le nombre de termes non nuls des polynômes qu'il prend en entrée. Nous nous focalisons donc sur la dépendance de cette complexité en le degré des polynômes donnés en entrée de l'algorithme E2. Comme l'entrée des algorithmes de cette section sera indépendante du degré des polynômes en entrée de l'algorithme E2, nous n'explicitons pas leur complexité. On remarque simplement qu'il s'agit d'algorithmes déterministes et inconditionnels.

2.2.1. Résultats préliminaires. — On démontre ici plusieurs résultats qui nous seront nécessaires pour justifier la correction de l'algorithme E2b décrit dans la section 2.2.2. On donne également une procédure, l'algorithme E2a, qui sera utilisée par l'algorithme E2b comme procédure auxiliaire.

Lemme 2.2.1. — *Soient r et s deux entiers positifs et $H_1, \ldots, H_s$ des hyperplans de $\mathbb{A}^r$. Il existe $\boldsymbol{\lambda} \in \{0, \ldots, s\}^r$ tel que $\boldsymbol{\lambda}$ n'appartient à aucun des s hyperplans $H_1, \ldots, H_s$.*

Démonstration : Chaque hyperplan contient au plus $(s+1)^{r-1}$ points de $\{0, \ldots, s\}^r$ puisque les coordonnées d'un tel point vérifient une relation de dépendance linéaire. La réunion des s hyperplans contient donc au plus

$$s(s+1)^{r-1} < (s+1)^r = \#\{0, \ldots, s\}^r.$$

Ainsi, au moins un des points de $\{0, \ldots, s\}^r$ n'est dans aucun des hyperplans. □

Soient $V, W \subset \mathbb{A}^k$ deux variétés équidimensionnelles et U un ouvert de $\mathbb{A}^k$. On dit que V et W s'*intersectent proprement* dans U si ou bien

$$V \cap W \cap U = \emptyset,$$

ou bien

$$\operatorname{codim}(\overline{V \cap W \cap U}) = \operatorname{codim}(\overline{V \cap U}) + \operatorname{codim}(\overline{W \cap U}).$$

Le lemme 2.2.1 permet de démontrer la variante suivante du [**Lec00**, Lemma 1].

Lemme 2.2.2. — *Soient $F_1, \ldots, F_r \in \mathbb{Z}[X_1, \ldots, X_k]$ de degré total majoré par δ et définissant une variété V de codimension 2. On écrit*

$$V = V_2 \cup V_{\geq 3},$$

où V_2 et $V_{\geq 3}$ désignent respectivement la réunion des composantes de V de codimension 2 et de codimension supérieure ou égale à 3. Alors il existe une matrice

$$L = (\lambda_{i,j})_{\substack{1 \leq i \leq 3 \\ 1 \leq j \leq r}}$$

dont les coefficients sont des entiers positifs majorés par $\delta^2 + \delta$, telle que les polynômes

$$G_i = \sum_{j=1}^{r} \lambda_{i,j} F_j, \qquad i = 1, 2, 3,$$

vérifient les conditions suivantes. La variété $V(G_1)$ intersecte proprement $V(G_2)$ et $V(G_3)$ dans $\mathbb{A}^k$ et les variétés $V(G_1, G_2)$ et $V(G_3)$ s'intersectent proprement dans $\mathbb{A}^k \backslash V_2$.

Remarque. On peut alors écrire :

$$V(G_1, G_2) = V_2 \cup W_2,$$
$$V(G_1, G_3) = V_2 \cup W_3,$$

où V_2, W_2 et W_3 sont pures de codimension 2 et n'ont deux à deux aucune composante irréductible commune.

Démonstration : On construit successivement G_1, G_2 puis G_3 de façon à ce qu'ils vérifient les conditions du lemme. Comme dans l'énoncé du lemme, on note V_2 et $V_{\geq 3}$ la réunion des composantes de V de codimension 2 et de codimension supérieure ou égale à 3 respectivement.

Comme la codimension de V est égale à 2, l'un au moins des polynômes F_j, par exemple F_1, est non nul. On pose alors $\boldsymbol{\lambda}_1 := (1, 0, \ldots, 0)$ et $G_1 := F_1$. On a dans ce cas :

$$V(G_1) = V_{1,1} \cup \cdots \cup V_{1,t_1},$$

où pour $1 \leq t \leq t_1$, la variété $V_{1,t}$ est une hypersurface irréductible de $\mathbb{A}^k$ et

$$t_1 \leq \deg(G_1) \leq \delta.$$

Comme V est de codimension 2, pour chaque t compris entre 1 et t_1, il existe un point $\boldsymbol{x}_{1,t} \in V_{1,t}$ et un entier $m_{1,t}$ compris entre 1 et r tels que

$$F_{m_{1,t}}(\boldsymbol{x}_{1,t}) \neq 0.$$

Pour $1 \leq t \leq t_1$, on considère alors l'hyperplan $H_{1,t}$ de $\mathbb{A}^r$ défini par :

$$F_1(\boldsymbol{x}_{1,t})Y_1 + \cdots + F_r(\boldsymbol{x}_{1,t})Y_r = 0. \tag{2.2.2}$$

D'après le lemme 2.2.1, il existe $\boldsymbol{\lambda}_2 = (\lambda_{2,1}, \ldots, \lambda_{2,r}) \in \{0, \ldots, \delta\}^r$ tel que $\boldsymbol{\lambda}_2$ n'appartient à aucun de ces hyperplans. On pose alors

$$G_2 := \sum_{j=1}^{r} \lambda_{2,j} F_j.$$

Par construction, $V(G_2)$ ne contient aucune des hypersurfaces $V_{1,t}$, pour t compris entre 1 et t_1. On a ainsi

$$\operatorname{codim}(V(G_1, G_2)) = 2,$$

et les variétés $V(G_1)$ et $V(G_2)$ s'intersectent proprement. De plus, comme la variété V est incluse dans $V(G_1, G_2)$, on peut écrire :

$$V(G_1, G_2) = V_2 \cup V_{2,1} \cup \cdots \cup V_{2,t_2},$$

où pour t compris entre 1 et t_2, la variété $V_{2,t}$ est irréductible de codimension 2 et n'est pas contenue dans V. On obtient la majoration

$$t_2 \leq \deg(V(G_1, G_2)) \leq \deg(G_1)\deg(G_2) \leq \delta^2,$$

par le théorème de Bézout. On pose

$$W_2 := V_{2,1} \cup \cdots \cup V_{2,t_2}.$$

Cette variété peut éventuellement être vide.

On cherche maintenant une troisième combinaison linéaire de $F_1, \ldots, F_r$ qui intersecte proprement chacune des variétés $V_{1,t}$ et chacune des variétés $V_{2,t}$. Fixons un entier t entre 1 et t_2. Comme $V_{2,t} \nsubseteq V$, il existe $\boldsymbol{x}_{2,t} \in V_{2,t}$ et $m_{2,t}$ un entier compris entre 1 et r tels que

$$F_{m_{2,t}}(\boldsymbol{x}_{2,t}) \neq 0.$$

On considère alors l'hyperplan $H_{2,t}$ de $\mathbb{A}^r$ défini par

$$F_1(\boldsymbol{x}_{2,t})Y_1 + \cdots + F_r(\boldsymbol{x}_{2,t})Y_r = 0. \tag{2.2.3}$$

D'après le lemme 2.2.1, comme $t_1 + t_2 \leq \delta^2 + \delta$, il existe

$$\boldsymbol{\lambda}_3 = (\lambda_{3,1}, \ldots, \lambda_{3,r}) \in \{0, \ldots, \delta^2 + \delta\}^r$$

n'appartenant à aucun des hyperplans $H_{1,t}$ définis en (2.2.2), pour $1 \leq t \leq t_1$ et à aucun des hyperplans $H_{2,t}$ définis en (2.2.3), pour $1 \leq t \leq t_2$. On pose finalement

$$G_3 := \sum_{j=1}^{r} \lambda_{3,j} F_j.$$

Par construction, $V(G_3)$ intersecte proprement $V(G_1)$ et W_2 et on peut écrire :

$$V(G_1, G_3) = V_2 \cup W_3,$$

où W_3 est pure de codimension 2 et n'a aucune composante en commun avec V_2. De plus, $V(G_3)$ ne contient également aucune des variétés $V_{2,t}$, pour t compris entre 1 et t_2, et intersecte donc proprement chacune d'elles. Ainsi, W_2 et W_3 n'ont aucune composante irréductible en commun, ce qu'on voulait. □

On sera amenés, au cours de l'exécution de l'algorithme E2b, à projeter des variétés de $\mathbb{A}^k$ de codimension 2 sur $\mathbb{A}^{k-1}$. Il nous faudra alors garantir que cette projection possède de « bonnes » propriétés, ce qui est assuré par le lemme suivant qui découle de [**KPS01**, Proposition 4.5].

Lemme 2.2.3. — *Soit $V \subset \mathbb{A}^k$ une variété équidimensionnelle de codimension 2 que l'on écrit comme réunion de ses composantes irréductibles :*

$$V = \bigcup_l C_l.$$

Alors il existe une matrice $M = (m_{i,j})_{\substack{1 \leq i \leq k-1 \\ 1 \leq j \leq k}}$ dont les coefficients sont des entiers positifs majorés par $2\deg(V)^2$ vérifiant la propriété suivante. Soit π l'application linéaire définie par

$$\begin{aligned} \pi : \ \mathbb{A}^k &\longrightarrow \mathbb{A}^{k-1} \\ \boldsymbol{x} &\longmapsto M\boldsymbol{x}, \end{aligned}$$

alors les $\overline{\pi(C_l)}$ sont des hypersurfaces irréductibles de $\mathbb{A}^{k-1}$ deux à deux distinctes.

Démonstration : D'après [**KPS01**, Proposition 4.5], il existe des entiers positifs

$$(m_{i,j})_{\substack{1 \leq i \leq k-1 \\ 1 \leq j \leq k}}$$

majorés par $2\deg(V)^2$ tels que si

$$L_i = m_{i,1}X_1 + \cdots + m_{i,k}X_k \qquad \text{pour } 1 \leq i \leq k-2,$$

alors

$$Z := V \cap V(L_1 + m_{1,0}, \ldots, L_{k-2} + m_{k-2,0})$$

est fini et de cardinal $\deg(V)$. On cherche maintenant une forme linéaire L_{k-1} qui sépare les éléments de Z, dans le sens où pour $\boldsymbol{\xi}, \boldsymbol{\xi}' \in Z$, on a

$$L_{k-1}(\boldsymbol{\xi}) \neq L_{k-1}(\boldsymbol{\xi}'), \qquad \text{si } \boldsymbol{\xi} \neq \boldsymbol{\xi}'.$$

Pour $\boldsymbol{\xi}, \boldsymbol{\xi}' \in Z$ tels que $\boldsymbol{\xi} \neq \boldsymbol{\xi}'$, on considère l'hyperplan de $\mathbb{A}^k$ défini par

$$(\xi_1 - \xi_1')X_1 + \cdots + (\xi_k - \xi_k')X_k = 0.$$

On peut former $\deg(V)(\deg(V)-1)/2$ tels hyperplans et d'après le lemme 2.2.1, il existe des entiers positifs $m_{k-1,1}, \ldots, m_{k-1,k}$ majorés par $\deg(V)(\deg(V)-1)/2$ tels que le point $(m_{k-1,1}, \ldots, m_{k-1,k})$ n'appartient à aucun de ces hyperplans. On pose alors

$$L_{k-1} = m_{k-1,1}X_1 + \cdots + m_{k-1,k}X_k$$

et il est clair que cette forme linéaire sépare bien tous les points de Z. On considère maintenant l'application linéaire π définie par :

$$\begin{aligned} \pi : \ \mathbb{A}^k &\longrightarrow \mathbb{A}^{k-1} \\ \boldsymbol{x} &\longmapsto (L_1(\boldsymbol{x}), \ldots, L_{k-1}(\boldsymbol{x})). \end{aligned}$$

Montrons qu'on a alors

$$\pi(V) \cap V(y_1 + m_{1,0}, \ldots, y_{k-2} + m_{k-2,0}) = \pi(Z). \tag{2.2.4}$$

L'inclusion $\pi(Z) \subseteq \pi(V) \cap V(y_1 + m_{1,0}, \ldots, y_{k-2} + m_{k-2,0})$ étant claire, on ne justifie que l'inclusion inverse. Si $\boldsymbol{\zeta} \in \pi(V) \cap V(y_1 + m_{1,0}, \ldots, y_{k-2} + m_{k-2,0})$, alors il existe $\boldsymbol{\xi} \in V$ tel que

$$\boldsymbol{\zeta} = \pi(\boldsymbol{\xi}).$$

On a alors, pour $1 \leq i \leq k-2$:

$$\zeta_i = L_i(\boldsymbol{\xi}).$$

Comme de plus $\boldsymbol{\zeta} \in V(y_1 + m_{1,0}, \ldots, y_{k-2} + m_{k-2,0})$, on a : $\zeta_i = -m_{i,0}$, pour i compris entre 1 et $k-2$. On a donc bien

$$\boldsymbol{\xi} \in V \cap V(L_1 + m_{1,0}, \ldots, L_{k-2} + m_{k-2,0}) = Z.$$

Comme la forme L_{k-1} sépare les points de Z, on a

$$\#\pi(Z) = \#Z = \deg(V).$$

Cette égalité combinée à l'égalité (2.2.4) montre que $\overline{\pi(V)}$ est une hypersurface de $\mathbb{A}^{k-1}$ de même degré que V.

On peut maintenant montrer que l'application π sépare les différentes composantes irréductibles de V. Écrivons V comme la réunion de ses composantes irréductibles :

$$V = \bigcup_l C_l.$$

On a alors :

$$\deg(V) = \sum_l \deg(C_l). \tag{2.2.5}$$

Pour chaque l, on pose

$$Z_l := C_l \cap V(L_1 + m_{1,0}, \ldots, L_{k-2} + m_{k-2,0}).$$

Comme

$$Z = \bigcup_l Z_l,$$

chaque Z_l est un ensemble fini de cardinal majoré par $\deg(C_l)$ mais d'après (2.2.5), on a nécessairement

$$\#Z_l = \deg(C_l)$$

pour chaque l et de plus, L_{k-1} sépare tous les points de Z_l. Ainsi, $\overline{\pi(C_l)}$ est une hypersurface de $\mathbb{A}^{k-1}$ de même degré que C_l. Finalement, on a

$$\pi(Z) = \bigcup_l \pi(Z_l),$$

et comme

$$\deg\left(\pi(Z)\right) = \sum_l \deg\left(\pi(Z_l)\right),$$

on a nécessairement, pour $l \neq l'$: $\pi(Z_l) \neq \pi(Z_{l'})$ et donc $\overline{\pi(C_l)} \neq \overline{\pi(C_{l'})}$. □

Nous aurons également besoin de l'algorithme suivant qui sera utilisé comme procédure auxiliaire par l'algorithme E2b. Comme celui-ci utilise des résultants, nous en détaillons d'abord quelques propriétés dont on trouvera les démonstrations dans [**CLO97**].

Etant donnés deux polynômes non nuls $G_1, G_2 \in \mathbb{Q}[X_1, \ldots, X_k]$, on écrit ces polynômes

$$\begin{aligned} G_1 &= A_{1,0}X_k^{\delta_1} + \cdots + A_{1,\delta_1}, \qquad & A_{1,0} \neq 0, \\ G_2 &= A_{2,0}X_k^{\delta_2} + \cdots + A_{2,\delta_1}, \qquad & A_{2,0} \neq 0, \end{aligned} \tag{2.2.6}$$

où les $A_{i,j}$ sont des polynômes de $\mathbb{Q}[X_1, \ldots, X_{k-1}]$. On définit alors le résultant par rapport X_k de G_1 et G_2 et on note $\mathrm{Res}_{X_k}(G_1, G_2)$, le déterminant de la matrice

carrée de taille $\delta_1 + \delta_2$:

$$\underbrace{\begin{pmatrix} A_{1,0} & & & & A_{2,0} & & & \\ A_{1,1} & A_{1,0} & & & A_{2,1} & A_{2,0} & & \\ & A_{1,1} & \ddots & & & A_{2,1} & \ddots & \\ \vdots & & \ddots & A_{1,0} & \vdots & & \ddots & A_{2,0} \\ & \vdots & & A_{1,1} & & \vdots & & A_{2,1} \\ A_{1,\delta_1} & & & & A_{2,\delta_2} & & & \\ & A_{1,\delta_1} & & \vdots & & A_{2,\delta_2} & & \vdots \\ & & \ddots & & & & \ddots & \\ & & & A_{1,\delta_1} & & & & A_{2,\delta_2} \end{pmatrix}}_{\delta_2 \text{ colonnes}} \underbrace{\qquad\qquad}_{\delta_1 \text{ colonnes}},$$

où les espaces vides sont complétés par des 0. On peut montrer que ce polynôme est un élément de l'idéal $(G_1, G_2) \cap \mathbb{Q}[X_1, \ldots, X_{k-1}]$ qui est nul si et seulement si les polynômes G_1 et G_2 ont un facteur commun dans $\mathbb{Q}[X_1, \ldots, X_k]$ ayant un degré non nul en X_k.

Nous utiliserons, pour justifier la correction de l'algorithme E2a, la proposition classique suivante que l'on peut trouver dans [**CLO97**, Chapter 3, §6] :

Proposition 2.2.4. — *Soient $G_1, G_2 \in \mathbb{Q}[X_1, \ldots, X_k]$ et soient $A_{1,0}$ et $A_{2,0}$ définis comme dans* (2.2.6). *Si $\mathrm{Res}_{X_k}(G_1, G_2)$ s'annule en un point $(x_1, \ldots, x_{k-1}) \in \mathbb{A}^{k-1}$, alors on a :*

1. *soit $A_{1,0}$ et $A_{2,0}$ s'annulent en $(x_1, \ldots, x_{k-1})$.*
2. *soit il existe $x_k \in \mathbb{A}^1$ tel que G_1 et G_2 s'annulent en $(x_1, \ldots, x_k)$.*

On peut maintenant décrire l'algorithme E2a.

Lemme 2.2.5. — *(Algorithme E2a) Il existe un algorithme qui a la propriété suivante : soient deux polynômes $G_1, G_2 \in \mathbb{Z}[X_1, \ldots, X_k]$ définissant une variété de codimension 2 et une matrice $M = (m_{s,t})_{\substack{1 \le s \le k-1 \\ 1 \le t \le k}} \in \mathcal{M}_{k-1 \times k}(\mathbb{Z})$ de rang $k-1$ définissant une application linéaire*

$$\begin{aligned} \pi : \ \mathbb{A}^k &\longrightarrow \mathbb{A}^{k-1} \\ \boldsymbol{x} &\longmapsto M\boldsymbol{x}. \end{aligned}$$

L'algorithme renvoie un polynôme $\widetilde{G} \in \mathbb{Z}[X_1, \ldots, X_{k-1}]$ ayant la propriété suivante. Si G_1, G_2 définissent une variété de codimension 2 et si pour toute composante irréductible C de $V(G_1, G_2)$, la variété $\overline{\pi(C)}$ est une hypersurface de $\mathbb{A}^{k-1}$,

alors

$$V(\widetilde{G}) = \overline{\pi(V(G_1, G_2))}.$$

Algorithme E2a

Entrée : Une matrice $M = (m_{s,t})_{\substack{1\leq s\leq k-1\\1\leq t\leq k}}$ à coefficients entiers et de rang $k-1$ et deux polynômes $G_1, G_2 \in \mathbb{Z}[X_1, \ldots, X_k]$.
Sortie : Un polynôme $\widetilde{G} \in \mathbb{Z}[X_1, \ldots, X_{k-1}]$ satisfaisant les conditions du lemme 2.2.5.

1. On choisit pour $\boldsymbol{m}_k = (m_{k,1}, \ldots, m_{k,k})$ l'un des k vecteurs de la base canonique de $\mathbb{Q}^k$, de telle sorte que la matrice $(m_{s,t})_{\substack{1\leq s\leq k\\1\leq t\leq k}}$ soit inversible et on fait
$$M' \longleftarrow (m_{s,t})_{\substack{1\leq s\leq k\\1\leq t\leq k}}.$$
2. Pour $i = 1, 2$, on fait $\widetilde{G}_i(\boldsymbol{X}) \longleftarrow G_i(M'^{-1}\boldsymbol{X})$.
3. Pour $i = 1, 2$, on pose $\delta_i = \deg_{X_k}(\widetilde{G}_i)$ et on définit $A_{i,0}(X_1, \ldots, X_{k-1})$ de sorte que
$$\widetilde{G}_i(X_1, \ldots, X_k) = A_{i,0}(X_1, \ldots, X_{k-1})X_k^{\delta_i} + G_i',$$
où $G_i' \in \mathbb{Z}[X_1, \ldots, X_k]$ et $\deg_{X_k}(G_i') < \delta_i$.
4. On fait
$$R \longleftarrow \mathrm{Res}_{X_k}(\widetilde{G}_1, \widetilde{G}_2) \in \mathbb{Z}[X_1, \ldots, X_{k-1}],$$
$$\widetilde{G} \longleftarrow 1.$$
5. On détermine la décomposition du polynôme R en facteurs irréductibles dans $\mathbb{Z}[X_1, \ldots, X_{k-1}]$ que l'on écrit :
$$R = R_1^{e_1} \cdots R_u^{e_u}.$$
6. Pour $1 \leq i \leq u$, si $\mathrm{codim}(V(\widetilde{G}_1, \widetilde{G}_2, R_i)) = 2$, alors on fait
$$\widetilde{G} \longleftarrow \widetilde{G} \cdot R_i.$$
7. On renvoie $\widetilde{G}$ et on termine.

Démonstration : Supposons que G_1, G_2 définissent une variété de codimension 2 et que pour toute composante irréductible C de $V(G_1, G_2)$, la variété $\overline{\pi(C)}$ est une hypersurface de $\mathbb{A}^{k-1}$ et montrons que, dans ce cas, le polynôme $\widetilde{G}$ renvoyé par cet algorithme vérifie bien :

$$V(\widetilde{G}) = \overline{\pi(V(G_1, G_2))}.$$

Comme la matrice M est de rang $k-1$, il existe un choix de

$$\boldsymbol{m}_k = (m_{k,1}, \ldots, m_{k,k})$$

parmi les k vecteurs de la base canonique de $\mathbb{Q}^k$ qui rend la matrice M' de l'étape 1 inversible. On peut calculer pour chacun de ces k choix possibles le déterminant de la matrice obtenue et choisir pour $\boldsymbol{m}_k$ un des vecteurs qui rend ce déterminant non nul. La matrice M' définit alors un isomorphisme ι de $\mathbb{A}^k$ et si on note

$$\begin{aligned} \pi' :\ \mathbb{A}^k &\longrightarrow \mathbb{A}^{k-1} \\ (x_1, \ldots, x_k) &\longmapsto (x_1, \ldots, x_{k-1}), \end{aligned}$$

on a

$$\pi' \circ \iota = \pi.$$

Les polynômes $\widetilde{G}_1$ et $\widetilde{G}_2$ définis à l'étape 2 vérifient alors :

$$\iota\left(V(G_1, G_2)\right) = V\left(\widetilde{G}_1, \widetilde{G}_2\right).$$

Il nous suffit donc de montrer que le polynôme $\widetilde{G}$ renvoyé par l'algorithme est tel que

$$V(\widetilde{G}) = \overline{\pi'\left(V\left(\widetilde{G}_1, \widetilde{G}_2\right)\right)}.$$

Soient $A_{1,0}$ et $A_{2,0}$ les polynômes définis à l'étape 3 et R le résultant par rapport à la variable X_k de $\widetilde{G}_1$ et $\widetilde{G}_2$. Remarquons tout d'abord que R est non nul puisqu'on a supposé que G_1 et G_2 définissent une variété de codimension 2. De plus, comme l'application π' envoie chaque composante irréductible de

$$V\left(\widetilde{G}_1, \widetilde{G}_2\right)$$

sur une hypersurface de $\mathbb{A}^{k-1}$, la sous-variété

$$\overline{\pi'\left(V\left(\widetilde{G}_1, \widetilde{G}_2\right)\right)}$$

est pure de codimension 1 et peut donc être définie par une seule équation $\widetilde{G}$.

