Journées Nationales de Calcul Formel

Rencontre organisée par :
Guillaume Chèze, Paola Boito, Clément Pernet et Mohab Safey el Din

2013

Grégoire Lecerf

Factorisation des polynômes à plusieurs variables

Vol. 3, n° 1 (2013), Cours n° I, p. 1-85.

<http://ccirm.cedram.org/item?id=CCIRM_2013__3_1_A1_0>

Centre international de rencontres mathématiques
U.M.S. 822 C.N.R.S./S.M.F.
Luminy (Marseille) France

Les cours du C.I.R.M.
Vol. **3** n° 1 (2013) 1-85
Cours n° I

Factorisation des polynômes à plusieurs variables

Support de cours des Journées Nationales de Calcul Formel 2013

par Grégoire Lecerf

Laboratoire d'informatique
UMR 7161 CNRS
Campus de l'École polytechnique
91128 Palaiseau Cedex, France

Courriel : gregoire.lecerf@math.cnrs.fr

Avant-Propos

La factorisation irréductible des polynômes à plusieurs variables est connue depuis les années cinquante pour ne pas être calculable pour un corps effectif de coefficients. Néanmoins pour les implémentations habituelles de nombreux corps, dont les nombres rationnels, les nombres algébriques, les corps finis, et les corps de fonctions, nous disposons d'algorithmes de bonne complexité théorique ainsi que d'implémentations performantes dans divers logiciels de calcul formel. Il reste néanmoins plusieurs questions ouvertes importantes puisque dans la plupart des cas on ne connaît pas d'algorithme en temps quasi-linéaire.

Nous commençons ce cours avec la factorisation séparable, qui permet de scinder un polynôme à une variable en facteurs dont les racines ont la même multiplicité. Nous montrons comment la factorisation irréductible à une variable se ramène au cas des polynômes séparables. Le cas des polynômes à une variable est en fait critique et nous présentons les algorithmes désormais classiques lorsque les coefficients sont des nombres rationnels ou appartiennent à un corps fini. Dans un deuxième temps, nous montrons comment ramener la factorisation des polynômes à deux variables à celle à une variable. Enfin, à partir de trois variables, il est souvent favorable de se ramener au cas de deux variables *via* un théorème dû à Hilbert. L'essentiel du coût de la factorisation devient alors lié à celui des opérations arithmétiques sur les polynômes et séries à plusieurs variables.

Le présent document est un support de cours et certains choix de présentation ont parfois favorisé la communication de certaines idées de façons simplifiées ou informelles. Par ailleurs ces notes de cours contiennent plusieurs exemples réalisés au moyen du logiciel de calcul formel et analytique Mathemagix (http://www.mathemagix.org).

Remerciements. Je tiens à remercier Jérémy Berthomieu et Guillaume Quintin pour leur commentaires sur ce manuscript. Je remercie aussi les concepteurs du logiciel GNU TEXMACS (http://www.texmacs.org) avec lequel le présent document a été rédigé.

Palaiseau, le 26 mai 2013

Table des matières

Introduction

Soit $\mathbb{A}$ un anneau commutatif intègre dont on note $U(\mathbb{A})$ le groupe des unités. Soient a et b des éléments de $\mathbb{A}$. On dit que a *divise* b, que l'on note $a \mid b$, s'il existe c dans $\mathbb{A}$ tel que $b = a\,c$. Comme $\mathbb{A}$ est intègre, si $a \neq 0$ alors c est unique. On dit que a est *irréductible* si $a \notin U(\mathbb{A})$ et si pour tout b et c dans $\mathbb{A}$ tels que $a = b\,c$ alors $b \in U(\mathbb{A})$ ou $c \in U(\mathbb{A})$. Notons que $0 = 0 \cdot 0$ n'est pas irréductible. On note $a \sim b$ lorsque les idéaux (a) et (b) engendrés respectivement par a et b sont égaux. On note $\mathbb{A}^* := \mathbb{A} \setminus \{0\}$.

Définition 1. *Un anneau commutatif intègre* $\mathbb{A}$ *est dit* factoriel *s'il vérifie les conditions suivantes :*

i. pour tout $a \in \mathbb{A}^*$*, il existe* $u \in U(\mathbb{A})$*,* $m \in \mathbb{N}$*, et des éléments irréductibles* $p_1, ..., p_m$ *de* $\mathbb{A}$ *tels que* $a = u\,p_1 \cdots p_m$ *;*

ii. si $p_1, ..., p_m$*,* $q_1, ..., q_n$ *sont des irréductibles de* $\mathbb{A}$*,* $u, v \in U(\mathbb{A})$*, et* $m, n \in \mathbb{N}$ *tels que* $u\,p_1 \cdots p_m = v\,q_1 \cdots q_n$*, alors* $m = n$ *et il existe une permutation* σ *de* $\{1, ..., n\}$ *telle que* $p_i \sim q_{\sigma(i)}$ *pour tout* $i \in \{1, ..., n\}$*.*

On dit alors que la décomposition en irréductibles $u\,p_1 \cdots p_m$ de a est *essentiellement unique*, ou *unique à des unités près*. Dans ce cours, nous supposons connu le fait que l'anneau des entiers $\mathbb{Z}$ est factoriel et que si $\mathbb{A}$ est un anneau factoriel alors l'anneau des polynômes $\mathbb{A}[x]$ l'est aussi. Nous nous intéressons au problème de calculer la décomposition en irréductibles des polynômes à plusieurs variables à coefficients dans un corps commutatif $\mathbb{K}$.

La recherche des racines d'un polynôme à une variable et plus généralement la factorisation de tels polynômes en irréductibles trouve ses origines dans des temps très anciens des mathématiques. Un bref historique se trouve par exemple dans [43]. Une présentation historique plus détaillée sur la factorisation des polynômes se trouve dans les sections « Notes » des chapitres de la partie III du livre [44]. Dans ce cours nous concentrons principalement sur des algorithmes récents, de bonne complexité, et qui permettent de couvrir tous les cas usuels.

Contrairement aux opérations élémentaires sur les polynômes telles que la somme, le produit, et le plus grand commun diviseur (noté *p.g.c.d.* dans le texte, et gcd dans les formules mathématiques), il n'existe pas d'algorithme de factorisation des polynômes de $\mathbb{K}[x]$ qui soit indépendant de $\mathbb{K}$. Afin de formaliser ce résultat nous devons considérer comme modèle de calcul une *machine de Turing* ou bien, de façon équivalente, une *machine à accès direct*. Nous dirons qu'un corps $\mathbb{K}$ est *effectif* si ses éléments peuvent être représentés dans ce modèle et si le test d'égalité de deux éléments de $\mathbb{K}$ est *calculable*. Concernant le formalisme et les concepts classiques de ce modèle de calcul, le lecteur pourra consulter des ouvrages généralistes comme [138, Chapitre 2].

Théorème 2. *[30, 31] Il existe des corps effectifs* $\mathbb{K}$ *pour lesquels la décomposition en irréductibles des polynômes de* $\mathbb{K}[x]$ *n'est pas calculable.*

Démonstration. Soit $\lambda\colon \mathbb{N}^* \to \mathbb{N}^*$ une fonction injective calculable. Soit p_i le i-ième nombre premier et soit $\mathbb{K}$ le corps $\mathbb{Q}(\sqrt{p_{\lambda(1)}}, \sqrt{p_{\lambda(2)}}, \sqrt{p_{\lambda(3)}}, \ldots)$. Chaque élément de $\mathbb{K}$ s'exprime comme un polynôme en un nombre fini de $\sqrt{p_i}$. Le test d'égalité peut être réalisé en utilisant par exemple des calculs de base standard *via* l'algorithme de Buchberger.

Pour une valeur de n donnée, factoriser $x^2 - p_n$ dans $\mathbb{K}[x]$ est équivalent à tester si n est dans l'image de λ. Pour prouver le théorème, il suffit donc de choisir λ telle que ce test ne soit pas calculable. Historiquement, de telles fonctions λ ont été construites dans [80].

À partir de résultats classiques sur les machines de Turing, nous pouvons décrire ici comment construire une telle fonction λ en utilisant le fait que le problème de l'arrêt d'un programme est indécidable. En fait, commençons par fixer un alphabet permettant de coder les machines de Turing et ainsi les couples (T, e) formés par une machine de Turing T et une entrée e. On peut ensuite construire l'encodage d'un tel couple en binaire de sorte à lui associer un entier $\varphi(T, e)$ avec φ injective. Par ailleurs, pour tout triplet d'entiers t, n et c, nous pouvons énumérer l'ensemble $P_{t,n,c}$ des couples (T, e) tels que T a pour taille t, e pour taille n et l'exécution de T sur l'entrée e prend c étapes. En ordonnant les triplets (t, n, c) et les éléments de $P_{t,n,c}$ nous obtenons un ordre calculable sur les couples (T, e) tels que T s'arrête sur e. On prend donc pour $\lambda(i)$, la valeur $\varphi(T, e)$ où (T, e) représente le i-ième couple pour cet ordre. □

Le précédent résultat nous indique que pour obtenir un algorithme de factorisation dans $\mathbb{K}[x]$ il faut examiner la nature de $\mathbb{K}$. Néanmoins, en pratique, nous pouvons nous contenter de manipuler des corps sous une forme particulière :

Définition 3. *Un corps* $\mathbb{K}$ *est* explicitement finiment engendré *sur un corps* $\mathbb{F}$ *lorsqu'il est donné par le corps des fractions de* $\mathbb{F}[x_1, \ldots, x_n]/\mathfrak{P}$*, où* $\mathfrak{P}$ *est un idéal premier de* $\mathbb{F}[x_1, \ldots, x_n]$ *donné par un ensemble fini de polynômes de* $\mathbb{F}[x_1, \ldots, x_n]$.

Un tel corps est bien effectif car on peut tester l'égalité grâce à une forme normale obtenue par un calcul de base standard. Avec un tel corps, de nombreux travaux ont contribué à montrer que la factorisation des polynômes est calculable sur $\mathbb{K}$. Les contributions majeures, d'un point de vue historique, remontent à Kronecker [83], puis, quarante ans plus tard, à Hermann [50], suivi par van der Waerden [143, 144], Fröhlich et Shepherdson [30, 31] dans les années cinquante, et enfin Seidenberg [131, 132, 133] et Richman [95, 121]. Rappelons qu'un corps et dit *premier* s'il n'a pas de sous-corps propre.

Théorème 4. *Si* $\mathbb{K}$ *est un corps explicitement finiment engendré sur son sous-corps premier* $\mathbb{F}$*, alors la factorisation dans* $\mathbb{K}[x_1, \ldots, x_n]$ *est calculable.*

L'objet de ce cours est de détailler ce résultat en présentant des algorithmes récents performants, issus de travaux de recherches principalement commencés dans les années 1970 en calcul formel.

L'approche que nous développons dans ce cours consiste d'abord à calculer la factorisation dite séparable pour un polynôme $F \in \mathbb{K}[x]$, où $\mathbb{K}$ est un corps. Il s'agit en fait de séparer les racines de F selon leur multiplicité. La factorisation séparable est au cœur des traitements modernes de la factorisation en mathématiques constructives, telles que développées par RICHMAN est ses collaborateurs [96]. Ensuite nous pourrons supposer que les polynômes à factoriser n'ont pas de racines multiples. Nous mentionnerons rapidement les méthodes de factorisation dans $\mathbb{Q}[x]$ et $\mathbb{F}_{p^k}[x]$, où $\mathbb{F}_{p^k}$ représente le corps fini à p^k éléments. Puis nous verrons comment la factorisation dans $\mathbb{K}[x]$ implique celle dans $\mathbb{K}(\alpha)[x]$, pour un nombre algébrique α sur $\mathbb{K}$, puis celle dans $\mathbb{K}[x, y]$. Enfin nous montrerons comment réduire le cas général de la factorisation dans $\mathbb{K}[x_1, ..., x_n]$ au cas de deux variables.

La plupart des logiciels de calcul formel offrent des fonctions pour factoriser les entiers et les polynômes. Néanmoins ce cadre général des polynômes à plusieurs variables sur un corps $\mathbb{K}$ explicitement finiment engendré sur son sous-corps premier est rarement disponible en toute généralité. Mentionnons que le logiciel MAGMA propose une implémentation efficace couvrant le cas général [136]. Parmi les logiciels libres, de nombreux cas particuliers de factorisation sont disponibles dans NTL [135], PARI/GP [117], SAGE [137], SINGULAR [26], et plus récemment dans MATHEMAGIX [56]. Dans ce cours nous illustrerons plusieurs algorithmes avec des exemples pour lesquels nous effectuerons les calculs avec l'interprète MMX-LIGHT de MATHEMAGIX. Le code source, écrit en C++, est regroupé dans la librairie appelée FACTORIX, mais utilise les opérations sur les matrices, polynômes et séries d'ALGEBRAMIX, les corps finis de FINITEFIELDZ, et les opérations sur les réseaux de LATTIZ.

1

Modèle de calcul et complexité

Dans l'introduction nous avons considéré le modèle de calcul des machines de Turing, et nous pourrions continuer à l'utiliser pour décrire les algorithmes de ce cours et analyser leur coût. Néanmoins, nous serions confrontés au problème qu'un grand nombre d'algorithmes récents sur les polynômes et matrices ne sont pas décris dans ce modèle. D'autre part l'analyse de complexité en fonction des opérations élémentaires sur une structure algébrique générique serait délicate. Par conséquent, nous avons choisi le modèle plus faible mais plus pratique des *arbres de calcul* que nous allons définir dans ce chapitre en suivant la présentation de [16, Chapter 4].

Le choix d'un modèle de calcul pour présenter et analyser des algorithmes est une question épineuse. Considérer un modèle trop proche des ordinateurs actuels, qui peuvent effectuer plusieurs tâches en parallèle, et qui ont une gestion de la mémoire par niveaux de cache, est bien trop compliqué. À l'inverse, considérer un modèle très simple comme les machines de Turing permet de bien formaliser la notion de programme et de complexité, mais il est très éloigné des architectures matérielles actuelles. Les arbres de calcul semblent aussi éloignés de la réalité, mais nous nous efforcerons de décrire les algorithmes dans un langage facilement compréhensible tout en gardant en tête, d'un point de vue théorique, qu'une telle description explicite la construction d'arbres de calcul pour lesquels nous pouvons énoncer des résultats avec rigueur.

Pour sûr, le travail nécessaire pour passer d'un algorithme décrit en terme d'arbres de calcul à une implémentation efficace est souvent important, surtout pour des opérations qui semblent simples comme le produit de polynômes ou le produit de matrices. En effet, ce travail consiste à concevoir de bonnes structures de données, à développer des stratégies d'utilisation de la mémoire favorables à l'architecture utilisée, et enfin à s'assurer que le code exécutable produit exploite correctement les ressources matérielles de calcul.

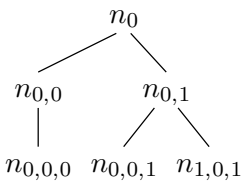
La lecture de ce chapitre n'est pas strictement nécessaire pour comprendre les algorithmes présentés dans les chapitres suivants. Il permet de formaliser les résultats de complexité et il contient aussi les rappels nécessaires sur le coût des opérations fondamentales sur les polynômes et matrices que nous utiliserons.

1.1 Arbre binaire

Définition 1.1. *Un* arbre binaire *est un graphe acyclique orienté dont tous les nœuds sauf*

un, appelé la racine*, a un parent et au plus deux fils.*

Exemple 1.2.



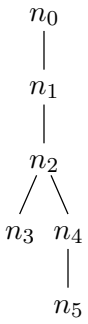
Un arbre binaire induit un *ordre partiel naturel* $\prec$ sur l'ensemble de ses nœuds, pour lequel la racine est le plus petit élément :

$$u \prec v \Leftrightarrow u \text{ est le grand-}\cdots\text{-grand-parent de } v.$$

Définition 1.3. *Un* chemin *est une suite finie de nœuds $v_0, v_1, ..., v_l$ tels que v_i est le parent de v_{i+1} pour tout $i \in \{0, ..., l-1\}$.*

Définition 1.4. *Un* segment initial *de longueur n d'un arbre binaire est un ensemble de nœuds $\mathcal{I} = \{v_1, ..., v_n\}$ tel que $v_1 \prec v_2 \prec \cdots \prec v_n$, et tel qu'il existe un nœud u qui est le successeur de v_n et le prédécesseur de tous les nœuds qui ne sont pas dans $\mathcal{I} \cup \{u\}$.*

Exemple 1.5. $\mathcal{I} = \{\}$, $\mathcal{I} = \{n_0\}$, $\mathcal{I} = \{n_0, n_1\}$ sont des segments initiaux de l'arbre suivant :



1.2 Syntaxe d'un arbre de calcul

Si $\mathbb{K}$ est un corps, nous définissons $\mathbb{K}^c := \{\mathbb{K}^0 \to \mathbb{K}, () \mapsto a \mid a \in \mathbb{K}\}$ comme l'ensemble des *fonctions d'arité zéro*, et notons Id la fonction identité sur $\mathbb{K}$. L'ensemble des *opérations élémentaires* Ω sur $\mathbb{K}$ est $\Omega := \mathbb{K}^c \cup \{\mathrm{Id}, +, -, \times, /\}$. L'ensemble des *prédicats* P est quant à lui défini par $P := \{=, \neq\}$.

Plus généralement nous serons amenés à considérer des structures algébriques $\mathbb{A}$ qui ne seront pas des corps et nous adapterons naturellement ces définitions. Par exemple, si $\mathbb{A}$ est un anneau, les opérations élémentaires seront $\mathbb{A}^c \cup \{\mathrm{Id}, +, -, \times\}$. Éventuellement rien n'empêche, si $\mathbb{A}$ est intègre d'y adjoindre la division exacte. Si $\mathbb{A}$ est ordonné, alors l'ensemble des prédicats pourra aussi contenir les tests $\{<, \leqslant, \geqslant, >\}$. Il conviendra, pour chaque énoncé, si le contexte n'est pas clair, de préciser les opérations élémentaires et prédicats utilisés.

Soit T un arbre binaire et $\mathcal{I}$ l'un de ses segments initiaux de longueur n. Les éléments de $\mathcal{I}$ sont appelés les nœuds d'entrée. L'ensemble des nœuds hors de $\mathcal{I}$ est partitionné selon leur degré sortant d :

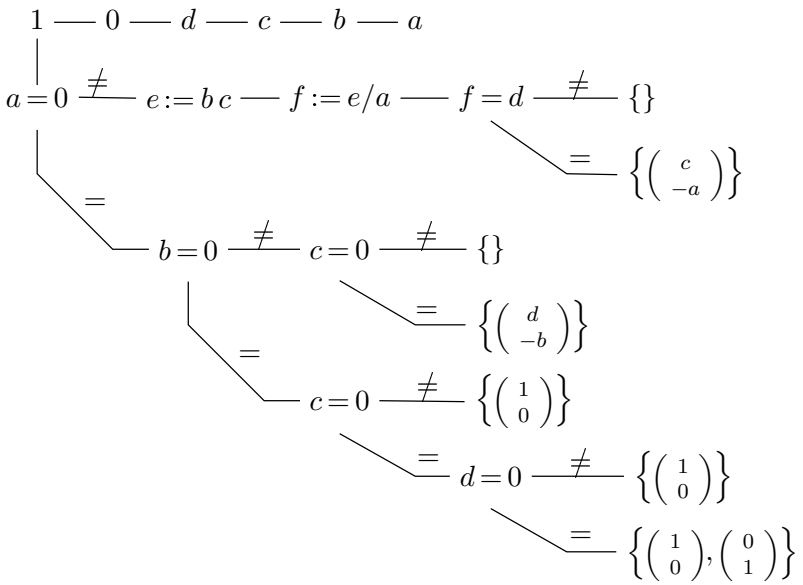
- les *nœuds de sortie* $\mathcal{O}$, qui correspondent à $d = 0$,
- les *nœuds de calcul* $\mathcal{C}$, qui correspondent à $d = 1$,

– les *nœuds de branchement* $\mathcal{B}$, qui correspondent à $d=2$.

Définition 1.6. *Un* arbre de calcul *sur une structure* $\mathbb{A}$ *munie des opérations élémentaires* Ω*, des prédicats* P*, et avec une entrée de taille* n*, est un arbre binaire muni :*

1. *d'une* fonction d'instruction *qui assigne :*
   – *à chaque nœud de calcul* v *une* instruction de calcul *de la forme* $(\omega; u_1, ..., u_m)$*, où* $\omega \in \Omega$ *est d'arité* m*, et* $u_1, ..., u_m \in \mathcal{I} \cup \mathcal{C}$ *sont des prédécesseurs de* v*,*
   – *à chaque nœud de branchement* v *une* instruction de test *de la forme* $(\rho; u_1, u_2)$*, où* $\rho \in P$ *et* $u_1, u_2 \in \mathcal{I} \cup \mathcal{C}$ *sont des prédécesseurs de of* v*,*
   – *à chaque nœud de sortie* v *une* instruction de sortie *de la forme* $(u_1, ..., u_m)$*, où* $m \in \mathbb{N}$ *et* $u_1, ..., u_m \in \mathcal{I} \cup \mathcal{C}$ *sont des prédécesseurs de* v*.*
2. *d'une* partition σ *de l'ensemble des nœuds de sortie telle que la longueur des nœuds de sortie est constante sur chaque partie de* σ*.*

Exemple 1.7. Pour le calcul du noyau d'une matrice de la forme $\begin{pmatrix} a & c \\ b & d \end{pmatrix}$ sur un corps $\mathbb{K}$, nous pouvons par exemple considérer l'arbre de calcul suivant avec $\{a, b, c, d\}$ pour segment initial :



1.3 Sémantique d'un arbre de calcul

Soit $e \in \mathbb{A}^n$ une *valeur d'entrée* de l'arbre de calcul T de segment initial $\mathcal{I}$ de longueur n. Nous construisons le chemin T_e dans T ainsi que les fonctions suivantes, en commençant par la racine de T :

– $\alpha : \mathcal{I} \cup \{\text{nœuds de calcul de } T_e\} \to \mathbb{A}$,
– $\beta : \{\text{nœuds de branchement de } T_e\} \to \{\text{faux}, \text{vrai}\}$.

Nous commençons par définir $T_e := \mathcal{I} = (v_1, ..., v_n)$ et $\alpha : \mathcal{I} \to \mathbb{A}$, $\alpha(v_i) := e_i$ pour tout $i \in \{1, ..., n\}$. Par récurrence, supposons le chemin construit jusqu'à un certain nœud w. Si w n'est pas un nœud de sortie alors le chemin se prolonge de la façon suivante :

– si w est un nœud de branchement, alors son successeur est son fils gauche ou droit selon la valeur $\beta(w)$;

– si w est un nœud de calcul, ou si $w = v_n$, alors son successeur v est défini comme étant son fils.