Montrons qu'on peut choisir comme équation le polynôme $\widetilde{G}$ calculé à l'étape 6. D'après la proposition 2.2.4, pour $\boldsymbol{x}' \in \mathbb{A}^{k-1}$, on a $R(\boldsymbol{x}') = 0$ si et seulement si ou bien il existe $x_k \in \mathbb{A}^1$ tel que

$$\widetilde{G}_1(\boldsymbol{x}', x_k) = \widetilde{G}_2(\boldsymbol{x}', x_k) = 0,$$

ou bien

$$A_{1,0}(\boldsymbol{x}') = A_{2,0}(\boldsymbol{x}') = 0.$$

Et de façon équivalente, on a $R(\boldsymbol{x}') = 0$ si et seulement si

$$\boldsymbol{x}' \in \pi'\left(V\left(\widetilde{G}_1, \widetilde{G}_2\right)\right) \qquad \text{ou} \qquad \boldsymbol{x}' \in V(A_{1,0}, A_{2,0}).$$

Ainsi, on a

$$\overline{\pi'\left(V\left(\widetilde{G}_1, \widetilde{G}_2\right)\right)} \subseteq V(R)$$

et donc il existe un diviseur $\widetilde{G}$ de R dans $\mathbb{Q}[X_1, \ldots, X_{k-1}]$ tel que

$$V(\widetilde{G}) = \overline{\pi'\left(V\left(\widetilde{G}_1, \widetilde{G}_2\right)\right)}.$$

On détermine alors la décomposition du polynôme R en produit de facteurs irréductibles dans $\mathbb{Q}[X_1, \ldots, X_{k-1}]$:

$$R = R_1^{e_1} \cdots R_u^{e_u}.$$

Il nous reste alors à déterminer, pour chaque facteur R_i de R, si

$$V(R_i) \subseteq \overline{\pi'\left(V\left(\widetilde{G}_1, \widetilde{G}_2\right)\right)}.$$

Cette condition est équivalente à ce que $V(R_i)$, vue comme sous-variété de $\mathbb{A}^k$, contienne une composante de $V(\widetilde{G}_1, \widetilde{G}_2)$ de codimension 2 et on teste cette condition en calculant

$$\operatorname{codim}(V(\widetilde{G}_1, \widetilde{G}_2, R_i)).$$

De nombreux algorithmes existent pour calculer la dimension d'une variété, on peut par exemple se référer à [**CLO97**, Chapitre 9]. Si cette codimension est égale à 2, c'est que $V(R_i)$ contient effectivement une composante irréductible de la variété $V(\widetilde{G}_1, \widetilde{G}_2)$ et dans ce cas, on ajoute le facteur R_i à $\widetilde{G}$. Ainsi construit, le polynôme $\widetilde{G}$ vérifie bien

$$V(\widetilde{G}) = \overline{\pi'\left(V\left(\widetilde{G}_1, \widetilde{G}_2\right)\right)}.$$

□

2.2.2. Intersection complète en dehors d'un ouvert. — Etant donnée une variété $V \subset \mathbb{A}^n$ de codimension i, où $0 \leq i \leq n$, une question naturelle est de savoir s'il est possible de définir la variété V par exactement i équations. On dit dans ce cas que la variété V est une *intersection complète*.

Ce n'est pas le cas en général mais on peut malgré tout décrire V comme une intersection complète à l'intérieur d'un ouvert dont le complémentaire est une hypersurface, comme le montre le résultat suivant. Comme on ne s'intéressera qu'au cas de variétés de codimension 2, on n'énonce le résultat que dans ce cas pour alléger les notations. On remarque cependant que les méthodes utilisées peuvent s'adapter pour traiter le cas où la variété est de codimension quelconque.

Proposition 2.2.6. — *(Algorithme E2b) Il existe un algorithme qui a la propriété suivante. Soient $F_1, \ldots, F_r \in \mathbb{Z}[X_1, \ldots, X_k]$ des polynômes définissant une*

variété V de codimension au moins 2. Si V est de codimension égale à 2, l'algorithme détermine trois polynômes $P_1, P_2, F_{r+1} \in \mathbb{Z}[X_1, \ldots, X_k]$ tels que :

1. $V \backslash V(F_{r+1}) = V(P_1, P_2) \backslash V(F_{r+1})$.
2. *$V(F_{r+1})$ ne contient aucune composante de V de codimension 2.*

Dans le cas où V est de codimension au moins 3, l'algorithme renvoie « RIEN ».

Remarque. Dans le cas où $k = 2$, les polynômes $F_1, \ldots, F_r$ définissent un nombre fini de points et la seconde condition de cette proposition assure qu'on a

$$V(P_1, P_2) = V,$$

puisque $V(F_{r+1})$ ne contient aucun point de $V(F_1, \ldots, F_r)$.

Algorithme E2b

Entrée : $F_1, \ldots, F_r \in \mathbb{Z}[X_1, \ldots, X_k]$ définissant une variété V de codimension au moins 2.
Sortie : $P_1, P_2, F_{r+1} \in \mathbb{Z}[X_1, \ldots, X_k]$ tels que $V \backslash V(F_{r+1}) = V(P_1, P_2) \backslash V(F_{r+1})$ et tels que $V(F_{r+1})$ ne contient aucune composante de V de codimension 2, ou « RIEN ».

Avant de décrire la procédure, on donne quelques notations : on note δ le maximum des degrés totaux des polynômes $F_1, \ldots, F_r$ et on écrit, comme dans l'énoncé du lemme 2.2.2,

$$V = V_2 \cup V_{\geq 3},$$

où V_2 et $V_{\geq 3}$ désignent respectivement la réunion des composantes de V de codimension 2 et de codimension supérieure ou égale à 3. On introduit également un ensemble E de couples de matrices qui nous servira de compteur.

1. On teste si V est de codimension au moins 3. Si c'est le cas, on renvoie « RIEN » et on termine. Sinon, on passe à l'étape 2.
2. On pose
$$E \longleftarrow \emptyset.$$
3. (a) On choisit une matrice $L = (\lambda_{i,j})_{\substack{1 \leq i \leq 3 \\ 1 \leq j \leq r}}$ dont les coefficients sont des entiers positifs et majorés par $\delta^2 + \delta$. On choisit une matrice $M = (m_{s,t})_{\substack{1 \leq s \leq k-1 \\ 1 \leq t \leq k}}$ de rang $k-1$ dont les coefficients sont des entiers positifs et majorés par $4\delta^4$ et on impose également que $(L, M) \notin E$.

(b) Pour $i = 1, 2, 3$, on fait

$$G_i \longleftarrow \sum_{j=1}^{r} \lambda_{i,j} F_j.$$

4. On applique l'algorithme E2a à G_1, G_2 et M (resp. à G_1, G_3 et M) et on appelle H_2 (resp. H_3) la sortie. On pose :

$$\begin{aligned} H_4 &\longleftarrow \frac{H_2}{\mathrm{pgcd}(H_2, H_3)}, \\ P_1 &\longleftarrow G_1, \\ P_2 &\longleftarrow G_2, \\ F_{r+1} &\longleftarrow H_4(M\boldsymbol{X}). \end{aligned}$$

5. Si les deux conditions

   $V \backslash V(F_{r+1}) = V(P_1, P_2) \backslash V(F_{r+1})$,

   $V(F_{r+1})$ ne contient aucune composante de V de codimension 2

   sont satisfaites, on renvoie (P_1, P_2, F_{r+1}) et on termine sinon on fait

$$E := E \cup \{(L, M)\}$$

   et on retourne à l'étape 3.

Démonstration : Montrons que cette procédure vérifie bien les conditions de la proposition 2.2.6. Dans un premier temps, on détaille le fonctionnement et la correction de la procédure. Dans un second temps, nous expliquerons comment on peut réaliser les tests effectués lors des différentes étapes et comment on peut choisir les entiers $\lambda_{i,j}$ et $m_{s,t}$.

Supposons tout d'abord que la variété V est de codimension au moins 3. Alors l'algorithme termine à l'étape 1 et le résultat renvoyé est correct.

Dans le cas contraire, la variété V est de codimension 2. Comme l'étape 5 consiste à tester les conditions imposées par la proposition 2.2.6, si la procédure renvoie un résultat, il sera correct. Il nous suffit donc de montrer qu'il existe un choix de (L, M) à l'étape 3a pour lequel les conditions de la propositions 2.2.6 sont remplies. Le lemme 2.2.2 garantit qu'il existe un choix de L pour lequel les polynômes G_1, G_2, G_3 formés à l'étape 3b vérifient les conditions suivantes :

$$\begin{aligned} V(G_1, G_2) &= V_2 \cup W_2, \\ V(G_1, G_3) &= V_2 \cup W_3, \end{aligned}$$

où V_2, W_2 et W_3 sont pures de codimension 2 et n'ont deux à deux aucune composante irréductible commune. Ecrivons

$$V_2 \cup W_2 \cup W_3 = \bigcup_l C_l,$$

où chaque C_l est irréductible de codimension 2. Alors d'après le lemme 2.2.3, il existe aussi un choix de M pour lequel les $\overline{\pi(C_l)}$ sont des hypersurfaces irréductibles de $\mathbb{A}^{k-1}$ deux à deux distinctes, où π désigne l'application linéaire de $\mathbb{A}^k$ dans $\mathbb{A}^{k-1}$ définie par la matrice M. D'après le lemme 2.2.5 (Algorithme E2a), les polynômes H_2 et H_3, calculés à l'étape 4, sont respectivement des équations pour les hypersurfaces $\overline{\pi(V_2 \cup W_2)}$ et $\overline{\pi(V_2 \cup W_3)}$. Ces conditions garantissent que le polynôme H_4 est une équation pour l'hypersurface $\overline{\pi(W_2)}$ de $\mathbb{A}^{k-1}$. Par construction, le polynôme F_{r+1} définit une hypersurface de $\mathbb{A}^k$ qui contient W_2 mais ne contient aucune composante de V_2 de codimension 2. Ainsi, la deuxième condition de la proposition 2.2.6 est satisfaite. Montrons maintenant la première, c'est à dire qu'on a bien :

$$V \backslash V(F_{r+1}) = V(P_1, P_2) \backslash V(F_{r+1}). \tag{2.2.7}$$

On a d'une part :

$$\begin{aligned} V(P_1, P_2) \backslash V(F_{r+1}) &= V(G_1, G_2) \backslash V(F_{r+1}) \\ &= (V_2 \cup W_2) \backslash V(F_{r+1}) \\ &= V_2 \backslash V(F_{r+1}). \end{aligned}$$

D'autre part, comme les polynômes G_1 et G_2 sont des combinaisons linéaires des polynômes $F_1, \ldots, F_r$, on a

$$V_2 \cup V_{\geq 3} \subseteq V(G_1, G_2) = V_2 \cup W_2.$$

Ceci montre que

$$V_{\geq 3} \subseteq W_2,$$

car une composante irréductible de $V_{\geq 3}$ ne peut être incluse dans V_2. Ainsi,

$$V \backslash V(F_{r+1}) = (V_2 \cup V_{\geq 3}) \backslash V(F_{r+1}) = V_2 \backslash V(F_{r+1}),$$

puisque

$$V_{\geq 3} \subseteq W_2 \subseteq V(F_{r+1}).$$

Ainsi, l'égalité (2.2.7) est vérifiée et la sortie de l'algorithme vérifie bien les conditions du lemme 2.2.6.

Expliquons maintenant comment on peut réaliser les différents tests effectués au cours de cette procédure. Pour le calcul de la dimension de V à l'étape 1, on peut utiliser la procédure décrite dans [**CLO97**, Chapitre 9]. A l'étape 5, l'inclusion $V \subseteq V(P_1, P_2)$ est toujours vérifiée puisque les polynômes P_1 et P_2

sont des combinaisons linéaires des polynômes $F_1, \ldots, F_r$ définissant V. Montrons donc comment on peut tester l'inclusion

$$V \backslash V(F_{r+1}) \supseteq V(P_1, P_2) \backslash V(F_{r+1}).$$

La condition

$$V(P_1, P_2) \backslash V(F_{r+1}) \subseteq V$$

est équivalente à

$$V(P_1, P_2) \backslash V(F_1 F_{r+1}, \ldots, F_r F_{r+1}) = \emptyset.$$

Soit Z une variable additionnelle, cette dernière égalité est équivalente à l'appartenance du polynôme 1 à l'idéal

$$\left(P_1, P_2, \prod_{i=1}^{r} (1 - Z F_{r+1} F_i) \right) \subseteq \mathbb{Q}[X_1, \ldots, X_k, Z]$$

et cette condition peut être testée en utilisant un test d'appartenance à un idéal comme celui décrit dans [**CLO97**, Chapitre 2].

Enfin, la sous-variété $V(F_{r+1})$ ne contient aucune composante de V de codimension 2 si et seulement si $V \cap V(F_{r+1}) = V(F_1, \ldots, F_r, F_{r+1})$ est de codimension au moins 3 et on peut encore calculer cette codimension en utilisant [**CLO97**, Chapitre 9]. □

Dans le cas où V est de codimension 2, on peut remarquer que l'on a nécessairement

$$\operatorname{codim}(V(F_1, \ldots, F_{r+1})) = 3 \quad \text{ou} \quad V(F_1, \ldots, F_{r+1}) = \emptyset. \tag{2.2.8}$$

En effet, si χ est une composante de la variété $V(F_1, \ldots, F_r)$, le point 2. de la proposition 2.2.6 garantit que χ n'est pas incluse dans $V(F_{r+1})$. Ainsi

$$\operatorname{codim}(\chi \cap V(F_{r+1})) = \operatorname{codim}(\chi) + 1 \quad \text{ou} \quad \chi \cap V(F_{r+1}) = \emptyset$$

et on a bien le résultat annoncé. Cette remarque sera importante pour justifier la correction de l'algorithme E2 du théorème 2.3.6.

2.3. Généralisation conditionnelle en 2 variables

Si on assume une version effective de la conjecture de Zilber (voir la conjecture 2.3.2 dans la section 2.3.1), on peut établir le résultat suivant :

Théorème 2.3.1. — *(Algorithme E) Il existe un algorithme qui a la propriété suivante : étant donnés $f_1, f_2, f_3 \in \mathbb{Z}[X, Y]$ donnés par leur représentation lacunaire, l'algorithme détermine un ensemble de polynômes $\{p_i\}_{i \in I}$ de $\mathbb{Z}[\mu_\infty][X, Y]$*

et un ensemble fini de triplets de polynômes $\{(q_{j,1}, q_{j,2}, q_{j,3})\}_{j\in J}$ *de* $\mathbb{Z}[\mu_\infty][X, Y]$ *donnés par leur représentation lacunaire tels que*

$$V(f_1, f_2, f_3) = \bigcup_{i\in I} V(p_i) \cup \bigcup_{j\in J} V(q_{j,1}, q_{j,2})\backslash V(q_{j,3}).$$

Cet algorithme nécessite au plus

$$O_{N,h}(\mathsf{I}(\log d) \log\log d)$$

opérations binaires, où d *est une borne pour le degré total des polynômes,* h *une borne pour leur hauteur et* N *une borne pour le nombre de termes non nuls de chacun de ces polynômes. De plus* I *et* J *possèdent* $O_{N,h}(1)$ *éléments, le nombre de termes non nuls et la hauteur des polynômes en sortie de l'algorithme sont majorés par* $O_{N,h}(1)$ *et ces polynômes sont de degré au plus* $O_{N,h}(d)$*.*

L'algorithme E correspondant à ce théorème sera décrit à la section 2.3.5.

Soit (σ, τ) un zéro commun de f_1, f_2 et f_3. On étudie d'abord le cas où au moins une des coordonnées de (σ, τ) est nulle, ce qui ramène simplement à des polynômes en une variable et donc peut être traité avec les algorithmes A et D décrits au chapitre 1. Une fois ces cas traités, on peut supposer que σ et τ sont tous deux non nuls, ce qui nous permet de nous placer dans $\mathbb{G}_m^2$. On va alors distinguer trois cas selon que (σ, τ) appartient à $\mathcal{H}_0$, à $\mathcal{H}_1\backslash\mathcal{H}_0$ ou à $\mathcal{H}_2\backslash\mathcal{H}_1$. Ces trois cas peuvent être traités dans un ordre quelconque. Pour faciliter la compréhension, nous traiterons d'abord le cas de $\mathcal{H}_0$ (Algorithme E0), puis le cas de $\mathcal{H}_2\backslash\mathcal{H}_1$ (Algorithme E2) et enfin le cas de $\mathcal{H}_1\backslash\mathcal{H}_0$ (Algorithme E1) qui est un peu plus technique.

On pourrait s'attendre à ce que les points isolés de $V(f_1, f_2, f_3)$ soient définis par seulement deux équations car une variété de dimension 0 est toujours intersection complète. La représentation que nous obtenons de ces points dans le théorème 2.3.1 est moins bonne mais on peut remarquer qu'on arrive tout de même à représenter les points isolés de $V(f_1, f_2, f_3) \cap \mathcal{H}_0$ et $V(f_1, f_2, f_3) \cap (\mathcal{H}_1\backslash\mathcal{H}_0)$ comme réunion d'intersections complètes comme on peut le voir dans les théorèmes 2.3.4 et 2.3.8. Nous ne sommes pas parvenus à représenter les points de $V(f_1, f_2, f_3) \cap (\mathcal{H}_2\backslash\mathcal{H}_1)$ de la même façon car la recherche de ces points fait intervenir la résolution de variétés de dimension positive, qui elles ne sont pas toujours des intersections complètes comme on l'a déjà remarqué avant d'énoncer la proposition 2.2.6. Il serait intéressant de comprendre si cette difficulté peut être contournée et si on peut représenter l'ensemble des points isolés de $V(f_1, f_2, f_3)$ comme réunion d'au plus $O_{n,h}(1)$ intersections complètes dans $\mathbb{A}^2$.

2.3.1. Conjecture de Zilber. — L'algorithme E2 est basé sur une version effective de la conjecture de Zilber, généralisation naturelle du théorème 2.1.3, comme nous l'expliquons maintenant.

Pour $\boldsymbol{x} \in \mathbb{G}_m^k$ et $\boldsymbol{b} \in \mathbb{Z}^k$, on note comme dans le premier chapitre :

$$\boldsymbol{x}^{\boldsymbol{b}} = x_1^{b_1} \cdots x_k^{b_k} \in \overline{\mathbb{Q}}^*.$$

Conjecture 2.3.2. — *Soit χ une variété de $\mathbb{G}_m^k$ de codimension au moins r. Alors il existe $B_2(\chi) > 0$ vérifiant la propriété suivante : soient $\sigma_1, \ldots, \sigma_{r-1} \in \overline{\mathbb{Q}}^*$ et $\boldsymbol{a}_1 \ldots, \boldsymbol{a}_{r-1} \in \mathbb{Z}^k$ tels que*

$$\boldsymbol{x} = \sigma_1^{\boldsymbol{a}_1} \cdots \sigma_{r-1}^{\boldsymbol{a}_{r-1}} \in \chi,$$

alors il existe $\boldsymbol{b} \in \mathbb{Z}^k \backslash \{0\}$ tel que

$$\boldsymbol{x}^{\boldsymbol{b}} = 1 \qquad et \qquad ||\boldsymbol{b}|| \leq B_2(\chi).$$

En particulier, si $\sigma_1, \ldots, \sigma_{r-1}$ sont multiplicativement indépendants, on a :

$$\langle \boldsymbol{a}_1, \boldsymbol{b} \rangle = \cdots = \langle \boldsymbol{a}_{r-1}, \boldsymbol{b} \rangle = 0.$$

Ce résultat est une conséquence de la conjecture suivante due à Zilber (voir [**BMZ07**, Conjecture 5.2]).

Conjecture 2.3.3. — *Soit χ une variété irréductible de $\mathbb{G}_m^k$ définie sur $\mathbb{C}$. Alors il existe un ensemble fini $\Omega = \Omega_\chi$ de translatés $\boldsymbol{\zeta}T$ de tores $T \subsetneq \mathbb{G}_m^k$ par des points de torsion $\boldsymbol{\zeta}$ satisfaisant la propriété suivante. Pour tout translaté K d'un sous-groupe algébrique par un point de torsion et pour toute composante $\mathcal{Y}$ de $\chi \cap K$ satisfaisant*

$$\dim \mathcal{Y} > \dim \chi + \dim K - k,$$

on a $\mathcal{Y} \subseteq \boldsymbol{\zeta}T$ pour un certain $\boldsymbol{\zeta}T \in \Omega$.

Vérifions que la conjecture 2.3.2 est bien une conséquence de la conjecture 2.3.3. Supposons dans un premier temps que la variété χ est irréductible. Considérons $\{\boldsymbol{\zeta}_1 T_1, \ldots, \boldsymbol{\zeta}_s T_s\} = \Omega_\chi$ un ensemble fini de translatés de sous-tores vérifiant la conjecture de Zilber 2.3.3, où les T_i sont strictement inclus dans $\mathbb{G}_m^k$. Ces sous-tores peuvent être définis par des équations monomiales. Ainsi, pour i compris entre 1 et s, on peut trouver $\boldsymbol{v}_i \in \mathbb{Z}^k$ tel que

$$\boldsymbol{z}^{\boldsymbol{v}_i} = z_1^{v_{i,1}} \cdots z_n^{v_{i,k}} = 1, \qquad \forall\, \boldsymbol{z} \in T_i.$$

On a alors

$$\boldsymbol{z}^{\mathrm{ord}(\boldsymbol{\zeta}_i)\boldsymbol{v}_i} = 1, \qquad \forall\, \boldsymbol{z} \in \boldsymbol{\zeta}_i T_i,$$

où $\mathrm{ord}(\boldsymbol{\zeta}_i)$ désigne l'ordre de $\boldsymbol{\zeta}_i$ dans le groupe multiplicatif μ_∞^k. On pose

$$B_2(\chi) := \max_{1 \leq i \leq s} ||\boldsymbol{v}_i|| \,\mathrm{ord}(\boldsymbol{\zeta}_i).$$

Soient maintenant $\sigma_1, \ldots, \sigma_{r-1} \in \overline{\mathbb{Q}}^*$ et $\boldsymbol{a}_1, \ldots \boldsymbol{a}_{r-1} \in \mathbb{Z}^k$ tels que

$$\boldsymbol{x} = \sigma_1^{\boldsymbol{a}_1} \cdots \sigma_{r-1}^{\boldsymbol{a}_{r-1}} \in \chi$$

et $\{e_1, \ldots, e_{k-r+1}\}$ une famille libre du sous-groupe de $\mathbb{Z}^k$ formé des vecteurs orthogonaux à $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_{r-1}$. Soient également K le sous-groupe algébrique de $\mathbb{G}_m^k$ de dimension $r-1$ défini par :

$$K := \{\boldsymbol{z} \in \mathbb{G}_m^k \,:\, \boldsymbol{z}^{\boldsymbol{e}_i} = 1, \forall\, 1 \leq i \leq k-r+1\}$$

et $\mathcal{Y}$ la composante irréductible de $K \cap \chi$ à laquelle appartient $\boldsymbol{x}$. On a $\dim \mathcal{Y} \geq 0$ et $\dim \chi + \dim K - k \leq k - r + r - 1 - k = -1$ donc

$$\dim \mathcal{Y} > \dim \chi + \dim K - k.$$

Par la conjecture 2.3.3, il existe $1 \leq i \leq s$ tel que $\mathcal{Y} \subset \boldsymbol{\zeta}_i T_i$. Ainsi $\boldsymbol{x} \in \boldsymbol{\zeta}_i T_i$ et $\boldsymbol{x}$ satisfait une équation monomiale de norme inférieure à $B_2(\chi)$. Ainsi, la conjecture 2.3.3 implique la conjecture 2.3.2 dans le cas où χ est irréductible.