Maintenant le successeur v connu, le chemin est prolongé avec v et les fonctions α et β sont prolongées de la façon suivante :

– si v est un nœud de branchement de la forme $(\rho; u_1, u_2)$, alors $\beta(v) := \rho(\alpha(u_1), \alpha(u_2))$,
– si v est un nœud de calcul de la forme $(\omega; u_1, ..., u_m)$, alors si $\omega(\alpha(u_1), ..., \alpha(u_m))$ est défini nous posons $\alpha(v) := \omega(\alpha(u_1), ..., \alpha(u_m))$, sinon le chemin s'arrête à w.

Si T_e s'arrête sur un nœud de sortie, alors T est dit *exécutable* sur e. Si $(u_1, ..., u_m)$ est un nœud de sortie de v, alors $(\alpha(u_1), ..., \alpha(u_m))$ est appelée la *valeur de sortie* de T sur l'entrée e. La partie de σ qui contient v est appelée la *classe de sortie* de e.

Exemple 1.8. Avec l'arbre de calcul de l'exemple 1.7, avec $\mathbb{A} := \mathbb{Q}$ et l'entrée $e := \begin{pmatrix} a & c \\ b & d \end{pmatrix} := \begin{pmatrix} 1 & 2 \\ 2 & 4 \end{pmatrix}$, le chemin T_e et les valeurs des fonctions α et β sont :

T_e	a	b	c	d	0	1	$a = 0$	$e := b\,c$	$f := e/a$	$f = d$
α	1	2	2	4	0	1		4	4	
β							faux			vrai

La valeur de sortie est $\left\{ \begin{pmatrix} 2 \\ -1 \end{pmatrix} \right\}$.

Si J est un sous-ensemble de $\mathbb{A}^n$, nous dirons que T est exécutable sur J si, et seulement si, T est exécutable sur chaque élément de J. La partition $\sigma = (\sigma_i)_{i \in \{1,...,t\}}$ des nœuds de sortie induit la partition $\pi := \{J_1, ..., J_t\}$ de J définie par :

$$J_i := \{e \in J \mid \text{la classe de sortie de } e \text{ est } \sigma_i\}.$$

Pour chaque i, nous définissons la fonction $\varphi_i : J_i \to \mathbb{A}^{m_i}$ qui retourne la valeur de sortie pour chaque $e \in J_i$. L'unique extension, notée φ, des φ_i définit une fonction sur J qui est appelée la fonction d'évaluation de T sur J.

Définition 1.9. *Une* collection *pour un sous-ensemble J de $\mathbb{A}^n$ est la donnée d'une partition $\pi = \{J_1, ..., J_t\}$ de J et d'une famille de fonctions $\varphi_i : J_i \to \mathbb{A}^{m_i}$ pour chaque $i \in \{1, ..., t\}$. On représente une telle collection par le couple (φ, π) où φ est l'unique fonction qui prolonge les φ_i sur J.*

Un arbre de calcul définit donc une collection naturelle induite par sa fonction d'évaluation.

1.4 Fonctions de coût

Continuons de considérer une structure algébrique munie des opérations élémentaires Ω et des prédicats P. Une fonction $c : \Omega \cup P \to \mathbb{N}$ est appelée une *fonction de coût*. Par exemple nous prendrons dans la suite c constante de valeur 1 : il s'agit de la fonction de *coût total*.

Soit T un arbre de calcul. Le *coût d'un nœud de sortie* v est obtenu en additionnant les coûts des instructions rencontrées sur le chemin partant de la racine et arrivant jusqu'à v. De la sorte, nous construisons une fonction de coût sur l'ensemble des nœuds de sortie de T. Si T est exécutable sur J, la fonction $t : J \to \mathbb{N}$, où $t(e)$ est le coût du nœud de sortie sur lequel se termine le chemin T_e, est appelée la fonction de coût sur J.

Définition 1.10. *Soit (φ, π) une collection associée à J, soit c une fonction de coût, et soit $\tau\colon J \to \mathbb{N}$. On dit que (φ, π) est* calculable *avec un coût τ lorsqu'il existe un arbre de calcul T qui calcule (φ, π) avec un coût borné par τ sur J.*

La complexité dans le pire cas*, notée $C^c(\varphi, \pi)$, pour une collection (φ, π) est définie comme le plus petit entier naturel r tel que (φ, π) est calculable avec le coût $\tau\colon J \to \mathbb{N}$, $a \mapsto r$.*

Nous dirons qu'un arbre calcule une collection (φ, π) en temps $\tau\colon J \to \mathbb{N}$ si, et seulement si, il a un coût au plus τ sur J. Finalement, dans ce formalisme, un *algorithme* est simplement une description en langage naturel de la façon dont on peut construire une famille d'arbres de calcul pour résoudre un problème donné. Lorsque nous parlerons du coût d'un algorithme, il s'agira du coût des arbres de calcul qu'il décrit.

Dès lors que les fonctions de coût deviennent compliquées, il est d'usage de simplifier les expressions en utilisant les notations suivantes :

Définition 1.11. *Si f et g sont deux fonctions de $\mathbb{N}$ dans $\mathbb{R}_+$ définies au voisinage de l'infini, l'écriture $f(n) \in O(g(n))$ signifie qu'il existe deux constantes N et C telles que $f(n) \leqslant C\, g(n)$ pout tout $n \geqslant N$. De plus, nous écrirons $f(n) \in \tilde{O}(g(n))$ lorsque $f(n) \in g(n) \log_2(3 + g(n))^{O(1)}$.*

Exemple 1.12. Si $\alpha > 0$, on a $n \log^\alpha n \log\log n \in \tilde{O}(n)$.

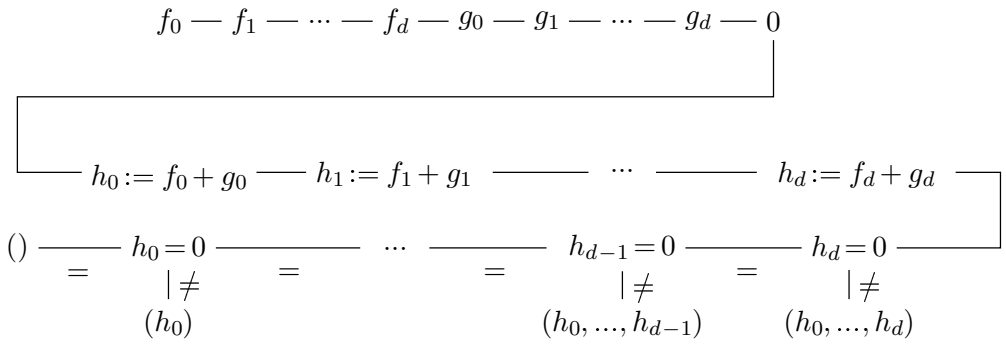
Si $f(n) \in \tilde{O}(n)$, alors f est dite au plus quasi-linéaire ; si $f(n) \in \tilde{O}(n^2)$, alors f est dite au plus quasi-quadratique, etc.

1.5 Opérations élémentaires sur les polynômes

Dans les paragraphes suivants nous illustrons les concepts précédents avec les opérations élémentaires sur les polynômes et matrices. Si $\mathbb{A}$ est un anneau, la *représentation dense* des polynômes de $\mathbb{A}[x]$ consiste à stocker un polynôme de degré d comme le vecteur de ses $d+1$ coefficients du degré 0 au degré d. Si $f = \sum_{i=0}^{d} f_i\, x^i$ est un polynôme de degré d, par convention, nous conviendrons que $f_i = 0$ si $i < 0$ ou $i > d$.

Proposition 1.13. *Il existe une constante σ telle que, pour tout anneau $\mathbb{A}$, il existe une famille d'arbres de calcul $(S_d)_{d\in\mathbb{N}}$ sur $\mathbb{A}$ telle que pour deux polynômes f et g de $\mathbb{A}[x]$ de degré au plus $d \geqslant 0$, l'arbre S_d évalué sur les entrées f et g calcule la somme $f + g$, avec une fonction de coût associée $d \mapsto c(T_d)$ bornée par $\sigma\, d$.*

Démonstration. Nous pouvons prendre l'arbre de calcul suivant pour S_d :



□

Plutôt que de décrire explicitement un arbre de calcul effectuant la somme de deux polynômes, nous préférons la description plus concise et flexible suivante :

Algorithme 1.1

Entrée : $f = \sum_{i=0}^{m} f_i x^i$ et $g = \sum_{i=0}^{n} g_i x^i$.
Sortie : $h := f + g$.

1. Pour i de 0 à $\max(m,n)$ calculer $h_i := f_i + g_i$.
2. Retourner $h = \sum_{i=0}^{\max(m,n)} h_i x^i$.

Aussi, afin d'être plus concis que dans la proposition précédente, nous dirons simplement que l'algorithme 1.1 nécessite au plus $O(\max(m,n))$ opérations d'anneaux. Ce qui sous-entend que la constante cachée dans le O ne dépend pas de l'anneau en question. Remarquons sur cet exemple que le coût de l'algorithme dans le modèle des arbres de calcul ne prend pas en compte les opérations arithmétiques sur le compteur de boucle i, mais seulement le nombre d'opérations effectuées dans l'anneau $\mathbb{A}$.

Proposition 1.14. *Il existe une constante μ telle que, pour tout anneau $\mathbb{A}$, il existe une famille d'arbres de calcul $(M_d)_d$ telle que pour deux polynômes f et g de $\mathbb{A}[x]$ de degré au plus d, l'arbre M_d évalué sur les entrées f et g calcule le produit fg, avec une fonction de coût associée $d \mapsto c(M_d)$ bornée par $\mu\, d^2$.*

Démonstration. Une telle famille d'arbres de calcul peut être construite à partie de l'algorithme naïf de multiplication des polynômes tel que décrit dans l'algorithme 1.2 ci-dessous. □

Algorithme 1.2

Entrée : $f = \sum_{i=0}^{m} f_i x^i$ et $g = \sum_{i=0}^{n} g_i x^i$.
Sortie : $h := fg$.

1. Si $f = 0$ ou $g = 0$ alors retourner $h := 0$.
2. Pour i de 0 à $m+n$ initialiser $h_i := 0$.
3. Pour i de 0 à m, pour j de 0 à n, calculer $h_{i+j} := h_{i+j} + f_i g_j$.

D'une façon concise nous dirons que le produit de deux polynômes de degré d coûte au plus $O(d^2)$ opérations dans l'anneau de base. Rappelons qu'il existe des algorithmes désormais classiques plus rapides, tels que l'algorithme de Karatsuba, conduisant à un coût en $O(d^{\log_2 3})$, ou bien des méthodes utilisant la transformée de Fourier rapide, conduisant à une borne de complexité quasi-linéaire $O(d \log d \log\log d)$. Nous n'entrerons pas dans les détails ici, mais renvoyons le lecteur à des ouvrages généralistes tels que le livre [44] ou bien les notes du cours de Bostan, donné à l'occasion des « Journées Nationales de Calcul Formel » en 2010 [14].

Dans la suite de ce cours nous analyserons fréquemment le coût des algorithmes de factorisation en fonction du coût du produit des polynômes à une variable. Afin d'énoncer les résultats d'une façon flexible, nous utiliserons une fonction de coût $\mathsf{M}\colon \mathbb{N} \to \mathbb{N}$ bornant le coût de l'algorithme choisi pour calculer le produit de deux polynômes à coefficients dans un anneau commutatif unitaire. Pour des raisons techniques, nous supposerons que M satisfait les propriétés suivantes :

– $\mathsf{M}(d)/d$ est croissante,

– $\mathsf{M}(m\,d) \leqslant m^2\,\mathsf{M}(d)$ pour tout $m \geqslant 1$.

Notons que ces conditions sont bien satisfaites pour les bornes des fonctions de coût usuelles. Les algorithmes de factorisation nécessitent de nombreuses opérations élémentaires sur les polynômes, que nous rappelons dans les propositions suivantes, et pour lesquelles nous fournissons des références dans le livre [44], plutôt que les références historiques.

Proposition 1.15. *[44, Chapter 11, Corollary 11.6] Le p.g.c.d. étendu de deux polynômes de degré au plus d sur un corps $\mathbb{K}$ peut se calculer avec $O(\mathsf{M}(d)\log d)$ opérations dans $\mathbb{K}$.*

Corollaire 1.16. *[44, Chapter 11, Corollary 11.8] Le coût d'une opération de corps dans $\mathbb{K}[x]/(q(x))$, où q est un polynôme irréductible de degré d est borné par $O(\mathsf{M}(d)\log d)$.*

Proposition 1.17. *[44, Chapter 11, Corollary 11.15] Le résultant de deux polynômes de degré au plus d sur un corps $\mathbb{K}$ peut se calculer avec $O(\mathsf{M}(d)\log d)$ opérations dans $\mathbb{K}$.*

Proposition 1.18. *[44, Chapter 10, Corollary 10.8] L'évaluation et l'interpolation d'un polynôme de degré au plus d sur un corps $\mathbb{K}$ en d points peut se faire en $O(\mathsf{M}(d)\ \log\ d)$ opérations dans $\mathbb{K}$.*

L'opération suivante est souvent appelée calcul de *réductions simultanées* :

Proposition 1.19. *[44, Chapter 10, Corollary 10.17] Si $g_1, ..., g_r$ sont des polynômes non constants à coefficients dans un corps $\mathbb{K}$ et dont la somme des degrés est au plus d, alors le calcul de $f \bmod g_1, ..., f \bmod g_r$ peut se faire en $O(\mathsf{M}(d)\log d)$ si f est de degré au plus d.*

L'opération inverse est souvent appelée le problème des *restes chinois* :

Proposition 1.20. *[44, Chapter 10, Corollary 10.23] Si $g_1, ..., g_r$ sont des polynômes non constants à coefficients dans un corps $\mathbb{K}$ premiers entre eux deux à deux, et dont la somme des degrés est d, et si $h_1, ..., h_r$ sont des polynômes tels que $\deg(h_i) \leqslant \deg(g_i) - 1$ pour tout $i \in \{1, ..., r\}$, alors le calcul du polynôme f de degré au plus $d-1$ tel que $f \bmod g_1 = h_1, ...,$ $f \bmod g_r = h_r$ peut se faire en $O(\mathsf{M}(d)\log d)$.*

1.6 Opérations élémentaires sur les entiers

Nous supposons tout au long de ce cours que les entiers sont représentés en base 2. La taille d'un entier est définie comme étant le nombre de bits utilisés dans cette représentation. En terme d'arbres de calcul, la structure algébrique sous-jacente est celle de $\mathbb{Z}/2\,\mathbb{Z}$.

Deux entiers de taille n peuvent être multipliés en temps $O(n^2)$ par la méthode naïve. Tout comme pour les polynômes nous serons amenés à analyser le coût de certains algorithmes en fonction du coût des opérations élémentaires sur les entiers. Nous introduisons par conséquent la fonction $\mathsf{I}(n)$ pour borner le coût de l'algorithme choisi pour multiplier deux entiers de taille n. Les meilleurs algorithmes connus à ce jour permettent de prendre $\mathsf{I}(n) = O(n \log n\, 2^{\log^* n})$ [32, 33], où $\log^*$ représente la fonction *logarithme itéré* – précisément $\log^* n$ est zéro si $n \leqslant 1$, et $1 + \log^*(\log n)$ si $n > 1$. Pour des raisons techniques, nous supposerons que I satisfait les propriétés suivantes :

– $\mathsf{I}(n)/n$ est croissante,
– $\mathsf{I}(m\,n) \leqslant m^2\,\mathsf{I}(n)$ pour tout $m \geqslant 1$.

Tout comme pour les polynômes nous rappelons les résultats suivants qui nous serons nécessaires dans la suite :

Proposition 1.21. *[128] Le p.g.c.d de deux entiers de taille au plus n peut se calculer en temps $O(\mathsf{I}(n)\log n)$.*

Corollaire 1.22. *Le coût d'une opération dans le corps fini $\mathbb{F}_p$ est dans $O(\mathsf{I}(\log p)\log\log p) \subseteq \tilde{O}(\log p)$.*

1.7 Opérations élémentaires sur les matrices

Nous supposons dans ce cours qu'une matrice M de taille $l \times c$ est représentée par le vecteur de ses l c coefficients. Nous parlerons ainsi d'une *représentation dense*. La quantité ω représentera un nombre réel dans l'intervalle $]2,3]$ tel que le coût de l'algorithme choisi pour multiplier deux matrices de taille $n \times n$ sur un anneau commutatif coûte $O(n^\omega)$ opérations d'anneau (c'est à dire $+, -, \times$). Il est classique que le coût de nombreuses opérations sur les matrices s'expriment simplement en fonction du coût d'un produit.

Proposition 1.23. *[16, Chapter 16, Theorem 16.7] Le déterminant d'une matrice de taille $n \times n$ sur un corps $\mathbb{K}$ peut être calculé avec $O(n^\omega)$ opérations dans $\mathbb{K}$.*

Proposition 1.24. *[16, Chapter 16, Theorem 16.6] L'inverse d'une matrice inversible de taille $n \times n$ peut être calculé avec $O(n^\omega)$ opérations dans $\mathbb{K}$.*

Définition 1.25. *Une matrice est dite* échelonnée en colonnes *si toutes ses colonnes non nulles sont toutes à gauche de ses colonnes nulles, et si le premier coefficient non nul d'une colonne non nulle, appelé le* pivot *de la colonne, est toujours strictement plus bas que le pivot de la colonne à sa gauche.*

Exemple 1.26. $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 2 & 2 & 0 & 0 \\ 3 & 5 & 0 & 0 \\ 4 & 2 & 3 & 0 \\ 8 & 5 & 2 & 0 \end{pmatrix}$ est échelonnée en colonnes.

Définition 1.27. *Une matrice est dite* échelonnée réduite en colonnes *si elle est échelonnée en colonnes, si les pivots valent* 1 *et si tous les autres coefficients dans la ligne d'un pivot sont nuls.*

Exemple 1.28. $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -2 & 5/2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -11/3 & 11/6 & 2/3 & 0 \end{pmatrix}$ est échelonnée réduite en colonnes.

Proposition 1.29. *[139, cas particulier de la proposition 2.11] Le calcul de la forme échelonnée réduite en colonne d'une matrice de taille $m \times n$ sur $\mathbb{K}$ de rang r coûte $O(m\,n\,r^{\omega-2})$ opérations dans $\mathbb{K}$.*

Une base d'un sous-espace vectoriel de $\mathbb{K}^n$ est dite échelonnée réduite si la matrice formée par les vecteurs de la base mis en colonne est échelonnée réduite.

Corollaire 1.30. *Si $m \geqslant n$, le calcul d'une base échelonnée réduite du noyau d'une matrice de taille $m \times n$ sur un corps $\mathbb{K}$ coûte au plus $O(m\,n^{\omega-1})$ opérations dans $\mathbb{K}$.*

2

Factorisation séparable

Dans ce chapitre $\mathbb{A}$ représente un anneau factoriel, $\mathbb{L}$ son corps des fractions, p sa caractéristique et F un polynôme de $\mathbb{A}[x]$ de degré $d \geqslant 1$. Nous nous intéressons à décomposer F en facteurs dont les racines ont toutes la même multiplicité, en suivant les méthodes de [87].

La *décomposition sans carré* de F, notée $\mathrm{sqr}(F)$ est l'ensemble des paires $(G,\, m)$ où G est le facteur sans carré de F de multiplicité $m \geqslant 1$, c'est-à-dire le produit de tous les facteurs irréductibles de F de multiplicité m. La *décomposition irréductible*, notée $\mathrm{irr}(F)$, est l'ensemble des paires (G, m), où G est un facteur irréductible de F de multiplicité m.

2.1 Séparabilité

Définition 2.1. *$F \in \mathbb{A}[x]$ est dit* séparable *s'il n'a aucune racine multiple dans une clôture algébrique $\bar{\mathbb{L}}$ de $\mathbb{L}$.*

Un polynôme F qui n'est pas constant est séparable si, et seulement si, son discriminant est non nul. Dans le suite nous noterons $\mathrm{res}(F,\, G)$ le *résultant* des polynômes F et G, et $\mathrm{disc}(F) = \mathrm{res}(F,\, F')$ le *discriminant* de F. Notons $\mathcal{B} := \{1, p, p^2, p^3, \ldots\}$ l'ensemble des puissances de p si $p > 0$, et $\mathcal{B} := \{1\}$ si $p = 0$.

Définition 2.2. *Le* degré d'inséparabilité *de F, noté $\deg^i(F)$, est le plus grand élément q de $\mathcal{B}$ tel que $F \in \mathbb{A}[x^q] \setminus \mathbb{A}[x^{pq}]$.*

Proposition 2.3. *Soient F et G deux polynômes primitifs de $\mathbb{A}[x] \setminus \mathbb{A}$.*

a) Le degré d'inséparabilité de F est la plus grande puissance de p qui divise la multiplicité de chaque racine de F.

b) $\deg^i(FG) \geqslant \min(\deg^i(F), \deg^i(G))$, l'inégalité devient une égalité dès lors que F et G sont premiers entre eux.

Démonstration. Si $q \in \mathcal{B}$ divise la multiplicité de chaque racine de F, alors F appartient bien à $\mathbb{A}[x^q]$. Réciproquement, si F peut s'écrire $G(x^q)$ avec $q \in \mathcal{B}$, alors la multiplicité d'une racine de F est bien un multiple de q. La partie (b) est une conséquence directe de la partie (a). □

Définition 2.4. *Le degré de séparabilité de $F \in \mathbb{A}[x]$, noté $\deg^s(F)$, est défini comme suit :*

$$\deg^s(F) := \sum_{(G,m)\in \mathrm{irr}(F)} \deg(G)/\deg^i(G) = \sum_{(G,m)\in \mathrm{irr}(F)} \deg^s(G).$$

Proposition 2.5. *Soient F et G des polynômes primitifs de $\mathbb{A}[x] \setminus \mathbb{A}$.*

a) $\deg^s(F)$ *est égal au nombre de racines de F dans $\bar{\mathbb{L}}$ sans compter les multiplicités.*
b) F est séparable si, et seulement si, $\deg^s(F) = \deg(F)$.
c) Pour tout $q \in \mathcal{B}$, $\deg^s(F(x^q)) = \deg^s(F)$.
d) $\deg^s(FG) \leqslant \deg^s(F) + \deg^s(G)$, *avec égalité si, et seulement si, F et G sont premiers entre eux.*

Démonstration. Ces propriétés découlent presque immédiatement des définitions et nous laissons au lecteur le soin de les vérifier. □

2.2 Déflation

Définition 2.6. *Si $F \in \mathbb{A}[x] \setminus \mathbb{A}$, le* polynôme déflaté *de F est l'unique polynôme $\tilde{F}$ défini par*

$$\tilde{F}(x^{\deg^i(F)}) := F(x).$$

Les multiplicités des racines de $\tilde{F}$ sont celles de F divisées $\deg^i(F)$, d'où la terminologie « déflation ». Puisque $\tilde{F}$ appartient à $\mathbb{A}[x] \setminus \mathbb{A}[x^p]$, la fonction suivante Φ est bien définie :

$$\begin{aligned} \Phi\colon \mathbb{A}[x] \setminus \mathbb{A} &\to (\mathbb{A}[x] \setminus \mathbb{A}[x^p]) \times \mathcal{B} \\ F &\mapsto (\tilde{F}, \deg^i(F)). \end{aligned}$$

Notons $\mathcal{Q}$ l'ensemble des puissances q-ièmes des polynômes irréductibles de $\mathbb{A}[x] \setminus \mathbb{A}$ pour $q \in \mathcal{B}$, et $\mathcal{S}$ l'ensemble des polynômes séparables irréductibles de $\mathbb{A}[x] \setminus \mathbb{A}$.

Proposition 2.7. *Φ est une bijection qui envoie $\mathcal{Q}$ surjectivement sur $\mathcal{S} \times \mathcal{B}$.*

Démonstration. Le fait que Φ est une bijection découle clairement des définitions. Si $F \in \mathbb{A}[x]$ est irréductible, alors $\tilde{F}$ est nécessairement irréductible et donc séparable car $\tilde{F} \notin \mathbb{A}[x^p]$. Puisque le polynôme déflaté de F^q coincide avec celui de $\tilde{F}^q$ pour tout $q \in \mathcal{B}$, le lemme 2.8 ci-dessous implique que $\Phi(\mathcal{Q})$ appartient bien à $\mathcal{S} \times \mathcal{B}$.

Réciproquement, si $(G, q) \in \mathcal{S} \times \mathcal{B}$. Nous devons montrer que $F(x) := G(x^q)$ appartient à $\mathcal{Q}$, et que $\deg^i(F) = q$. Cette dernière égalité est claire puisque G est séparable. Soit $h \leqslant q$ le plus grand élément de $\mathcal{B}$ tel que $G(x^h)$ est une puissance h-ième, et soit $\tilde{G}$ la racine h-ième correspondante, de sorte que $\tilde{G}^h(x) = G(x^h)$. Par le lemme 2.8 ci-dessous, $\tilde{G}$ et donc $\tilde{G}(x^{q/h})$ sont irréductibles et séparables, ce qui prouve que $F(x) = \tilde{G}(x^{q/h})^h \in \mathcal{Q}$. □

Lemme 2.8. *Soient F et G deux polynômes de $\mathbb{A}[x] \setminus \mathbb{A}$ tels que $G(x^p) = F(x)^p$.*

a) G est primitif si, et seulement si, F est primitif.
b) G est séparable si, et seulement si, F est séparable.
c) Si G est irréductible alors F est irréductible. La réciproque est vraie si F ou G est séparable.

Démonstration. La preuve de la partie (a) est immédiate. Les multiplicité de $G(x^p)$ (resp. de $F(x)^p$) sont toutes p si, et seulement si, G (resp. F) est séparable. La partie (b) est donc une conséquence de la proposition 2.5. Concernant la partie (c), si G est irréductible, alors pour n'importe quel facteur irréductible A de F, le polynôme A^p divise F^p, et donc B divise G, si nous définissons B par $B(x^p) = A(x)^p$. Par conséquent B et donc A sont des unités de $\mathbb{A}$. Réciproquement, supposons F irréductible, et soit A un facteur irréductible de G. Puisque $A(x^p)$ divise $F(x)^p$, le polynôme $A(x^p)$ peut s'écrire $F(x)^\alpha$ pour un certain $\alpha \in \{0, \ldots, p-1\}$. En dérivant, nous obtenons $\alpha F' F^{\alpha-1} = 0$, d'où $\alpha = 0$ dès que F est séparable. □

2.3 Décomposition séparable

Définition 2.9. *La* décomposition séparable *d'un polynôme primitif* $F \in \mathbb{A}[x] \setminus \mathbb{A}$, *notée* $\mathrm{sep}(F)$, *est l'ensemble des triplets* $(G_1, q_1, m_1), ..., (G_s, q_s, m_s)$ *de* $\mathbb{A}[x] \times \mathcal{B} \times \mathbb{N}^*$ *vérifiant les propriétés suivantes :*

(S_1) $F(x) = \prod_{i=1}^{s} G_i^{m_i}(x^{q_i})$,
(S_2) *les* $G_i(x^{q_i})$ *sont premiers entre eux deux à deux,*
(S_3) *les* m_i *ne sont pas divisibles par* p,
(S_4) *les* G_i *sont primitifs et separables de degré au moins* 1,
(S_5) *les couples* (q_i, m_i) *sont deux à deux distincts.*

Notons que dans le cas où $p = 0$ la factorisation séparable n'est rien d'autre que la factorisation sans carré.

Théorème 2.10. *Tout polynôme primitif* F *de* $\mathbb{A}[x]$ *admet une unique décomposition séparable (à permutation et unités près dans* $\mathbb{A}$*).*

Démonstration. Soient $(F_1, e_1), ..., (F_r, e_r)$ la décomposition irréductible de F dans $\mathbb{A}[x]$. Pour chaque $i \in \{1, ..., s\}$, soit a_i la plus grande puissance de p qui divise e_i, et soit $m_i := e_i/a_i$. En définissant $(G_i, q_i) := \Phi(F_i^{a_i})$, nous obtenons des triplets

$$(G_1, q_1, m_1), ..., (G_s, q_s, m_s)$$

qui satisfont (S_1), (S_2) et (S_3). La propriété (S_4) est aussi satisfaite par la proposition 2.7. Pour obtenir (S_5) il suffit de regrouper les triplets qui partagent les mêmes deuxièmes et troisièmes coordonnées de la façon suivante : si $(A_1, q, m), ..., (A_t, q, m)$ sont de tels triplets alors nous les fusionnons en le triplet $(A_1 \cdots A_r, q, m)$. Les propriétés de (S_1) à (S_4) sont préservées, ce qui prouve l'existence de la décomposition séparable.

Supposons maintenant que nous disposons de triplets $(G_1, q_1, m_1), ..., (G_s, q_s, m_s)$ qui satisfont les propriétés de (S_1) à (S_5). Pour chaque $i \in \{1, ..., s\}$, n'importe quelle racine de $G_i(x^{q_i})$ dans $\bar{\mathbb{L}}$ admet q_i comme multiplicité, et donc $q_i m_i$ comme multiplicité dans F. La propriété (S_5) implique alors que les racines de $G_i(x^{q_i})$ sont exactement celles de F de multiplicité $q_i m_i$, ce qui conduit à l'unicité de la décomposition séparable. □

Corollaire 2.11. *Si* $\mathbb{A}$ *est contenu dans un anneau factoriel* $\mathbb{B}$, *alors la décomposition séparable de* $F \in \mathbb{A}[x]$ *dans* $\mathbb{A}$ *coïncide avec celle dans* $\mathbb{B}[x]$.

Exemple 2.12. Lorsque $\mathbb{A} := \mathbb{F}_3$ et $F := x^2 (x+1)^3 (x+2)^4 = x^9 + 2x^8 + 2x^3 + x^2$, nous avons $\mathrm{sep}(F) = \{(x, 1, 2), (x+1, 3, 1), (x+2, 1, 4)\}$.

Exemple 2.13. Lorsque $\mathbb{A} := \mathbb{F}_3[t]$ et $F := (x + 2t)^7 (x^3 + 2t)^3 (x^6 + t)$, nous avons $\mathrm{sep}(F) = \{(x + 2t, 1, 7), (x + 2t^3, 9, 1), (x^2 + t, 3, 1)\}$.

Exemple 2.14. La factorisation séparable est implémentée en C++ dans la librairie FACTORIX de MATHEMAGIX. L'affichage par défaut d'un triplet (G, q, m) est $G(x^q)^m$. Les composantes du triplet son accessibles *via* les fonctions `factor`, `ideg` et `mul`.

```
Mmx] use "factorix";
     x == polynomial (0, 1);
Mmx] F == (x^9 + x^3 + 1)^2 * (x^4 - 1)
```

$$x^{22} - x^{18} + 2\,x^{16} + 2\,x^{13} - 2\,x^{12} + x^{10} - 2\,x^9 + 2\,x^7 - x^6 + x^4 - 2\,x^3 - 1$$

```
Mmx] separable_factorization F
```

$$(x^4 - 1)\,(x^9 + x^3 + 1)^2$$

```
Mmx] S == separable_factorization (F mod modulus 3)
```

$$(x^3 + x^2 + x + 1)\,((x^3)^2 + x^3 + 2)^2\,(x + 2)^7$$

```
Mmx] [ [factor g, ideg g, mul g] | g in S ]
```

$$[[x^3 + x^2 + x + 1, 1, 1], [x^2 + x + 2, 3, 2], [x + 2, 1, 7]]$$

La preuve du théorème 2.10 utilise l'existence de la décomposition irréductible. Pour une autre preuve constructive le lecteur pourra consulter celle du livre [96, Chapter VI, Theorem 6.3].

2.4 Algorithme naïf

Nous commençons par présenter un algorithme simple pour calculer la décomposition séparable. Le cœur de celui-ci repose sur le calcul suivant, similaire à celui utilisé pour la factorisation sans carrée dans le cas de la caractéristique zéro :

Algorithme 2.1

Entrée : un polynôme primitif $F \in \mathbb{A}[x]$ de degré $d \geqslant 1$.
Sortie : la liste L des facteurs séparables de F de la forme $(G, 1, m)$, et un polynôme C qui est le produit des autres facteurs séparables de F.

1. Initialiser L avec la liste vide et i à l'entier 1.
2. Calculer $C := \gcd(F, F')$ et $G := F/C$.
3. Tant que $\deg(G) \geqslant 1$ faire :
   a) calculer $P := \gcd(C, G)$ et $C := G/P$,
   b) si $\deg(G) > \deg(P)$ alors adjoindre $(G/P, 1, i)$ à L,
   c) $G := P$, et $i := i + 1$.
4. Retourner L et C.

Proposition 2.15. *L'algorithme 2.1 est correct. Si $\mathbb{A}$ est un corps, alors son coût est borné par $O(d\,\mathsf{M}(d)\log d)$.*

Démonstration. Notons F_0 le facteur de F composé des racines de multiplicité qui ne sont pas multiples de p et notons $F_1 := F/F_0$. Notons $(G_1, 1, m_1), \ldots, (G_s, 1, m_s)$ la décomposition séparable de F_0.

Comme

$$F'(x) = F_1(x) \sum_{i=1}^{s} m_i\,G_i'(x)\,G_i^{m_i-1}(x)\,\frac{F_0(x)}{G_i^{m_i}(x)},$$

alors, à l'étape 2, nous avons $C = F_1 \prod_{i=1}^{s} G_i^{m_i-1}$, et $G = \prod_{i=1}^{s} G_i$. Dans la première étape de la boucle nous obtenons $P = \prod_{i=1,m_i\geqslant 2}^{s} G_i$, $C = F_1 \prod_{i=1,m_i\geqslant 2}^{s} G_i^{m_i-2}$, et G/P correspond au G_i pour lequel $m_i = 1$. À la deuxième étape de la boucle nous avons $P = \prod_{i=1,m_i\geqslant 3}^{s} G_i$, $C = F_1 \prod_{i=1,m_i\geqslant 3}^{s} G_i^{m_i-3}$, et G/P correspond au G_i pour lequel $m_i = 2$. Par récurrence nous déduisons que l'algorithme est correct.

Le nombre d'étapes de l'algorithme est borné par d, le coût de chaque étape est dominé par celui du p.g.c.d. qui est en $O(\mathsf{M}(d)\log d)$ comme rappelé dans la proposition 1.15. □

Finalement la décomposition séparable s'obtient par appels successifs de l'algorithme précédent. L'algorithme suivant a été proposé par Gianni et Trager [46], et est dérivé de l'algorithme attribué à Musser dans le cas de la caractéristique nulle [99].

Algorithme 2.2

Entrée : un polynôme primitif $F \in \mathbb{A}[x]$ de degré $d \geqslant 1$.
Sortie : la décomposition séparable de F.

1. Appeler l'algorithme 2.1 pour obtenir l'ensemble $L = \{(G_1, 1, m_1), ..., (G_r, 1, m_r)\}$ des facteurs séparables de F contenant toutes les racines de F de multiplicité non divisible par p, et le facteur C restant de F.
2. Calculer $\tilde{C}$ défini par $\tilde{C}(x^p) = C(x)$.
3. Appeler récursivement le présent algorithme pour obtenir la décomposition séparable $\tilde{L} = \{(G_{r+1}, q_{r+1}, m_{r+1}), ..., (G_s, q_s, m_s)\}$ de $\tilde{C}$.
4. Retourner $\{(G_1, 1, m_1), ..., (G_r, 1, m_r), (G_{r+1}, p\,q_{r+1}, m_{r+1}), ..., (G_s, p\,q_s, m_s)\}$.

Proposition 2.16. *L'algorithme 2.2 est correct. Si $\mathbb{A}$ est un corps, alors son coût est borné par $O(d\,\mathsf{M}(d)\log d)$.*

Démonstration. Par construction, les racines de C ont une multiplicité divisible par p donc $\tilde{C}$ est bien défini à l'étape 2. Comme $\tilde{L}$ est la décomposition séparable de $\tilde{C}$, alors $\{(G_{r+1}, p\,q_{r+1}, m_{r+1}), ..., (G_s, p\,q_s, m_s)\}$ est la décomposition séparable de $C(x)$. Comme le degré de $\tilde{C}$ est borné par d/p. Le coût total est borné par

$$O\left(\sum_{i\geqslant 1} \frac{d}{p^i}\,\mathsf{M}\left(\frac{d}{p^i}\right)\log\frac{d}{p^i}\right) \subseteq O(d\,\mathsf{M}(d)\log d). \qquad \square$$

2.5 Algorithme rapide

L'algorithme rapide de factorisation séparable repose sur une récurrence différente de l'algorithme précédent.

Lemme 2.17. *Supposons $p > 0$, et soit F un polynôme primitif de $\mathbb{A}[x]$. Il existe des polynômes primitifs S_0 et S_1 de $\mathbb{A}[x]$, uniques à des unités près de $\mathbb{A}$, avec les propriétés suivantes :*

- *les facteurs irréductibles de S_0 sont séparables de multiplicité au plus $p-1$,*
- *$F(x) = S_0(x)\,S_1(x^p)$.*

Démonstration. Notons $\mathrm{irr}(F)$ l'ensemble des couples (G, m) représentant les facteurs irréductibles G de F de multiplicité m. Le polynôme S_0 est défini par sa décomposition en irréductibles :

$$\mathrm{irr}(S_0) := \{(G, m \bmod p) \mid (G, m) \in \mathrm{irr}(F), G \text{ est séparable et } m \bmod p \neq 0\}.$$

Un facteur irréductible G de F/S_0 est soit inséparable soit de multiplicité m dans F divisible par p. Par conséquent $F/S_0 \in \mathbb{A}[x^p]$ et nous définissons $S_1(x^p) := F(x)/S_0(x)$. □

Exemple 2.18. Avec l'exemple 2.12 nous avons $S_0 = (x+2)\,x^2$ et $S_1 = (x+1)\,(x+2)$.

Exemple 2.19. Avec l'exemple 2.13 nous avons $S_0 = x + 2\,t$ et $S_1 = (x^2+t)\,(x+2\,t^3)^2\,(x+2\,t)^3$.

Pour calculer la décomposition séparable de F nous commençons par celle de S_0.

Algorithme 2.3

Entrée : un polynôme primitif $F \in \mathbb{A}[x]$ de degré $d \geqslant 1$.
Sortie : $\mathrm{sqr}(S_0)$ et S_1.

1. Poser $l := 1$ et initialiser L avec la liste vide.
2. Calculer $U := \gcd(F, F')$, $V := F/U$, et $W := F'/U$.
3. Tant que $\deg(V) \geqslant 1$ faire :
   a) Calculer $H := \gcd(V, W - V')$, $W = (W - V')/H$, et $V := V/H$.
   b) Si $\deg(H) \geqslant 1$ alors adjoindre (H, l) à L.
   c) Incrémenter l de 1.
4. Calculer $S_0 := \prod_{(H,l)\in L} H^l$.
5. Calculer F/S_0 et S_1 défini par $S_1(x^p) = (F/S_0)(x)$.
6. Retourner L et S_1.

Lemme 2.20. *L'algorithme 2.3 est correct. Si $\mathbb{A}$ est un corps, alors son coût est borné par $O(\mathsf{M}(d)\log d)$.*

Démonstration. Notons V_l et W_l les valeurs respectives de V et W au départ de l'étape l de la boucle « tant que », et notons H_l la valeur de H calculée durant l'étape l. Notons n la valeur de l à la sortie de la boucle. Définissons aussi V_n et W_n comme les valeurs respectives de V et W à la sortie de la boucle. Nous allons prouver la propriété suivante par récurrence sur l de 1 à n :

$$V_l = \prod_{\substack{(G,m)\in \mathrm{sqr}(S_0)\\ m\geqslant l}} G, \qquad W_l = \sum_{\substack{(G,m)\in \mathrm{sqr}(S_0)\\ m\geqslant l}} (m-l+1)\,\frac{V_l}{G}\,G'.$$

À l'étape 2 nous avons :

$$U = \gcd(S_0, S_0')\,S_1(x^p), \qquad V = \frac{S_0}{\gcd(S_0, S_0')}, \qquad W = \frac{S_0'}{\gcd(S_0, S_0')}.$$

Si E_0 représente la partie sans carré de S_0, alors nous obtenons :

$$\begin{aligned}\gcd(S_0, S_0') &= \gcd\left(\prod_{(G,m)\in \mathrm{sqr}(S_0)} G^m, \sum_{(G,m)\in \mathrm{sqr}(S_0)} m\,\frac{S_0}{G}\,G'\right)\\ &= \gcd\left(E_0, \sum_{(G,m)\in \mathrm{sqr}(S_0)} m\,\frac{E_0}{G}\,G'\right)\prod_{(G,m)\in \mathrm{sqr}(S_0)} G^{m-1}\\ &= \prod_{(G,m)\in \mathrm{sqr}(S_0)} G^{m-1}.\end{aligned}$$

Cette dernière égalité utilise le lemme 2.21 ci-dessous. Par conséquent l'hypothèse de récurrence est satisfaite pour $l = 1$. Supposons l'hypothèse de récurrence satisfaite pour une certaine valeur de l dans $\{1, ..., n-1\}$. En utilisant à nouveau le lemme 2.21 nous déduisons :

$$\begin{aligned} H_l =& \gcd(V_l, W_l - V_l') = \gcd\left(\prod_{\substack{(G,m)\in \mathrm{sqr}(S_0) \\ m \geqslant l}} G, \sum_{\substack{(G,m)\in \mathrm{sqr}(S_0) \\ m \geqslant l}} (m-l)\frac{V_l}{G} G' \right) \\ =& \prod_{\substack{(G,m)\in \mathrm{sqr}(S_0) \\ m = l}} G, \end{aligned}$$

ce qui conduit aux formules attendues pour V_{l+1} et W_{l+1}. Puisque n est le premier entier tel que $\deg(V_n) = 0$, nous obtenons que $n-1$ est la plus grande des multiplicités des facteurs de S_0. Finalement L contient tous les facteurs sans carré de S_0 à la fin de la boucle.

L'étape 2 utilise $O(\mathsf{M}(d)\log d)$ opérations dans $\mathbb{L}$. L'étape l de la boucle coûte au plus $O(\mathsf{M}(\deg(V_l))\log(\deg(V_l)))$ puisque $\deg(W_l) \leqslant \deg(V_l) - 1$. Par la super-additivité de M, l'étape 3 totalise au plus $O\big(\mathsf{M}\big(\sum_{l\geqslant 1}^{n-1} \deg(V_l)\big)\log d\big)$ operations dans $\mathbb{L}$, ce qui conduit à la borne de complexité attendue puisque $\prod_{l=1}^{n-1} V_l = S_0$. À la fin, $O(\mathsf{M}(d))$ opérations supplémentaires suffisent pour déduire S_1. □

Lemme 2.21. *Soient G_1, ..., G_s des polynômes primitifs séparables de $\mathbb{A}[x] \setminus \mathbb{A}$, soit $G := G_1 \cdots G_s$, et soit $(c_1, ..., c_s) \in \mathbb{A}^s$. Si $G_1, ..., G_s$ sont premiers entre eux deux à deux, alors nous avons l'égalité suivante :*

$$\gcd\left(G, \sum_{i=1}^{s} c_i \frac{G}{G_i} G_i'\right) = \prod_{c_i = 0} G_i.$$

Démonstration. La preuve découle du calcul suivant :

$$\gcd\left(G, \sum_{i=1}^{s} c_i \frac{G}{G_i} G_i'\right) = \prod_{j=1}^{s} \gcd\left(G_j, \sum_{i=1}^{s} c_i \frac{G}{G_i} G_i'\right) = \prod_{j=1}^{s} \gcd\left(G_j, c_j \frac{G}{G_j} G_j'\right).$$

Ce dernier p.g.c.d. est égal à G_j si $c_j = 0$ et 1 si $c_j \neq 0$ puisque G_j' est premier avec G_j. □

Pour obtenir la factorisation séparable de F, nous commençons par calculer $\mathrm{sqr}(S_0)$ et S_1 par l'algorithme précédent. Ensuite nous calculons récursivement $\mathrm{sep}(S_1)$, pour finalement fusionner ces décompositions et obtenir $\mathrm{sep}(F)$. Expliquons maintenant comment cette fusion peut être réalisée rapidement. Si

$$\begin{aligned} \mathrm{sqr}(S_0) =& \{(G_1, m_1), ..., (G_r, m_r)\}, \\ \mathrm{seq}(S_1) =& \{(H_1, q_1, n_1), ..., (H_s, q_s, n_s)\}, \end{aligned}$$

alors nous commençons par calculer tous les $U_{i,j} := \gcd(G_i, H_j(x^{pq_j}))$ pour $i \in \{1, ..., r\}$ et $j \in \{1, ..., s\}$. Nous verrons que $(U_{i,j}, 1, m_i + p\, q_j\, n_j)$ est un élément de $\mathrm{sep}(F)$ dès que $\deg(U_{i,j}) \geqslant 1$. Nous aurons aussi besoin de la notation suivante

$$A \operatorname{pmod} B := \mathrm{lc}(B)^{\max(0, \deg(A) - \deg(B))} A \bmod B$$

pour représenter le *pseudo-reste* de la division de A par B. Ici, $\mathrm{lc}(B)$ représente le coefficient de tête de B. Dans l'algorithme suivant nous supposons que la division exacte dans $\mathbb{A}$ est disponible comme opération élémentaire.