Dans le cas général, on écrit χ comme réunion de composantes irréductibles :

$$\chi = \chi_1 \cup \cdots \cup \chi_s.$$

Pour chaque $1 \leq j \leq s$, on peut trouver $B_2(\chi_j)$ satisfaisant la conclusion de la conjecture 2.3.2 appliquée à χ_j. Si

$$\boldsymbol{x} = \sigma_1^{\boldsymbol{a}_1} \cdots \sigma_{r-1}^{\boldsymbol{a}_{r-1}} \in \chi,$$

alors $\boldsymbol{x}$ appartient à l'un des χ_j, où $1 \leq j \leq s$. Ainsi, $\boldsymbol{x}$ vérifie une équation monomiale de norme majorée par

$$B_2(\chi) := \max_{1 \leq j \leq s} B_2(\chi_j)$$

et on a bien le résultat voulu.

Remarque. Dans la suite, nous utiliserons uniquement la conjecture 2.3.2 dans le cas où $r = 3$. Celle-ci sera nécessaire pour garantir la validité des résultats de la section 2.3.3 et en particulier la validité de l'algorithme E2.

2.3.2. Points de $\mathcal{H}_0$. — On cherche dans un premier temps à déterminer les points de $V(f_1, f_2, f_3)$ dont les deux coordonnées sont des racines de l'unité, i.e. les points de $V(f_1, f_2, f_3) \cap \mathcal{H}_0$. On peut pour cela utiliser l'algorithme C décrit dans le premier chapitre dans le cas particulier de trois polynômes à deux variables. On obtient :

Théorème 2.3.4. — *(Algorithme E0) Il existe un algorithme qui a la propriété suivante. Soient $f_1, f_2, f_3 \in \mathbb{Q}[X, Y]$ donnés par leur représentation lacunaire. L'algorithme détermine deux familles finies de sous-variétés de torsion*

$$\begin{aligned} V_i &= V\left(X^{\alpha_{i,1}}Y^{\alpha_{i,2}} - \zeta_i\right) && i \in I, \\ W_j &= V(X^{\beta_{j,1}}Y^{\beta_{j,2}} - \zeta_j, X^{\beta'_{j,1}}Y^{\beta'_{j,2}} - \zeta'_j) && j \in J \end{aligned}$$

telles que

$$V(f_1, f_2, f_3) \cap \mathcal{H}_0 \subseteq \bigcup_{i \in I} V_i \cup \bigcup_{j \in J} W_j \subseteq V(f_1, f_2, f_3).$$

Cet algorithme nécessite

$$O_{N,h}\left(\mathsf{I}(\log d) \log\log d\right)$$

opérations binaires, où d est une borne pour le degré total des polynômes, h une borne pour leur hauteur et N une borne pour le nombre de termes non nuls de chacun de ces polynômes. De plus, les ensembles I et J possèdent au plus $O_{N,h}(1)$ éléments et tous les polynômes en sortie sont de degré majoré par $O_{N,h}(d)$, de hauteur majorée par $O_{N,h}(1)$ et possèdent au plus $O_{N,h}(1)$ termes non nuls.

Remarque. Les polynômes $X^{\alpha_{i,1}}Y^{\alpha_{i,2}} - \zeta_i$ qui définissent les sous-variétés V_i sont des facteurs communs aux polynômes f_1, f_2 et f_3 et les sous-variétés W_j décrivent les points isolés de $V(f_1, f_2, f_3) \cap \mathcal{H}_0$.

2.3.3. Points de $\mathcal{H}_2 \backslash \mathcal{H}_1$. — On suppose dans tout ce paragraphe que la conjecture 2.3.2 est établie dans le cas où $r = 3$. Expliquons maintenant comment on peut obtenir une représentation des points de $V(f_1, f_2, f_3) \cap (\mathcal{H}_2 \backslash \mathcal{H}_1)$. Pour pouvoir décrire la procédure, que nous appellerons Algorithme E2, qui détermine une représentation de ces points, nous avons besoin de la procédure auxiliaire suivante qui est l'analogue en deux variables de l'algorithme D2 de la proposition 2.1.5.

Proposition 2.3.5. — *(Algorithme E2c) Il existe un algorithme qui a la propriété suivante. Soient $k \geq 3$, $F_1, \ldots, F_r \in \mathbb{Z}[X_1, \ldots, X_k]$ de degré au plus δ et de hauteur au plus h et $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{Z}^k$ satisfaisant $||\boldsymbol{a}||, ||\boldsymbol{b}|| \leq d$. S'il existe $\sigma, \tau \in \overline{\mathbb{Q}}^*$ multiplicativement indépendants tels que $\sigma^{\boldsymbol{a}}\tau^{\boldsymbol{b}}$ appartient à une composante irréductible de $V(F_1, \ldots, F_r) \subseteq \mathbb{G}_m^k$ de codimension au moins 3, l'algorithme renvoie un ensemble de r polynômes $F'_1, \ldots, F'_r \in \mathbb{Z}[X_1, \ldots, X_{k-1}]$ et $\boldsymbol{a}', \boldsymbol{b}' \in \mathbb{Z}^{k-1}$ tels que*

$$\begin{aligned} F_1\left(X^{\boldsymbol{a}}Y^{\boldsymbol{b}}\right) &= F'_1(X^{\boldsymbol{a}'}Y^{\boldsymbol{b}'}) \\ &\vdots \\ F_r\left(X^{\boldsymbol{a}}Y^{\boldsymbol{b}}\right) &= F'_r(X^{\boldsymbol{a}'}Y^{\boldsymbol{b}'}) \end{aligned} \tag{2.3.1}$$

en au plus

$$O_{k,h,\delta}(\mathsf{I}(\log d)\log\log d)$$

opérations binaires. De plus, les polynômes $F'_1,\dots,F'_r$ sont de degré et de hauteur majorés par $O_{k,h,\delta}(1)$ et $||\boldsymbol{a}'||, ||\boldsymbol{b}'|| = O_{k,h,\delta}(d)$.

Décrivons tout d'abord la procédure :

Algorithme E2c

Entrée : $F_1,\dots,F_r \in \mathbb{Z}[X_1,\dots,X_k]$ et $\boldsymbol{a},\boldsymbol{b} \in \mathbb{Z}^k$.
Sortie : $F'_1,\dots,F'_r \in \mathbb{Z}[X_1,\dots,X_{k-1}]$ et $\boldsymbol{a}',\boldsymbol{b}' \in \mathbb{Z}^{k-1}$ satisfaisant les conditions de la proposition 2.3.5 ou « RIEN ».

1. Tester parmi tous les vecteurs de $\mathbb{Z}^k\backslash\{0\}$ de norme au plus $B_2(V(F_1,\dots,F_r))$ (où B_2 est défini à la conjecture 2.3.2), s'il en existe un qui est orthogonal à $\boldsymbol{a}$ et $\boldsymbol{b}$. Si oui, appeler $\boldsymbol{c}$ un tel vecteur, sinon renvoyer « RIEN » et terminer.
2. Déterminer une base $\{\boldsymbol{c}_1,\dots,\boldsymbol{c}_{k-1}\}$ du sous-groupe de $\mathbb{Z}^k$ formé des vecteurs orthogonaux à $\boldsymbol{c}$ à l'aide de l'algorithme $D1$.
3. Déterminer les coordonnées $\boldsymbol{a}'$ et $\boldsymbol{b}'$ de $\boldsymbol{a}$ et $\boldsymbol{b}$ dans cette base.
4. Pour $1 \le i \le r$, faire
$$F'_i(X_1,\dots,X_{k-1}) \longleftarrow F_i\left(X_1^{c_{1,1}}\cdots X_{k-1}^{c_{k-1,1}},\dots,X_1^{c_{1,k}}\cdots X_{k-1}^{c_{k-1,k}}\right),$$
5. Renvoyer $(F'_1,\dots,F'_r,\boldsymbol{a}',\boldsymbol{b}')$ et terminer.

Démonstration de la proposition 2.3.5 : Montrons que la procédure E2c satisfait bien les conditions de cette proposition. Supposons qu'il existe $\sigma,\tau \in \overline{\mathbb{Q}}^*$ multiplicativement indépendants tels que $\sigma^{\boldsymbol{a}}\tau^{\boldsymbol{b}}$ appartient à une composante irréductible de $V(F_1,\dots,F_r)$ de codimension au moins 3. D'après la conjecture 2.3.2, il existe un vecteur $\boldsymbol{c} \in \mathbb{Z}^k\backslash\{0\}$ orthogonal à $\boldsymbol{a}$ et $\boldsymbol{b}$ vérifiant $||\boldsymbol{c}|| \le B_2(V(F_1,\dots,F_r))$. Supposons qu'on choisisse ce vecteur à l'étape 1. Comme $\boldsymbol{a}$ et $\boldsymbol{b}$ sont orthogonaux à $\boldsymbol{c}$, ces vecteurs sont bien combinaisons entières des vecteurs $\boldsymbol{c}_1,\dots,\boldsymbol{c}_{k-1}$ calculés à l'étape 2. Pour voir que les polynômes $F'_1,\dots,F'_r$ définis à l'étape 4 satisfont la condition (2.3.1) de la proposition 2.3.5, il suffit de faire la substitution $X_j := X^{a'_j}Y^{b'_j}$, pour $1 \le j \le k-1$.

Evaluons maintenant le nombre d'opérations binaires effectuées par l'algorithme E2c ainsi que la taille de la sortie. La première étape de l'algorithme nécessite de tester pour $O_{k,h,\delta}(1)$ vecteurs s'ils sont orthogonaux ou non à $\boldsymbol{a}$ et $\boldsymbol{b}$ et nécessite d'effectuer $O_{k,h,\delta}(\log d)$ opérations binaires. Remarquons que si l'algorithme ne s'arrête pas à l'étape 1, on a $||\boldsymbol{c}|| \le B_2(V(F_1,\dots,F_r)) = O_{k,h,\delta}(1)$.

A l'étape 2, l'application de l'algorithme D1 au vecteur $\boldsymbol{c}$ nécessite $O_{k,h,\delta}(1)$ opérations binaires. On obtient alors $||\boldsymbol{c}_i|| = O_{k,h,\delta}(1)$, pour tout $1 \leq i \leq k-1$, d'après le lemme 2.1.4. Pour calculer les coordonnées de $\boldsymbol{a}$ et $\boldsymbol{b}$ à l'étape 3, il suffit d'inverser la matrice des $\boldsymbol{c}_i$, ce qui nécessite $O_{k,h,\delta}(1)$ opérations binaires, puis de multiplier le résultat par $\boldsymbol{a}$ et $\boldsymbol{b}$. La complexité binaire de ces deux multiplications est majorée par $O_{k,h,\delta}(\log d)$ qui est également une borne pour le nombre d'opérations effectuées à cette étape. On obtient ainsi les deux majorations $||\boldsymbol{a}'|| = O_{k,h,\delta}(d)$ et $||\boldsymbol{b}'|| = O_{k,h,\delta}(d)$. L'étape 4 de l'algorithme n'est qu'une substitution qui nécessite $O_{k,h,\delta}(1)$ opérations binaires. On observe également que les polynômes $F'_1, \ldots, F'_r$ sont de hauteur et de degré majorés par $O_{k,h,\delta}(1)$ comme annoncé dans la proposition 2.3.5. □

Remarque. Tester tous les vecteurs de norme au plus $B_2(V(F_1, \ldots, F_r))$ à la première étape n'est pas réalisable en pratique et on peut apporter à cette étape la même modification que l'on avait suggérée à la fin de l'algorithme D2. Pour éviter cette recherche exhaustive, on peut déterminer une base LLL-réduite du sous-groupe de $\mathbb{Z}^k$ formé des vecteurs orthogonaux à $\boldsymbol{a}$ et $\boldsymbol{b}$, à l'aide de l'algorithme D1, et appeler $\boldsymbol{c}$ le plus petit vecteur de cette base. Le lemme [**GG03**, Theorem 16.9, p. 466] et la conjecture 2.3.2 garantissent que s'il existe $\sigma, \tau \in \overline{\mathbb{Q}}^*$ multiplicativement indépendants et tels que $\sigma^{\boldsymbol{a}}\tau^{\boldsymbol{b}}$ appartient à une composante irréductible de $V(F_1, \ldots, F_r)$ de codimension au moins 3, alors $||\boldsymbol{c}|| \leq 2^{k-1}B_2(V(F_1, \ldots, F_r))$. On obtient ainsi un vecteur $\boldsymbol{c}$ non nul, orthogonal aux vecteurs $\boldsymbol{a}, \boldsymbol{b}$ et de norme contrôlée sans avoir à parcourir l'ensemble de tous les vecteurs de norme majorée par $B_2(V(F_1, \ldots, F_r))$. De la même façon que pour l'algorithme D, la complexité théorique de l'algorithme qui découle de cette modification n'est plus linéaire en $\log d$ mais seulement polynomiale en $\log d$ à cause de l'application de l'algorithme LLL, c'est pourquoi on a préféré ne pas inclure cette modification ici.

On peut maintenant démontrer le théorème suivant :

Théorème 2.3.6. — *(Algorithme E2) Il existe un algorithme qui a la propriété suivante. Soient $f_1, f_2, f_3 \in \mathbb{Z}[X, Y]$ des polynômes donnés par leur représentation lacunaire, de degré au plus d, de hauteur au plus h et possédant au plus N termes non nuls. L'algorithme détermine la représentation lacunaire d'un polynôme $p \in \mathbb{Z}[X, Y]$ et un ensemble fini S de triplets de polynômes de $\mathbb{Z}[X, Y]$ tels que :*

$$V(f_1, f_2, f_3) \cap (\mathcal{H}_2 \backslash \mathcal{H}_1) \subseteq V(p) \cup \bigcup_{(p_1,p_2,p_3)\in S} V(p_1, p_2) \backslash V(p_3) \subseteq V(f_1, f_2, f_3). \tag{2.3.2}$$

en au plus

$$O_{N,h}\left(\mathsf{I}(\log d)\log\log d\right)$$

opérations binaires, où d est une borne pour le degré total des polynômes, h une borne pour leur hauteur et N une borne pour le nombre de termes non nuls de chacun de ces polynômes. De plus, $\#S = O_{N,h}(1)$ et tous les polynômes en sortie sont de degré majoré par $O_{N,h}(d)$, de hauteur majorée par $O_{N,h}(1)$ et possèdent au plus $O_{N,h}(1)$ termes non nuls.

Décrivons tout d'abord la procédure. On rappelle la notation suivante déjà introduite dans la section 1.3.1. Pour $F(\boldsymbol{X}) = \sum_{i=1}^{N} a_i \boldsymbol{X}^{\boldsymbol{\alpha}_i} \in \mathbb{C}[X_1^{\pm 1}, \ldots, X_n^{\pm 1}]$, on note

$$JF(\boldsymbol{X}) = X_1^{u_1} \cdots X_n^{u_n} F(\boldsymbol{X}),$$

où les entiers $u_1, \ldots, u_n$ sont aussi petits que possibles et de telle sorte que JF soit un polynôme.

Algorithme E2

Entrée : Trois polynômes $f_1, f_2, f_3 \in \mathbb{Z}[X, Y]$ donnés par leur représentation lacunaire :

$$f_1(X,Y) = \sum_{i=1}^{n} f_{1,i} X^{a_i} Y^{b_i},\ f_2(X,Y) = \sum_{i=1}^{n} f_{2,i} X^{a_i} Y^{b_i},\ f_3(X,Y) = \sum_{i=1}^{n} f_{3,i} X^{a_i} Y^{b_i}.$$

On autorise certains coefficients de f_1, f_2 et f_3 à être nuls de façon à ce qu'ils aient le même support. Si f_1, f_2 et f_3 ont chacun au plus N termes non nuls, on a $n \leq 3N$.
Sortie : Un polynôme $p \in \mathbb{Z}[X, Y]$ et un ensemble fini S de triplets de polynômes de $\mathbb{Z}[X, Y]$ donnés par leur représentation lacunaire satisfaisant les conditions du théorème 2.3.6.

1. Faire

$$\begin{aligned} F_1(X_1,\ldots,X_n) &\longleftarrow \sum_{i=1}^{n} f_{1,i}X_i \\ F_2(X_1,\ldots,X_n) &\longleftarrow \sum_{i=1}^{n} f_{2,i}X_i \\ F_3(X_1,\ldots,X_n) &\longleftarrow \sum_{i=1}^{n} f_{3,i}X_i \\ \boldsymbol{a} &\longleftarrow (a_1,\ldots,a_n) \\ \boldsymbol{b} &\longleftarrow (b_1,\ldots,b_n) \\ r &\longleftarrow 3 \\ k &\longleftarrow n \\ p &\longleftarrow 1 \\ S &\longleftarrow \emptyset. \end{aligned}$$

2. Si $k = 2$ aller à l'étape 3, sinon appliquer l'algorithme E2c à $(F_1,\ldots,F_r,\boldsymbol{a},\boldsymbol{b})$. S'il renvoie « RIEN », aller à l'étape 3. Sinon, appeler $(F'_1,\ldots,F'_r,\boldsymbol{a}',\boldsymbol{b}')$ le quintuplet obtenu en sortie, faire $(F_1,\ldots,F_r,\boldsymbol{a},\boldsymbol{b}) \longleftarrow (F'_1,\ldots,F'_r,\boldsymbol{a}',\boldsymbol{b}')$, $k \longleftarrow k-1$ et recommencer l'étape 2.

3. Faire

$$\begin{aligned} D &\longleftarrow \operatorname{pgcd}(JF_1,\ldots,JF_r) \\ p &\longleftarrow p \cdot D(X^{\boldsymbol{a}}Y^{\boldsymbol{b}}), \\ F_i &\longleftarrow JF_i/D, \qquad \text{pour } 1 \le i \le r. \end{aligned}$$

4. Appliquer l'algorithme E2b à $(F_1,\ldots,F_r)$. S'il renvoie « RIEN », aller à l'étape 6, sinon appeler (P_1,P_2,F_{r+1}) le triplet renvoyé par cet algorithme, puis faire :

$$\begin{aligned} p_1(X,Y) &\longleftarrow P_1(X^{\boldsymbol{a}}Y^{\boldsymbol{b}}) \\ p_2(X,Y) &\longleftarrow P_2(X^{\boldsymbol{a}}Y^{\boldsymbol{b}}) \\ p_3(X,Y) &\longleftarrow F_{r+1}(X^{\boldsymbol{a}}Y^{\boldsymbol{b}}) \\ S &\longleftarrow S \cup \{(p_1,p_2,p_3)\}. \end{aligned}$$

5. Si F_{r+1} est constant ou si $k = 2$, aller à l'étape 6, sinon faire $r \longleftarrow r+1$ et aller à l'étape 2.

6. Renvoyer (p,S) et terminer.

Démonstration du théorème 2.3.6 : Montrons que l'algorithme E2 satisfait bien les conditions de ce théorème. Expliquons tout d'abord pourquoi il termine. Remarquons que l'étape 2 est exécutée au plus $n-2$ fois puisqu'à chaque passage, le nombre k de variables diminue d'une unité. Si, à l'étape 2, on obtient effectivement $k=2$, on passe à l'étape 3 où on divise les polynômes $F_1,\dots,F_r$ par leur pgcd. Ces polynômes définissent alors soit la variété vide soit une variété de dimension 0 puisque $k=2$. Dans le premier cas, l'algorithme E2c à l'étape 4 renvoie « RIEN », on passe à l'étape 6 et on termine. Dans le second cas, on applique l'étape 4 puis l'étape 5 qui renvoie à l'étape 6 (car $k=2$) où on termine à nouveau. Supposons maintenant qu'on passe à l'étape 3 avec $k>2$, ce sera le cas si l'algorithme E2c renvoie « RIEN » à l'étape 2 au bout d'un certain temps. On exécute alors les étapes 3 puis 4. Si l'algorithme E2c renvoie « RIEN » à l'étape 4, on passe à l'étape 6 et on termine. Sinon on passe à l'étape 5. Si F_{r+1} est constant, on va à l'étape 6 et on termine, sinon on retourne à l'étape 2 en remplaçant r par $r+1$. D'après la remarque faite après la démonstration de la proposition 2.3.5, on a alors

$$\operatorname{codim}(V(F_1,\dots,F_r,F_{r+1}))=3.$$

Ainsi, lorsqu'on applique à nouveau l'algorithme E2c à l'étape 2, celui-ci va à nouveau faire diminuer le nombre k de variables. Ceci garantit que l'algorithme termine.