Algorithme 2.4

Entrée : un polynôme primitif $F \in \mathbb{A}[x]$ de degré d, $\mathrm{sqr}(S_0)$ et $\mathrm{sep}(S_1)$.
Sortie : $\mathrm{sep}(F)$.

1. Si $S_1 \in \mathbb{A}$ alors retourner $\{(G_1, 1, m_1), ..., (G_r, 1, m_r)\}$.
   Sinon initialiser L avec la liste vide.
2. Pour tout $i \in \{1, ..., r\}$ faire
   a) Pour tout $j \in \{1, ..., s\}$ calculer $R_{i,j}(x) := \tilde{G}_i(x) \bmod H_j(x^{q_j})$, où $\tilde{G}_i$ est le polynôme défini par $\tilde{G}(x^p) = G_i(x)^p$.
   b) Pour tout $j \in \{1, ..., s\}$ calculer $S_{i,j}(x) := \gcd(R_{i,j}(x), H_j(x^{q_j}))$.
   c) Pour tout $j \in \{1, ..., s\}$ calculer $T_{i,j}(x) := G_i(x) \bmod S_{i,j}(x^p)$.
   d) Pour tout $j \in \{1, ..., s\}$ calculer $U_{i,j}(x) := \gcd(T_{i,j}(x), S_{i,j}(x^p))$ et adjoindre $(U_{i,j}, 1, m_i + p\, q_j\, n_j)$ à L si $\deg(U_{i,j}) \geqslant 1$.
   e) Calculer $V_i(x) := \prod_{j=1}^{s} U_{i,j}(x)$.
   f) Calculer $U_{i,0}(x) := G_i(x)/V_i(x)$ et adjoindre $(U_{i,0}, 1, m_i)$ à L si $\deg(U_{i,0}) \geqslant 1$.
3. Pour tout $j \in \{1, ..., s\}$ faire
   a) Calculer $W_j(x^{q_j}) := \prod_{i=1}^{r} S_{i,j}(x)^{q_j}$.
   b) Calculer $U_{0,j}(x) := H_j(x)/W_j(x)$ et adjoindre $(U_{0,j}, p\, q_j, n_j)$ à L si $\deg(U_{0,j}) \geqslant 1$.
4. Retourner L.

Lemme 2.22. *L'Algorithme 2.4 est correct. Si $\mathbb{A}$ est un corps, alors il consomme $O(\mathsf{M}(d) \log d)$ opérations arithmétiques dans $\mathbb{A}$.*

Démonstration. Nous devons vérifier que la liste L retournée satisfait les propriétés (S_1) à (S_5). Pour tout $i \in \{1, ..., r\}$ et tout $j \in \{1, ..., s\}$ nous avons

$$R_{i,j}(x^p) = G_i(x)^p \bmod H_j(x^{pq_j}), \quad S_{i,j}(x^p) = \gcd(G_i(x)^p, H_j(x^{pq_j})),$$

et par conséquent

$$\begin{aligned} U_{i,j}(x) &= \gcd(G_i(x), S_{i,j}(x^p)) = \gcd(G_i(x), \gcd(G_i(x)^p, H_j(x^{pq_j}))) \\ &= \gcd(G_i(x), H_j(x^{pq_j})). \end{aligned}$$

À la fin de l'étape 2, pour tout $i \in \{1, ..., r\}$, il est clair que $G_i = \prod_{j=0}^{s} U_{i,j}$, et que les $U_{i,j}$ sont séparables et premiers entre eux. Dans l'étape 3 nous avons

$$S_{i,j}(x)^{q_j} = \gcd(\tilde{G}_i(x)^{q_j}, H_j(x^{q_j})^{q_j}) = \gcd(\tilde{G}_i(x)^{q_j}, H_j(x^{q_j}))$$

puisque les $\tilde{G}_i$ sont séparables par le lemme 2.8. Par conséquent les $U_{0,j}$ sont bien définis et nous avons

$$W_j(x^{pq_j}) = \prod_{i=1}^{r} S_{i,j}(x^p)^{q_j} = \prod_{i=1}^{r} \gcd(G_i(x)^{pq_j}, H_j(x^{pq_j})) = \prod_{i=1}^{r} U_{i,j}(x)^{pq_j}.$$

Nous en déduisons que $H_j(x^{pq_j}) = U_{0,j}(x^{pq_j}) \prod_{i=1}^{r} U_{i,j}(x)^{pq_j}$ est vrai pour tout $j \in \{1, ..., s\}$. Nous pouvons réécrire F en :

$$\begin{aligned} F(x) &= S_0(x)\, S_1(x^p) = \left(\prod_{i=1}^{r} \prod_{j=0}^{s} U_{i,j}(x)^{m_i} \right) \prod_{j=1}^{s} \left(U_{0,j}(x^{pq_j})^{n_j} \prod_{i=1}^{r} U_{i,j}(x)^{pq_j n_j} \right) \\ &= \left(\prod_{i=1}^{r} \prod_{j=1}^{s} U_{i,j}(x)^{m_i + pq_j n_j} \right) \left(\prod_{i=1}^{r} U_{i,0}(x)^{m_i} \right) \left(\prod_{j=1}^{s} U_{0,j}(x^{pq_j})^{n_j} \right). \end{aligned}$$

La décomposition retournée dans L satisfait bien (S_1). Les autres propriétés sont immédiatement satisfaites par construction.

Si $p \geqslant d+1$ alors $S_1 \in \mathbb{A}$, et l'algorithme ne fait rien. Nous pouvons donc supposer à partir de maintenant que $p \leqslant d$. Le coût suivant est une conséquence directe de la super-additivité de M, de l'inégalité $r \leqslant p-1$, et de $\deg(S_0) + p\deg(S_1) = d$. L'étape a calcule des puissances p-ièmes et des réductions simultanées, totalisant ainsi

$$\begin{aligned} &O\left(\sum_{i=1}^{r} \deg(G_i)\log p + \sum_{i=1}^{r} \left(\mathsf{M}(\deg(S_1))\log s + \mathsf{M}(\max(\deg(G_i), \deg(S_1)))\right)\right) \\ &\quad \subseteq O(\deg(S_0)\log d + p\,\mathsf{M}(S_1)\log d + \mathsf{M}(\deg(S_0))) \subseteq O(\mathsf{M}(d)\log d). \end{aligned}$$

Le coût total des p.g.c.d. dans les étapes b tombe dans

$$\begin{aligned} &O\left(\sum_{i=1}^{r}\sum_{j=1}^{s} \mathsf{M}(\deg(H_j(x^{q_j})))\log(\deg(H_j(x^{q_j})))\right) \\ &\quad \subseteq O(p\,\mathsf{M}(\deg(S_1))\log d) \subseteq O(\mathsf{M}(d)\log d). \end{aligned}$$

Le coût total des réductions simultanées des étapes c tombe dans

$$\begin{aligned} &O\left(\sum_{i=1}^{r}\left(\mathsf{M}(\deg(G_i)) + \mathsf{M}\left(p\sum_{j=1}^{s}\deg(S_{i,j})\right)\log d\right)\right) \\ &\quad \subseteq O(\mathsf{M}(\deg(S_0)) + \mathsf{M}(p\deg(S_1))\log d) \subseteq O(\mathsf{M}(d)\log d). \end{aligned}$$

Le coût total des étapes d est borné par

$$O\left(\sum_{i=1}^{r}\sum_{j=1}^{s} \mathsf{M}(p\deg(S_{i,j}))\log d\right) \subseteq O(\mathsf{M}(d)\log d).$$

Les étapes e et f coûtent

$$O\left(\sum_{i=1}^{r} \mathsf{M}(\deg(G_i))\log d\right) \subseteq O(\mathsf{M}(d)\log d).$$

Finalement le coût de l'étape 3 est au plus

$$O\left(\sum_{j=1}^{s} \mathsf{M}(\deg(H_j(x^{q_j})))\log d\right) \subseteq O(\mathsf{M}(d)\log d). \qquad \square$$

Exemple 2.23. Avec l'exemple 2.12 nous entrons dans l'algorithme 2.4 avec $\mathrm{sqr}(S_0) = \{(x+2, 1), (x, 2)\}$ et $\mathrm{sep}(S_1) = \{((x+1)\,(x+2), 1, 1)\}$, et obtenons les valeurs suivantes pour les $U_{i,j}$:

$i \setminus j$	0	1
0		$x+1$
1	1	$x+2$
2	x	1

Exemple 2.24. Avec l'exemple 2.13 nous entrons dans l'algorithme 2.4 avec $\mathrm{sqr}(S_0) = \{(x + 2\,x, 1)\}$ et $\mathrm{sqr}(S_1) = \{(x^2 + t, 1, 1), (x + 2\,t^3, 1, 2), (x + 2\,t^3, 3, 1)\}$, et obtenons les valeurs suivantes pour les $U_{i,j}$:

$i \setminus j$	0	1	2	3
0		x^2+t	1	$x+2\,t^3$
1	1	1	$x+2\,t$	1

Algorithme 2.5

Entrée : un polynôme primitif $F \in \mathbb{A}[x]$ de degré d.
Sortie : $\mathrm{sep}(F)$.

1. Calculer $\mathrm{sqr}(S_0)$ et S_1 avec l'algorithme 2.3.
2. Utiliser récursivement le présent algorithme pour calculer $\mathrm{sep}(S_1)$.
3. Appeler l'algorithme 2.4 avec $\mathrm{sqr}(S_0)$ et $\mathrm{sep}(S_1)$ de sorte à obtenir $\mathrm{sep}(F)$.

Proposition 2.25. *L'algorithme 2.5 est correct. Si $\mathbb{A}$ est un corps, alors il effectue $O(\mathsf{M}(d) \log d)$ opérations dans $\mathbb{A}$.*

Démonstration. La preuve est une conséquence directe des lemmes 2.20 et 2.22, et de la borne $\sum_{k \geqslant 0} \mathsf{M}(d/p^k) \log (d/p^k) \in O(\mathsf{M}(d) \log d)$. □

L'algorithme rapide de cette section est extrait de [87]. Il peut être vu comme une extension de l'algorithme de Yun [44, Theorem 14.23]. Par ailleurs, lorsque $\mathbb{A}$ est parfait, il coïncide essentiellement avec l'algorithme de décomposition sans carré proposé dans [82, Section 4.6.3, Exercise 36] [44, Exercise 14.30].

2.6 Polynômes à deux variables

Dans le cas où $\mathbb{A} = \mathbb{K}[t]$, avec $\mathbb{K}$ un corps, nous énonçons ici les résultats de complexité qui nous serons utiles par la suite. L'utilisation directe de l'algorithme rapide est assez délicate si l'on souhaite éviter que des polynômes de grands degrés en t interviennent dans les calculs intermédiaires. Une manière efficace pour obtenir la décomposition séparable consiste à calculer les décompositions séparables de $F(a, x)$ pour suffisamment de valeurs de a, à éliminer celles pour lesquelles la spécialisation en $t = a$ ne commute pas avec le calcul de la décomposition séparable de $F(t, x)$, puis à interpoler les facteurs séparables avec les valeurs restantes. Il s'agit d'une technique classique mais dont les détails précis sont assez longs à écrire. Nous ne donnons pas la preuve de la proposition suivante, et renvoyons le lecteur à l'article [87]. Notons d_t (resp. d_x) le degré partiel de F en t (resp. en x).

Proposition 2.26. *[87, Proposition 8] Si $\mathbb{K}$ contient au moins $d_t\,(2\,d_x + 1) + 1$ éléments, alors $\mathrm{sep}(F)$ peut être calculé avec au plus $O(d_x\,(d_x\,\mathsf{M}(d_t) \log d_t + d_t\,\mathsf{M}(d_x) \log d_x)$ ou $\tilde{O}(d_t\,d_x^2)$ opérations dans $\mathbb{K}$.*

L'article [87] contient aussi une variante probabiliste plus rapide, qui conduit à un temps quasi-linéaire en moyenne, mais nous n'aurons pas besoin d'un tel résultat dans la suite. L'approche par évaluation et interpolation pour les polynômes à deux variables est de même nature que celle multi-modulaire conçue par Gerhard lorsque $\mathbb{A} = \mathbb{Z}$ [45].

2.7 Réduction à la factorisation de polynômes séparables

Dans cette section nous expliquons comment le problème de la factorisation irréductible se réduit au cas des polynômes séparables. Nous commençons par le calcul de Φ^{-1}, où Φ est la fonction définie dans la section 2.2. Nous supposons que $\mathbb{A}$ dispose d'une opération élémentaire permettant d'extraire une racine p-ième.

Algorithme 2.6

Entrée : $(G, q) \in \mathcal{S} \times \mathcal{B}$.
Sortie : $(H, h) \in \mathbb{A}[x] \times \mathcal{B}$ avec H irréductible tel que $H^h = \Phi^{-1}(G, q)$.

1. Poser $\tilde{G} := G$ et $h := 1$.
2. Tant que $\tilde{G}(x^p)$ est une puissance p-ième et tant que $h < q$ faire :
   Remplacer $\tilde{G}$ par la racine p-ième de $\tilde{G}(x^p)$ et multiplier h par p.
3. Retourner $(\tilde{G}(x^{q/h}), h)$.

Lemme 2.27. *L'algorithme 2.6 est correct et coûte $O(\deg(G) \log_p q)$ extractions de racines p-ièmes de $\mathbb{A}$.*

Démonstration. Les quantités $\tilde{G}$ est h calculées par l'algorithme coïncident avec celles de la preuve de la proposition 2.7. □

Dans l'algorithme suivant nous supposons disposer d'un algorithme de factorisation irréductible pour les polynômes séparables.

Algorithme 2.7

Entrée : un polynôme primitif $F \in \mathbb{A}[x]$ de degré $d \geqslant 1$.
Sortie : $\mathrm{irr}(F)$.

1. Calculer la décomposition séparable $\mathrm{sep}(F)$ de F.
2. Pour chaque $(G, q, m) \in \mathrm{sep}(F)$ calculer la décomposition irréductible G.
3. Retourner

$$\bigcup_{(G,q,m)\in\mathrm{sep}(F)} \{(H(x^{q/h}), h\,m) |\ H^h := \Phi^{-1}(\bar{G}, q),$$
pour chaque facteur irréductible $\bar{G}$ de $G\}$.

Proposition 2.28. *L'algorithme 2.7 est correct. Si $\mathbb{A}$ est un corps, alors il effectue des factorisations irreducibles de polynômes séparables de $\mathbb{A}[x]$ dont la somme des degrés est au plus d, ainsi que $O(\mathsf{M}(d) \log d)$ opérations arithmétiques dans $\mathbb{A}$ et $O(d)$ extractions de racines p-ièmes dans $\mathbb{A}$.*

Démonstration. Le coût de la première étape est donné par la proposition 2.25 et le coût de la dernière étape par le lemme précédant. □

Il est souvent suggéré, dans des articles en calcul formel, que la factorisation irréductible se réduit à factoriser des polynômes sans carré [9, 25, 41]. Bien que ce point de vue convienne bien au cas de la caractéristique zéro, il conduit souvent à des difficultés techniques en caractéristique positive. En fait si $\mathbb{A} = \mathbb{K}[t]$, avec $\mathbb{K}$ un corps, nous verrons dans le chapitre 5 que nous pouvons réduire la factorisation irréductible dans $\mathbb{A}[x]$ à celle dans $\mathbb{K}[x]$ au moyen de la remontée de Hensel. Si la caractéristique p est nulle ou suffisamment grande et que F est sans carré alors cette réduction commence par choisir un point de spécialisation $a \in \mathbb{K}$ tel que $F(a, x)$ soit sans carré. Maintenant si $p > 0$ cette stratégie ne s'applique plus. Par exemple, considérons $F(t, x) := (x^p + t)\,(x + t^p)$ en prenant pour $\mathbb{K}$ la clôture algébrique de $\mathbb{F}_p$. Il est clair que F est sans carré et que $F(a, x)$ n'est pas sans carré quelle que soit la valeur choisie pour $a \in \mathbb{K}$. Par symmétrie il en va de même si l'on permute t et x. Par ailleurs nous avons $\mathrm{sep}(F) = \{(x + t^p, 1, 1), (x + t, p, 1)\}$, et pour tout $a \in \mathbb{K}$, les polynômes $x + a^p$ et $x + a$ sont séparables.

Dans leur algorithme de factorisation irréductible des polynômes à plusieurs variables sur un corps fini, Bernardin et Monagan [9], résolvent ce problème du choix de la valeur a pour la remontée de Hensel en utilisant la factorisation séparable de manière un peu cachée. L'utilisation explicite de la factorisation séparable comme un prétraitement systématique à la factorisation irréductible est due à Steel [136].

Remarque 2.29. La réduction présentée dans cette section s'adapte au calcul de la décomposition sans carré comme expliqué dans [87].

3

Polynômes à une variable sur un corps fini

Dans ce chapitre nous rappelons et illustrons les résultats classiques pour factoriser les polynômes dans $\mathbb{F}_q[x]$, où $q = p^k$ est une puissance d'un nombre premier p. Grâce aux résultats du chapitre précédent nous supposons dans tout ce chapitre que le polynôme à factoriser est séparable. Les premières techniques connues remontent aux travaux de Gauss (1798), Galois (1830), et Arwins (1918). Les premiers algorithmes performants sont dus à Berlekamp [7, 8], Zassenhaus [151], puis Cantor et Zassenhaus [17]. Ils regroupent les principales idées mathématiques que nous présentons ici.

Exemple 3.1. Plusieurs des algorithmes présentés dans ce chapitres ont été programmés en C++ dans la librairie finitefieldz de Mathemagix.

```
Mmx] use "factorix";
     x == polynomial (0, 1);
     f == (x^9 + x^3 + 1)^2 * (x^4 - 1)
```

$$x^{22} - x^{18} + 2\,x^{16} + 2\,x^{13} - 2\,x^{12} + x^{10} - 2\,x^9 + 2\,x^7 - x^6 + x^4 - 2\,x^3 - 1$$

```
Mmx] f mod modulus 3
```

$$x^{22} + 2\,x^{18} + 2\,x^{16} + 2\,x^{13} + x^{12} + x^{10} + x^9 + 2\,x^7 + 2\,x^6 + x^4 + x^3 + 2$$

```
Mmx] irreducible_factorization (f mod modulus 3)
```

$$(x+1)\,(x^2+1)\,(x^2+x+2)^6\,(x+2)^7$$

3.1 Résidus

Soit $\mathbb{K}$ un corps, soit f un polynôme séparable de degré d dans $\mathbb{K}[x]$, soit $g \in \mathbb{K}[x]$ de degré au plus $d-1$, et soit $\mathbb{F}$ un sous-corps fini de $\mathbb{K}$ de cardinal χ. Notons $\varphi_1, \ldots, \varphi_d$ les racines de f. La dérivée l-ième de $h \in \mathbb{K}[x]$ est notée $h^{(l)}$. Les facteurs irréductibles de f sont notés $f_1, \ldots, f_r$.

Proposition 3.2. *Les trois assertions suivantes sont équivalentes :*

a) $g(\varphi_i)/f'(\varphi_i) \in \mathbb{F}$, *pour tout* $i \in \{1, \ldots, d\}$.
b) $g^\chi - (f')^{\chi-1}\, g \in (f)$ *(méthode de Berlekamp).*
c) $g^\chi + (f^{\chi-1}\, g)^{(\chi-1)} = 0$ *(méthode de Niederreiter).*

Un polynôme $g \in \mathbb{K}[x]$ *de degré au plus* $d-1$ *satisfait l'une de ces conditions si, et seulement si, il est une* $\mathbb{F}$*-combinaison linéaire de* $f_1' \hat{f}_1, \ldots, f_r' \hat{f}_r$, *où* $\hat{f}_i := f/f_i$.

Démonstration. Posons $\rho_i := g(\varphi_i)/f'(\varphi_i)$ pour chaque $i \in \{1, ..., d\}$, de sorte que la décomposition en éléments simples de g/f sur $\overline{\mathbb{K}}$ s'écrive :

$$\frac{g}{f} = \sum_{i=1}^{d} \frac{\rho_i}{x - \varphi_i}. \tag{3.1}$$

Pour chaque $i \in \{1, ..., d\}$, nous avons $(g^\chi - (f')^{\chi-1}\, g)(\varphi_i) = (f')^\chi(\varphi_i)\ (\rho_i^\chi - \rho_i)$, ce qui donne l'équivalence entre (a) et (b). L'équivalence entre (a) et (c) provient du calcul suivant :

$$\begin{aligned} g^p + (f^{p-1}\, g)^{(p-1)} &= g^p + \left(f^p \frac{g}{f} \right)^{(p-1)} = g^p + f^p \left(\frac{g}{f} \right)^{(p-1)} \\ &= f^p \left(\left(\frac{g}{f} \right)^p + \left(\frac{g}{f} \right)^{(p-1)} \right) \\ &= f^p \left(\sum_{i=1}^{d} \frac{\rho_i^p}{(x-\varphi_i)^p} + \sum_{i=1}^{d} \frac{-\rho_i}{(x-\varphi_i)^p} \right), \end{aligned}$$

où la dernière égalité utilise le théorème classique dit de Wilson, qui affirme que p divise $(p-1)! + 1$.

Comme $f' = f'_1\ \hat{f}_1 + \cdots + f'_r\ \hat{f}_r$, si $g = f'_j\ \hat{f}_j$ alors $g(\varphi_i)/f'(\varphi_i)$ vaut 1 lorsque $i = j$ et 0 sinon. Réciproquement, tout $g \in \mathbb{K}[x]_{d-1}$ s'écrit sous la forme $\sum_{j=0}^{d} \gamma_j \frac{f}{(x-\varphi_j)}$. La condition (a) implique que $\frac{\gamma_i \frac{f(x)}{(x-\varphi_i)}(\varphi_i)}{f'(\varphi_i)} = \gamma_i \in \mathbb{F}$ pour tout i. Comme g est à coefficients dans $\mathbb{K}$, les γ_i dont les indices correspondent à des φ_i racines d'un même facteur irréductible de f sont égaux. □

Définition 3.3. *Le polynôme de la k-ième trace sur* $\mathbb{F}_2$ *est* $\mathrm{Tr}_k(x) := x^{2^{k-1}} + x^{2^{k-2}} + \cdots + x^4 + x^2 + x + 1 \in \mathbb{F}_{2^k}[x]$.

Lemme 3.4. *Pour tout* $a \in \mathbb{F}_{2^k}$ *nous avons* $\mathrm{Tr}_k(a) \in \mathbb{F}_2$. *De plus* $\left|\mathrm{Tr}_k^{-1}(0)\right| = \left|\mathrm{Tr}_k^{-1}(1)\right| = 2^{k-1}$.

Démonstration. Comme $x^{2^k} + x = \mathrm{Tr}_k^2(x) + \mathrm{Tr}_k(x) = \mathrm{Tr}_k(x)\ (\mathrm{Tr}_k(x) + 1)$ nous obtenons bien $\mathrm{Tr}_k(a) \in \mathbb{F}_2$. Nous obtenons aussi le fait que Tr_k a toutes ses racines dans $\mathbb{F}_{2^k}$. Comme elles sont au nombre de 2^{k-1} nous en déduisons la dernière assertion. □

Si $\mathbb{K}$ est le corps fini $\mathbb{F}_q$ à q éléments alors nous pouvons déjà montrer que la connaissance d'une base de $\left\langle f'_1\ \hat{f}_1, ..., f'_r\ \hat{f}_r \right\rangle_{\mathbb{F}_q}$ permet de déduire la décomposition irréductible de f, à l'aide de l'algorithme suivant :

Algorithme 3.1

Entrée : $f \in \mathbb{F}_q[x]$ séparable, et une base $g_1, ..., g_r$ de $\left\langle f'_1\ \hat{f}_1, ..., f'_r\ \hat{f}_r \right\rangle_{\mathbb{F}_q}$.
Sortie : les facteurs irréductibles $f_1, ..., f_r$ de f.

1. Si $r = 1$ alors retourner f.
2. Choisir des éléments $c_1, ..., c_r$ dans $\mathbb{F}_q$ à l'aide d'un générateur aléatoire, et calculer $g := c_1\, g_1 + \cdots + c_r\, g_r$.
3. Si $h_1 := \gcd(f, g)$ n'est pas constant alors aller à l'étape 6.
4. Calculer $h := (g/f')^{\frac{q-1}{2}} \bmod f$ si q est impair, et $h := \mathrm{Tr}_k(g/f') \bmod f$ sinon.
5. Si $h_1 := \gcd(f, h - 1)$ est constant alors retourner à l'étape 2.
6. Calculer $h_2 := f/h_1$.

7. Calculer une base $g_{1,1}, ..., g_{r_1,1}$ des $g_i/h_2 \bmod h_1$ pour $i \in \{1, ..., r\}$.
8. Calculer une base $g_{1,2}, ..., g_{r_2,2}$ des $g_i/h_1 \bmod h_2$ pour $i \in \{1, ..., r\}$.
9. Retourner la réunion des facteurs obtenus par l'appel récursif avec $h_1, g_{1,1}, ..., g_{r_1,1}$, d'une part, puis $h_2, g_{1,2}, ..., g_{r_2,2}$ d'autre part.

Les générateurs de nombres aléatoires ne sont pas prévus dans les arbres de calculs tels que définis au chapitre 1. Lorsqu'un tel générateur intervient alors nous considérons chaque variable aléatoire comme entrée de l'arbre de calcul. Pour chaque jeu de valeurs de ces entrées nous pouvons considérer alors le coût de l'arbre. Le *coût moyen* est simplement défini comme la moyenne de ces coûts sur l'ensemble des valeurs des variables aléatoires.

Proposition 3.5. *L'algorithme 3.1 est correct et nécessite au plus* $O(d\ r^{\omega-1} \log r + (r + \log d + \log q)\, \mathsf{M}(d) \log r)$ *opérations dans* $\mathbb{F}_q$ *en moyenne.*

Démonstration. Si $\gcd(f, g)$ à l'étape 2 n'est pas constant alors c'est un facteur propre de f. Sinon les résidus de g/f sont tous non nuls. Lorsque q est impair, modulo chaque f_i, la quantité g/f' tombe alors dans $\mathbb{F}_q$ et par conséquent la classe de $(g/f')^{\frac{q-1}{2}}$ est -1 ou 1. Les facteurs irréductibles de h_1 sont les f_i de f tel que cette dernière classe soit 1. La distribution de ces -1 et 1 se fait de façon aléatoire dès lors que celle des c_i l'est, ce qui montre déjà que la profondeur moyenne des appels récursifs est dans $O(\log r)$. Si $g = \rho_1\, f_1'\, \hat{f}_1 + \cdots + \rho_r\, f_r'\, \hat{f}_r$, et $h_1 = f_1 \cdots f_l$ alors $g \bmod h_1 = h_2 \left(\rho_1\, f_1'\, \hat{f}_1 + \cdots + \rho_l\, f_l'\, \hat{f}_l\right)$. Donc les $g_{i,1}$ forment bien un ensemble générateur de $\left\langle f_1'\, \hat{f}_1, ..., f_l'\, \hat{f}_l \right\rangle_{\mathbb{F}_q}$. Il en va de même concernant h_2. L'algorithme fonctionne donc correctement si q est impair. Si q est pair (et donc une puissance de 2), alors les arguments sont similaires en utilisant le lemme précédent.

L'étape 2 coûte $O(r\, d)$ opérations. L'étape 4 coûte $O((\log d + \log q)\, \mathsf{M}(d))$ opérations. Les étapes 3, 5, et 6 utilisent $O(\mathsf{M}(d) \log d)$ opérations. Les étapes 7 et 8 nécessitent $O(d\, r^{\omega-1})$ opérations. □

3.2 Algorithme de Berlekamp

Avec la propriété (b) de la proposition 3.2, en prenant pour $\mathbb{K}$ le corps fini $\mathbb{F}_q$ et pour $\mathbb{F} := \mathbb{K}$, nous obtenons l'algorithme de Berlekamp :

Algorithme 3.2

Entrée : $f \in \mathbb{F}_q[x]$ séparable de degré $d \geqslant 1$
Sortie : les facteurs irréductibles $f_1, ..., f_r$ de f.

1. Calculer, dans la base $1, x, ..., x^{d-1}$, la matrice B de l'endomorphisme de Frobenius $\alpha \mapsto \alpha^q$ dans $\mathbb{F}_q[x]/(f)$.
2. Calculer une base $\rho_1, ..., \rho_r$ du noyau de B.
3. Calculer $g_i = \rho_i\, f' \bmod f$ pour $i \in \{1, ..., r\}$.
4. Appeler l'algorithme 3.1 avec $f, g_1, ..., g_r$ et retourner la factorisation obtenue.

Proposition 3.6. *L'algorithme 3.2 est correct et utilise en moyenne* $O(d^\omega \log d + (d + \log q)\, \mathsf{M}(d) \log d)$ *opérations dans* $\mathbb{F}_q$.

Démonstration. Le fait que l'algorithme est correct provient des propositions 3.2 et 3.5. Le coût de la première étape est dans $O((d + \log q)\, \mathsf{M}(d))$. L'étape 2 effectue $O(d^\omega)$ opérations. Le reste du coût provient essentiellement de la proposition précédente. □

Remarque 3.7. L'algorithme de Berlekamp est implémenté dans Mathemagix dans `finitefieldz/berlekamp.hpp`.

3.3 Algorithme de Niederreiter

L'algorithme de Niederreiter consiste essentiellement à remplacer dans l'algorithme de Berlekamp le calcul du noyau de l'endomorphisme de Frobenius par le noyau de l'application suivante, grâce à la proposition 3.2 : $g \mapsto g^q + (f^{q-1}\, g)^{(q-1)}$. Lorsque $q = 2$, la matrice correspondante est un peu plus rapide à calculer. Pour plus de détails nous renvoyons le lecteurs aux articles [104, 105]. Le reste de cette sous-section peut être omis en première lecture. Nous explicitons le lien entre les matrices de Berlekamp et Niederreiter et examinons la complexité lorsque $q = p$ dans un cadre plus général qui nous sera utile dans le chapitre 5.

Soit $\mathbb{A}$ un anneau commutatif de caractéristique p, et soit f un polynôme de $\mathbb{A}[x]$ de degré d. Nous étudions ici le coût du calcul de $g^p + (f^{p-1}\, g)^{(p-1)}$ pour $g \in \mathbb{A}[x]_{d-1}$. Dans cette dernière expression la partie la plus difficile est $(f^{p-1}\, g)^{(p-1)}$. Nous nous concentrons sur la construction de la matrice N de taille $d \times d$ de l'application suivante :

$$\begin{aligned} \mathbb{A}[x]_{d-1} &\to \mathbb{A}[x^p]_{d-1} \\ g &\mapsto (f^{p-1}\, g)^{(p-1)}, \end{aligned}$$

où $\mathbb{A}[x^p]_{d-1}$ représente l'espace des polynômes en x^p de degré au plus $d-1$ en x^p.

Proposition 3.8. *Si f est unitaire, alors N peut-être calculée avec $O((d + \log p)\, \mathsf{M}(d))$ opérations dans $\mathbb{A}$.*

Démonstration. Notons $\mathrm{coeff}(A, x^i)$ le coefficient de x^i de $A \in \mathbb{A}[x]$. Nous souhaitons calculer $-\mathrm{coeff}\,(f^{p-1}\, g, x^{ip+p-1})$ pour tout $i \in \{0, ..., d-1\}$. Notons $\mathrm{rev}\,(n, .)$ l'opération consistant à prendre le polynôme aux inverses des racines, c'est à dire $\mathrm{rev}\,(n, A) = \sum_{k=0}^{n} a_{n-k}\, x^k$, si $A := \sum_{k=0}^{n} a_k \in \mathbb{A}[x]_n$. En fait nous allons calculer

$$-\mathrm{coeff}(\mathrm{rev}(d\,p-1,\, f^{p-1}\, g), x^{ip}), \text{ pour tout } i \in \{0, ..., d-1\}.$$

Puisque f est unitaire de degré d, le terme constant de $\mathrm{rev}(d, f)$ est 1. Par conséquent nous pouvons définir $\sum_{i \geqslant 0} u_i\, x^i$ comme le développement en séries de $\mathrm{rev}(d-1, g)/\mathrm{rev}(d, f)$, de sorte que :

$$\begin{aligned} \mathrm{rev}(d\,p-1,\, f^{p-1}\, g) &= \mathrm{rev}(d, f)^{p-1}\, \mathrm{rev}(d-1, g) \\ &= \mathrm{rev}(d, f)^{p}\, \mathrm{rev}(d-1, g)/\mathrm{rev}(d, f) \\ &= \mathrm{rev}(d, f)^{p} \sum_{i \geqslant 0} u_i\, x^i. \end{aligned}$$

Nous sommes ramenés au calcul de u_0, u_p, ..., $u_{(d-1)p}$. Le calcul de $\mathrm{rev}(d,\ f)^p$ utilise $O(d \log p)$ opérations dans $\mathbb{A}$, et le produit

$$\mathrm{rev}(d, f)^{p} \sum_{i=0}^{d-1} u_{ip}\, x^{ip}$$

coûte $O(\mathsf{M}(d))$. Pour calculer $u_0, u_p, ..., u_{(d-1)p}$, nous utilisons le fait que $(u_i)_{i \in \mathbb{N}}$ satisfait la récurrence linéaire suivante :

$$u_i := -(f_{d-1}\, u_{i-1} + \cdots + f_0\, u_{i-d}), \text{ pour } i \geqslant d,$$

où $f_j := \text{coeff}(f, x^j)$. Pour sûr les valeurs initiales $u_0, ..., u_{d-1}$ peuvent être calculées à partir de $\text{rev}(d-1, g)/\text{rev}(d, f)$ à la précision (x^d) avec $O(\mathsf{M}(d))$ opérations dans $\mathbb{A}$.

Pour tout $i \geqslant 0$, si $a_{d-1}\, x^{d-1} + \cdots + a_1\, x + a_0$ représente $\text{rem}(x^i,\ f)$, alors $u_i = a_0\, u_0 + \cdots + a_{d-1}\, u_{d-1}$. Soit B la matrice de Berlekamp, de taille $d \times d$ dont la i-ième colonne est $\text{rem}\,(x^{p(i-1)}, f)$ exprimé dans la base $1, x, ..., x^{d-1}$. La construction de B peut se faire en $O(\mathsf{M}(d)\,(d + \log p))$. Finalement nous venons de montrer que la matrice N se factorise en $N = H_2\, B^t\, H_1$, où H_1, H_2 sont les matrices de Hankel des applications suivantes :

$$\begin{aligned} H_1 &: \mathbb{A}[x]_{d-1} \to \mathbb{A}^d \\ &\quad g \mapsto (u_0, ..., u_{d-1}), \\ H_2 &: \mathbb{A}^d \to \mathbb{A}[x^p]_{d-1} \\ &\quad (u_0, ..., u_{(d-1)p}) \mapsto -\text{rev}\left(d\,p - 1, \left[\text{rev}(d, f)^p \sum_{i=0}^{d-1} u_{ip}\, x^{ip} \right]_0^{dp} \right), \end{aligned}$$

où $\left[\sum_{i \geqslant 0} a_i\, x^i\right]_0^l := \sum_{0 \leqslant i \leqslant l-1} a_i\, x^i$.

Puisque H_1 est symétrique, $(B^t\, H_1)^t = H_1\, B$ peut être calculée avec $O(d\, \mathsf{M}(d))$ opérations dans $\mathbb{A}$. Le deuxième produit $H_2\,(B^t\, H_1)$ coûte aussi $O(d\, \mathsf{M}(d))$. □

La décomposition $N = H_2\, B^t\, H_1$ obtenue dans cette preuve provient de [105, Theorem 1]. Supposons à partir de maintenant que $\mathbb{A} := \mathbb{K}[z]/(q(z))$, où q est un polynôme non constant, unitaire et séparable de $\mathbb{K}[z]$ de degré n. Soient $g_1, ..., g_t$ des polynômes de $\mathbb{A}[x]_{d-1}$ avec $1 \leqslant t \leqslant d$.

Proposition 3.9. *Le calcul de* $g_1^p + (f^{p-1}\, g_1)^{(p-1)}, ..., g_t^p + (f^{p-1}\, g_t)^{(p-1)}$ *peut être effectué avec*

$$O(\mathsf{M}(n)\,(\mathsf{M}(d)\,(d + \log p) + d^2\, t^{\omega-2} + t\, d\,(\log n + \log p)))$$

opérations dans $\mathbb{K}$.

Démonstration. Puisque p est la caractéristique, le calcul de $g_1^p, ..., g_t^p$ requiert $O(t\, d \log p)$ opérations dans $\mathbb{A}$, ce qui totalise $O(t\, d\, \mathsf{M}(n) \log p)$ opérations dans $\mathbb{K}$. Nous pouvons donc nous concentrer sur le calcul de $(f^{p-1}\, g_1)^{(p-1)}, ..., (f^{p-1}\, g_t)^{(p-1)}$. Si f est unitaire alors ce calcul peut se faire à l'aide de la matrice N de la proposition 3.8 qui peut être construite en $O(\mathsf{M}(d)\,(d + \log p))$ opérations dans $\mathbb{A}$: il suffit en effet de calculer le produit de N avec la matrice de taille $d \times t$ dont les colonnes sont les coefficients des g_i. Ce produit prend $O(d^2\, t^{\omega-2})$ opérations dans $\mathbb{A}$.

Si f n'est pas unitaire, $\mathbb{A}$ peut se décomposer en $\mathbb{A}_1 \times \cdots \times \mathbb{A}_r$, de sorte que f puisse être rendu unitaire dans chaque $\mathbb{A}_i$. Chaque $\mathbb{A}_i$ est de la forme $\mathbb{A}_i := \mathbb{K}[z]/(q_i(z))$, où q_i est unitaire de degré $n_i \geqslant 1$. Nous effectuons les calculs dans chaque $\mathbb{A}_i$ et reconstruisons le résultat souhaité *via* la méthode des restes chinois. Une telle décomposition est obtenue comme suit. Pour tout $i \in \{0, ..., d\}$, écrivons f_i pour la préimage dans $\mathbb{K}[z]$ du coefficient de x^i dans f. Nous prenons $\mathbb{A}_1 := \mathbb{K}[z]/(q_1(z))$ de sorte que la projection de f dans $\mathbb{A}_1[x]$ a un coefficient de tête inversible : en fait q_1 est obtenu comme $q_1 := q/\gcd\,(f_d, q)$. Puisque q est séparable, q/q_1 est premier avec q_1. Soit $k_1 \leqslant d - 1$ le plus grand entier tel que f_{k_1} est non nul modulo q/q_1. Cet entier peut être déterminé au moyen de $(d - k_1)\ \mathsf{M}(n)$ opérations dans $\mathbb{K}$. À partir de f_{k_1} et q/q_1 nous calculons q_2 se sorte que f_{k_1} soit inversible dans $\mathbb{A}_2 := \mathbb{K}[z]/(q_2(z))$ et se réduise à zéro modulo $q/(q_1\, q_2)$. En itérant ce processus nous obtenons la décomposition souhaitée au moyen d'un nombre total de $O(d\, \mathsf{M}(n) \log n)$ opérations dans $\mathbb{K}$.

Par la proposition 1.19 chaque élément de $\mathbb{A}$ peut être projeté dans $\mathbb{A}_1 \times \cdots \times \mathbb{A}_r$ en utilisant $O(\mathsf{M}(n)\log n)$ opérations dans $\mathbb{K}$. Par conséquent toutes les projections de f et des g_i dans $\mathbb{A}_1[x] \times \cdots \times \mathbb{A}_r[x]$ totalisent $O(t\,d\,\mathsf{M}(n)\log n)$ opérations dans $\mathbb{K}$. Fixons $i \in \{1, ..., r\}$ et analysons le coût du calcul de

$$(f^{p-1}\,g_1)^{(p-1)}, ..., (f^{p-1}\,g_t)^{(p-1)}$$

dans $\mathbb{A}_i[x]$. Pour chaque $j \in \{1, ..., t\}$ nous introduisons π_j et ρ_j pour les quotient et reste de la division de g_j par f dans $\mathbb{A}_i[x]$, de sorte que $g_j = \pi_j\,f + \sigma_j$ soit vrai dans $\mathbb{A}_i$. Par construction, les divisions sont bien définies puisque le coefficient de tête de f est inversible dans $\mathbb{A}_i$. Cette inversion coûte $O(\mathsf{M}(n_i)\log n_i)$ opérations dans $\mathbb{K}$. Ensuite le calcul de tous les π_j et σ_j totalise $O(t\,\mathsf{M}(d))$ opérations dans $\mathbb{A}_i$. Pour chaque j nous devons calculer

$$(f^{p-1}\,g_j)^{(p-1)} = (f^p\,\pi_j)^{(p-1)} + (f^{p-1}\,\sigma_j)^{(p-1)} = f^p\,\pi_j^{(p-1)} + (f^{p-1}\,\sigma_j)^{(p-1)}.$$

Le calcul de tous les $(f^{p-1}\,\sigma_j)^{(p-1)}$ coûte $O(d^2\,t^{\omega-2} + \mathsf{M}(d)\,(d + \log p))$ opérations dans $\mathbb{A}_i$. Calculer tous les $f^p\,\pi_j^{(p-1)}$ coûte $O(d\log p + t\,\mathsf{M}(d))$ opérations dans $\mathbb{A}_i$. Le calcul de $(f^{p-1}\,g_1)^{(p-1)}, ..., (f^{p-1}\,g_s)^{(p-1)}$ nécessite au total $O(d^2\,t^{\omega-2} + \mathsf{M}(d)\,(d + \log p))$ opérations dans $\mathbb{A}_i$ plus $O(\mathsf{M}(n_i)\log n_i)$ opérations dans $\mathbb{K}$.

La somme de tous ces coûts pour chaque $i \in \{1, ..., r\}$ conduit à $O(\mathsf{M}(n)\,(\mathsf{M}(d)\,(d + \log p) + d^2\,t^{\omega-2} + \log n))$ grâce à la super-additivité de M. Finalement, tous les $(f^{p-1}\,g_i)^{(p-1)}$ sont remontés dans $\mathbb{A}[x]$ au moyen de $O(t\,d\,\mathsf{M}(n)\log n)$ opérations dans $\mathbb{K}$. □

3.4 Algorithme de Cantor-Zassenhaus

Une composante commune à de nombreux algorithmes de factorisation sur les corps finis est la suite des itérés de Frobenius $(\Phi_i)_{i\geqslant 0}$, relative à f définie par $\Phi_i = x^{q^i} \bmod f$. D'une façon naïve Φ_0, ..., Φ_d peuvent être calculés en utilisant $O(d\,\mathsf{M}(d)\log q)$ opérations dans $\mathbb{F}_q$. Néanmoins il est pertinent de considérer l'opération de composition modulaire définie comme le calcul de $g \circ h \bmod f$ pour g et h deux polynômes de degré au plus $d-1$. Nous pouvons calculer Φ_1 en $O(\mathsf{M}(d)\log q)$ opérations puis obtenir Φ_{i+1} comme $\Phi_1 \circ \Phi_i \bmod f$. Il reste alors à s'intéresser au coût de la composition modulaire qui peut être rendu indépendant de q. Néanmoins, si l'on souhaite calculer les $d+1$ premiers itérés, il existe une technique spécifique exploitant l'évaluation et interpolation rapide d'un polynôme en plusieurs points, et donnant le résultat suivant :

Proposition 3.10. *Avec les notations précédentes, une fois Φ_1 connu, $\Phi_0, ..., \Phi_l$, pour $l \leqslant d$, peuvent être calculés avec $O(\mathsf{M}(d)^2 \log d \log l)$ opérations dans $\mathbb{F}_q$.*

Démonstration. L'évaluation de Φ_i en $\Phi_1, ..., \Phi_i$ modulo f donne $\Phi_{i+1}, ..., \Phi_{2i}$. Le coût total découle alors de la proposition 1.18. □

L'algorithme de Cantor-Zassenhaus comporte deux étapes. La première consiste à décomposer f en un produit de polynômes $\prod_{i\geqslant 1} e_i$ où e_i ne comporte que des facteurs irréductibles de degré i. Nous parlons alors de *décomposition par degré* des facteurs. La deuxième étape consiste à factoriser chaque e_i en irréductibles.