Montrons maintenant que le polynôme p et l'ensemble S renvoyés par l'algorithme vérifient bien la condition (2.3.2) du théorème 2.3.6. Soient $\sigma,\tau\in\overline{\mathbb{Q}}^*$ multiplicativement indépendants tels que

$$f_1(\sigma,\tau)=f_2(\sigma,\tau)=f_3(\sigma,\tau)=0.$$

Alors $\sigma^{\boldsymbol{a}}\tau^{\boldsymbol{b}}\in V(F_1,F_2,F_3)$, où F_1,F_2,F_3 sont les polynômes à n variables définis à l'étape 1. De plus, tant qu'on ne passe pas pour la première fois à l'étape 3, d'après la proposition 2.3.5 (Algorithme E2c), on a après chaque application de l'étape 2 :

$$\begin{aligned}f_1(X,Y)&=F_1(X^{\boldsymbol{a}}Y^{\boldsymbol{b}})\\ f_2(X,Y)&=F_2(X^{\boldsymbol{a}}Y^{\boldsymbol{b}})\\ f_3(X,Y)&=F_3(X^{\boldsymbol{a}}Y^{\boldsymbol{b}}).\end{aligned}$$

Comme on l'a déjà vu, après au plus $n-2$ itérations de l'étape 2, on passe à l'étape 3. Si $k=2$, on exécute les étapes 3 puis 4 et on obtient :

$$V(F_1,F_2,F_3)=V(D),$$

si l'algorithme E2c renvoie « RIEN » et sinon

$$V(F_1, F_2, F_3) = V(D) \cup \big(V(P_1, P_2) \backslash V(F_4)\big) \cup V(F_1, F_2, F_3, F_4),$$

où P_1, P_2 et F_4 sont les polynômes calculés à l'étape 4 qui vérifient la proposition 2.3.5 appliquée aux polynômes F_1, F_2, F_3. La deuxième condition de cette proposition implique dans ce cas que

$$V(F_1, F_2, F_3, F_4) = \emptyset,$$

puisque $V(F_4)$ ne contient aucune composante de codimension 2 (et donc aucun point) de $V(F_1, F_2, F_3)$. On a alors

$$V(F_1, F_2, F_3) = V(D) \cup \big(V(P_1, P_2) \backslash V(F_4)\big).$$

Dans les deux cas, l'algorithme termine, la première inclusion de (2.3.2) est vérifiée et la seconde est en fait une égalité. Supposons maintenant $k \geq 3$. D'après la proposition 2.3.5 (Algorithme E2c), tous les points de

$$V(F_1, F_2, F_3) \cap (\mathcal{H}_2 \backslash \mathcal{H}_1)$$

sont dans une composante irréductible de $V(F_1, F_2, F_3)$ de codimension au plus 2. Si $\sigma^{\boldsymbol{a}}\tau^{\boldsymbol{b}}$ est dans une composante de codimension 1 de $V(F_1, F_2, F_3)$, il annule le pgcd de F_1, F_2 et F_3, c'est à dire le polynôme D déterminé à l'étape 3. Ainsi, (σ, τ) annule le polynôme p renvoyé par l'algorithme $E2$ qui, par construction, est un diviseur des polynômes f_1, f_2 et f_3. Supposons maintenant que $\sigma^{\boldsymbol{a}}\tau^{\boldsymbol{b}}$ appartient à une composante de $V(F_1, F_2, F_3)$ de codimension 2 et appelons encore F_1, F_2 et F_3 les polynômes obtenus après division par leur pgcd à l'étape 3. On a alors après l'étape 4 :

$$\begin{aligned} V(F_1, F_2, F_3) &= \Big(V(F_1, F_2, F_3) \backslash V(F_4)\Big) \cup V(F_1, F_2, F_3, F_4) \\ &= \Big(V(P_1, P_2) \backslash V(F_4)\Big) \cup V(F_1, F_2, F_3, F_4), \end{aligned} \tag{2.3.3}$$

d'après la proposition 2.3.5 (Algorithme E2c). Si $\sigma^{\boldsymbol{a}}\tau^{\boldsymbol{b}} \in V(P_1, P_2) \backslash V(F_4)$, alors on a $(\sigma, \tau) \in V(p_1, p_2) \backslash V(p_3)$. Par ailleurs, $V(p_1, p_2) \backslash V(p_3) \subseteq V(f_1, f_2, f_3)$ d'après l'égalité (2.3.3). Supposons maintenant que $\sigma^{\boldsymbol{a}}\tau^{\boldsymbol{b}} \in V(F_1, F_2, F_3, F_4)$. On a remarqué en (2.2.8), après la proposition 2.2.6 (Algorithme E2b), que $V(F_1, F_2, F_3, F_4)$ est nécessairement de codimension 3. On retourne alors à l'étape 2 et comme de plus la variété $V(F_1, F_2, F_3, F_4)$ possède un point de $\mathcal{H}_2 \backslash \mathcal{H}_1$ (le point $\sigma^{\boldsymbol{a}}\tau^{\boldsymbol{b}}$) dans une de ses composantes de codimension au moins 3, l'application de l'algorithme E2c au sextuplet $(F_1, F_2, F_3, F_4, \boldsymbol{a}, \boldsymbol{b})$ fait diminuer le nombre de variables d'une unité. On peut alors appliquer à nouveau le raisonnement précédent et finalement, (σ, τ) appartient à $V(p_1, p_2) \backslash V(p_3)$, où (p_1, p_2, p_3) est l'un des triplets de l'ensemble S renvoyé par l'algorithme E2. Ainsi, la première inclusion de (2.3.2)

est toujours vérifiée. De plus, après chaque application de l'étape 4, on a de façon analogue à (2.3.3) :

$$\begin{aligned} V(F_1,\dots,F_r) &= \Big(V(F_1,\dots,F_r)\backslash V(F_{r+1})\Big) \cup V(F_1,\dots,F_r,F_{r+1}) \\ &= \Big(V(P_1,P_2)\backslash V(F_{r+1})\Big) \cup V(F_1\dots,F_r,F_{r+1}), \end{aligned}$$

d'après la proposition 2.3.5 (Algorithme E2c). Ceci garantit la seconde inclusion de (2.3.2) et achève la correction de l'algorithme.

Il ne nous reste donc qu'à évaluer le temps d'exécution de l'algorithme E2. Remarquons tout d'abord que les étapes 2 à 5 sont répétées au plus $n-2$ fois car à chaque application de l'étape 2, le nombre de variables k diminue de 1. Remarquons également que le nombre r de polynômes n'excède jamais n. Il nous faut maintenant déterminer une borne pour le nombre d'opérations binaires nécessaires à l'exécution de chaque étape.

Il est clair que l'étape 1 nécessite

$$O_{N,h}(\log d)$$

opérations binaires.

A l'étape 2, on doit appliquer l'algorithme E2c à $r \leq n \leq 3N$ polynômes de degré et de hauteur $O_{N,h}(1)$ et à deux vecteurs dont les coordonnées sont majorées en valeur absolue par $O_{N,h}(d)$. Ceci nécessite donc

$$O_{N,h}(\mathsf{I}(\log d)\log\log d)$$

opérations binaires d'après la proposition 2.3.5.

Le calcul de pgcd à l'étape 3 nécessite $O_{N,h}(1)$ opérations binaires puisque les polynômes $F_1,\dots,F_r$ sont de degré et de hauteur $O_{N,h}(1)$. Le calcul du polynôme p nécessite ensuite $O_{N,h}(\log d)$ opérations binaires et on vérifie facilement que ce polynôme possède $O_{N,h}(1)$ termes non nuls, est de hauteur $O_{N,h}(1)$ et est de degré majoré par $O_{N,h}(d)$. Enfin, les divisions de polynômes nécessitent encore $O_{N,h}(1)$ opérations binaires. Le nombre d'opérations binaires effectuées à cette étape peut donc être majoré par

$$O_{N,h}(\log d).$$

Appliquer l'algorithme E2c aux polynômes $F_1,\dots,F_r$ à l'étape 4 nécessite $O_{N,h}(1)$ opérations binaires puisque ces polynômes sont de degré et de hauteur $O_{N,h}(1)$. L'évaluation des polynômes P_1, P_2 et F_{r+1} requiert quant à elle

$$O_{N,h}(\log d)$$

opérations binaires ce qui est aussi une borne pour le nombre total d'opérations binaires effectuées à cette étape.

Finalement, l'exécution de l'algorithme E2 nécessite au total

$$O_{N,h}(\mathsf{I}(\log d)\log\log d)$$

opérations binaires. □

Remarque. Dans le cas où les polynômes f_1, f_2 et f_3 n'ont pas de facteur cyclotomique généralisé en commun, on a :

$$p = \mathrm{pgcd}(f_1, f_2, f_3).$$

Dans le cas contraire, c'est à dire si les polynômes f_1, f_2 et f_3 ont un facteur cyclotomique généralisé en commun, on peut juste assurer que p est un diviseur de leur pgcd et que le polynôme $\mathrm{pgcd}(f_1, f_2, f_3)/p$ est produit de polynômes cyclotomiques généralisés. En effet, par construction, le polynôme p est un facteur de $\mathrm{pgcd}(f_1, f_2, f_3)$ et d'après le théorème 2.3.6, le polynôme $\mathrm{pgcd}(f_1, f_2, f_3)/p$ ne s'annule qu'en un nombre fini de points de $\mathcal{H}_2\backslash\mathcal{H}_1$ et ne peut donc pas avoir de facteur non cyclotomique généralisé.

2.3.4. Points de $\mathcal{H}_1\backslash\mathcal{H}_0$. — Remarquons tout d'abord que les résultats de cette section sont inconditionnels. En particulier, ils ne dépendent pas de la conjecture de Zilber.

Ce cas est de loin le plus technique et il faudrait certainement pouvoir améliorer cette partie pour obtenir une implémentation pratique de l'algorithme E.

La méthode est analogue à celle utilisée pour déterminer les points de $\mathcal{H}_2\backslash\mathcal{H}_1$: on décrit d'abord un algorithme qui permet de diminuer le nombre de variables à chaque étape avant de donner la procédure pour déterminer une représentation des points de $V(f_1, f_2, f_3) \cap (\mathcal{H}_1\backslash\mathcal{H}_0)$ que nous appellerons Algorithme E1. Les coefficients des polynômes auxquels nous appliquons cette procédure auxiliaire ne sont pas entiers mais appartiennent à une extension cyclotomique de $\mathbb{Z}$. Pour conserver malgré tout des polynômes à coefficients entiers, on a préféré introduire une variable additionnelle Z que l'on évaluera parfois en des racines de l'unité. On est alors amenés à introduire la notation suivante : pour $F \in \mathbb{Q}[X_1, \ldots, X_k, Z]$, et $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{Z}^k$, on note

$$F(X^{\boldsymbol{a}}Y^{\boldsymbol{b}}, Z) := F(X^{a_1}Y^{b_1}, \ldots, X^{a_k}Y^{b_k}, Z).$$

On peut maintenant décrire la procédure auxiliaire, analogue de l'algorithme de la proposition 2.3.5 pour les points de $\mathcal{H}_1\backslash\mathcal{H}_0$.

Proposition 2.3.7. — *(Algorithme E1a) Il existe un algorithme qui a la propriété suivante. Soient $\zeta \in \mu_\infty$, $F_1, F_2, F_3 \in \mathbb{Z}[X_1, \ldots, X_k, Z]$, $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{Z}^k$ et*

$\alpha, \beta \in \mathbb{Z}$ tels que $||\boldsymbol{a}||, ||\boldsymbol{b}||, |\alpha|, |\beta| \leq d$. On note

$$V := V\left(F_1(X^{\boldsymbol{a}}Y^{\boldsymbol{b}}, \zeta), F_2(X^{\boldsymbol{a}}Y^{\boldsymbol{b}}, \zeta), F_3(X^{\boldsymbol{a}}Y^{\boldsymbol{b}}, \zeta), X^\alpha Y^\beta - \zeta\right) \subseteq \mathbb{G}_m^2.$$

Supposons qu'il existe $(\sigma, \tau) \in (\overline{\mathbb{Q}}^*)^2 \backslash \mu_\infty^2$ tel que $\sigma^\alpha \tau^\beta = \zeta$ et $\sigma^{\boldsymbol{a}}\tau^{\boldsymbol{b}}$ appartient à une composante irréductible de

$$V(F_1(X_1, \ldots, X_k, \zeta), F_2(X_1, \ldots, X_k, \zeta), F_3(X_1, \ldots, X_k, \zeta))$$

de codimension au moins 2. Alors l'algorithme renvoie un ensemble fini S dont les éléments sont des octuplets $(F_1', F_2', F_3', \boldsymbol{a}', \boldsymbol{b}', \alpha', \beta', \zeta')$ tels que :

$$\begin{aligned} &F_1', F_2', F_3' \in \mathbb{Z}[X_1, \ldots, X_{k-1}, Z], \\ &\boldsymbol{a}', \boldsymbol{b}' \in \mathbb{Z}^{k-1}, \\ &\alpha', \beta' \in \mathbb{Z}, \\ &\zeta' \in \mu_\infty \end{aligned}$$

et tels que $V \cap (\mathcal{H}_1 \backslash \mathcal{H}_0)$ soit égale à la réunion pour $(F_1', F_2', F_3', \boldsymbol{a}', \boldsymbol{b}', \alpha', \beta', \zeta') \in S$ de

$$V\left(F_1'(X^{\boldsymbol{a}'}Y^{\boldsymbol{b}'}, \zeta'), F_2'(X^{\boldsymbol{a}'}Y^{\boldsymbol{b}'}, \zeta'), F_3'(X^{\boldsymbol{a}'}Y^{\boldsymbol{b}'}, \zeta'), X^{\alpha'}Y^{\beta'} - \zeta'\right) \cap (\mathcal{H}_1 \backslash \mathcal{H}_0), \tag{2.3.4}$$

et tels que cette réunion soit incluse dans V.

Dans le cas contraire l'algorithme renvoie « RIEN ».

Cet algorithme nécessite $O(\log d)$ opérations binaires. De plus, le cardinal de S est majoré par $O(1)$ et pour chaque élément $(F_1', F_2', F_3', \boldsymbol{a}', \boldsymbol{b}', \alpha', \beta', \zeta') \in S$, on a :

$$\begin{aligned} &F_1', F_2', F_3' \text{ sont de degré et de hauteur majorés par } O(1), \\ &||\boldsymbol{a}'||, ||\boldsymbol{b}'||, |\alpha'|, |\beta'| = O(d), \\ &\operatorname{ord}(\zeta') = O(1), \end{aligned} \tag{2.3.5}$$

où toutes les constantes implicites dans les O dépendent de ζ, f_1, f_2, f_3.

Décrivons tout d'abord la procédure :

Algorithme E1a

Entrée : $(F_1, F_2, F_3, \boldsymbol{a}, \boldsymbol{b}, \alpha, \beta, \zeta)$, où $F_1, F_2, F_3 \in \mathbb{Q}[X_1, \ldots, X_k, Z]$, $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{Z}^k$ et $\alpha, \beta \in \mathbb{Z}$, $\zeta \in \mu_\infty$.
Sortie : Un ensemble fini S dont les éléments sont des octuplets satisfaisant les conditions de la proposition 2.3.7 ou « RIEN ».

1. Faire $S \longleftarrow \emptyset$.
2. Déterminer l'ensemble C des vecteurs $\boldsymbol{c} \in \mathbb{Z}^k \backslash \{0\}$ tels que

$$||\boldsymbol{c}|| \leq B_1\big(V(F_1(X_1, \ldots, X_k, \zeta), F_2(X_1, \ldots, X_k, \zeta), F_3(X_1, \ldots, X_k, \zeta))\big),$$

où B_1 est défini au théorème 2.1.3), et tels que (α, β) et $(\langle \boldsymbol{a}, \boldsymbol{c}\rangle, \langle \boldsymbol{b}, \boldsymbol{c}\rangle)$ sont colinéaires.

3. Si $C = \emptyset$, renvoyer « RIEN » et terminer. Sinon pour chaque $\boldsymbol{c} \in C$ faire :

   (a) Déterminer une base du sous-groupe de $\mathbb{Z}^{k+1}$ formé des vecteurs orthogonaux au vecteur $(c_1, \ldots, c_k, -1)$ avec l'algorithme D1, puis déterminer la forme normale d'Hermite de cette base et noter cette matrice $(k+1) \times k$:

$$H = \begin{pmatrix} c_{1,1} & c_{1,2} & \cdots & c_{1,k-1} & c_{1,k} \\ c_{2,1} & c_{2,2} & \cdots & c_{2,k-1} & c_{2,k} \\ 0 & c_{3,2} & \cdots & c_{3,k-1} & c_{3,k} \\ \vdots & \ddots & \ddots & \vdots & \vdots \\ \vdots & & \ddots & c_{k,k-1} & c_{k,k} \\ 0 & \cdots & \cdots & 0 & m \end{pmatrix}.$$

   (b) Déterminer les coordonnées $\boldsymbol{a}'$ et $\boldsymbol{b}'$ des vecteurs $(a_1, \ldots, a_k, \langle \boldsymbol{a}, \boldsymbol{c}\rangle)$ et $(b_1, \ldots, b_k, \langle \boldsymbol{b}, \boldsymbol{c}\rangle)$ dans la base formée par les colonnes de la matrice H du sous-groupe $M \subseteq \mathbb{Z}^{k+1}$ qu'elles engendrent.

   (c) Déterminer un générateur (α', β') du sous-groupe de $\mathbb{Z}^2 : N = \mathbb{Z}(\alpha, \beta) + \mathbb{Z}(a'_k, b'_k)$, qui est de rang 1, et deux entiers u et v tels que

$$(\alpha', \beta') = u(\alpha, \beta) + v(a'_k, b'_k).$$

   Déterminer les deux entiers t_1 et t_2 tels que

$$\begin{aligned} (\alpha, \beta) &= t_1(\alpha', \beta'), \\ (a'_k, b'_k) &= t_2(\alpha', \beta'). \end{aligned}$$

   (d) Pour chaque racine m-ième de l'unité η :

      (i) Faire

$$\begin{aligned} \zeta' &\longleftarrow \zeta^u \eta^v, \\ F'_1(X_1, \ldots, X_{k-1}, Z) &\longleftarrow F_1(Z^{t_2 c_{1,k}} Y_1, \ldots, Z^{t_2 c_{k,k}} Y_k, Z^{t_1}), \\ F'_2(X_1, \ldots, X_{k-1}, Z) &\longleftarrow F_2(Z^{t_2 c_{1,k}} Y_1, \ldots, Z^{t_2 c_{k,k}} Y_k, Z^{t_1}), \\ F'_3(X_1, \ldots, X_{k-1}, Z) &\longleftarrow F_3(Z^{t_2 c_{1,k}} Y_1, \ldots, Z^{t_2 c_{k,k}} Y_k, Z^{t_1}), \end{aligned}$$

où

$$\begin{aligned} Y_1 &= X_1^{c_{1,1}} X_2^{c_{1,2}} \cdots X_{k-1}^{c_{1,k-1}}, \\ Y_2 &= X_1^{c_{2,1}} X_2^{c_{2,2}} \cdots X_{k-1}^{c_{2,k-1}}, \\ Y_3 &= X_2^{c_{3,2}} \cdots X_{k-1}^{c_{3,k-1}}, \\ &\vdots \\ Y_k &= X_{k-1}^{c_{k,k-1}}. \end{aligned}$$

(ii) Faire $S \longleftarrow S \cup \{(F_1', F_2', F_3', \boldsymbol{a}', \boldsymbol{b}', \alpha', \beta', \zeta')\}$.

4. Renvoyer S et terminer.

Démonstration de la proposition 2.3.7 : Montrons que cet algorithme vérifie bien les conditions de cette proposition. Supposons qu'il existe $(\sigma, \tau) \in V \cap (\mathcal{H}_1 \backslash \mathcal{H}_0)$ tel que $\sigma^{\boldsymbol{a}} \tau^{\boldsymbol{b}}$ appartient à une composante de

$$V(F_1(X_1, \ldots, X_k, \zeta), F_2(X_1, \ldots, X_k, \zeta), F_3(X_1, \ldots, X_k, \zeta))$$

de codimension au moins 2. On a alors :

$$\sigma^{\alpha} \tau^{\beta} = \zeta$$

et d'après le théorème 2.1.3, il existe $\boldsymbol{c}_0 \in \mathbb{Z}^k \backslash \{0\}$ dont la norme est majorée par

$$B_1\big(V(F_1(X_1, \ldots, X_k, \zeta), F_2(X_1, \ldots, X_k, \zeta), F_3(X_1, \ldots, X_k, \zeta))\big)$$

tel que

$$\sigma^{\langle \boldsymbol{a}, \boldsymbol{c}_0 \rangle} \tau^{\langle \boldsymbol{b}, \boldsymbol{c}_0 \rangle} = 1.$$

De plus, comme les nombres σ et τ ne sont pas tous deux des racines de l'unité, les vecteurs $(\langle \boldsymbol{a}, \boldsymbol{c}_0 \rangle, \langle \boldsymbol{b}, \boldsymbol{c}_0 \rangle)$ et (α, β) sont nécessairement colinéaires. Ainsi le vecteur $\boldsymbol{c}_0$ appartient à l'ensemble C déterminé à l'étape 2. Considérons maintenant l'étape 3 en supposant que le vecteur $\boldsymbol{c} = \boldsymbol{c}_0 = (c_1, \ldots, c_k)$ a été choisi. Le vecteur $(c_1, \ldots, c_k, -1)$ est alors orthogonal au vecteur $(a_1, \ldots, a_k, \langle \boldsymbol{a}, \boldsymbol{c} \rangle)$ et au vecteur $(b_1, \ldots, b_k, \langle \boldsymbol{b}, \boldsymbol{c} \rangle)$. Ces deux vecteurs appartiennent donc au sous-groupe de $\mathbb{Z}^{k+1}$ engendré par les vecteurs colonnes de la matrice H calculée à l'étape 3a. Il est alors possible de déterminer les coordonnées $\boldsymbol{a}'$ et $\boldsymbol{b}'$ de $(a_1, \ldots, a_k, \langle \boldsymbol{a}, \boldsymbol{c} \rangle)$ et $(b_1, \ldots, b_k, \langle \boldsymbol{b}, \boldsymbol{c} \rangle)$ dans cette base. En regardant la dernière coordonnée de ces vecteurs, on observe que

$$m a_k' = \langle \boldsymbol{a}, \boldsymbol{c} \rangle \quad \text{et} \quad m b_k' = \langle \boldsymbol{b}, \boldsymbol{c} \rangle,$$

ce qui prouve que $N = \mathbb{Z}(\alpha, \beta) + \mathbb{Z}(a_k', b_k')$ est de rang 1. Par ailleurs, comme σ et τ vérifient l'équation binomiale $\sigma^{\langle \boldsymbol{a}, \boldsymbol{c} \rangle} \tau^{\langle \boldsymbol{b}, \boldsymbol{c} \rangle} = 1$, on a

$$\sigma^{a_k'} \tau^{b_k'} = \eta_0,$$

pour une certaine racine m-ième de l'unité η_0. Supposons qu'on choisisse $\eta = \eta_0$ à l'étape 3d. Alors le système

$$\begin{cases} \sigma^{\alpha}\tau^{\beta} = \zeta \\ \sigma^{a'_k}\tau^{b'_k} = \eta \end{cases}$$

est équivalent à l'unique équation

$$\sigma^{\alpha'}\tau^{\beta'} = \zeta', \tag{2.3.6}$$

où (α', β') est le générateur de N déterminé à l'étape 3c et $\zeta' = \zeta^u\eta^v$ est la racine de l'unité déterminée à l'étape 3(d)i. Considérons les polynômes F'_1, F'_2, F'_3 calculés à l'étape 3(d)i et montrons que (σ, τ) appartient à

$$V\left(F'_1(X^{\boldsymbol{a}'}Y^{\boldsymbol{b}'}, \zeta'), F'_2(X^{\boldsymbol{a}'}Y^{\boldsymbol{b}'}, \zeta'), F'_3(X^{\boldsymbol{a}'}Y^{\boldsymbol{b}'}, \zeta'), X^{\alpha'}Y^{\beta'} - \zeta'\right) \subseteq \mathbb{G}_m^2.$$

D'après l'égalité 2.3.6, (σ, τ) vérifie l'équation

$$X^{\boldsymbol{\alpha}'}Y^{\boldsymbol{\beta}'} = \zeta',$$

et il nous reste à montrer que, pour $j = 1, 2, 3$, on a

$$F'_j\left(\sigma^{\boldsymbol{a}'}\tau^{\boldsymbol{b}'}, \zeta'\right) = 0.$$

D'après la définition de F'_j à l'étape 3(d)i, $F'_j\left(\sigma^{\boldsymbol{a}'}\tau^{\boldsymbol{b}'}, \zeta'\right)$ est égal à :

$$F_j\left(\zeta'^{t_2 c_{1,k}}\sigma^{\langle \boldsymbol{a}', \boldsymbol{c}_1\rangle - a'_k c_{1,k}}\tau^{\langle \boldsymbol{b}', \boldsymbol{c}_1\rangle - b'_k c_{1,k}}, \ldots, \zeta'^{t_2 c_{k,k}}\sigma^{\langle \boldsymbol{a}', \boldsymbol{c}_k\rangle - a'_k c_{k,k}}\tau^{\langle \boldsymbol{b}', \boldsymbol{c}_k\rangle - b'_k c_{k,k}}, \zeta'^{t_1}\right),$$

où $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_k$ désignent les k premières lignes de la matrice H déterminée lors de l'étape 3a. Or, d'après la définition de t_1 et t_2 à l'étape 3c, on a

$$\zeta'^{t_1} = \zeta \qquad \text{et} \qquad \zeta'^{t_2} = \eta.$$

De plus, pour $1 \leq i \leq k$, on a également :

$$\langle \boldsymbol{a}', \boldsymbol{c}_i\rangle = a_i \qquad \text{et} \qquad \langle \boldsymbol{b}', \boldsymbol{c}_i\rangle = b_i,$$

par définition de $\boldsymbol{a}'$ et $\boldsymbol{b}'$. Ainsi, $F'_j\left(\sigma^{\boldsymbol{a}'}\tau^{\boldsymbol{b}'}, \zeta'\right)$ est égal à :

$$F_j\left(\eta^{c_{1,k}}\sigma^{a_1 - a'_k c_{1,k}}\tau^{b_1 - b'_k c_{1,k}}, \ldots \eta^{c_{k,k}}\sigma^{a_k - a'_k c_{k,k}}\tau^{b_k - b'_k c_{k,k}}, \zeta\right).$$

En utilisant le fait que

$$\sigma^{a'_k}\tau^{b'_k} = \eta,$$

on obtient finalement :

$$\begin{aligned} F'_j\left(\sigma^{\boldsymbol{a}'}\tau^{\boldsymbol{b}'}, \zeta'\right) &= F_j\left(\sigma^{a_1}\tau^{b_1}, \ldots, \sigma^{a_k}\tau^{b_k}, \zeta\right) \\ &= F_j\left(\sigma^{\boldsymbol{a}}\tau^{\boldsymbol{b}}, \zeta\right) \\ &= 0, \end{aligned}$$

et on a bien le résultat voulu.

Tout ceci prouve que les points de $V \cap (\mathcal{H}_1 \backslash \mathcal{H}_0)$ sont inclus dans la réunion pour $(F_1', F_2', F_3', \boldsymbol{a}', \boldsymbol{b}', \alpha', \beta', \zeta') \in S$ de

$$V\left(F_1'(X^{\boldsymbol{a}'}Y^{\boldsymbol{b}'}, \zeta'), F_2'(X^{\boldsymbol{a}'}Y^{\boldsymbol{b}'}, \zeta'), F_3'(X^{\boldsymbol{a}'}Y^{\boldsymbol{b}'}, \zeta'), X^{\alpha'}Y^{\beta'} - \zeta'\right) \cap (\mathcal{H}_1 \backslash \mathcal{H}_0).$$

Pour démontrer l'inclusion inverse, il suffit de constater que

$$\begin{aligned}
F_1'\left(X^{\boldsymbol{a}'}Y^{\boldsymbol{b}'}, \zeta'\right) &= F_1\left(X^{\boldsymbol{a}}Y^{\boldsymbol{b}}, \zeta\right) \mod X^{\alpha'}Y^{\beta'} - \zeta', \\
F_2'\left(X^{\boldsymbol{a}'}Y^{\boldsymbol{b}'}, \zeta'\right) &= F_2\left(X^{\boldsymbol{a}}Y^{\boldsymbol{b}}, \zeta\right) \mod X^{\alpha'}Y^{\beta'} - \zeta', \\
F_3'\left(X^{\boldsymbol{a}'}Y^{\boldsymbol{b}'}, \zeta'\right) &= F_3\left(X^{\boldsymbol{a}}Y^{\boldsymbol{b}}, \zeta\right) \mod X^{\alpha'}Y^{\beta'} - \zeta', \\
X^{\alpha}Y^{\beta} - \zeta &= 0 \mod X^{\alpha'}Y^{\beta'} - \zeta'.
\end{aligned}$$

On a donc bien la deuxième inclusion et l'algorithme E1 satisfait les conditions de la proposition 2.3.7, ce qui achève sa correction.

Evaluons maintenant le temps d'exécution de cet algorithme. Toutes les constantes implicites dans les O qui vont suivre dépendent éventuellement de ζ et des polynômes F_1, F_2, et F_3 mais pour alléger les notations, on ne les fait pas apparaître comme indice dans les O.

A l'étape 2, on doit considérer tous les vecteurs $\boldsymbol{c}$ de $\mathbb{Z}^k \backslash \{0\}$ de norme majorée par

$$B_1\big(V(F_1(X_1, \ldots, X_k, \zeta), F_2(X_1, \ldots, X_k, \zeta), F_3(X_1, \ldots, X_k, \zeta))\big)$$

et tester pour chacun d'eux si les vecteurs (α, β) et $(\langle \boldsymbol{a}, \boldsymbol{c}\rangle, \langle \boldsymbol{b}, \boldsymbol{c}\rangle)$ sont colinéaires. Tester cette condition nécessite d'évaluer les deux produits scalaires puis de calculer un déterminant 2×2 et donc requiert $O(\mathsf{I}(\log d) \log\log d)$ opérations binaires. Comme il y a $O(1)$ vecteurs $\boldsymbol{c}$ à considérer, cette étape nécessite d'effectuer $O(\mathsf{I}(\log d) \log\log d)$ opérations binaires.

Passons maintenant à l'étape 3 et fixons $\boldsymbol{c} \in C$. Comme $||\boldsymbol{c}|| = O(1)$, tous les calculs de l'étape 3a nécessitent $O(1)$ opérations binaires. A l'étape 3b, pour obtenir les coordonnées $\boldsymbol{a}'$ et $\boldsymbol{b}'$ de $(a_1, \ldots, a_k, \langle \boldsymbol{a}, \boldsymbol{c}\rangle)$ et $(b_1, \ldots, b_k, \langle \boldsymbol{b}, \boldsymbol{c}\rangle)$ dans la base H, il suffit d'inverser la matrice H, ce qui coûte $O(1)$ opérations binaires, et de multiplier le résultat obtenu respectivement par $(a_1, \ldots, a_k, \langle \boldsymbol{a}, \boldsymbol{c}\rangle)$ et $(b_1, \ldots, b_k, \langle \boldsymbol{b}, \boldsymbol{c}\rangle)$. Tout ceci nécessite donc encore $O(\log d)$ opérations binaires. A l'étape 3c, on doit appliquer l'algorithme d'Euclide étendu qui nécessite dans ce cas $O(\mathsf{I}(\log d) \log\log d)$ opérations binaires. Comme $m = O(1)$ à l'étape 3d, tous les calculs effectués à l'étape 3(d)i nécessitent $O(1)$ opérations binaires et écrire les résultats à l'étape 3(d)ii nécessite $O(\log d)$ opérations binaires. Finalement, comme $\#C = O(1)$, les étapes 3a à 3d sont réalisées $O(1)$ fois et le nombre d'opérations binaires effectuées est $O(\mathsf{I}(\log d) \log\log d)$.

Finalement, l'exécution de cet algorithme nécessite donc bien

$$O(\mathsf{I}(\log d)\log\log d)$$

opérations binaires. □

On est maintenant en mesure de décrire la procédure qui permet de déterminer une représentation des points de $V(f_1,f_2,f_3)\cap(\mathcal{H}_1\backslash\mathcal{H}_0)$. On va pour cela démontrer le résultat suivant :

Théorème 2.3.8. — *(Algorithme E1) Il existe un algorithme qui a la propriété suivante. Soient $f_1,f_2,f_3\in\mathbb{Z}[X,Y]$ donnés par leur représentation lacunaire, de degré au plus d, de hauteur au plus h et possédant chacun au plus N termes non nuls. L'algorithme détermine un ensemble fini S de quadruplets (p,α,β,ζ) tels que $p\in\mathbb{Z}[X,Y,Z]$ est donné par sa représentation lacunaire, $\alpha,\beta\in\mathbb{Z}$, $\zeta\in\mu_\infty$ et*

$$V(f_1,f_2,f_3)\cap(\mathcal{H}_1\backslash\mathcal{H}_0)\subseteq\bigcup_{(p,\alpha,\beta,\zeta)\in S}V\left(p(X,Y,\zeta),X^\alpha Y^\beta-\zeta\right)\subseteq V(f_1,f_2,f_3). \tag{2.3.7}$$

Cet algorithme nécessite au plus

$$O_{N,h}\left(\mathsf{I}(\log d)\log\log d\right)$$

opérations binaires.

De plus, $\#S=O_{N,h}(1)$ et pour chaque $(p,\alpha,\beta,\zeta)\in S$, le polynôme p est de degré majoré par $O_{N,h}(d)$, de hauteur majorée par $O_{N,h}(1)$ et possède au plus $O_{N,h}(1)$ termes non nuls. On a également, $|\alpha|,|\beta|=O_{N,h}(d)$ et $\mathrm{ord}(\zeta)=O_{N,h}(1)$.

Décrivons la procédure :

Algorithme E1

Entrée : Trois polynômes $f_1,f_2,f_3\in\mathbb{Z}[X,Y]$ donnés par leur représentation lacunaire :

$$f_1(X,Y)=\sum_{i=1}^{n}f_{1,i}X^{a_i}Y^{b_i},\ \ f_2(X,Y)=\sum_{i=1}^{n}f_{2,i}X^{a_i}Y^{b_i},\ \ f_3(X,Y)=\sum_{i=1}^{n}f_{3,i}X^{a_i}Y^{b_i}.$$

On autorise ici certains coefficients de f_1, f_2 et f_3 à être nuls de façon à ce qu'ils aient le même support. Si f_1,f_2 et f_3 ont chacun au plus N termes non nuls, on a $n\leq 3N$.

Sortie : Un ensemble fini S de quadruplets (p,α,β,ζ) satisfaisant les conditions du théorème 2.3.8.

Au cours de l'exécution de la procédure, on introduit deux ensembles S_1 et S_2 dont les éléments sont des nonuplets.

1. Faire
$$\begin{aligned} F_1(X_1,\dots,X_n,Z) &\longleftarrow \sum_{i=1}^{n} f_{1,i}X_i,\\ F_2(X_1,\dots,X_n,Z) &\longleftarrow \sum_{i=1}^{n} f_{2,i}X_i,\\ F_3(X_1,\dots,X_n,Z) &\longleftarrow \sum_{i=1}^{n} f_{3,i}X_i,\\ \boldsymbol{a} &\longleftarrow (a_1,\dots,a_n),\\ \boldsymbol{b} &\longleftarrow (b_1,\dots,b_n),\\ S_1 &\longleftarrow \{(F_1,F_2,F_3,\boldsymbol{a},\boldsymbol{b},0,0,1,n)\},\\ S_2 &\longleftarrow \emptyset,\\ S &\longleftarrow \emptyset. \end{aligned}$$

2. Si $S_1 = \emptyset$, aller à l'étape 3, sinon choisir $(F_1,F_2,F_3,\boldsymbol{a},\boldsymbol{b},\alpha,\beta,\zeta,k) \in S_1$ et faire
$$S_1 \longleftarrow S_1\backslash\{(F_1,F_2,F_3,\boldsymbol{a},\boldsymbol{b},\alpha,\beta,\zeta,k)\}.$$
Si $k = 1$, faire
$$S_2 \longleftarrow S_2 \cup \{(F_1,F_2,F_3,\boldsymbol{a},\boldsymbol{b},\alpha,\beta,\zeta,1)\}$$
puis retourner à l'étape 2. Si $k > 1$, appliquer l'algorithme E1a à l'octuplet $(F_1,F_2,F_3,\boldsymbol{a},\boldsymbol{b},\alpha,\beta,\zeta)$. S'il renvoie « RIEN », faire
$$S_2 \longleftarrow S_2 \cup \{(F_1,F_2,F_3,\boldsymbol{a},\boldsymbol{b},\alpha,\beta,\zeta,k)\},$$
sinon appeler T la sortie (dont les éléments sont des octuplets) et faire
$$S_1 \longleftarrow S_1 \cup (T \times \{k-1\}),$$
puis retourner à l'étape 2.

3. Pour chaque $(F_1,F_2,F_3,\boldsymbol{a},\boldsymbol{b},\alpha,\beta,\zeta,k) \in S_2$:
   (a) Faire
$$\begin{aligned} \widetilde{F}_1(X_1,\dots,X_k) &\longleftarrow F_1(X_1,\dots,X_k,\zeta),\\ \widetilde{F}_2(X_1,\dots,X_k) &\longleftarrow F_2(X_1,\dots,X_k,\zeta),\\ \widetilde{F}_3(X_1,\dots,X_k) &\longleftarrow F_3(X_1,\dots,X_k,\zeta),\\ \widetilde{D}(X_1,\dots,X_k) &\longleftarrow \operatorname{pgcd}(J\widetilde{F}_1, J\widetilde{F}_2, J\widetilde{F}_3). \end{aligned}$$
   (b) Déterminer un polynôme $D \in \mathbb{Z}[X_1,\dots,X_k,Z]$ pour lequel on a $D(X_1,\dots,X_k,\zeta) = \widetilde{D}(X_1,\dots,X_k)$.

(c) Si D n'est pas constant, faire

$$p(X,Y,Z) \longleftarrow JD(X^{\boldsymbol{a}}Y^{\boldsymbol{b}},Z)$$
$$S \longleftarrow S \cup \{(p,\alpha,\beta,\zeta)\}.$$

4. Renvoyer S et terminer.

Démonstration du théorème 2.3.8 : Montrons que la procédure E1 satisfait bien les conditions de ce théorème. Justifions tout d'abord pourquoi cet algorithme termine. Seule l'étape 2 est récursive et nécessite donc des explications. Lors de la première exécution de l'étape 2, l'ensemble S_1 ne possède qu'un élément dont la dernière coordonnée est n. Remarquons ensuite que l'application de l'algorithme E1a à un élément de S_1 ajoute au plus $O_{N,h}(1)$ éléments à S_1 dont la dernière coordonnée a diminué d'une unité. Ainsi, l'étape 2 de l'algorithme E1 n'est répétée qu'un nombre fini de fois et cet algorithme termine.

Montrons maintenant la correction de cet algorithme. On va d'abord montrer par récurrence, qu'après chaque application de l'étape 2, on a égalité entre

$$V(f_1,f_2,f_3) \cap (\mathcal{H}_1 \backslash \mathcal{H}_0)$$

et

$$\bigcup_{(F_1,F_2,F_3,\boldsymbol{a},\boldsymbol{b},\alpha,\beta,\zeta,k)\in S_1\cup S_2} V(F_1(X^{\boldsymbol{a}}Y^{\boldsymbol{b}},\zeta),F_2(X^{\boldsymbol{a}}Y^{\boldsymbol{b}},\zeta),F_3(X^{\boldsymbol{a}}Y^{\boldsymbol{b}},\zeta),X^{\alpha}Y^{\beta}-\zeta) \cap (\mathcal{H}_1 \backslash \mathcal{H}_0).$$

Avant la première application de l'étape 2, l'ensemble S_2 est vide et l'ensemble S_1 ne possède que l'élément $(F_1,F_2,F_3,\boldsymbol{a},\boldsymbol{b},0,0,1,n)$ qui vérifie :

$$F_1(X^{\boldsymbol{a}}Y^{\boldsymbol{b}},1) = f_1(X,Y)$$
$$F_2(X^{\boldsymbol{a}}Y^{\boldsymbol{b}},1) = f_2(X,Y)$$
$$F_3(X^{\boldsymbol{a}}Y^{\boldsymbol{b}},1) = f_3(X,Y).$$

L'égalité est donc vraie avant la première exécution de l'étape 2. Supposons maintenant que l'égalité est vérifiée et appliquons une nouvelle fois l'étape 2. Choisissons $(F_1,F_2,F_3,\boldsymbol{a},\boldsymbol{b},\alpha,\beta,\zeta,k) \in S_1$. Si $k=1$, on ajoute cet élément à S_2 et l'égalité est encore vérifiée puisque l'ensemble $S_1 \cup S_2$ reste inchangé. Supposons maintenant que $k>1$ et appliquons l'algorithme E2c à $(F_1,F_2,F_3,\boldsymbol{a},\boldsymbol{b},\alpha,\beta,\zeta)$. S'il renvoie « RIEN », on ajoute l'élément $(F_1,F_2,F_3,\boldsymbol{a},\boldsymbol{b},\alpha,\beta,\zeta,k)$ à S_2 et comme l'ensemble $S_1 \cup S_2$ est encore inchangé, le même argument s'applique. Il nous reste à considérer le cas où l'algorithme E2c renvoie un ensemble T. Alors d'après la proposition 2.3.7, on a égalité entre

$$V\left(F_1(X^{\boldsymbol{a}}Y^{\boldsymbol{b}},\zeta),F_2(X^{\boldsymbol{a}}Y^{\boldsymbol{b}},\zeta),F_3(X^{\boldsymbol{a}}Y^{\boldsymbol{b}},\zeta),X^{\alpha}Y^{\beta}-\zeta\right) \cap (\mathcal{H}_1 \backslash \mathcal{H}_0)$$

et la réunion pour $(F_1', F_2', F_3', \boldsymbol{a}', \boldsymbol{b}', \alpha', \beta', \zeta') \in T$ de

$$V\left(F_1'(X^{\boldsymbol{a}'}Y^{\boldsymbol{b}'}, \zeta'), F_2'(X^{\boldsymbol{a}'}Y^{\boldsymbol{b}'}, \zeta'), F_3'(X^{\boldsymbol{a}'}Y^{\boldsymbol{b}'}, \zeta'), X^{\alpha'}Y^{\beta'} - \zeta'\right) \cap (\mathcal{H}_1 \backslash \mathcal{H}_0).$$

Ainsi, on a bien encore égalité entre

$$V(f_1, f_2, f_3) \cap (\mathcal{H}_1 \backslash \mathcal{H}_0)$$

et

$$\bigcup_{(F_1,F_2,F_3,\boldsymbol{a},\boldsymbol{b},\alpha,\beta,\zeta,k)\in S_1\cup S_2} V(F_1(X^{\boldsymbol{a}}Y^{\boldsymbol{b}}, \zeta), F_2(X^{\boldsymbol{a}}Y^{\boldsymbol{b}}, \zeta), F_3(X^{\boldsymbol{a}}Y^{\boldsymbol{b}}, \zeta), X^{\alpha}Y^{\beta} - \zeta) \cap (\mathcal{H}_1 \backslash \mathcal{H}_0).$$

Lorsqu'on passe à l'étape 3, puisque $S_1 = \emptyset$, on a égalité entre

$$V(f_1, f_2, f_3) \cap (\mathcal{H}_1 \backslash \mathcal{H}_0)$$

et

$$\bigcup_{(F_1,F_2,F_3,\boldsymbol{a},\boldsymbol{b},\alpha,\beta,\zeta,k)\in S_2} V(F_1(X^{\boldsymbol{a}}Y^{\boldsymbol{b}}, \zeta), F_2(X^{\boldsymbol{a}}Y^{\boldsymbol{b}}, \zeta), F_3(X^{\boldsymbol{a}}Y^{\boldsymbol{b}}, \zeta), X^{\alpha}Y^{\beta} - \zeta) \cap (\mathcal{H}_1 \backslash \mathcal{H}_0).$$

Soient $(F_1, F_2, F_3, \boldsymbol{a}, \boldsymbol{b}, \alpha, \beta, \zeta, k) \in S_2$ et $(\sigma, \tau) \in (\overline{\mathbb{Q}}^*)^2 \backslash \mu_\infty^2$ tel que $\sigma^{\boldsymbol{a}}\tau^{\boldsymbol{b}} \in V(F_1, F_2, F_3)$ et tels que $\sigma^\alpha\tau^\beta = \zeta$. Alors d'après la proposition 2.3.7, le point $\sigma^{\boldsymbol{a}}\tau^{\boldsymbol{b}}$ est dans une composante de $V(F_1, F_2, F_3)$ de codimension 1 et donc annule le pgcd $\widetilde{D}$ de $J\widetilde{F_1}, J\widetilde{F_2}$ et $J\widetilde{F_3}$ calculé à l'étape 3a. Ainsi, on a $(\sigma, \tau) \in V(p(X, Y, \zeta), X^\alpha Y^\beta - \zeta)$, où $p(X, Y, Z) = JD(X^{\boldsymbol{a}}Y^{\boldsymbol{b}}, Z)$ est le polynôme calculé à l'étape 3c. Ainsi, on a finalement

$$V(f_1, f_2, f_3) \cap (\mathcal{H}_1 \backslash \mathcal{H}_0) \subseteq \bigcup_{(p,\alpha,\beta,\zeta)\in S} V\left(p(X, Y, \zeta), X^\alpha Y^\beta - \zeta\right)$$

ce qui prouve la correction de l'algorithme.

Il nous reste maintenant à déterminer son temps d'exécution. On peut d'abord remarquer que tout au long de l'algorithme, on a $\#S_1 = O_{N,h}(1)$. En effet, on a lors de la première exécution de l'étape 2, $\#S_1 = 1$ puis, d'après la proposition 2.3.7, chaque application de l'algorithme E1a à l'étape 2 ajoute à S_1 au plus $O_{N,h}(1)$ éléments. Ainsi, l'étape 2 est exécutée au plus $O_{N,h}(1)$ fois. Chacune de ces exécutions requiert d'appliquer l'algorithme E1a à un octuplet de la forme $(F_1, F_2, F_3, \boldsymbol{a}, \boldsymbol{b}, \alpha, \beta, \zeta, k)$ où F_1, F_2 et F_3 sont de degré et de hauteur $O_{N,h}(1)$, où $||\boldsymbol{a}||, ||\boldsymbol{b}||, |\alpha|, |\beta| = O_{N,h}(d)$ et où $\mathrm{ord}(\zeta) = O_{N,h}(1)$. Cette exécution de l'algorithme E2c nécessite donc $O_{N,h}(\mathsf{I}(\log d) \log\log d)$ opérations binaires. Ainsi, avant d'arriver à l'étape 3, on a effectué au plus

$$O_{N,h}(\mathsf{I}(\log d) \log\log d)$$

opérations binaires.