Algorithme 3.3

Entrée : f de degré $d \geqslant 1$.
Sortie : la décomposition $e_1, ..., e_d$ par degré des facteurs irréductibles de f.

1. Pour chaque i de 1 à d :
   calculer $e_i := \gcd(\Phi_i - x, f)$ puis remplacer f par f/e_i.
2. Retourner $e_1, ..., e_d$.

Proposition 3.11. *L'algorithme 3.3 est correct et nécessite* $O(O(\mathsf{M}(d)\,(\mathsf{M}(d)\log^2 d + \log q))$ *opérations dans* $\mathbb{F}_q$.

Démonstration. La formule utilisée pour calculer e_i repose sur le fait classique de la théorie des corps finis que $x^{q^i} - x$ est le produit de tous les facteurs irréductibles unitaires de $\mathbb{F}_q[x]$ dont le degré divise i. Par la proposition précédente nous obtenons un coût total en $O(\mathsf{M}(d)\log q + \mathsf{M}(d)^2\log^2 d + d\,\mathsf{M}(d)\log d)$. □

Notons que l'algorithme précédent ne fait pas intervenir de valeurs aléatoires. L'étape suivante consiste à supposer que f est le produit de r polynômes irréductibles de degré d/r.

Algorithme 3.4

Entrée : f séparable ayant tous ses facteurs irréductibles de degré d/r.
Sortie : la décomposition en irréductibles $f_1, ..., f_r$.

1. Si $r = 1$ alors retourner f.
2. Répéter :
   a) Choisir un polynôme h de degré au plus $d-1$ d'une façon aléatoire.
   b) Calculer $f_1 := \gcd(f, h)$. Si f_1 n'est pas constant alors aller à l'étape 3.
   c) Calculer $g := h^{\frac{q^{d/r}-1}{2}} \bmod f$ si q est impair et $g := \mathrm{Tr}_{kd/r}(h) \bmod f$ si q est pair.
   d) Calculer $f_1 := \gcd(f, g-1)$.
   e) Si f_1 n'est pas constant alors aller à l'étape 3.
3. Calculer $f_2 := f/f_1$ et appeler récursivement la fonction sur f_1 et f_2 et retourner la réunion des facteurs.

Proposition 3.12. *L'algorithme 3.4 est correct et effectue* $O(d\,\mathsf{M}(d)\log q \log d)$ *opérations dans* $\mathbb{F}_q$. *Lorsque* q *est impair le coût tombe dans* $O(\mathsf{M}(d)^2\log^3 d + \mathsf{M}(d)\log q \log d)$.

Démonstration. Le choix aléatoire d'un polynôme h conduit à des choix aléatoires indépendants des polynômes $h \bmod f_i$ pour $i \in \{1, ..., r\}$. Comme f_i est de degré d/r alors le polynôme $g \bmod f_i$ de l'étape 2 est -1 ou 1 (resp. 0 ou 1) lorsque q est impair (resp. pair *via* le lemme 3.4). Les -1 et 1 (resp. 0 ou 1) se distribuent de façon aléatoire. La profondeur moyenne des appels récursifs est par conséquent dans $O(\log d)$. Pour les détails nous renvoyons le lecteur à l'article [29].

Le coût du calcul de g à l'étape 2 peut se faire en $O(d\,\mathsf{M}(d)\log q)$. Les autres opérations totalisent un coût dans $O(\mathsf{M}(d)\log d)$. Lorsque q est impair il est intéressant de procéder comme suit. En notant $n := d/r$ et $\alpha := h \bmod f$, comme

$$\frac{q^n - 1}{2} = (q^{n-1} + q^{n-2} + \cdots + q + 1)\,\frac{q-1}{2},$$

nous pouvons calculer $\alpha^{\frac{q^n-1}{2}} = \left(\alpha^{q^{n-1}} \cdots \alpha^q \alpha\right)^{\frac{q-1}{2}}$ grâce à la proposition 3.10. □

Théorème 3.13. *Si q est impair, un polynôme de degré d dans $\mathbb{F}_q[x]$ peut être factorisé avec un coût moyen dans $\tilde{O}(d^2 + d \log q)$.*

Démonstration. Il s'agit d'une conséquence des propositions 3.11 et 3.12. □

Remarque 3.14. La présentation choisie ici est inspirée de [44, Chapter 14]. Le calcul des itérés de Frobenius constitue une brique importante en terme des performances pratiques. L'algorithme de Cantor-Zassenhaus est implémenté dans Mathemagix dans `finitefieldz/cantor_zassenhaus.hpp`. Aussi, d'un point de vue pratique, dans nos implémentations nous avons choisi d'utiliser des générateurs pseudo-aléatoires se comportant comme des itérateurs parcourant tous les choix possibles dans chaque situation. De la sorte, nous sommes certains que nos algorithmes se terminent avec un résultat correct.

3.5 Algorithmes plus rapides

Les algorithmes précédents peuvent être améliorés et adaptés pour la recherche des racines, pour tester l'irréductibilité, ou bien encore pour construire des polynômes irréductibles de degré donné. Encore une fois nous renvoyons le lecteur vers [44, Chapter 14]. Pour d'avantage de résultats sur les polynômes à coefficients dans un corps fini, nous invitons le lecteur à consulter le livre à paraître [97].

Dans l'article [73], Kaltofen et Shoup ont relié le problème de factorisation au problème de composition modulaire et ont obtenu une borne de complexité dans $O(d^{1,815} \log q)$ en terme d'opérations dans $\mathbb{F}_q$. Sur un corps fini, Kedlaya et Umans ont récemment conçu un algorithme avec un coût quasi-linéaire [77], permettant de déduire de [73] un algorithme de factorisation sur $\mathbb{F}_q[x]$ ayant un coût binaire moyen dans $\tilde{O}((d^{1,5} + d \log q) \log q)$, ce qui représente la meilleure borne de complexité à notre connaissance.

Défi 3.1. Implémenter un algorithme de composition modulaire de sorte à observer un temps quasi-linéaire et à être plus efficace que les autres implémentations.

Problème ouvert 3.1. Existe-il un algorithme de composition modulaire de coût quasi-linéaire pour tout corps $\mathbb{K}$?

Problème ouvert 3.2. Existe-il un algorithme de factorisation dans $\mathbb{F}_q[x]$ en temps polynomial ?

4

Polynômes à une variable sur les nombres rationnels

Dans ce chapitre nous présentons les principales techniques connues pour factoriser un polynôme à une variable à coefficients dans $\mathbb{Q}$. Nous présentons brièvement une approche simple mais de coût exponentiel, puis les deux approches historiques conduisant à des temps polynomiaux en utilisant l'algorithme LLL. Enfin nous décrivons une méthode plus récente offrant de meilleurs performances pratiques.

Nous examinons aussi la factorisation dans $\mathbb{K}(\alpha)[x]$ en fonction de celle dans $\mathbb{K}[x]$, où α est algébrique sur un corps $\mathbb{K}$. Plusieurs énoncés, comme la remontée de Hensel, concernent néanmoins un cadre plus large, qui est utile pour le chapitre suivant. Grâce aux résultats du chapitre 2 nous supposons dans tout ce chapitre que le polynôme à factoriser est séparable.

4.1 Préliminaires

Rappelons le résultat suivant classique :

Théorème 4.1. *Soit $\mathbb{A}$ un anneau factoriel de corps de fraction $\mathbb{K}$. Un polynôme primitif f de $\mathbb{A}[x]$ est irréductible dans $\mathbb{A}[x]$ si, et seulement si, il est irréductible dans $\mathbb{K}[x]$.*

Par conséquent, nous pouvons supposer tout au long de ce chapitre que les polynômes à factoriser dans $\mathbb{Q}[x]$ sont en fait dans $\mathbb{Z}[x]$, séparables et primitifs. Dans le cas de polynômes à factoriser dans $\mathbb{Z}[x]$ nous n'aborderons pas la question de la factorisation de son contenu. La factorisation des entiers est un problème difficile qui fait appel a des techniques bien différentes de celles abordées dans ce cours.

Exemple 4.2. La factorisation des entiers étant coûteuse, l'utilisateur dispose dans MATHEMAGIX de deux types de fonctions : l'une retournant la factorisation irréductible, et l'autre, préfixée par `probable`, utilisant un certain nombre d'heuristiques pour être plus rapide, mais sans garantir que les facteurs trouvés sont bien irréductibles. Dans l'exemple suivant nous illustrons cet aspect, et aussi montrons la différence entre les factorisations dans $\mathbb{Z}[x]$ et $\mathbb{Q}[x]$. Notons que les facteurs trouvés sont affichés sous forme d'un produit.

```
Mmx] use "factorix"; type_mode? := true;
Mmx] irreducible_factorization (2^128 - 1)
```

3 5 17 257 641 65537 274177 6700417 67280421310721 : Vector(Irreducible_ factor(Integer))

```
Mmx] probable_irreducible_factorization (2^128 - 1)
```

3 5 17 257 641 65537 274177 6700417 67280421310721 : Vector(Irreducible_factor(Integer))

```
Mmx] x == polynomial (0 :> Integer, 1 :> Integer)
```

x : Polynomial(Integer)

```
Mmx] irreducible_factorization (15 * (x^2 - 1))
```

$3\,5\,(x+1)\,(x-1)$: Vector(Irreducible_factor(Polynomial(Integer)))

```
Mmx] x == polynomial (0 :> Rational, 1 :> Rational)
```

x : Polynomial(Rational)

```
Mmx] irreducible_factorization (15 * (x^2 - 1))
```

$15\,(x+1)\,(x-1)$: Vector(Irreducible_factor(Polynomial(Rational)))

4.2 Algorithme naïf et taille des coefficients

Soit f un polynôme séparable et primitif de $\mathbb{Z}[x]$ de degré d. Si g est un *facteur propre* de f, c'est à dire non constant et primitif, de degré au plus $m \leqslant d-1$, alors, pour tout entier $i \in \{0,\dots,m\}$, nécessairement $g(i)$ divise $f(i)$. Cette simple remarque fournit un algorithme pour trouver un tel facteur g ou bien montrer qu'il n'en existe pas :

1. Évaluer f en tous les points de $\{0,\dots,m\}$.
2. Énumérer tous les vecteurs $(b_0,\dots,b_m)$ de $\mathbb{Z}^m$ tels que b_i divise $f(i)$ pour tout $i \in \{0,\dots,m\}$.
3. Pour chaque tel vecteur $(b_0,\dots,b_m)$, interpoler le polynôme g de degré au plus m tel que $g(i)=b_i$ pour tout $i \in \{0,\dots,m\}$. Si g est dans $\mathbb{Z}[x] \setminus \mathbb{Z}$ et divise f alors c'est un facteur de f.
4. Si aucun facteur de f n'est trouvé à l'étape précédente alors f est irréductible.

Le coût de cette méthode dans le pire des cas est exponentiel. Néanmoins elle conduit à un premier résultat concernant la taille des coefficients des facteurs de f. Notons $\|f\|_\infty := \max_{i\in\{0,\dots,d\}} |f_i|$.

Proposition 4.3. *Soit f un polynôme séparable primitif de $\mathbb{Z}[x]$ de degré d, et soit g un facteur propre de f de degré $m \leqslant d-1$. Alors $\|g\|_\infty \leqslant (m+1)\,(d+1)\,m^{d+2m}\,\|f\|_\infty$.*

Démonstration. À partir de la formule d'interpolation de Lagrange

$$g(x) = \sum_{i=0}^{m} g(i) \prod_{j=0, j\neq i}^{m} \frac{x-j}{i-j},$$

nous obtenons la majoration grossière suivante :

$$\|g\|_\infty \leqslant \left(\max_{i\in\{0,\dots,m\}} |g(i)| \right) \left\| \sum_{i=0}^{m} \prod_{j=0, j\neq i}^{m} (x+j) \right\|_\infty \leqslant (m+1)\,m^{2m} \max_{i\in\{0,\dots,m\}} |g(i)|.$$

La conclusion provient de $\max_{i\in\{0,\dots,m\}} |g(i)| \leqslant \max_{i\in\{0,\dots,m\}} |f(i)| \leqslant (d+1)\,m^d\,\|f\|_\infty$. □

En d'autres termes, la taille des coefficients de g est au plus la taille de ceux de f plus $O(d \log d)$.

Exemple 4.4. Dans cet exemple la taille des coefficients des facteurs dépasse légèrement celle de f.

```
Mmx] f == polynomial (-1 :> Integer, (0 | i in 0..104), 1)
```

$x^{105}-1$

```
Mmx] irreducible_factorization f
```

$(x-1)\,(x^2+x+1)\,(x^4+x^3+x^2+x+1)\,(x^6+x^5+x^4+x^3+x^2+x+1)\,(x^8-x^7+x^5-x^4+x^3-x+1)\,(x^{12}-x^{11}+x^9-x^8+x^6-x^4+x^3-x+1)\,(x^{24}-x^{23}+x^{19}-x^{18}+x^{17}-x^{16}+x^{14}-x^{13}+x^{12}-x^{11}+x^{10}-x^8+x^7-x^6+x^5-x+1)\,(x^{48}+x^{47}+x^{46}-x^{43}-x^{42}-2\,x^{41}-x^{40}-x^{39}+x^{36}+x^{35}+x^{34}+x^{33}+x^{32}+x^{31}-x^{28}-x^{26}-x^{24}-x^{22}-x^{20}+x^{17}+x^{16}+x^{15}+x^{14}+x^{13}+x^{12}-x^9-x^8-2\,x^7-x^6-x^5+x^2+x+1)$

L'idée pour borner plus finement la taille des coefficients d'un facteur de f est de considérer une fonction multiplicative sur les polynômes, que l'on puisse facilement relier à la norme $\|.\|_\infty$. La *mesure de Mahler* de $f\in\mathbb{C}[x]\setminus\{0\}$, notée $M(f)$, est définie par

$$\ln M(f) := \frac{1}{2\,\pi}\int_0^{2\pi} \ln|f(e^{\iota\theta})|\,\mathrm{d}\theta,$$

où ln représente le logarithme népérien. Pour les preuves des résultats suivants nous renvoyons le lecteur à [44, Chapter 6, Section 6.6] qui prouve une borne due à Mignotte [94]. Si $f=g\,h$, alors nous avons $M(f)=M(g)\,M(h)$. Par ailleurs, en calculant $M(x-\varphi)$ nous obtenons :

Proposition 4.5. *Si* $f=f_d\,(x-\varphi_1)\cdots(x-\varphi_d)$ *alors* $M(f)=|f_d|\prod_{i=1}^{d}\max(1,|\varphi_i|)$.

Par ailleurs nous pouvons borner $M(f)$ comme suit :

Proposition 4.6. $M(f)\leqslant\|f\|_2$.

À partir de formules classiques reliant les coefficients de f à ses racines nous avons :

Proposition 4.7. *Si* g *est un facteur de degré* m *de* f, *alors*

$$\|g\|_2\leqslant 2^m\,M(g)\leqslant 2^m\left|\frac{g_m}{f_d}\right|\|f\|_2.$$

En combinant ces résultats nous obtenons :

Corollaire 4.8. (Mignotte) *Si* g *est un facteur de* f *de degré* m, *alors* $\|g\|_\infty\leqslant\sqrt{d+1}\,2^m\,\|f\|_\infty$.

4.3 Remontée de Newton-Hensel

Dans cette section $\mathbb{A}$ représente un anneau commutatif avec une unité, I est un idéal propre de $\mathbb{A}$, et f est un polynôme de $\mathbb{A}[x]$ de coefficient de tête, noté $\mathrm{lc}(f)$, inversible modulo I. Le lemme suivant est une forme du lemme de Hensel, mais avec le point de vue de l'itération de Newton.

Lemme 4.9. *Si* $\mathfrak{g} \in \mathbb{A}[x]$ *divise* f *modulo* I *et que* $\mathrm{lc}(\mathfrak{g}) = 1$*, alors il existe un unique polynôme* $\mathfrak{G}$ *congru à* $\mathfrak{g}$ *modulo* I*, unitaire et divisant* f *modulo* I^2*.*

Démonstration. Posons $\mathfrak{h} := f/\mathfrak{g}$. Nous cherchons deux polynômes $\mathfrak{G}$ et $\mathfrak{H}$ congrus respectivement à $\mathfrak{g}$ et $\mathfrak{h}$ modulo I, avec $\mathfrak{G}$ unitaire, et tels que $f = \mathfrak{G}\,\mathfrak{H} \bmod I^2$. Cette dernière équation se réécrit en $f = \mathfrak{g}\,\mathfrak{h} + \mathfrak{g}\,(\mathfrak{H} - \mathfrak{h}) + \mathfrak{h}\,(\mathfrak{G} - \mathfrak{g}) \bmod I^2$. En prenant les restes de la division par $\mathfrak{g}$ nous obtenons $\mathrm{rem}(f, \mathfrak{g}) = \mathrm{rem}(\mathfrak{h}\,(\mathfrak{G} - \mathfrak{g}), \mathfrak{g}) \bmod I^2$. Puisque f est séparable, le résultant de $\mathfrak{g}$ et $\mathfrak{h}$ est inversible modulo I, et donc $\mathfrak{h}$ est inversible modulo $\mathfrak{g}$ dans $(\mathbb{A}/I^2)[x]$. En notant $\mathfrak{u}$ son inverse, nous en déduisons la formule suivante :

$$\mathfrak{G} = \mathfrak{g} + \mathrm{rem}(\mathfrak{u}\, f, \mathfrak{g}) \bmod I^2. \qquad \square$$

Comme $f' = \mathfrak{g}'\,\mathfrak{h} + \mathfrak{g}\,\mathfrak{h}'$, alors $\mathfrak{h} = f'/\mathfrak{g}' \bmod \mathfrak{g}$ et nous obtenons la formule

$$\mathfrak{G} - \mathfrak{g} = \frac{\mathfrak{g}'}{f'}\, f \bmod \mathfrak{g} \bmod I^2,$$

et par conséquent un algorithme pour calculer $\mathfrak{G}$. Un partie importante du calcul consiste à évaluer f et f' modulo $\mathfrak{g}$ tout comme dans le cas de l'opérateur de Newton. Si f s'évalue rapidement à l'aide de peu d'opérations, alors il est inutile de développer f.

Comme $f \bmod \mathfrak{g}$ a tous ses coefficients dans I, il est suffisant de connaître l'inverse de f' modulo $\mathfrak{g}$ à la précision I. Lorsque I est maximal, $\mathbb{A}/I$ est un corps et nous pouvons obtenir cet inverse à partir de l'algorithme du p.g.c.d. étendu. En général, l'inverse $\mathfrak{U}$ de f' modulo $\mathfrak{G}$ à la précision I^2 peut être déduit de l'inverse $\mathfrak{u}$ de f' à la précision I par la formule suivante :

$$\mathfrak{U} = \mathfrak{u} - \mathfrak{u}\,(\mathfrak{u}\, f' - 1) \bmod \mathfrak{G},$$

puisque $\mathfrak{U}\, f' = 1 - (\mathfrak{u}\, f' - 1)^2 = 1 \bmod \mathfrak{G}$ à la précision I^2. Nous obtenons alors l'algorithme suivant :

Algorithme 4.1

Entrée : $f \in \mathbb{A}[x]$, et $\mathfrak{f}_1, \ldots, \mathfrak{f}_s$ unitaires dans $(\mathbb{A}/I)[x]$ tels que $f = \mathrm{lc}(f)\,\mathfrak{f}_1 \cdots \mathfrak{f}_s$ dans $(\mathbb{A}/I)[x]$.

Sortie : $\mathfrak{F}_1, \ldots, \mathfrak{F}_s$ unitaires dans $(\mathbb{A}/I^2)[x]$ tels que $f = \mathrm{lc}(f)\,\mathfrak{F}_1 \cdots \mathfrak{F}_s$ dans $(\mathbb{A}/I^2)[x]$ et $\mathfrak{F}_i = \mathfrak{f}_i \bmod I$ pour tout i.

1. Si $s = 1$ retourner $f/\mathrm{lc}(f)$.
2. Calculer $f \bmod \mathfrak{f}_1, \ldots, f \bmod \mathfrak{f}_s$ et $f' \bmod \mathfrak{f}_1, \ldots, f' \bmod \mathfrak{f}_s$ modulo I^2.
3. Calculer $\mathfrak{r}_1 = \mathfrak{f}_1' \frac{f}{f'} \bmod \mathfrak{f}_1, \ldots, \mathfrak{r}_s = \mathfrak{f}_s' \frac{f}{f'} \bmod \mathfrak{f}_s$ modulo I^2.
4. Retourner $\mathfrak{f}_1 + r_1, \ldots, \mathfrak{f}_s + r_s$.

Proposition 4.10. *Soient* $\mathbb{A} = \mathbb{Z}$ *et* $I = (p)$ *avec* p *premier. À partir d'une décomposition* $f = \mathrm{lc}(f)\,\mathfrak{f}_1 \cdots \mathfrak{f}_s$ *modulo* p*, avec des* $\mathfrak{f}_i$ *unitaires, nous pouvons calculer la decomposition* $f = \mathrm{lc}(f)\,\mathfrak{F}_1 \cdots \mathfrak{F}_s$ *modulo* p^σ *telle que* $\mathfrak{F}_i = \mathfrak{f}_i \bmod p$ *pour tout* i*, en temps* $O(\mathsf{I}(\sigma \log p)\,\mathsf{M}(d) \log d)$*.*

Démonstration. Il suffit d'appliquer successivement l'algorithme 4.1 pour arriver en précision p^2, p^4,... jusqu'à dépasser p^σ. Le coût découle principalement de la proposition 1.19, auquel il faut ajouter le coût pour obtenir les inverses de $f' \bmod \mathfrak{f}_i$ pour tout i, c'est à dire $O(\mathsf{M}(d) \log d + \mathsf{I}(\sigma \log p)\,\mathsf{M}(d))$. $\square$

Proposition 4.11. *Soient* $\mathbb{K}$ *un corps,* $\mathbb{A} = \mathbb{K}[t]$ *et* $I = (t)$*. À partir d'une décomposition* $f = \mathrm{lc}(f)\,\mathfrak{f}_1 \cdots \mathfrak{f}_s$ *modulo* t*, avec des* $\mathfrak{f}_i$ *unitaires, nous pouvons calculer la decomposition* $f = \mathrm{lc}(f)\,\mathfrak{F}_1 \cdots \mathfrak{F}_s$ *modulo* t^σ *telle que* $\mathfrak{F}_i = \mathfrak{f}_i \bmod t$ *pour tout* i*, en temps* $O(\mathsf{M}(\sigma)\,\mathsf{M}(d) \log d)$*.*

Démonstration. La preuve est similaire à celle de la proposition précédente. □

4.4 Recombinaison des facteurs p-adiques

Une fois les facteurs de f calculés dans $\mathbb{Z}_p$ à une précision suffisante déterminée à l'aide du Corollaire 4.8, nous pouvons tester toutes les recombinaisons possibles pour déduire les facteurs de f dans $\mathbb{Q}[x]$. Cette technique, principalement due à Hensel (1918), a commencé à être utilisée en calcul formel par Zassenhaus [151].