Avant de débuter l'étape 3, on a $\#S_2 = O_{N,h}(1)$ qui est donc une borne pour le nombre de répétitions de cette étape. Fixons maintenant $(F_1, F_2, F_3, \boldsymbol{a}, \boldsymbol{b}, \alpha, \beta, \zeta, k)$ dans S_2. Comme les polynômes F_1, F_2 et F_3 sont de degré et de hauteur majorés par $O_{N,h}(1)$, l'étape 3a nécessite $O_{N,h}(1)$ opérations binaires. Pour obtenir le polynôme D à l'étape 3b, il suffit de remplacer tous les ζ qui apparaissent dans l'expression de $\widetilde{D}$ par des Z. Ceci nécessite $O_{N,h}(1)$ opérations binaires puisque $\widetilde{D}$ est de degré et de hauteur majorés par $O_{N,h}(1)$. L'évaluation à l'étape 3c nécessite elle $O_{N,h}(\log d)$ opérations binaires car $||\boldsymbol{a}||, ||\boldsymbol{b}|| = O_{N,h}(d)$. Finalement, l'étape 3 nécessite en tout

$$O_{N,h}(\log d)$$

opérations binaires.

Au total, l'exécution de cet algorithme nécessite bien

$$O_{N,h}(\mathsf{I}(\log d) \log\log d)$$

opérations binaires. □

2.3.5. Algorithme E. — L'algorithme E correspondant au théorème 2.3.1 est maintenant clair : il suffit de chercher successivement les points dont l'une des coordonnée est nulle, puis les points de $V(f_1, f_2, f_3) \cap \mathcal{H}_0$, de $V(f_1, f_2, f_3) \cap (\mathcal{H}_1 \backslash \mathcal{H}_0)$ et enfin de $V(f_1, f_2, f_3) \cap (\mathcal{H}_2 \backslash \mathcal{H}_1)$. Tout ceci peut-être résumé dans la procédure suivante :

Algorithme E

Entrée : Trois polynômes $f_1, f_2, f_3 \in \mathbb{Z}[X, Y]$ donnés par leur représentation lacunaire :

$$f_1(X,Y) = \sum_{i=1}^{n} f_{1,i} X^{a_i} Y^{b_i}, \quad f_2(X,Y) = \sum_{i=1}^{n} f_{2,i} X^{a_i} Y^{b_i}, \quad f_3(X,Y) = \sum_{i=1}^{n} f_{3,i} X^{a_i} Y^{b_i}.$$

On autorise ici certains coefficients de f_1, f_2 et f_3 à être nuls de façon à ce qu'ils aient le même support. Si f_1, f_2 et f_3 ont chacun au plus N termes non nuls, on a $n \leq 3N$.

Sortie : Un ensemble fini de polynômes $\{p_i\}_{i \in I}$ et un ensemble fini de triplets de polynômes $\{(q_{j,1}, q_{j,2}, q_{j,3})\}_{j \in J}$ donnés par leur représentation lacunaire tels que

$$V(f_1, f_2, f_3) = \bigcup_{i \in I} V(p_i) \cup \bigcup_{j \in J} V(q_{j,1}, q_{j,2}) \backslash V(q_{j,3}).$$

1. Si chaque monôme de f_1, f_2 et f_3 est divisible par X et par Y, poser $T_2 \longleftarrow \{(X, Y, 1)\}$, sinon faire $T_2 \longleftarrow \emptyset$.

2. Appliquer l'algorithme A au triplet $(f_1(X,0), f_2(X,0), f_3(X,0))$ puis au triplet $(f_1(0,Y), f_2(0,Y), f_3(0,Y))$ et appeler S_1 et S_2 les ensembles de couples d'entiers renvoyés dans chacun des cas.
   Faire
   $$T_2 \longleftarrow T_2 \cup \left\{\left(\Phi_m\left(X^e\right), Y, 1\right)\right\}_{(m,e)\in S_1} \cup \left\{\left(X, \Phi_m\left(Y^e\right), 1\right)\right\}_{(m,e)\in S_2}.$$

3. Appliquer l'algorithme D au triplet $(f_1(X,0), f_2(X,0), f_3(X,0))$ puis au triplet $(f_1(0,Y), f_2(0,Y), f_3(0,Y))$ et appeler $q_1(X)$ et $q_2(Y)$ les polynômes renvoyés dans chacun des cas.
   Faire
   $$T_2 \longleftarrow T_2 \cup \left\{\left(q_1(X), Y, 1\right), \left(X, q_2(Y), 1\right)\right\}.$$

4. Appliquer l'algorithme E0, au triplet (f_1, f_2, f_3). Appeler T_1 l'ensemble de polynômes renvoyé par cet algorithme et S l'ensemble de paires de polynômes renvoyé par cet algorithme.
   Faire
   $$T_2 \longleftarrow T_2 \cup \left\{(p_1, p_2, 1)\right\}_{(p_1,p_2)\in S}.$$

5. Appliquer l'algorithme E1, au triplet (f_1, f_2, f_3). Appeler S l'ensemble de quadruplets renvoyé par cet algorithme.
   Faire
   $$T_2 \longleftarrow T_2 \cup \left\{\left(p(X,Y,\zeta), X^\alpha Y^\beta - \zeta, 1\right)\right\}_{(p,\alpha,\beta,\zeta)\in S}.$$

6. Appliquer l'algorithme E2, au triplet (f_1, f_2, f_3). Appeler p le polynôme renvoyé par cet algorithme et S l'ensemble de triplets de polynômes renvoyés par cet algorithme.
   Faire
   $$T_1 \longleftarrow T_1 \cup \{p\}$$
   $$T_2 \longleftarrow T_2 \cup S.$$

7. Renvoyer (T_1, T_2) et terminer.

Démonstration du théorème 2.3.1 : Montrons que l'algorithme E satisfait bien les conditions de ce théorème. La première étape consiste uniquement à tester si le point $(0,0)$ appartient à $V(f_1, f_2, f_3)$. La seconde étape permet d'obtenir une représentation des points de $V(f_1, f_2, f_3)$ dont l'une des coordonnée est nulle et l'autre est une racine de l'unité. La troisième étape permet quant à elle, de déterminer une représentation des points de $V(f_1, f_2, f_3)$ dont l'une des coordonnée

est nulle et l'autre n'est pas une racine de l'unité. Après ces trois premières étapes, on a donc une représentation des points de

$$V(f_1, f_2, f_3) \cap V(XY).$$

Les étapes $4, 5$ et 6 permettent d'obtenir une représentation des points de

$$V(f_1, f_2, f_3) \cap \mathbb{G}_m^2$$

et on obtient bien finalement tous les points de $V(f_1, f_2, f_3)$. Ceci prouve la correction de l'algorithme E.

Il nous reste maintenant à vérifier que le temps d'exécution de cet algorithme est conforme à l'énoncé du théorème 2.3.1.

L'étape 1 est une simple vérification et nécessite

$$O_{N,h}(\log d)$$

opérations binaires.

L'étape 2 consiste à appliquer deux fois l'algorithme A à des polynômes de degré au plus d, de hauteur au plus h et ayant au plus n termes non nuls, ceci nécessite donc

$$O_{N,h}(\mathsf{I}(\log d) \log\log d)$$

opérations binaires.

L'étape 3 nécessite d'appliquer deux fois l'algorithme D aux mêmes polynômes et ceci nécessite

$$O_{N,h}(\log d)$$

opérations binaires.

A l'étape 4, on applique l'algorithme E0 au triplet de polynômes (f_1, f_2, f_3). Comme ces polynômes sont de degré au plus d, de hauteur au plus h et ont au plus N termes non nuls, ceci nécessite au total

$$O_{N,h}(\mathsf{I}(\log d) \log\log d)$$

opérations binaires.

A l'étape 5, on applique l'algorithme E1 au triplet de polynômes (f_1, f_2, f_3). Comme ces polynômes sont de degré au plus d, de hauteur au plus h et ont au plus N termes non nuls, ceci nécessite au total

$$O_{N,h}(\mathsf{I}(\log d) \log\log d)$$

opérations binaires.

A l'étape 6, on applique enfin l'algorithme E2 au triplet de polynômes (f_1, f_2, f_3). Comme ces polynômes sont de degré au plus d, de hauteur au plus h et ont au

plus N termes non nuls, ceci nécessite au total

$$O_{N,h}(\mathsf{I}(\log d)\log\log d)$$

opérations binaires. Finalement, on ne dépasse pas la complexité annoncée et ceci achève la démonstration du théorème 2.3.1. □

ANNEXE A

EXEMPLES

A.1. Un exemple pour l'algorithme A

On détaille ici les calculs qu'effectue l'algorithme A pour déterminer les facteurs cyclotomiques du polynôme

$$f(X) = X^{2\,011} + 3X^{1\,486} + X^{1\,300} + 2X^{961} + X^{650} + 1.$$

L'algorithme A consiste à considérer toutes les partitions possibles du support de f et tous les choix possibles de racines de l'unité dont l'ordre vérifie les conditions du théorème 1.2.3. Comme le polynôme f possède 6 termes non nuls, il y a dans ce cas 41 partitions à considérer et, pour chaque partition, on a au plus 6 racines de l'unité à considérer. On ne va détailler les calculs que pour une seule partition et pour un seul choix de racine de l'unité, mais l'algorithme A doit lui, considérer tous les choix possibles.

Etape 1 : On pose $S_f \leftarrow \emptyset$.
Etape 2 : On fixe la partition $\{\{1,2,4\},\{3,5,6\}\}$ du support de f, ce qui correspond à écrire $f(X) = f_1(X) + f_2(X)$, où

$$f_1(X) = X^{2\,011} + 3X^{1\,486} + 2X^{961},$$
$$f_2(X) = X^{1\,300} + X^{650} + 1.$$

Etape 2a : On écrit :

$$f_1(X) = X^{961} g_1(X^{e_1}) \quad \text{avec} \quad g_1(X) = X^2 + 3X + 2 \quad \text{et} \quad e_1 = 525,$$
$$f_2(X) = g_2(X^{e_2}) \quad \text{avec} \quad g_2(X) = X^2 + X + 1 \quad \text{et} \quad e_2 = 650.$$

Etape 2b : On calcule

$$e \leftarrow \text{pgcd}(e_1, e_2) = 25, \quad e'_1 \leftarrow e_1/e = 525/25 = 21, \quad e'_2 \leftarrow e_2/e = 650/25 = 26.$$

Etape 2c : On fait le choix $m' = 6$.
Etape 2ci : On calcule :

$$m_1 \leftarrow \frac{m'}{\mathrm{pgcd}(m', e_1')} = \frac{6}{3} = 2 \quad \text{et} \quad m_2 \leftarrow \frac{m'}{\mathrm{pgcd}(m', e_2')} = \frac{6}{2} = 3.$$

Etape 2cii : On teste si G_1 s'annule en une racine de l'unité d'ordre $m_1 = 2$, i.e. si $G_1(-1) = 0$ et on teste si G_2 s'annule en une racine de l'unité d'ordre $m_2 = 3$. Comme c'est le cas, on ajoute à S_f la paire $(m, e) = (6, 25)$.
Etape 3 : On renvoie S_f et on termine.

On rappelle que la paire $(6, 25)$ renvoyée en sortie signifie que le polynôme f est divisible par $\Phi_6(X^{25})$ et donc possède les facteurs cyclotomiques $\Phi_6(X), \Phi_{30}(X)$ et $\Phi_{150}(X)$ d'après le lemme 1.2.1. On peut consulter la section A.3 où l'on détaille comment obtenir la liste complète des facteurs cyclotomiques à partir de la sortie de l'algorithme A.

L'application complète de l'algorithme A à ce polynôme révèle que celui-ci ne possède pas d'autre facteur cyclotomique. Ainsi, la paire $(6, 25)$ suffit dans ce cas à représenter tous ses facteurs cyclotomiques.

A.2. Temps d'exécution de l'algorithme A

Dans le tableau suivant, on donne le temps d'exécution de l'algorithme A en fonction du nombre de termes N du polynôme en entrée. Les polynômes utilisés pour obtenir ces résultats ont été choisis au hasard. Ils sont de hauteur inférieure à 1000 et ont des degrés de l'ordre de 10^{100}. Ces résultats ont été obtenus grâce à une implémentation de l'algorithme A en Maple sur un eMachines E3034, Processeur AMD Athlon 64 3500+, 2.2 GHz, 1024 Mo.

N	Partitions	Conway-Jones	Boucles	Temps
2	1	2	2	0.003 s
3	1	4	4	0.005 s
4	4	4	10	0.01 s
5	11	6	46	0.03 s
6	41	8	138	0.1 s
7	162	10	766	0.5 s
8	715	12	3 134	2.5 s
9	3 425	12	19 020	15 s
10	17 722	14	100 316	1 min 40 s
11	98 253	18	652 692	19 min 20 s
12	580 317	20	16 364 456	6 h 40 min

1ère colonne : nombre de termes non nuls du polynôme.
2ème colonne : nombre de partitions du support du polynôme à considérer pour l'algorithme A.
3ème colonne : nombre d'entiers m sans facteur carré vérifiant $\Psi(m) \leq N$ (voir le théorème 1.2.3 de Conway et Jones).
4ème colonne : nombre total de boucles effectuées par l'algorithme A.
5ème colonne : temps d'exécution de l'algorithme A.

On constate que le temps d'exécution de l'algorithme A dépend beaucoup du nombre de termes non nuls du polynôme donné en entrée. Si l'algorithme est très efficace jusqu'à $N = 8$ ou 9, les calculs deviennent vite impraticables au-delà. Cependant, les calculs menés par l'algorithme A peuvent très facilement être menés en parallèle. Il suffit par exemple de répartir l'ensemble des partitions du polynôme à considérer entre les différents opérateurs. Ceci permettrait d'appliquer l'algorithme A à des polynômes ayant plus de termes non nuls. Notons que cette dernière remarque s'applique également aux algorithmes B et C.

A.3. Liste complète des facteurs cyclotomiques

On rappelle qu'en général, on ne peut pas déterminer la liste complète des facteurs cyclotomiques d'un polynôme en temps polynomial en le logarithme de son degré comme on l'a remarqué au début de la section 1.2. Détaillons malgré tout, sur un exemple, comment on peut obtenir cette liste complète de facteurs cyclotomiques à partir de la sortie de l'algorithme A en utilisant le lemme 1.2.1.

Sur la page personnelle de Michael Filaseta à l'adresse `http://maple.math.sc.edu/research/Cyclotomic.html`, on peut utiliser un applet, basé sur l'algorithme de [**FS04**], qui permet de déterminer si un polynôme f possède un facteur cyclotomique. Dans le cas où le polynôme possède effectivement un facteur cyclotomique, cet algorithme renvoie un entier m pour lequel $\Phi_m(X)|f(X)$. Dans cet applet, le polynôme

$$f(X) = 2X^{10^{100}} - X^{128\,000} + 3X^{64\,000} - 1$$

est donné en exemple. L'applet détermine que ce polynôme est divisible par $\Phi_{3\,072}(X)$.

Si on applique l'algorithme A au polynôme f, celui-ci renvoie la paire d'entiers $(6, 64\,000)$, ce qui signifie que le polynôme f est divisible par $\Phi_6(X^{64\,000})$. On peut alors retrouver tous ses facteurs cyclotomiques à l'aide du lemme 1.2.1. Pour utiliser les mêmes notations que dans ce lemme, on pose $m \leftarrow 6$ et $e \leftarrow 64\,000$.

Comme $64\,000 = 2^9 \cdot 5^3$, on a $e_1 = 2^9 = 512$, $e_2 = 5^3 = 125$ et $me_1 = 3\,072$. On obtient alors la factorisation :

$$\begin{aligned}\Phi_6\left(X^{64\,000}\right) &= \Phi_{3\,072}\left(X^{125}\right)\\ &= \prod_{d|125} \Phi_{3\,072d}(X)\\ &= \Phi_{3\,072}(X)\Phi_{15\,360}(X)\Phi_{76\,800}(X)\Phi_{384\,000}(X),\end{aligned}$$

et on récupère ainsi la liste complète des facteurs cyclotomiques du polynôme f.

A.4. Un exemple pour l'algorithme B

On détaille ici les calculs qu'effectue l'algorithme B pour déterminer une représentation des points de torsion appartenant à l'hypersurface définie par

$$f(X,Y) = X^{398}Y^{240} + X^{262} - X^{131}Y^{43} + X^5Y^{369} + Y^{86}.$$

L'algorithme B consiste à considérer toutes les partitions possibles du support de f et tous les choix possibles de points de torsion qui vérifient les conditions du corollaire 1.3.1. Comme le polynôme f est la somme de 5 monômes, il y a dans ce cas 11 partitions à considérer et, pour chaque partition, on a au plus 124 points de torsion à considérer d'après la corollaire 1.3.1. On ne va détailler les calculs que pour une seule partition et pour un seul choix de point de torsion, mais l'algorithme B doit lui, considérer tous les choix possibles.

Etape 1 : On pose $S_f \leftarrow \emptyset$.
Etape 2 : On fixe la partition $\{\{1,3\},\{2,4,5\}\}$ du support de f, ce qui correspond à écrire $f(X,Y) = f_1(X,Y) + f_2(X,Y)$, où

$$\begin{aligned}f_1(X,Y) &= X^{398}Y^{240} - X^{131}Y^{43},\\ f_2(X,Y) &= X^{262} + X^5Y^{369} + Y^{86}.\end{aligned}$$

Etape 2a : On écrit :

$$\begin{aligned}f_1(X,Y) &= X^{131}Y^{43}g_1(X^{267}Y^{197}) && \text{avec} && g_1(X) = -1 + X,\\ f_2(X,Y) &= X^{262}g_2(X^{-257}Y^{369}, X^{-131}Y^{43}) && \text{avec} && g_2(X) = 1 + X + Y^2.\end{aligned}$$

Etape 2b : On calcule une base du sous-groupe de $\mathbb{Z}^2$ engendré par $\boldsymbol{\lambda}_{1,1} = \begin{pmatrix} 267 \\ 197 \end{pmatrix}$, $\boldsymbol{\lambda}_{2,1} = \begin{pmatrix} -257 \\ 369 \end{pmatrix}$ et $\boldsymbol{\lambda}_{2,2} = \begin{pmatrix} -131 \\ 43 \end{pmatrix}$. L'algorithme détermine en fait la seule base qui soit triangulaire supérieure grâce à un calcul de forme normale

d'Hermite mais une base quelconque nous suffirait. On obtient dans ce cas la base $\{\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2\}$, où :

$$\boldsymbol{\lambda}_1 = \begin{pmatrix} 1 \\ 29\,887 \end{pmatrix} \quad \text{et} \quad \boldsymbol{\lambda}_2 = \begin{pmatrix} 0 \\ 37\,288 \end{pmatrix}.$$

On doit ensuite calculer les coordonnées de $\boldsymbol{\lambda}_{1,1}, \boldsymbol{\lambda}_{2,1}$ et $\boldsymbol{\lambda}_{2,2}$ dans la base $\{\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2\}$:

$$\begin{aligned} \boldsymbol{\lambda}_{1,1} &= 267\boldsymbol{\lambda}_1 - 214\boldsymbol{\lambda}_2, \\ \boldsymbol{\lambda}_{2,1} &= -257\boldsymbol{\lambda}_1 + 206\boldsymbol{\lambda}_2, \\ \boldsymbol{\lambda}_{2,2} &= -131\boldsymbol{\lambda}_1 + 105\boldsymbol{\lambda}_2. \end{aligned}$$

Etape 2c : On fait le choix $m' = 3$ et on choisit le point d'ordre 3 : $(\zeta_3, 1)$, où ζ_3 désigne une racine 3-ième de l'unité.
Etape 2ci : On calcule :

$$\begin{aligned} \omega_{1,1} &\leftarrow \zeta_3^{267} 1^{-214} = 1, \\ \omega_{2,1} &\leftarrow \zeta_3^{-257} 1^{206} = \zeta_3, \\ \omega_{2,2} &\leftarrow \zeta_3^{-131} 1^{105} = \zeta_3^2. \end{aligned}$$

Etape 2cii : Comme

$$g_1(\omega_{1,1}) = g_2(\omega_{2,1}, \omega_{2,2}) = 0,$$

on ajoute à S_f la paire $\left(\begin{pmatrix} 1 & 29\,887 \\ 0 & 37\,288 \end{pmatrix}, (\zeta_3, 1) \right)$.
Etape 3 : On renvoie S_f et on termine.

On rappelle que le fait que la paire $\left(\begin{pmatrix} 1 & 29\,887 \\ 0 & 37\,288 \end{pmatrix}, (\zeta_3, 1) \right)$ appartienne à S_f signifie que les points (σ, τ) vérifiant

$$\begin{cases} \sigma\tau^{29\,887} = \zeta_3 \\ \tau^{37\,288} = 1 \end{cases}$$

appartiennent à l'hypersurface définie par f.

Pour cette hypersurface, l'algorithme B renvoie les quatre paires :

$$\begin{aligned} G_1 &= \left(\begin{pmatrix} 1 & 29\,887 \\ 0 & 37\,288 \end{pmatrix}, (\zeta_3, 1) \right), & G_2 &= \left(\begin{pmatrix} 1 & 29\,887 \\ 0 & 37\,288 \end{pmatrix}, (\zeta_3^2, 1) \right), \\ G_3 &= \left(\begin{pmatrix} 131 & -43 \end{pmatrix}, (\zeta_6) \right), & G_4 &= \left(\begin{pmatrix} 131 & -43 \end{pmatrix}, (\zeta_6^5) \right). \end{aligned}$$

Comme on pouvait s'y attendre, si l'algorithme B renvoie une paire (L, ω), il renvoie aussi toutes les paires (L, ω'), où ω' est conjugué de ω, puisque l'hypersurface est définie par une équation rationnelle. Ainsi, les paires G_1 et G_2, G_3 et G_4 sont « conjuguées » deux à deux.

Les paires G_1 et G_2 représentent des translatés de sous-groupes de $\mathbb{G}_m^2$ de dimension 0 et donc des points isolés de l'hypersurface définie par f. Ces paires représentent chacune 37 288 points de torsion. Comme ces deux sous-variétés de torsion sont des translatés du même sous-groupe algébrique de $\mathbb{G}_m^2$ par des points de torsion différents, elles ne possèdent aucun point en commun. Ainsi, l'hypersurface définie par f contient exactement 74 576 points isolés de torsion.

Les paires G_5 et G_6 quant à elles, représentent des translatés de sous-groupes de dimension 1, c'est à dire des facteurs cyclotomiques généralisés de f. En effet, le polynôme f admet le facteur

$$J\Phi_6\left(X^{131}Y^{-43}\right) = X^{262} - Y^{43}X^{131} + Y^{86}.$$

On se rend compte sur ce simple exemple que notre représentation est beaucoup plus compacte que l'énumération de toutes les composantes irréductibles de torsion et qu'il est très facile, à partir de cette représentation, de retrouver toutes les composantes irréductibles de torsion. C'est d'ailleurs pour faciliter cette recherche des composantes irréductibles de torsion que l'on a préféré renvoyer des matrices triangulaires.

Ce n'est pas le cas dans l'exemple que nous venons de détailler, mais il peut y avoir des répétitions dans les résultats renvoyés par cet algorithme, comme c'était déjà le cas pour l'algorithme A. Dans ce cas, cela signifie que l'algorithme B peut renvoyer deux paires distinctes (L, ω) et (L', ω') définissant deux sous-variétés de torsion différentes mais ayant des points de torsion en commun.

A.5. Un exemple pour l'algorithme D2

Soient m et n deux entiers. On détaille ici les calculs qu'effectue l'algorithme D2 appliqué à

$$\begin{aligned} F_1(X_1, X_2, X_3, X_4) &= 2X_1X_2 - 5X_1 + 2X_3 + 6X_2 - 5X_4 - 15, \\ F_2(X_1, X_2, X_3, X_4) &= 3X_1X_2 - 2X_1 + 3X_3 + 9X_2 - 2X_4 - 6, \\ \boldsymbol{a} &= (n, m, m+1, 1). \end{aligned}$$

Cet exemple sera utilisé dans l'exemple pour l'algorithme D dans la prochaine section.

Etape 1 : La difficulté ici est qu'on ne connaît pas de borne effective pour $B_1(F_1, F_2)$. Cependant, dans ce cas, on peut trouver « à la main » un vecteur de petite norme orthogonal à $\boldsymbol{a}$ comme par exemple le vecteur $\boldsymbol{b} = (0, 1, -1, 1)$.

Etape 2 : On détermine une base du sous-groupe de $\mathbb{Z}^4$ formé des vecteurs orthogonaux à $\boldsymbol{b}$. On a écrit ces vecteurs en colonne dans la matrice : $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$.

Etape 3 : On détermine les coordonnées $\boldsymbol{a}'$ de $\boldsymbol{a}$ dans la base obtenue à l'étape précédente. On obtient le vecteur $\boldsymbol{a}' = (n, m, 1)$.

Etape 4 : On pose

$$\begin{aligned} F_1'(X_1, X_2, X_3) &= F_1(X_1, X_2, X_2X_3, X_3) \\ &= 2X_1X_2 - 5X_1 + 2X_2X_3 + 6X_2 - 5X_3 - 15, \\ F_2'(X_1, X_2, X_3) &= F_2(X_1, X_2, X_2X_3, X_3) \\ &= 3X_1X_2 - 2X_1 + 3X_2X_3 + 9X_2 - 2X_3 - 6. \end{aligned}$$

Etape 5 : On renvoie $(F_1', F_2', \boldsymbol{a}')$ et on termine.

En sortie, on vérifie que l'on a bien :

$$\begin{aligned} F_1(X^n, X^m, X^{m+1}, X) &= F_1'(X^n, X^m, X), \\ F_2(X^n, X^m, X^{m+1}, X) &= F_2'(X^n, X^m, X), \end{aligned}$$

conformément à la proposition 2.1.5.

A.6. Un exemple pour l'algorithme D

Soient m et n deux entiers premiers entre eux tels que $m, n \gg 1$. On détaille ici les calculs effectués par l'algorithme D appliqué aux polynômes f_1 et f_2, où

$$\begin{aligned} f_1(X) &= 2X^{n+m} - 5X^n + 2X^{m+1} + 6X^m - 5X - 15, \\ f_2(X) &= 3X^{n+m} - 2X^n + 3X^{m+1} + 9X^m - 2X - 6. \end{aligned}$$

Etape 1 : On linéarise les polynômes f_1 et f_2 et on note F_1 et F_2 les polynômes obtenus. On appelle $\boldsymbol{a}$ le vecteur d'exposants correspondant et k le nombre de variables. On fait donc :

$$\begin{aligned} F_1(X_1, X_2, X_3, X_4, X_5) &\leftarrow 2X_1 - 5X_2 + 2X_3 + 6X_4 - 5X_5 - 15, \\ F_2(X_1, X_2, X_3, X_4, X_5) &\leftarrow 3X_1 - 2X_2 + 3X_3 + 9X_4 - 2X_5 - 6, \\ \boldsymbol{a} &\leftarrow (n + m, n, m + 1, m, 1), \\ k &\leftarrow 5. \end{aligned}$$

On a ainsi

$$F_1(X^{n+m}, X^n, X^{m+1}, X^m, X) = f_1(X),$$
$$F_2(X^{n+m}, X^n, X^{m+1}, X^m, X) = f_2(X).$$

Etape 2 : Comme $k \neq 1$, on applique l'algorithme D2 au triplet $(F_1, F_2, \boldsymbol{a})$ qui renvoie le triplet $(F_1', F_2', \boldsymbol{a}')$, où

$$F_1'(X_1, X_2, X_3, X_4) = 2X_1X_2 - 5X_1 + 2X_3 + 6X_2 - 5X_4 - 15,$$
$$F_2'(X_1, X_2, X_3, X_4) = 3X_1X_2 - 2X_1 + 3X_3 + 9X_2 - 2X_4 - 6,$$
$$\boldsymbol{a}' = (n, m, m+1, 1).$$

On fait alors :

$$(F_1, F_2, \boldsymbol{a}) \leftarrow (F_1', F_2', \boldsymbol{a}'),$$
$$k \leftarrow 4,$$

et on recommence l'étape 2.
Etape 2 : Comme $k \neq 1$, on applique l'algorithme D2 au triplet $(F_1, F_2, \boldsymbol{a})$ qui renvoie le triplet $(F_1', F_2', \boldsymbol{a}')$, où

$$F_1'(X_1, X_2, X_3) = 2X_1X_2 - 5X_1 + 2X_2X_3 + 6X_2 - 5X_3 - 15,$$
$$F_2'(X_1, X_2, X_3) = 3X_1X_2 - 2X_1 + 3X_2X_3 + 9X_2 - 2X_3 - 6,$$
$$\boldsymbol{a}' = (n, m, 1).$$

Cet exemple d'application de l'algorithme D2 est détaillé dans la section précédente. On fait alors :

$$(F_1, F_2, \boldsymbol{a}) \leftarrow (F_1', F_2', \boldsymbol{a}'),$$
$$k \leftarrow 3,$$

et on recommence l'étape 2.
Etape 2 : Comme $k \neq 1$, on applique l'algorithme D2 au triplet $(F_1, G_3, \boldsymbol{a}_3)$ et celui-ci renvoie « RIEN ». En effet, comme n et m sont premiers entre eux, un vecteur orthogonal à $\boldsymbol{a}_3 = (n, m, 1)$ est de norme au moins égale à $\min(n, m)$ et si on a choisi m et n suffisamment grands, ceci excède $B_1(F_1, G_3)$. Ainsi, le vecteur $\boldsymbol{a}$ ne possède pas d'orthogonal non nul de petite norme ce qui explique que l'algorithme D2 ne renvoie rien dans ce cas. On passe alors à l'étape 3.
Etape 3 : On calcule D, le pgcd de F_1 et F_2 dans $\mathbb{Z}[X_1, X_2, X_3]$ et on obtient

$$D(X_1, X_2, X_3) = X_1 + X_3 + 3.$$

On renvoie finalement le polynôme

$$p(X) = D(X^n, X^m, X) = X^n + X + 3.$$

D'après le théorème 2.1.1, on sait que le polynôme p renvoyé par l'algorithme D est tel que

1. $p|\operatorname{pgcd}(f_1, f_2)$
2. $\operatorname{pgcd}(f_1, f_2)/p$ est produit de facteurs cyclotomiques.

Or dans ce cas, les polynômes f_1 et f_2 n'ont pas de facteur cyclotomique comme on peut le voir par exemple en leur appliquant l'algorithme A. Ainsi, le polynôme $X^n + X + 3$ est exactement le pgcd de f_1 et f_2.

A.7. Un exemple pour l'algorithme E1a

On va appliquer l'algorithme E1a à l'octuplet $(F_1, F_2, F_3, \boldsymbol{a}, \boldsymbol{b}, 3n, 3n, 1)$, où

$$\begin{aligned} F_1(X_1, X_2, Z) &= 1 + X_1X_2 + 5X_2, \\ F_2(X_1, X_2, Z) &= 1 + 5X_2 + 25X_2^2, \\ F_1(X_1, X_2, Z) &= 1 + X_1X_2 + 25X_1X_2^3, \\ \boldsymbol{a}_2 &= (2, -1), \\ \boldsymbol{b}_2 &= (1, 0). \end{aligned}$$

Cet exemple sera utilisé dans l'exemple pour l'algorithme E1 dans la prochaine section.

Etape 1 : On fait $S \leftarrow \emptyset$.
Etape 2 : On cherche parmi tous les vecteurs $\boldsymbol{c}$ de $\mathbb{Z}^2\backslash\{0\}$ de norme majorée par $B_1(V(F_1, F_2, F_3))$ (cf. Conjecture 2.3.2), ceux pour lesquels les vecteurs

$$(\langle \boldsymbol{a}, \boldsymbol{c}\rangle, \langle \boldsymbol{b}, \boldsymbol{c}\rangle) \quad \text{et} \quad (3n, 3n)$$

sont colinéaires. On note C cet ensemble. On ne peut en réalité pas déterminer l'ensemble C puisqu'on ne connaît pas de borne explicite pour $B_1(V(F_1, F_2, F_3))$. Cependant, on peut déterminer certains de ces vecteurs « à la main » et dans ce cas, on obtient par exemple que le vecteur $\boldsymbol{c}_0 = (3, 3)$ appartient à l'ensemble C. Nous ne détaillons dans la suite que les calculs effectués par l'algorithme E1a pour ce vecteur particulier.
Etape 3 : On choisit le vecteur $\boldsymbol{c} = (3, 3)$.
Etape 3a : On détermine une base du sous-groupe de $\mathbb{Z}^3$ formé des vecteurs orthogonaux à $(3, 3, -1)$. On écrit les deux vecteurs de cette base en colonne dans une matrice que l'on appelle H. On a alors

$$H = \begin{pmatrix} 1 & 1 \\ -1 & 0 \\ 0 & 3 \end{pmatrix}.$$

Etape 3b : On détermine les coordonnées $\boldsymbol{a}'$ et $\boldsymbol{b}'$ des vecteurs $(2,-1,3)$ et $(1,0,3)$ dans cette base. On obtient

$$\begin{aligned}\boldsymbol{a}' &= (1,1),\\ \boldsymbol{b}' &= (0,1).\end{aligned}$$

Etape 3c : On détermine un générateur de $\mathbb{Z}(3n,3n)+\mathbb{Z}(3,3)$. On obtient le vecteur $(3,3)$ et on a

$$(\alpha',\beta') = (3,3) = 1\times(3,3) + 0\times(3n,3n),$$

c'est à dire $u=1$ et $v=0$. On détermine aussi les deux entiers t_1 et t_2 tels que

$$\begin{aligned}(\alpha,\beta) &= t_1(\alpha',\beta'),\\ (a'_k,b'_k) &= t_2(\alpha',\beta').\end{aligned}$$

On obtient $t_1=n$ et $t_2=2$.
Etape 3d : On a $m=3$ (l'entier en dernière position dans la matrice H). On doit donc effectuer cette étape pour chacune des 3 racines de l'unité 1, j et j^2, où $j=\exp(\frac{2i\pi}{3})$.
Etape 3d(i) : On choisit $\eta=1$. On pose alors

$$\begin{aligned}\zeta' &\leftarrow \zeta^0\eta^1 = 1,\\ F'_1(X_1,Z) &\leftarrow F_1(X_1Z,X_1^{-1},Z^n) = (1+Z)+5X_1^{-1},\\ F'_2(X_1,Z) &\leftarrow F_2(X_1Z,X_1^{-1},Z^n) = 1+5X_1^{-1}+25X_1^{-2},\\ F'_3(X_1,Z) &\leftarrow F_3(X_1Z,X_1^{-1},Z^n) = (1+Z)+25X_1^{-2}.\end{aligned}$$

Etape 3d(ii) : On ajoute à S l'octuplet

$$(F'_1,F'_2,F'_3,(1),(0),3,3,1).$$

Etape 3d(i) : On choisit $\eta=j$. On pose alors

$$\begin{aligned}\zeta' &\leftarrow \zeta^0\eta^1 = j,\\ F'_1(X_1,Z) &\leftarrow F_1(X_1Z,X_1^{-1},Z^n) = (1+Z)+5X_1^{-1},\\ F'_2(X_1,Z) &\leftarrow F_2(X_1Z,X_1^{-1},Z^n) = 1+5X_1^{-1}+25X_1^{-2},\\ F'_3(X_1,Z) &\leftarrow F_3(X_1Z,X_1^{-1},Z^n) = (1+Z)+25X_1^{-2}.\end{aligned}$$

Etape 3d(ii) : On ajoute à S l'octuplet

$$(F'_1,F'_2,F'_3,(1),(0),3,3,j).$$

Etape 3d(i) : On choisit $\eta = j^2$. On pose alors

$$\begin{aligned}\zeta' \leftarrow & \zeta^0\eta^1 = j^2,\\ F_1'(X_1, Z) \leftarrow & F_1(X_1Z, X_1^{-1}, Z^n) = (1+Z) + 5X_1^{-1},\\ F_2'(X_1, Z) \leftarrow & F_2(X_1Z, X_1^{-1}, Z^n) = 1 + 5X_1^{-1} + 25X_1^{-2},\\ F_3'(X_1, Z) \leftarrow & F_3(X_1Z, X_1^{-1}, Z^n) = (1+Z) + 25X_1^{-2}.\end{aligned}$$

Etape 3d(ii) : On ajoute à S l'octuplet

$$(F_1', F_2', F_3', (1), (0), 3, 3, j^2).$$

Etape 4 : On renvoie S et on termine.

A.8. Un exemple pour l'algorithme E1

Soit n un entier vérifiant $n \gg 1$. On détaille ici les calculs qu'effectue l'algorithme E1 pour déterminer une représentation des points de $\mathcal{H}_1 \backslash \mathcal{H}_0$ inclus dans la variété définie par les trois polynômes :

$$\begin{aligned}f_1(X, Y) =& X^{3n+1}Y^{3n} + X^2Y + 5,\\ f_2(X, Y) =& X^{3n+2}Y^{3n} + 5X + 25,\\ f_3(X, Y) =& X + X^2Y + 25Y.\end{aligned}$$

Etape 1 : On linéarise les trois polynômes, on définit les deux vecteurs d'exposants et les ensembles S_1, S_2 et S :

$$\begin{aligned}F_1(X_1, X_2, X_3, X_4, X_5, Z) \leftarrow & X_1 + X_3 + 5,\\ F_2(X_1, X_2, X_3, X_4, X_5, Z) \leftarrow & X_2 + 5X_4 + 25,\\ F_3(X_1, X_2, X_3, X_4, X_5, Z) \leftarrow & X_3 + X_4 + 25X_5,\\ \boldsymbol{a}_5 \leftarrow & (3n+1, 3n+2, 2, 1, 0),\\ \boldsymbol{b}_5 \leftarrow & (3n, 3n, 1, 0, 1),\\ S_1 \leftarrow & \{(F_1, F_2, F_3, \boldsymbol{a}_5, \boldsymbol{b}_5, 0, 0, 1, 5)\},\\ S_2 \leftarrow & \emptyset,\\ S \leftarrow & \emptyset.\end{aligned}$$

Etape 2 : Comme l'ensemble S_1 ne possède qu'un élément, on n'a pas d'autre choix que de choisir l'élément $(F_1, F_2, F_3, \boldsymbol{a}, \boldsymbol{b}, 0, 0, 1, 5)$ de S_1 et on le retire de S_1. Comme $k = 5 > 1$, on applique l'algorithme E1a à $(F_1, F_2, F_3, \boldsymbol{a}, \boldsymbol{b}, 0, 0, 1)$. Celui-ci

renvoie un ensemble possédant un seul élément $(F_1', F_2', F_3', \boldsymbol{a}', \boldsymbol{b}', 0, 0, 1)$ où :

$$\begin{aligned}
F_1'(X_1, X_2, X_3, X_4, Z) &= F_1(X_1X_2, X_1, X_3, X_2^{-1}, X_4, Z) \\
&= X_1X_2 + X_3 + 5, \\
F_2'(X_1, X_2, X_3, X_4, Z) &= F_2(X_1X_2, X_1, X_3, X_2^{-1}, X_4, Z) \\
&= X_1 + 5X_2^{-1} + 25, \\
F_3'(X_1, X_2, X_3, X_4, Z) &= F_3(X_1X_2, X_1, X_3, X_2^{-1}, X_4, Z) \\
&= X_2^{-1} + X_3 + 25X_4, \\
\boldsymbol{a} &= (3n + 2, -1, 2, 0), \\
\boldsymbol{b} &= (3n, 0, 1, 1).
\end{aligned}$$

Pour alléger les notations dans la suite, on pose

$$(F_1, F_2, F_3, \boldsymbol{a}, \boldsymbol{b}) := (F_1', F_2', F_3', \boldsymbol{a}', \boldsymbol{b}').$$

On ajoute à S_1 l'élément $(F_1, F_2, F_3, \boldsymbol{a}, \boldsymbol{b}, 0, 0, 1, 4)$ et on recommence l'étape 2.
Etape 2 : Comme l'ensemble S_1 ne possède qu'un élément, on n'a pas d'autre choix que de choisir l'élément $(F_1, F_2, F_3, \boldsymbol{a}, \boldsymbol{b}, 0, 0, 1, 4)$ de S_1 et on le retire de S_1.
Comme $k = 4 > 1$, on applique l'algorithme E1a à $(F_1', F_2', F_3', \boldsymbol{a}', \boldsymbol{b}', 0, 0, 1)$. Celui-ci renvoie un ensemble possédant un seul élément $(F_1', F_2', F_3', \boldsymbol{a}', \boldsymbol{b}', 0, 0, 1)$ où :

$$\begin{aligned}
F_1'(X_1, X_2, X_3, Z) &= F_1(X_1, X_3, X_2, X_2X_3^2, Z) \\
&= X_1X_3 + X_2 + 5, \\
F_2'(X_1, X_2, X_3, Z) &= F_2(X_1, X_3, X_2, X_2X_3^2, Z) \\
&= X_1 + 5X_3^{-1} + 25, \\
F_3'(X_1, X_2, X_3, Z) &= F_3(X_1, X_3, X_2, X_2X_3^2, Z) \\
&= X_3^{-1} + X_2 + 25X_2X_3^2, \\
\boldsymbol{a}' &= (3n + 2, 2, -1), \\
\boldsymbol{b}' &= (3n, 1, 0).
\end{aligned}$$

Pour alléger les notations dans la suite, on pose

$$(F_1, F_2, F_3, \boldsymbol{a}, \boldsymbol{b}) := (F_1', F_2', F_3', \boldsymbol{a}', \boldsymbol{b}').$$

On ajoute à S_1 l'élément $(F_1, F_2, F_3, \boldsymbol{a}, \boldsymbol{b}, 0, 0, 1, 3)$ et on recommence l'étape 2.
Comme l'ensemble S_1 ne possède qu'un élément, on n'a pas d'autre choix que de choisir l'élément $(F_1, F_2, F_3, \boldsymbol{a}, \boldsymbol{b}, 0, 0, 1, 3)$ de S_1 et on le retire de S_1.
Comme $k = 3 > 1$, on applique l'algorithme E1a à $(F_1, F_2, F_3, \boldsymbol{a}, \boldsymbol{b}, 0, 0, 1)$. Celui-ci

renvoie un ensemble contenant l'octuplet $(F_1', F_2', F_3', \boldsymbol{a}', \boldsymbol{b}', 3n, 3n, 1)$, où

$$\begin{aligned} F_1'(X_1, X_2, Z) &= 1 + X_1X_2 + 5X_2, \\ F_2'(X_1, X_2, Z) &= 1 + 5X_2 + 25X_2^2, \\ F_3'(X_1, X_2, Z) &= 1 + X_1X_2 + 25X_1X_2^3, \\ \boldsymbol{a}' &= (2, -1), \\ \boldsymbol{b}' &= (1, 0). \end{aligned}$$

Pour alléger les notations dans la suite, on pose

$$(F_1, F_2, F_3, \boldsymbol{a}, \boldsymbol{b}) := (F_1', F_2', F_3', \boldsymbol{a}', \boldsymbol{b}').$$

On ajoute à S_1 l'élément $(F_1, F_2, F_3, \boldsymbol{a}, \boldsymbol{b}, 3n, 3n, 1, 2)$. On ajoute aussi tous les autres octuplets renvoyés par l'algorithme E1a mais on ne détaille les calculs effectués dans la suite de l'algorithme E1 que pour l'octuplet $(F_1, F_2, F_3, \boldsymbol{a}, \boldsymbol{b}, 3n, 3n, 1, 2)$. On recommence une nouvelle fois l'étape 2.

Etape 2 : On choisit dans S_1 l'octuplet $(F_1, F_2, F_3, \boldsymbol{a}, \boldsymbol{b}, 3n, 3n, 1, 2)$ et on le retire de S_1.

Comme $k = 2 > 1$, on applique l'algorithme E1a à $(F_1, F_2, F_3, \boldsymbol{a}_2, \boldsymbol{b}_2, 3n, 3n, 1)$ (c'est l'exemple que l'on a détaillé dans la section A.7). Celui-ci renvoie un ensemble possédant plusieurs éléments dont les trois que l'on a obtenu dans la section A.7 en détaillant l'algorithme A1a sur cet exemple. On ajoute tous ces éléments à S_1 avant de retourner à l'étape 2. Dans la suite, on ne détaillera que la fin de l'algorithme E1 pour ces trois octuplets.

Etape 2 : Choisissons dans S_1 le premier octuplet renvoyé par l'algorithme E1a dans la section A.7 que l'on récrit $(F_{1,1}, F_{2,1}, F_{3,1}, \boldsymbol{a}, \boldsymbol{b}, 3, 3, 1)$ où :

$$\begin{aligned} F_{1,1}(X_1, Z) =& (1 + Z) + 5X_1^{-1}, \\ F_{2,1}(X_1, Z) =& 1 + 5X_1^{-1} + 25X_1^{-2}, \\ F_{3,1}(X_1, Z) =& (1 + Z) + 25X_1^{-2}, \\ \boldsymbol{a} =& (1), \\ \boldsymbol{b} =& (0). \end{aligned}$$

On retire cet élément de S_1.

Cette fois-ci, on a $k = 1$. On ajoute donc $(F_{1,1}, G_{1,1}, H_{1,1}, \boldsymbol{a}, \boldsymbol{b}, 3, 3, 1)$ à l'ensemble S_2. Appelons $(F_{1,2}, F_{2,2}, F_{3,2}, \boldsymbol{a}, \boldsymbol{b}, 3, 3, 1)$ et $(F_{3,1}, F_{3,2}, F_{3,3}, \boldsymbol{a}, \boldsymbol{b}, 3, 3, 1)$ les deux autres octuplets que l'on a obtenu à la section A.7. L'étape 2 se déroule de façon analogue pour ces deux octuplets et on les ajoute également à S_2 avant de passer à l'étape 3.

Etape 3 : On choisit $(F_{1,1}, F_{2,1}, F_{3,1}, \boldsymbol{a}, \boldsymbol{b}, 3, 3, 1, 1)$ dans S_2.