Algorithme 4.2

Entrée : $f \in \mathbb{Z}[x]$ primitif et séparable de degré $d \geqslant 1$.
Sortie : les facteurs irréductibles de f dans $\mathbb{Z}[x]$.

1. Choisir un nombre premier $p > d$ ne divisant ni le coefficient de tête de f, ni le discriminant de f, et calculer la factorisation irréductible $\mathfrak{F}_1, ..., \mathfrak{F}_s$ de f dans $\mathbb{Z}_p$ à la précision σ telle que $p^\sigma \geqslant 2\sqrt{d+1}\,2^{d/2}\,|f_d|\,\|f\|_\infty$.
2. Parcourir les sous-ensembles $\{e_1, ..., e_k\}$ de $\{1, ..., s\}$ par taille croissante, tels que $\deg \mathfrak{F}_{e_1} + \cdots + \deg \mathfrak{F}_{e_k} \leqslant d/2$:
   a) calculer $\mathfrak{G} := \mathrm{lc}(f)\,\mathfrak{F}_{e_1} \cdots \mathfrak{F}_{e_k}$ à la précision p^σ ;
   b) à partir de chaque coefficient $\mathfrak{G}_i$ de $\mathfrak{G}$, calculer le nombre entier g_i de la façon suivante : prendre la préimage $\tilde{\mathfrak{G}}_i$ de $\mathfrak{G}_i$ dans $\{0, ..., p^\sigma - 1\}$, si $\tilde{\mathfrak{G}}_i < p^\sigma/2$ alors poser $g_i := \tilde{\mathfrak{G}}_i$, sinon poser $g_i := -\tilde{\mathfrak{G}}_i$;
   c) calculer $g \in \mathbb{Z}[x]$ comme la partie primitive de $\sum_{i=0}^{\deg \mathfrak{G}_i} g_i\,x^i$.
   d) si g divise f alors l'ajouter à la liste des facteurs irréductibles de f.

Exemple 4.12. Dans l'exemple suivant nous montrons la nature des calculs de l'algorithme précédent en ignorant la borne théorique pour la précision. Mentionnons que les calculs sont réalisés dans Mathemagix d'une façon détendue [10, 53].

```
Mmx] use "factorix";
     p == modulus 5;
     F == polynomial (-1, 0, 0, 0, 1)
```

$x^4 - 1$

```
Mmx] irreducible_factorization (F mod p)
```

$(x+4)\,(x+1)\,(x+3)\,(x+2)$

```
Mmx] Fp == polynomial (p_adic (z, p) | z in @F)
```

$(1+O(p^{10}))\,x^4 + O(p^{10})\,x^3 + O(p^{10})\,x^2 + O(p^{10})\,x + 4 + 4\,p + 4\,p^2 + 4\,p^3 + 4\,p^4 + 4\,p^5 + 4\,p^6 + 4\,p^7 + 4\,p^8 + 4\,p^9 + O(p^{10})$

```
Mmx] v == irreducible_factorization (Fp)
```

$((1+O(p^{10}))\,x + 1 + O(p^{10}))\;((1+O(p^{10}))\,x + 2 + p + 2\,p^2 + p^3 + 3\,p^4 + 4\,p^5 + 2\,p^6 + 3\,p^7 + 3\,p^9 + O(p^{10}))\;((1+O(p^{10}))\,x + 3 + 3\,p + 2\,p^2 + 3\,p^3 + p^4 + 2\,p^6 + p^7 + 4\,p^8 + p^9 + O(p^{10}))\;((1+O(p^{10}))\,x + 4 + 4\,p + 4\,p^2 + 4\,p^3 + 4\,p^4 + 4\,p^5 + 4\,p^6 + 4\,p^7 + 4\,p^8 + 4\,p^9 + O(p^{10}))$

```
Mmx] g == factor v[1] * factor v[2]
```

$$(1+O(p^{10}))\,x^2+O(p^{10})\,x+1+O(p^{10})$$

```
Mmx] polynomial (@map (z :-> integer (z[0,10]), [@g]))
```

$$x^2+1$$

Remarque 4.13. En pratique il est utile de prendre p de la taille proche d'un mot machine pour que l'arithmétique des polynômes de $(\mathbb{Z}/p\,\mathbb{Z})[x]$ soit plus efficace.

Remarque 4.14. Pour le parcours des sous-ensembles il existe des algorithmes classiques permettant d'énumérer tous les sous-ensembles d'une taille donnée les uns après les autres sans avoir besoin de les stocker tous en mémoire. Nous renvoyons le lecteur par exemple au livre [106].

Remarque 4.15. Souvent, lorsque f n'est pas irréductible il admet des facteurs avec des coefficients de taille plus petite que celle prévue par la borne de Mignotte. Il est alors pertinent de tester l'existence de facteurs bien avant d'atteindre cette borne. De nombreuses optimisations doivent être mises en œuvres dans une implémentation de l'algorithme 4.2 pour le rendre efficace, comme expliqué dans l'article [1].

4.5 Polynômes de Swinnerton-Dyer

L'inconvénient de la méthode précédente est son coût exponentiel dans le pire des cas. Bien que dans beaucoup de cas il existe des nombres premiers p pour lesquels f a peu de facteurs p-adiques, il existe des exemples comme les polynômes de Swinnerton-Dyer pour lesquels la borne exponentielle est atteinte.

Définition 4.16. *Soit p_i le i-ième nombre premier pour $i \geqslant 1$. Le n-ième polynôme de Swinnerton-Dyer est défini par*

$$S_n(x) = \prod \left(x \pm \sqrt{2} \pm \sqrt{3} \pm \sqrt{5} \pm \cdots \pm \sqrt{p_n}\right),$$

où le produit est pris sur les 2^n possibilités de signes $+$ et $-$.

Le degré de S_n est 2^n. Ces polynômes peuvent être calculés numériquement (par exemple avec une arithmétique d'intervalle pour garantir le calcul) ou bien avec une cascade de résultants, comme par exemple : $S_2(x) = \mathrm{res}_{z_2}\left(\mathrm{res}_{z_1}(x - z_1 - z_2, z_1^2 - 2), z_2^2 - 3\right)$. Par cette dernière formule nous constatons que S_n a tous ses coefficients dans $\mathbb{Z}$ et qu'il est unitaire. Par des arguments classiques de théorie de Galois, nous savons que S_n est irréductible dans $\mathbb{Z}[x]$. Par ailleurs, si p est un nombre premier alors $\mathbb{F}_{p^2}$ contient toutes les racines carrées modulo p. Par conséquent $S_n(x)$ se factorise en facteurs de degrés 1 ou 2 modulo p. Le temps pour retrouver les recombinaisons des facteurs p-adiques est donc exponentiel en $d = 2^n$.

Exemple 4.17. Les calculs suivants montrent les facteurs irréductibles de S_3 modulo quelques nombres premiers. À la fin, le logiciel confirme l'irréductibilité de S_3 dans $\mathbb{Z}[x]$.

```
Mmx] use "factorix"
```

```
Mmx] f == swinnerton_dyer_polynomial 3
```

$$x^8 - 40\,x^6 + 352\,x^4 - 960\,x^2 + 576$$

```
Mmx] irreducible_factorization (f mod modulus 2)
```

$$x^8$$

```
Mmx] irreducible_factorization (f mod modulus 3)
```

$$(x^2+1)^2\,x^4$$

```
Mmx] irreducible_factorization (f mod modulus 5)
```

$$(x^2+2)^2\,(x^2+3)^2$$

```
Mmx] irreducible_factorization (f mod modulus 7)
```

$$(x^2+6\,x+3)\,(x^2+x+6)\,(x^2+6\,x+6)\,(x^2+x+3)$$

```
Mmx] irreducible_factorization (f mod modulus 11)
```

$$(x^2+7\,x+2)\,(x^2+4\,x+2)\,(x^2+2\,x+10)\,(x^2+9\,x+10)$$

```
Mmx] irreducible_factorization f
```

$$x^8 - 40\,x^6 + 352\,x^4 - 960\,x^2 + 576$$

4.6 Borne de complexité polynomiale

L'approche proposée dans cette section pour obtenir un algorithme en temps polynomial pour factoriser dans $\mathbb{Q}[x]$ suit la démarche historique.

Théorème 4.18. *[90] Si $f \in \mathbb{Z}[x]$ est un polynôme primitif de degré d alors sa décomposition irréductible peut être calculée en temps polynomial en d et $\log \| f \|_\infty$.*

Pour la preuve de ce théorème nous renvoyons le lecteur à l'article original [90] ou bien à [44, Chapter 16] qui contient une variante très détaillée. Nous rappelons ici juste quelques définitions et montrons les principales idées sur un exemple.

Définition 4.19. *Un* réseau *est un $\mathbb{Z}$-module engendré par une base de $\mathbb{R}^n$.*

Définition 4.20. *Une* base réduite *pour un réseau L est une base $f_1, \ldots, f_n$ du réseau tels que, si nous notons $f_1^*, \ldots, f_n^*$ la suite des vecteurs obtenus par le procédé d'orthogonalisation de Gram-Schmidt, nous avons $\| f_i^* \|_2 \leqslant 2 \, \| f_{i+1}^* \|_2$ pour tout $i \in \{1, \ldots, n-1\}$.*

Théorème 4.21. *[90] Une base réduite d'un réseau peut-être calculée en temps polynomial.*

Pour plus de détails nous renvoyons à nouveau le lecteur vers [44, Chapter 16]. Pour l'historique, les algorithmes récents et des applications, le lecteur pourra consulter [103]. À notre connaissance, la meilleure borne de complexité connue est présentée dans l'article [109]. L'une des propriétés simple, mais importante, des bases réduites est la suivante :

Proposition 4.22. *Si $f_1, \ldots, f_n$ est une base réduite d'un réseau L, alors pour tout vecteur f de L nous avons l'inégalité $\| f_1 \|_2 \leqslant 2^{(n-1)/2} \, \| f \|_2$.*

En d'autres termes, la norme du plus petit vecteur d'une base réduite n'est éloignée que de $O(n)$ bits de la norme du plus petit vecteur du réseau. À partir de ces outils nous pouvons décrire brièvement comment obtenir la factorisation dans $\mathbb{Q}[x]$ en temps polynomial :

1. Tout d'abord, pour un polynôme $f \in \mathbb{Z}[x]$, nous pouvons choisir un nombre premier p ne divisant ni $\mathrm{lc}(f)$ ni $\mathrm{disc}(f)$ en temps polynomial tel que $p \in \tilde{O}(d \log \|f\|_2)$.
2. Nous calculons ensuite un facteur h de f dans $(\mathbb{Z}/p\,\mathbb{Z})[x]$ en temps polynomial en d et p, et donc polynomial en d et $\log \|f\|_2$.
3. Par la méthode de Newton-Hensel, nous obtenons un facteur $\mathfrak{F}$ de f dans $\mathbb{Z}/p^\sigma\,\mathbb{Z}[x]$ unitaire à une précision σ que nous déterminons à l'aide de la borne de Mignotte.
4. Soit $l := \deg(\mathfrak{h})$, pour $n = d-1$, nous calculons une base réduite du réseau engendré par les d lignes de la matrice carrée suivante :
$$L := \begin{pmatrix} p^\sigma & & & & & \\ & \ddots & & & & \\ & & p^\sigma & & & \\ \mathfrak{F}_0 & \cdots & \mathfrak{F}_{l-1} & 1 & & \\ & \ddots & & \ddots & \ddots & \\ & & \mathfrak{F}_0 & \cdots & \mathfrak{F}_{l-1} & 1 \end{pmatrix}.$$
5. Le premier vecteur de cette base réduite représente un polynôme g de degré au plus n tel que $g = q\,\mathfrak{F} + p^\sigma\, r$, avec $r \in \mathbb{Z}[x]$ de degré au plus $l-1$, et $q \in \mathbb{Z}[x]$ de degré au plus $n-l$. Si la norme de g est trop grande, alors f est irréductible. Sinon nous pouvons construire un facteur propre de f qui est divisible par $\mathfrak{F}$ modulo p^σ.

Exemple 4.23. Dans l'exemple suivant nous montrons la nature des calculs, sans tenir compte des bornes théoriques, pour trouver un facteur de x^6-1.

```
Mmx] use "factorix";
     p == 7;
     f == polynomial (-1, 0, 0, 0, 0, 0, 1)
```

$x^6 - 1$

```
Mmx] fp == polynomial (p_adic (z, modulus p) | z in @f)
```

$(1+O(p^{10}))\,x^6 + O(p^{10})\,x^5 + O(p^{10})\,x^4 + O(p^{10})\,x^3 + O(p^{10})\,x^2 + O(p^{10})\,x + 6 + 6\,p + 6\,p^2 + 6\,p^3 + 6\,p^4 + 6\,p^5 + 6\,p^6 + 6\,p^7 + 6\,p^8 + 6\,p^9 + O(p^{10})$

```
Mmx] h == factor (irreducible_factorization (fp)[1])
```

$(1+O(p^{10}))\,x + 2 + 4\,p + 6\,p^2 + 3\,p^3 + 2\,p^5 + 6\,p^6 + 2\,p^7 + 4\,p^8 + 3\,p^9 + O(p^{10})$

```
Mmx] s == 10; l == deg h; d == deg f;
Mmx] L == vertical_join (
       [ if i=j then p^s else 0 | j in 0..d || i in 0..l ],
       [ (0 | j in 0..i), (integer (h[j][0,s]) | j in 0..l+1),
         (0 | j in 0..d-l-i-1) || i in 0..d-l ])
```

$$\begin{bmatrix} 282475249 & 0 & 0 & 0 & 0 & 0 \\ 146507972 & 1 & 0 & 0 & 0 & 0 \\ 0 & 146507972 & 1 & 0 & 0 & 0 \\ 0 & 0 & 146507972 & 1 & 0 & 0 \\ 0 & 0 & 0 & 146507972 & 1 & 0 \\ 0 & 0 & 0 & 0 & 146507972 & 1 \end{bmatrix}$$

```
Mmx] L == row_lll L
```

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & -1 & 1 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ -5498 & -3681 & 1818 & 5498 & 3680 & -1818 \\ -3680 & 1818 & 5498 & 3681 & -1818 & -5498 \end{bmatrix}$$

```
Mmx] g == [ polynomial (@row (L, i)) | i in 0..4 ]
```

$[x^3+1, -x^3+x^2-x, x^4+x, x^5+x^2]$

```
Mmx] f1 == gcd (g[0], gcd (g[1], gcd (g[2], g[3])))
```

x^2-x+1

```
Mmx] f rem f1
```

0

```
Mmx] f quo f1
```

x^4+x^3-x-1

4.7 Calcul des racines complexes

Dans cette section nous expliquons comment calculer une approximation numérique d'une racine d'un polynôme à coefficients entiers en temps polynomial. Il existe de nombreuses méthodes pour prouver ce résultat. Celle que nous avons retenue utilise la construction suivante :

Définition 4.24. *Soit* $f = \sum_{i=0}^{d} f_i\, x^i \in \mathbb{C}[x]$, *unitaire de degré* d. *Nous définissons* $\bar{f} := \sum_{i=0}^{d} \overline{f_i} x^i$, $\mathrm{rev}(d, f) := \sum_{i=0}^{d} f_{d-i} x^i$ *(le polynôme aux inverses des racines), et*

$$\mathtt{B}_f(x, y) := \frac{\mathrm{rev}(d, \bar{f})(x)\,\mathrm{rev}(d, f)(y) - f(x)\,\bar{f}(y)}{1 - x\,y} = \sum_{0 \leqslant i,j < d} b_{i,j}\, x^i\, y^j.$$

Nous notons $B_f := (b_{i,j})_{i,j}$ *la matrice symétrique de taille* $d \times d$ *des* $b_{i,j}$.

Le résultat suivant est classique et nous renvoyons le lecteur par exemple à l'ouvrage [49, Chapter 6] pour la preuve.

Théorème 4.25. (Schur-Cohn) *Si* $f \in \mathbb{C}[x]$ *est unitaire de degré* d, *alors la signature de* B_f *est égale à la différence* $\pi_+(f) - \pi_-(f)$, *où* $\pi_+(f)$ *(resp.* $\pi_-(f)$*) représente le nombre de racines de* f *incluses dans le disque ouvert* $B(0,1)$ *(resp. hors du disque fermé* $\bar{B}(0,1)$*).*

Exemple 4.26. Si $f(x) = (x - 1/2)(x - 2\,\iota)$, où $\iota^2 = -1$, alors nous pouvons effectuer les calculs suivants d'une façon numérique en affichant seulement 3 chiffres significatifs :

```
Mmx] use "multimix"; significant_digits := 3;
Mmx] X == coordinate ('x); Y == coordinate ('y);
```

```
Mmx] f == polynomial (-complex (1/2, 0.0), 1.0)
        * polynomial (-complex (0.0,2.0), 1.0)
```

$$1.00\,x^2 + (-0.500 - 2.00\,\mathrm{i})\,x + 1.00\,\mathrm{i}$$

```
Mmx] x == mvpolynomial (complex (1.0, 0.0)) * monomial X;
     y == mvpolynomial (complex (1.0, 0.0)) * monomial Y;
Mmx] B == sum (mvpolynomial (conj f[deg(f) - i]) * x^i | i in [0..deg f])
        * sum (mvpolynomial (f[deg(f) - i]) * y^i | i in [0..deg f])
        - sum (mvpolynomial (f[i]) * x^i | i in [0..deg f])
        * sum (mvpolynomial (conj f[i]) * y^i | i in [0..deg f])
```

$$(0e-16+0e-16\,\mathrm{i})\,x\,y + (1.50-1.50\,\mathrm{i})\,y + (1.50+1.50\,\mathrm{i})\,x + 0e-17+0e-17\,\mathrm{i}$$

```
Mmx] B_f == [ B[monomial(X)^i * monomial (Y)^j] | i in [0..deg f]
                                                || j in [0..deg f] ]
```

$$\begin{bmatrix} 0e-17+0e-17\,\mathrm{i} & 1.50+1.50\,\mathrm{i} \\ 1.50-1.50\,\mathrm{i} & 0e-16+0e-16\,\mathrm{i} \end{bmatrix}$$

```
Mmx] [ det B_f [0,i,0,i] | i in [0..1 + deg f] ]
```

$$[1.00, 1.00, 1.00]$$

Puisqu'il y n'y a pas de changement de signe dans la séquence des mineurs principaux de B_f, nous en déduisons que $n_+ = n_-$. En calculant $\gcd(f, x^2-1)$ nous pouvons vérifier que f n'a pas de racine sur le cercle unité. Par conséquent f admet une racine dans le disque unité et l'autre en dehors.

Proposition 4.27. *Soit f un polynôme de $\mathbb{Z}[x]$ sans carré de degré d. Il existe un algorithme permettant de déterminer si f admet un zéro dans $B(0,1)$ en temps polynomial en d et $\log \|f\|_\infty$.*

Démonstration. La matrice B_f se calcule en temps polynomial. Il suffit de vérifier que B_f est définie positive, ce qui peut se faire en vérifiant que tous les mineurs principaux sont strictement positifs. □

Pour une version beaucoup plus précise de ce résultat nous renvoyons le lecteur à l'article de Saux Picart [118].

Proposition 4.28. *Soit f un polynôme de $\mathbb{Z}[x]$ séparable de degré d. Le calcul d'une approximation d'un zéro de f dans $\mathbb{C}$ à une distance au plus $\varepsilon := 2^{-\sigma} < 1$ peut se faire en temps $(d \log(\|f\|_\infty/\varepsilon))^{O(1)}$.*

Démonstration. Il est classique, par exemple par la proposition 4.6, que les racines de f sont contenues dans un disque centré à l'origine et de rayon $R = 2^\rho \in (d\,\|f\|_\infty)^{O(1)}$, avec $\rho \in \mathbb{N}$. Si $a \in \bar{B}(0,R) \cap (\varepsilon\,\mathbb{Z} + \iota\,\varepsilon\,\mathbb{Z})$, et si $\eta \in [0,R] \cap (3/4)^\tau R\,\mathbb{Z}$, avec τ le plus petit entier tel que $(3/4)^\tau R < 2^{-\sigma}$, alors l'existence d'un zéro de f dans le disque $\bar{B}(a,\eta)$ est équivalent à l'existence d'un zéro de $f(\eta\,x + a)$ dans $\bar{B}(0,1)$. Comme $f(\eta\,x+a)$ peut se calculer de façon exacte (c'est à dire sans arrondi sur les nombres décimaux) en temps polynomial, alors la proposition précédente permet de tester en temps polynomial l'existence d'un zéro de f dans $\bar{B}(a,\eta)$.

Afin de trouver un tel zéro nous procédons par dichotomie. Pour commencer nous recouvrons le disque $\bar{B}(0, R)$ par les quatre disques $\bar{B}(R/2 + \iota\, R/2, 3\, R/4)$, $\bar{B}(-R/2 + \iota\, R/2, 3\, R/4)$, $\bar{B}(R/2 - \iota\, R/2, 3\, R/4)$, $\bar{B}(-R/2 - \iota\, R/2, 3\, R/4)$. Nous pouvons choisir un de ces disques qui contient au moins une racine de f en temps polynomial. Nous pouvons itérer ce processus sur le disque choisi et, ainsi de suite, jusqu'à obtenir un disque de rayon $(3/4)^\tau$ contenant au moins une racine de f. □

Pour un survol historique précis des méthodes pour calculer des approximations des racines complexes d'un polynôme à coefficients rationnels ou plus généralement des nombres complexes, nous renvoyons le lecteur aux livres [49, 93] et à l'article [114]. Dans les paragraphes suivants nous mentionnons quelques résultats à propos des bornes de complexité connues.