Etape 3a : On fait

$$\begin{aligned}
\widetilde{F}_1(X_1) \longleftarrow & F_1(X_1,1) = 2 + 5X_1^{-1},\\
\widetilde{F}_2(X_1) \longleftarrow & F_2(X_1,1) = 1 + 5X_1^{-1} + 25X_1^{-2},\\
\widetilde{F}_3(X_1) \longleftarrow & F_3(X_1,1) = 2 + 25X_1^{-2},\\
\widetilde{D}(X_1) \longleftarrow & \operatorname{pgcd}(J\widetilde{F}_1, J\widetilde{F}_2, J\widetilde{F}_3)\\
& = \operatorname{pgcd}(2X_1 + 5, X_1^2 + 5X_1 + 25, 2X_1^2 + 25)\\
& = 1.
\end{aligned}$$

Etape 3b : On a $D = 1$.
Etape 3c : Comme D est constant, on ne fait rien.
Etape 3 : On choisit $(F_{1,2}, F_{2,2}, F_{3,2}, \boldsymbol{a}, \boldsymbol{b}, 3, 3, j, 1)$ dans S_2.
Etape 3a : On fait

$$\begin{aligned}
\widetilde{F}_1(X_1) \longleftarrow & F_1(X_1,j) = 1 + j + 5X_1^{-1},\\
\widetilde{F}_2(X_1) \longleftarrow & F_2(X_1,1) = 1 + 5X_1^{-1} + 25X_1^{-2},\\
\widetilde{F}_3(X_1) \longleftarrow & F_3(X_1,1) = 1 + j + 25X_1^{-2},\\
\widetilde{D}(X_1) \longleftarrow & \operatorname{pgcd}(J\widetilde{F}_1, J\widetilde{F}_2, J\widetilde{F}_3)\\
& = \operatorname{pgcd}((1+j)X_1 + 5, X_1^2 + 5X_1 + 25, (1+j)X_1^2 + 25)\\
& = X_1 - 5j.
\end{aligned}$$

Etape 3b : On a $D = X_1 - 5Z$.
Etape 3c : Comme D est non constant, on fait :

$$\begin{aligned}
p(X,Y,Z) \longleftarrow & JD(X,Z) = X - 5Z,\\
S \longleftarrow & S \cup \{(X - 5Z, 1, 1, j)\}.
\end{aligned}$$

Etape 3 : On choisit $(F_{1,3}, F_{2,3}, F_{3,3}, \boldsymbol{a}, \boldsymbol{b}, 3, 3, j^2, 1)$ dans S_2.
Etape 3a : On fait

$$\begin{aligned}
\widetilde{F}_1(X_1) \longleftarrow & F_1(X_1,j) = 1 + j^2 + 5X_1^{-1},\\
\widetilde{F}_2(X_1) \longleftarrow & F_2(X_1,1) = 1 + 5X_1^{-1} + 25X_1^{-2},\\
\widetilde{F}_3(X_1) \longleftarrow & F_3(X_1,1) = 1 + j^2 + 25X_1^{-2},\\
\widetilde{D}(X_1) \longleftarrow & \operatorname{pgcd}(J\widetilde{F}_1, J\widetilde{F}_2, J\widetilde{F}_3)\\
& = \operatorname{pgcd}((1+j)X_1 + 5, X_1^2 + 5X_1 + 25, (1+j)X_1^2 + 25)\\
& = X_1 - 5j^2.
\end{aligned}$$

Etape 3b : On a $D = X_1 - 5Z^2$.

Etape 3c : Comme D est non constant, on fait :

$$p(X,Y,Z) \longleftarrow JD(X,Z) = X - 5Z^2,$$
$$S \longleftarrow S \cup \{(X - 5Z^2, 1, 1, j^2)\}.$$

On doit ensuite répéter l'étape 3 pour tous les autres éléments de S_2, ce qu'on ne détaille pas ici.
Etape 4 : On renvoie S et on termine.

On rappelle que les paires $(X - 5Z, 1, 1, j)$, et $(X - 5Z^2, 1, 1, j^2)$ renvoyées par cet algorithmes correspondent respectivement aux variétés $V(XY - j, X - 5j)$ et $V(XY - j^2, X - 5j^2)$ qui définissent respectivement les points $(5j, \frac{1}{5})$ et $(5j^2, \frac{1}{5})$. On vérifie facilement que ce sont bien des points de $V(f_1, f_2, f_3) \cap \mathcal{H}_1 \backslash \mathcal{H}_0$.

A.9. Un exemple pour l'algorithme E2c

On détaille ici les calculs qu'effectue l'algorithme E2c appliqué au quintuplet $(F_1, F_2, H, \boldsymbol{a}, \boldsymbol{b})$ où :

$$F_1(X_1, X_2, X_3, X_4) = X_2X_3 - 5X_2 - 2X_3 + 10,$$
$$F_2(X_1, X_2, X_3, X_4) = X_1 - 2X_2 - 2X_4 + 4,$$
$$F_3(X_1, X_2, X_3, X_4) = X_1 + X_2X_3 - 7X_2 - 2X_3 - 2X_4 + 14,$$
$$\boldsymbol{a} = (n+1, n, 0, 1),$$
$$\boldsymbol{b} = (1, 1, 1, 0).$$

Cet exemple sera utilisé pour l'exemple de l'algorithme E2 dans la section A.10.

Etape 1 : On teste parmi tous les vecteurs $\boldsymbol{c}$ de $\mathbb{Z}^4 \backslash \{0\}$ de norme au plus $B_2(V(F_1, F_2, F_3))$ (cf Conjecture 2.3.2) s'il en existe un qui soit orthogonal à $\boldsymbol{a}$ et $\boldsymbol{b}$. On trouve dans ce cas le vecteur $\boldsymbol{c} = (-1, 1, 0, 1)$.
Etape 2 : On détermine une base du sous-groupe de $\mathbb{Z}^4$ formé des vecteurs orthogonaux à $\boldsymbol{c}$. On écrit les trois vecteurs de cette base en colonne dans la matrice suivante :

$$\begin{pmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Etape 3 : On détermine les coordonnées $\boldsymbol{a}'$ et $\boldsymbol{b}'$ de $\boldsymbol{a}$ et $\boldsymbol{b}$ dans la base déterminée à l'étape précédente. On obtient

$$\begin{aligned}\boldsymbol{a}' &= (n, 0, 1),\\ \boldsymbol{b}' &= (1, 1, 0).\end{aligned}$$

Etape 4 : On pose

$$\begin{aligned}F_1'(X_1, X_2, X_3) &\leftarrow F_1(X_1X_3, X_1, X_2, X_3)\\ &= X_1X_2 - 5X_1 - 2X_2 + 10,\\ F_2'(X_1, X_2, X_3) &\leftarrow F_2(X_1X_3, X_1, X_2, X_3)\\ &= X_1X_3 - 2X_1 - 2X_3 + 4,\\ F_3'(X_1, X_2, X_3) &\leftarrow F_3(X_1X_3, X_1, X_2, X_3)\\ &= X_1X_3 + X_1X_2 - 7X_1 - 2X_2 - 2X_3 + 14.\end{aligned}$$

Etape 5 : On renvoie $(F_1', F_2', F_3', \boldsymbol{a}', \boldsymbol{b}')$ et on termine.

On vérifie que l'on a bien

$$\begin{aligned}F_1(X^{n+1}Y, X^nY, Y, X) &= F_1'(X^nY, X, Y),\\ F_2(X^{n+1}Y, X^nY, Y, X) &= F_2'(X^nY, X, Y),\\ F_3(X^{n+1}Y, X^nY, Y, X) &= F_3'(X^nY, X, Y).\end{aligned}$$

A.10. Un exemple pour l'algorithme E2

Soit n un entier tel que $n \gg 1$. On détaille ici les calculs qu'effectue l'algorithme E2 pour déterminer une représentation des points de $\mathcal{H}_2\backslash\mathcal{H}_1$ appartenant à la variété définie par les polynômes :

$$\begin{aligned}f_1(X, Y) =& X^nY^2 - 5X^nY - 2Y + 10,\\ f_2(X, Y) =& X^{n+1}Y - 2X^nY - 2X + 4,\\ f_3(X, Y) =& X^{n+1}Y + X^nY^2 - 7X^nY - 2X - 2Y + 14.\end{aligned}$$

Etape 1 : On linéarise les trois polynômes, on définit les deux vecteurs d'exposants $\boldsymbol{a}$ et $\boldsymbol{b}$, le nombre de variables k, le nombre de polynômes r, on initialise S à

l'ensemble vide et p à 1 :

$$\begin{aligned}
F_1(X_1, X_2, X_3, X_4, X_5) &\leftarrow X_2 - 5X_3 - 2X_4 + 10,\\
F_2(X_1, X_2, X_3, X_4, X_5) &\leftarrow X_1 - 2X_3 - 2X_5 + 4,\\
F_3(X_1, X_2, X_3, X_4, X_5) &\leftarrow X_1 + X_2 - 7X_3 - 2X_4 - 2X_5 + 14,\\
\boldsymbol{a} &\leftarrow (n+1, n, n, 0, 1),\\
\boldsymbol{b} &\leftarrow (1, 2, 1, 1, 0),\\
k &\leftarrow 5,\\
r &\leftarrow 3,\\
p &\leftarrow 1,\\
S &\leftarrow \emptyset.
\end{aligned}$$

Etape 2 : Comme $k = 5 \neq 2$, on applique l'algorithme E2c au quintuplet $(F_1, F_2, F_3, \boldsymbol{a}, \boldsymbol{b})$. Celui-ci renvoie le quintuplet $(F'_1, F'_2, F'_3, \boldsymbol{a}', \boldsymbol{b}')$ où :

$$\begin{aligned}
F'_1(X_1, X_2, X_3, X_4) &= X_2X_3 - 5X_2 - 2X_3 + 10,\\
F'_2(X_1, X_2, X_3, X_4) &= X_1 - 2X_2 - 2X_4 + 4,\\
F'_3(X_1, X_2, X_3, X_4) &= X_1 + X_2X_3 - 7X_2 - 2X_3 - 2X_4 + 14,\\
\boldsymbol{a}' &= (n+1, n, 0, 1),\\
\boldsymbol{b}' &= (1, 1, 1, 0).
\end{aligned}$$

On fait $(F_1, F_2, F_3, \boldsymbol{a}, \boldsymbol{b}) \leftarrow (F'_1, F'_2, F'_3, \boldsymbol{a}', \boldsymbol{b}')$, $k \leftarrow 4$ et on recommence l'étape 2.

Etape 2 : Comme $k = 4 \neq 2$, on applique l'algorithme E2c au quintuplet $(F_1, F_2, F_3, \boldsymbol{a}, \boldsymbol{b})$. C'est l'exemple que l'on a détaillé à la section A.9. Celui-ci renvoie alors le quintuplet $(F'_1, F'_2, F'_3, \boldsymbol{a}', \boldsymbol{b}')$ où :

$$\begin{aligned}
F'_1(X_1, X_2, X_3) &= X_1X_2 - 5X_1 - 2X_2 + 10,\\
F'_2(X_1, X_2, X_3) &= X_1X_3 - 2X_1 - 2X_3 + 4,\\
F'_3(X_1, X_2, X_3) &= X_1X_3 + X_1X_2 - 7X_1 - 2X_2 - 2X_3 + 14,\\
\boldsymbol{a}' &= (n, 0, 1),\\
\boldsymbol{b}' &= (1, 1, 0).
\end{aligned}$$

On fait $(F_1, F_2, F_3, \boldsymbol{a}, \boldsymbol{b}) \leftarrow (F'_1, F'_2, F'_3, \boldsymbol{a}', \boldsymbol{b}')$, $k \leftarrow 3$ et on recommence l'étape 2.

Etape 2 : Comme $k = 3 \neq 2$, on applique l'algorithme E2c au quintuplet $(F_1, F_2, F_3, \boldsymbol{a}, \boldsymbol{b})$. Comme celui-ci renvoie « RIEN », on va à l'étape 3.

Etape 3 : On calcule

$$\begin{aligned} D(X_1, X_2, X_3) &\leftarrow \text{pgcd}(F_1, F_2, F_3) \\ &= X_1 - 2, \\ p(X, Y) &\leftarrow p(X, Y) \cdot D(X^n Y, Y, X) = X^n Y - 2, \\ F_1(X_1, X_2, X_3) &\leftarrow F_1/D = X_2 - 5, \\ F_2(X_1, X_2, X_3) &\leftarrow F_2/D = X_3 - 2, \\ F_3(X_1, X_2, X_3) &\leftarrow F_3/D = X_2 + X_3 - 7. \end{aligned}$$

Etape 4 : On applique l'algorithme E2b à (F_1, F_2, F_3) pour résoudre le système

$$F_1 = F_2 = F_3 = 0.$$

Celui-ci renvoie le triplet (P_1, P_2, F_4), où $P_1 = (X_2 - 5)$, $P_2 = (X_3 - 2)$ et $F_4 = 1$. On pose :

$$\begin{aligned} p_1(X, Y) &\leftarrow P_1(X^n Y, Y, X) \\ &= Y - 5, \\ p_2(X, Y) &\leftarrow P_2(X^n Y, Y, X) \\ &= X - 2, \\ p_3(X, Y) &\leftarrow F_4(X^n Y, Y, X) = 1, \\ S &\leftarrow S \cup \{(p_1, p_2, p_3)\}. \end{aligned}$$

Etape 5 : Comme $F_4 = 1$, on va à l'étape 6.

Etape 6 : On renvoie (p, S) et on termine.

On vérifie que $p(X, Y) = X^n Y - 2$ est bien un facteur commun de f_1, f_2 et f_3. Dans ce cas, ce polynôme est même leur pgcd puisqu'on peut montrer que ces trois polynômes n'ont pas de facteur cyclotomique en commun.

La variété définie par f_1, f_2 et f_3 possède aussi un point isolé défini par la paire $(Y - 5, X - 2)$. C'est à dire le point $(2, 5)$ qui est bien un point de $V(f_1, f_2, f_3) \cap \mathcal{H}_2 \backslash \mathcal{H}_1$.

INDEX DES ALGORITHMES

NOTATIONS UTILISÉES

BIBLIOGRAPHIE

[AS08] I. Aliev et C. Smyth, *Solving algebraic equations in roots of unity*, preprint, arXiv : 0704.1747v3., (2008).

[AD00] F. Amoroso et S. David, *Minoration de la hauteur normalisée des hypersurfaces*, Acta Arith. **92**, (2000), 339-366.

[AD03] F. Amoroso et S. David, *Minoration de la hauteur normalisée dans un tore*, J. Inst. Math. Jussieu **2**, (2003), 335-381.

[AD06] F. Amoroso et S. David, *Points de petite hauteur sur une sous-variété d'un tore*, Compos. Math. **142**, (2006), 551-562.

[AV09] F. Amoroso et E. Viada, *Small points on subvarieties of tori*, Duke Math. J. **150** no 3, (2009), 407-442.

[AZ00] F. Amoroso et U. Zannier, *A relative Dobrowolski lower bound over Abelian varieties*, Ann. Scuola Norm. Sup. Pisa Cl. Sci. **4** XXIX, (2000), 711-727.

[AKS07] M. Avendaño, T. Krick et M. Sombra, *Factoring bivariate sparse (lacunary) polynomials*, J. Complexity **23** no 2, (2007), 193-216.

[BS02] F. Beukers et C. Smyth, *Cyclotomic points on curves*, Number theory for the millenium, **1** (Urbana, Il., 2000), (2002), 67-85.

[BMZ07] E. Bombieri, D. Masser, et U. Zannier, *Anomalous Subvarieties - Structure Theorems and Applications*, Int. Math. Res. Not. **19**, (2007), 1-33.

[BZ95] E. Bombieri et U. Zannier. *Algebraic points on subvarieties of* $\mathbb{G}_m^n$, Int. Math. Res. Not. **7**, (1995), 333-347.

[BZ98] E. Bombieri et U. Zannier. *Intersections of varieties with 1-dimensional tori and a conjecture of Schinzel*, manuscrit, (1998).

[CTV09] Q. Cheng, S. P. Tarasov et M. N. Vyalyi, *Efficient algorithms for sparse cyclotomic integer zero testing*, Theory of Computing Systems **46** no 1, (2010), 120-142.

[C93] C. H. Cohen. *A Course in Computational Algebraic Number Theory*, Graduate Text in Mathematics 138, Springer-Verlag, (1993).

[CJ76] J. H. Conway et A. J. Jones, *Trigonometric Diophantine equations (On vanishing sums of roots of unity)*, Acta Arith. **30**, (1976), 229-240.

[CLO97] D. Cox, J. Little, D. O'Shea, *Ideals, varieties, and algorithms. An introduction to computational algebraic geometry and commutative algebra. Second edition*, Springer-Verlag, 1997.

[DP07] S. David et P. Philippon, *Minorations des hauteurs normalisées des sous-variétés des puissances de courbes elliptiques*, Int. Math. Res. Pap. **3**, (2007), ID rpm006.

[FGS08] M. Filaseta, A. Granville et A. Schinzel, *Irreducibility and Greatest Common Divisor Algorithms for Sparse Polynomials*, Number Theory and Polynomials (ed. James McKee and Chris Smyth), LMS Lecture Note Series **352**, Cambridge Univ. Press, (2008), 155-176.

[FS04] M. Filaseta et A. Schinzel, *On testing the divisibility of lacunary polynomials by cyclotomic polynomials*, Math. Comp. **73**, (2004), 957-965.

[F07] M. Fürer,

[GG03] J. von zur Gathen et J. Gerhard, *Modern Computer Algebra*, Cambridge University Press, 2nd edition, (2003).

[Ha53] G. Hajós, *Solution of Problem 41*, (Hongrois), Mat. Lapok 4, (1953), 40-41.

[Hi88] M. Hindry, *Autour d'une conjecture de S. Lang*, Invent. Math. **94**, (1988), 575-603.

[HL10] J. Hoeven et G. Lecerf, *On the bit-complexity of sparse polynomial and series multiplication*, preprint, arXiv : 0901.4323v1, (2010).

[KK06] E. Kaltofen et P. Koiran, *Finding small degree factors of multivariate supersparse (lacunary) polynomials over algebraic number fields*, ISSAC 2006, Proc. 2006 Internat. Symp. Symbolic Algebraic comput, (2006), 162-168.

[Knu70] D. E. Knuth, *The analysis of algorithms*, Actes du Congrés International des Mathématiciens, (1970), 269-274.

[Kro82] L. Kronecker, *Grundzüge einer arithmetischen Theorie der algebraischen Grössen*, J. reine angew. Math., **92**, (1882), 1-122.

[KPS01] T. Krick, L. M. Pardo, M. Sombra, *Sharp estimates for the arithmetic Nullstellensatz*, Duke Math. J. **109** (2001) 521-598.

[LS96] G. Labahn et A. Storjohann, *Asymptotically fast computation of Hermite normal forms of integer matrices*, In Proc. Internat. Symp. on Symbolic and Algebraic Computation : ISSAC '96, (1996), 259-266.

[Lan83] S. Lang, *Fundamentals of diophantine geometry*, Springer-Verlag, (1983).

[Lau84] M. Laurent, *Equations diophantiennes exponentielles*, Invent. Math. **78**, (1984), 299-327.

[Lec00] G. Lecerf, *Computing an equidimensional decomposition of an algebraic variety by means of geometric resolutions*, Proceedings of ISSAC'2000 (ACM), (2000).

[Len99] H. Lenstra, *Finding small degee factors of lacunary polynomials*, In Győry et al. GIU99, (1999), 267-276.

[Ler10] L. Leroux, *Computing the torsion points of a variety defined by lacunary polynomials*, preprint, arXiv : 0911.2594v1, (2010).

[MS04] T. Mulders et A. Storjohann, *Certified dense linear system solving*, J. Symbolic Comput., **37**, (2004), 485-510.

[P84] D. A. Plaisted, *New NP-hard and NP-complete polynomial and integer divisibility problems*, Theoret. Comput. Sci. **31**, (1984), 125-138.

[PR98] B. Poonen, M. Rubinstein, *The number of intersection points made by the diagonals of a regular polygon*, SIAM J. on Disc. Math. **11** no 1, (1998), 133-156.

[Ra83] M. Raynaud, *Courbes sur une variété abélienne et points de torsion*, Invent. Math **71**, (1983), 207-233.

[Re02] G. Rémond, *Sur les sous-variétés des tores*, Comp. Math., (2002), 337-366.

[Ro07] J. M. Rojas, *Efficiently detecting subtori and torsion points*, proceedings of MAGIC 2005, Contemporary Mathematics **448**, (2007), 213-233.

[RS62] J. B. Rosser et L. Schoenfeld, *Approximate formulas for some functions of prime numbers*, Illinois J. Math. **6**, (1962), 64-94.

[Ru93] W. M. Ruppert, *Solving algebraic equations in roots of unity*, J. Reine Angew. Math. **435**, (1993), 119-156.

[Schi02] A. Schinzel, *On the greatest common divisor of two univariate polynomials, I*, A panorama of number theory or the view from Baker's garden (Zürich, 1999), Cambridge Univ. Press, Cambridge, (2002), 337-352.

[S96] W. M. Schmidt, *Heights of points on subvarieties of* $\mathbb{G}_m^n$, Number Theory 1993-1994, edition S. David, London Math. Soc. Ser., **235**, Cambridge University Press, (1996), 157-187.

[SS71] A. Schönhage et V. Strassen, *Schnelle Multiplikation grosser Zahlen*, Computing **7**, (1971), 281-292.

[Z00] U. Zannier, Appendix in A. Schinzel's *Polynomials with Special Regard to Reducibility. With an Appendix by Umberto Zannier*, in Encyclopedia of Mathematics and its Applications, **77**, Cambridge University Press, (2000), 517-539.

[Z09] U. Zannier, *Lecture notes on Diophantine Analysis*, Publications of the Scuola Normale Superiore, (2009).

Algorithmes pour les polynômes lacunaires

Le but de cette thèse est d'utiliser plusieurs résultats profonds de géométrie diophantienne et de géométrie algébrique pour obtenir des applications à la factorisation des polynômes lacunaires. Dans la première partie, on décrit un algorithme qui détermine une représentation des points de torsion d'une sous-variété de $\mathbb{G}_m^n$ définie par des polynômes lacunaires. La complexité de cet algorithme est quasi-linéaire en le logarithme du degré des polynômes définissant cette sous-variété. Dans la seconde partie, on s'intéresse à des systèmes surdéterminés d'équations polynomiales. On décrit un algorithme qui permet d'écrire les zéros communs de trois polynômes à deux variables comme une réunion finie d'intersections complètes en dehors d'un ouvert de $\mathbb{A}^2$. La complexité de cet algorithme est encore quasi-linéaire en le logarithme du degré des polynômes en entrée mais cet algorithme dépend de la validité de la conjecture de Zilber qui est encore à ce jour un problème ouvert.

Algorithms for lacunary polynomials

The aim of this thesis is to use some results from Diophantine geometry and from algebraic geometry to obtain applications to the the factorization of lacunary polynomials. In the first part, we describe an algorithm which computes a representation of the torsion points of a subvariety of $\mathbb{G}_m^n$ defined by lacunary polynomials. The complexity of this algorithm is quasi-linear in the logarithm of the degree of the polynomials defining the subvariety. In the second part, we focus on systems of three lacunary polynomial equations in two variables. We describe an algorithm that computes a representation of the common zeroes of those polynomials as a finite union of complete intersections outside an open subset of $\mathbb{A}^2$. The complexity of this algorithm is quasi-linear in the logarithm of the degree of the input polynomials. However this algorithm depends on the conjecture of Zilber which is still an open problem.

Mots clés *(Indexation Rameau)* : Algorithmes, Polynômes, Factorisation.

Spécialité : Mathématiques et leurs intéractions
Laboratoire de Mathématiques Nicolas Oresme, CNRS UMR 6139
Université de Caen Basse-Normandie BP 5186
14 032 CAEN Cedex, FRANCE
louis.leroux@math.unicaen.fr