Il existe des algorithmes essentiellement optimaux pour calculer les racines complexes d'un polynôme [79, 101, 102, 112, 113, 116], qui sont dans la continuation d'idées développées plus tôt par Schönhage [129]. Les algorithmes les plus rapides du point de vue théorique sont ceux présentés dans les articles [102, 116], et qui utilisent une méthode de scindage équilibré et des généralisations d'un théorème de Grace et Heawood [24], qui peut être vu comme une version dans le cadre complexe du théorème de Rolle. Plus précisément, si $k + 1$ racines d'un polynôme p de degré n sont contenues dans une boule de rayon ρ, alors pour tout $\ell \leqslant k$, au moins $k + 1 - \ell$ racines de $p^{(\ell)}$ appartiennent à la boule de rayon $O(\rho)$, centrée au barycentre des $k+1$ racines. Pour les problèmes dits mal conditionnés, un scindage équilibré des racines demande une précision accrue (voir la discussions dans [114, Section 7]). Une autre approche moins équilibrée a été proposée par Pan pour traiter plus efficacement les grappes de racines [116] : son algorithme est presque optimal dans le pire des cas, mais aussi optimal dans les situations bien conditionnées. L'idée repose sur l'utilisation du théorème de Turan pour approcher les distances d'un point à l'ensemble des zéros d'un polynôme.

Par ailleurs, il existe d'autres techniques pour les problèmes mal conditionnés, qui sont théoriquement plus lentes que les méthodes précédentes mais qui sont souvent efficaces en pratique. Dans l'article [120], Renegar accélère la méthode de Weyl par subdivision en combinant l'algorithme de Schur-Cohn à l'α-theory pour les racines simples. Dans [115], Pan revisite la stratégie de Renegar pour traiter les grappes de racine plus efficacement grâce à l'opérateur de Schröder et à un critère d'arrêt reposant sur le théorème de Turan. Dans [12], Bini et Fiorentino décrivent leur implémentation utilisant de nombreuses heuristiques. Leur logiciel est bien adapté au cas où les polynômes ont peu de monômes et profite même des situations mal conditionnées. La littérature concernant le calcul des zéros d'un polynôme est très vaste et nous ne pourrions en faire ici une présentation exhaustive. Mentionnons juste quelques études de complexité pour différentes autres méthodes [27, 142, 149, 150].

Exemple 4.29. Mathemagix dispose de fonctions numériques robustes pour calculer des approximations des racines des polynômes. Des détails algorithmiques sont présentés dans les notes du cours de van der Hoeven, donné à l'occasion des « Journées Nationales de Calcul Formel » en 2011 [54].

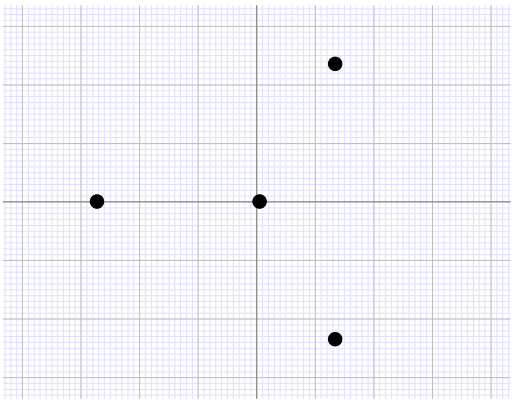
```
Mmx] use "analyziz"; type_mode? := true;
Mmx] x == polynomial (ball complex 0.0, ball complex 1.0);
Mmx] f == x^4 + 20 * x - 1
```

$1.00000000000000000\,x^4 + (0e-323228494 + 0e-323228494\,\mathrm{i})\,x^3 + (0e-323228494 + 0e-323228494\,\mathrm{i})\,x^2 + 20.0000000000000000\,x - 1.00000000000000000$: Polynomial(Ball(Floating, Complex(Floating)))

```
Mmx] v == roots f
```

[0.04999968750781223, −2.7308837023864973, 1.3404420074393425 − 2.350935485644\
7136 i, 1.3404420074393425 + 2.3509354856447136 i] : Vector(Ball(Floating, Complex(Floating)))

```
Mmx] include "graphix/points.mmx";
Mmx] $points (map (center, v))
```



4.8 Approximant algébrique numérique

La méthode que nous présentons ici provient de l'article [76]. L'idée principale consiste à calculer une approximation β suffisamment précise d'une racine complexe α de f, et d'utiliser l'algorithme de réduction de réseau pour calculer le polynôme minimal de α, d'où la terminologie d'*approximant algébrique*.

Algorithme 4.3

Entrée : $f \in \mathbb{Z}[x]$, séparable et primitif de degré $d \geqslant 1$, et un entier $m \leqslant d-1$.
Sortie : un facteur irréductible de f de degré au plus m s'il existe et sinon rien.

1. Poser $M := d^{d+2}\,\|f\|_2$, $c := 2^{(3/2)d^2+2d-1}\,M^{3m}$, $\varepsilon = 2^{-(2d^2+3d+4d\log_2 M)}$.
2. Calculer β telle qu'il existe une racine α de f avec $|\beta-\alpha| \leqslant \varepsilon/(2\,n\,\|f\|_2^n)$.
3. Pour i de 0 à m faire :
   Calculer $\tilde{\alpha}_i$ comme approximation de β^i avec $\left\lceil -\frac{1}{2}\log\varepsilon \right\rceil$ bits après la virgule.
4. Construire le réseau $\tilde{L}_c$ formé à partir des lignes de la matrice suivante :
$$\begin{pmatrix} 1 & & & c\,\mathrm{Re}(\tilde{\alpha}_0) & c\,\mathrm{Im}(\tilde{\alpha}_0) \\ & 1 & & c\,\mathrm{Re}(\tilde{\alpha}_1) & c\,\mathrm{Im}(\tilde{\alpha}_1) \\ & & \ddots & \vdots & \vdots \\ & & & 1 \quad c\,\mathrm{Re}(\tilde{\alpha}_m) & c\,\mathrm{Im}(\tilde{\alpha}_m) \end{pmatrix}$$
5. Appeler l'algorithme de réduction de base LLL. Notons $\tilde{v}$ le premier vecteur de la base réduite.
6. Si $\|\tilde{v}\|_2 \leqslant 2\,M^2$ alors retourner le polynôme correspondant.

Pour la preuve que cet algorithme fonctionne correctement et a un coût polynomial nous renvoyons de lecteur à l'article original [76].

Exemple 4.30. Dans cet exemple nous trouvons une racine de f de façon heuristique en utilisant la méthode de Newton. Les quantités M, c et ε sont aussi choisies de façon heuristique. Pour le calcul du polynôme minimal de α nous utilisons l'algorithme LLL disponible dans la librairie LATTIZ de MATHEMAGIX.

```
Mmx] use "lattiz"; type_mode? := true;
Mmx] f == polynomial (6 :> Integer, 0, -5, 0, 1)
```

$x^4 - 5\,x^2 + 6$: Polynomial(Integer)

```
Mmx] N z == z - evaluate (f, z) / evaluate (derive f, z);
Mmx] a == N N N N N 1.0
```

1.41421356237303701652: Floating

```
Mmx] c == 10^5
```

100000: Integer

```
Mmx] L == [ 1, 0, c;
            0, 1, as_integer trunc (c * a )]
```

$$\begin{bmatrix} 1 & 0 & 100000 \\ 0 & 1 & 141421 \end{bmatrix}$$: Matrix(Integer)

```
Mmx] row_lll L
```

$$\begin{bmatrix} -239 & 169 & 149 \\ 99 & -70 & 530 \end{bmatrix}$$: Matrix(Integer)

```
Mmx] L == [ 1, 0, 0, c;
            0, 1, 0, as_integer trunc (c * a );
            0, 0, 1, as_integer trunc (c * a^2)]
```

$$\begin{bmatrix} 1 & 0 & 0 & 100000 \\ 0 & 1 & 0 & 141421 \\ 0 & 0 & 1 & 199999 \end{bmatrix}$$: Matrix(Integer)

```
Mmx] R == row_lll L
```

$$\begin{bmatrix} -2 & 0 & 1 & -1 \\ -129 & 169 & -55 & 204 \\ -14 & -239 & 176 & 205 \end{bmatrix}$$: Matrix(Integer)

```
Mmx] g == polynomial (@row (R, 0)[0,3])
```

$x^2 - 2$: Polynomial(Integer)

```
Mmx] f rem g
```

0: Polynomial(Rational)

```
Mmx] f div g
```

$x^2 - 3$: Polynomial(Rational)

4.9 Dérivée logarithmique

L'inconvénient des méthodes précédentes reposant sur l'algorithme de réduction de réseau est que la taille du plus petit vecteur attendu (en l'occurrence un facteur de f) croît avec le degré et la taille des coefficients de f. Ces méthodes n'utilisent qu'un seul facteur approché de f (numérique ou p-adique) et sont par conséquent très utiles pour calculer un facteur irréductible de petite taille connue à l'avance. En revanche elles n'exploitent pas le fait que nous pouvons connaître tous les facteurs p-adiques de f. Par ailleurs bien qu'en temps polynomial, les exposants de la borne de complexité sont plutôt grands et rendent ces méthodes finalement peu praticables.

Une avancée majeure pour obtenir un algorithme de factorisation dans $\mathbb{Q}[x]$ en temps polynomial efficace en pratique est due à van Hoeij [52]. Cette avancée a été complétée ensuite par Belabas, van Hoeij, Klüners et Steel [6] puis améliorée par Novocin dans sa thèse de doctorat [107]. L'idée principale est la suivante : si f se factorise en $\mathrm{lc}(f)\, f_1 \cdots f_r$ alors l'égalité

$$f' = f\,\frac{f_1'}{f_1} + \cdots + f\,\frac{f_r'}{f_r},$$

permet, d'une certaine façon, de « linéariser » le problème. Plus précisément, si $\mathfrak{F}_1$, ..., $\mathfrak{F}_s$ sont les facteurs p-adiques unitaires de f, alors pour chaque $i \in \{1, ..., r\}$, il existe un vecteur $\mu_i \in \{0,1\}^s$ tel que $f_i = \mathrm{lc}(f_r)\, \mathfrak{F}_1^{\mu_{i,1}} \cdots \mathfrak{F}_s^{\mu_{r,s}}$. En prenant la dérivée logarithmique et en multipliant par f, nous obtenons :

$$f\,\frac{f_i'}{f_i} = \mu_{i,1}\, f\,\frac{\mathfrak{F}_1'}{\mathfrak{F}_1} + \cdots + \mu_{i,s}\, f\,\frac{\mathfrak{F}_r'}{\mathfrak{F}_r}.$$

Notons $\sigma \in \mathbb{N}$ la précision connue pour les facteurs p-adiques, et $\mathfrak{G}_i$ la préimage de $f\,\frac{\mathfrak{F}_1'}{\mathfrak{F}_1}$ dans $\mathbb{Z}[x]$, telle que $\mathfrak{G}_i = f\,\frac{\mathfrak{F}_1'}{\mathfrak{F}_1} \bmod p^\sigma$ et tous les coefficients de $\mathfrak{G}_i$ sont dans $\{0, ..., p^\sigma - 1\}$. Il existe un entier $c \in \mathbb{Z}$ tel que

$$f\,\frac{f_i'}{f_i} = \mu_{i,1}\,\mathfrak{G}_1 + \cdots + \mu_{i,s}\,\mathfrak{G}_s + c\,p^\sigma.$$

La borne de Mignotte permet de déterminer un entier τ tel que $\left\| f\,\frac{f_i'}{f_i} \right\|_\infty < p^\tau$. Si $\mathfrak{a} = \sum_{i \geqslant 0} a_i\, p^i$ est un nombre p-adique, nous notons $[\mathfrak{a}]_\tau^\sigma := \sum_{i=\tau}^{\sigma-1} a_i\, p^{i-\tau} \in \mathbb{N}$. Si $\mathfrak{Q} = \sum_{i=0}^d \mathfrak{Q}_i\, x^i \in \mathbb{Z}_p[x]$ alors notons, par extension, $[\mathfrak{Q}]_\tau^\sigma := \sum_{i=0}^d [\mathfrak{Q}_i]_\tau^\sigma\, x^i \in \mathbb{N}[x]$.

Considérons maintenant le développement p-adique de $f\,\frac{f_i'}{f_i}$ à la précision p^σ : il existe un polynôme $\beta_i \in \mathbb{Z}[x]$ à coefficients 0 ou 1 tel que $f\,\frac{f_i'}{f_i} + p^\tau\,\beta_i$ ait tous ses coefficients positifs et strictement inférieurs à p^τ, ce qui permet d'écrire $\left[f\,\frac{f_i'}{f_i} + \beta_i \right]_\tau^\sigma = 0$ et donc

$$0 = \left[f\,\frac{f_i'}{f_i} + p^\tau\,\beta_i \right]_\tau^\sigma = [\mu_{i,1}\,\mathfrak{G}_1 + \cdots + \mu_{i,s}\,\mathfrak{G}_s + p^\tau\,\beta_i]_\tau^\sigma.$$

À partir de $\mathfrak{G}_i = [\mathfrak{G}_i]_0^\tau + [\mathfrak{G}_i]_\tau^\sigma + [\mathfrak{G}_i]_\sigma^\infty$, et en posant $\varsigma_i := [\mu_{i,1}\,[\mathfrak{G}_1]_0^\tau + \cdots + \mu_{i,s}\,[\mathfrak{G}_s]_0^\tau]_\tau^\sigma$ nous avons :

$$\begin{aligned}[\mu_{i,1}\,\mathfrak{G}_1 + \cdots + \mu_{i,s}\,\mathfrak{G}_s + p^\tau\,\beta_i]_\tau^\sigma \;=\; & \mu_{i,1}\,[\mathfrak{G}_1]_\tau^\sigma + \cdots + \mu_{i,s}\,[\mathfrak{G}_s]_\tau^\sigma + \beta_i + \varsigma_i \\ & - p^{\sigma-\tau}[\mu_{i,1}\,[\mathfrak{G}_1]_\tau^\sigma + \cdots + \mu_{i,s}\,[\mathfrak{G}_s]_\tau^\sigma + \beta_i + \varsigma_i]_{\sigma-\tau}^\infty.\end{aligned}$$

Si $\sigma > \tau$ nous obtenons finalement des polynômes $\delta_i := \beta_i + \varsigma_i$ et $\varepsilon_i := [\mu_{i,1}\,[\mathfrak{G}_1]_\tau^\sigma + \cdots + \mu_{i,s}\,[\mathfrak{G}_s]_\tau^\sigma + \beta_i + \varsigma_i]_\sigma^\infty$ dans $\mathbb{Z}[x]$, de degré au plus $d-1$, tels que $\|\delta_i\|_\infty \leqslant s+1$, $\|\varepsilon_i\|_\infty \leqslant s+2$ et

$$\mu_{i,1}\,[\mathfrak{G}_1]_\tau^\sigma + \cdots + \mu_{i,s}\,[\mathfrak{G}_s]_\tau^\sigma + p^{\sigma-\tau}\delta_i = \varepsilon_i. \tag{4.1}$$

Il est alors naturel de considérer le réseau L engendré par les lignes de la matrice suivante :

$$\begin{pmatrix} 1 & & [\mathfrak{G}_{1,0}]_\tau^\sigma & \cdots & [\mathfrak{G}_{1,d-1}]_\tau^\sigma \\ & \ddots & \vdots & & \vdots \\ & & 1 & [\mathfrak{G}_{s,0}]_\tau^\sigma & \cdots & [\mathfrak{G}_{s,d-1}]_\tau^\sigma \\ & & & p^{\sigma-\tau} & & \\ & & & & \ddots & \\ & & & & & p^{\sigma-\tau} \end{pmatrix}.$$

L'équation (4.1) montre que $(\mu_{i,1}, \ldots, \mu_{i,s}, \varepsilon_{i,0}, \ldots, \varepsilon_{i,d-1})$ est un vecteur de « petite norme » dans ce réseau. Pour obtenir un algorithme complet de factorisation il reste à prouver qu'une base réduite de L permet bien de trouver tous les μ_i dès que σ est suffisamment grand. La présente description de l'algorithme est très simplifiée. De nombreuses optimisations sont importantes pour obtenir une implémentation vraiment efficace. Nous renvoyons le lecteur aux articles [47, 108, 109].

Exemple 4.31. Dans le calcul suivant nous illustrons la méthode pour la factorisation du produit des deux premiers polynômes de Swinnerton-Dyer.

```
Mmx] use "factorix"; p == modulus 9973;
     f == swinnerton_dyer_polynomial 2 * swinnerton_dyer_polynomial 3
```

$$x^{12} - 50\,x^{10} + 753\,x^8 - 4520\,x^6 + 10528\,x^4 - 6720\,x^2 + 576$$

```
Mmx] irreducible_factorization (f mod p)
```

$$(x^2 + 3391\,x + 342)\,(x^2 + 3391\,x + 1)\,(x^2 + 6582\,x + 342)\,(x^2 + 6582\,x + 1)\,(x^2 + 6582\,x + 9623)\,(x^2 + 3391\,x + 9623)$$

```
Mmx] f_p == polynomial (p_adic (z, p) | z in @f);
     facts_p == map (factor, irreducible_factorization (f_p));
Mmx] G == [(f_p div facts_p[i]) * derive facts_p[i] | i in 0..#facts_p];
Mmx] tau == 3; sigma == 4; s == #G; d == deg f - 1;
     L == horizontal_join (
       [ if i = j then 1 else 0 | j in 0..s || i in 0..s ],
       [ integer (G[i][j][tau, sigma]) | j in 0..d || i in 0..s ])
```

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 4366 & 4827 & 1629 & 6630 & 9707 & 7406 & 8181 & 3859 & 5821 & 3109 & 8349 \\ 0 & 1 & 0 & 0 & 0 & 0 & 7641 & 9972 & 9540 & 0 & 5771 & 9972 & 1863 & 0 & 6725 & 9972 & 8349 \\ 0 & 0 & 1 & 0 & 0 & 0 & 8153 & 5145 & 499 & 3342 & 8123 & 2566 & 5279 & 6113 & 8788 & 6863 & 8349 \\ 0 & 0 & 0 & 1 & 0 & 0 & 2331 & 9972 & 432 & 0 & 4201 & 9972 & 8109 & 0 & 3247 & 9972 & 1623 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1819 & 5145 & 9473 & 3342 & 1849 & 2566 & 4693 & 6113 & 1184 & 6863 & 1623 \\ 0 & 0 & 0 & 0 & 0 & 1 & 5606 & 4827 & 8343 & 6630 & 265 & 7406 & 1791 & 3859 & 4151 & 3109 & 1623 \end{bmatrix}$$

```
Mmx] L == vertical_join (L, horizontal_join (
      [ 0 | j in 0..s || i in 0..d ],
      [ if i = j then p^(sigma-tau) else 0 | j in 0..d || i in 0..d ]))
```

$$\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 4366 & 4827 & 1629 & 6630 & 9707 & 7406 & 8181 & 3859 & 5821 & 3109 & 8349 \\
0 & 1 & 0 & 0 & 0 & 0 & 7641 & 9972 & 9540 & 0 & 5771 & 9972 & 1863 & 0 & 6725 & 9972 & 8349 \\
0 & 0 & 1 & 0 & 0 & 0 & 8153 & 5145 & 499 & 3342 & 8123 & 2566 & 5279 & 6113 & 8788 & 6863 & 8349 \\
0 & 0 & 0 & 1 & 0 & 0 & 2331 & 9972 & 432 & 0 & 4201 & 9972 & 8109 & 0 & 3247 & 9972 & 1623 \\
0 & 0 & 0 & 0 & 1 & 0 & 1819 & 5145 & 9473 & 3342 & 1849 & 2566 & 4693 & 6113 & 1184 & 6863 & 1623 \\
0 & 0 & 0 & 0 & 0 & 1 & 5606 & 4827 & 8343 & 6630 & 265 & 7406 & 1791 & 3859 & 4151 & 3109 & 1623 \\
0 & 0 & 0 & 0 & 0 & 0 & 9973 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 9973 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 9973 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 9973 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 9973 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 9973 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 9973 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 9973 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 9973 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 9973 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 9973
\end{bmatrix}$$

```
Mmx] R == row_lll L
```

$$\begin{bmatrix}
-1 & 1 & -1 & 1 & -1 & -1 & 1 & 0 & 1 & 2 & 1 & 0 & 1 & 2 & 1 & 0 & 1 \\
0 & -1 & 0 & -1 & 0 & 0 & 1 & 2 & 1 & 0 & 1 & 2 & 1 & 0 & 1 & 2 & 1 \\
-35 & 37 & 85 & -37 & -81 & 39 & 147 & -4 & 136 & -4 & 95 & -4 & -32 & -4 & -86 & -4 & -234 \\
87 & -58 & -33 & 59 & 90 & -30 & 134 & -58 & 244 & -57 & -194 & -58 & 13 & -57 & -24 & -58 & 173 \\
-14 & 10 & -87 & -11 & 58 & -15 & -144 & 30 & -67 & 29 & 196 & 30 & -127 & 29 & -297 & 30 & 260 \\
-22 & -49 & 151 & 48 & -52 & 121 & 210 & -98 & -9 & -99 & 154 & -98 & 170 & -99 & -71 & -98 & 133 \\
-91 & 80 & 241 & -79 & -248 & 84 & -56 & 6 & 35 & 7 & -106 & 6 & 131 & 7 & -53 & 6 & 12 \\
-34 & -87 & -28 & 89 & 50 & 56 & -311 & -24 & 184 & -22 & 54 & -24 & -131 & -22 & 371 & -24 & -27 \\
131 & -8 & -119 & 7 & 145 & -105 & -26 & -25 & -142 & -26 & -70 & -25 & 445 & -26 & -33 & -25 & -26 \\
-232 & 26 & -81 & -27 & -9 & 142 & 53 & 91 & -38 & 90 & -101 & 91 & -194 & 90 & -113 & 91 & -146 \\
15 & 344 & -171 & -343 & 288 & 102 & 302 & -118 & -91 & -117 & 17 & -118 & 4 & -117 & 125 & -118 & 348 \\
-383 & 23 & 542 & 120 & 339 & -240 & 108 & -488 & -111 & 599 & 52 & 193 & 68 & -531 & -173 & 377 & 31 \\
43 & -547 & -341 & -72 & -422 & -75 & 33 & -523 & 73 & 415 & 195 & -171 & -22 & -667 & 40 & 217 & 8 \\
488 & -376 & -161 & -416 & -176 & 586 & 73 & 467 & 70 & 593 & -215 & -709 & 218 & 128 & -91 & -192 & 27 \\
-535 & -309 & 247 & -678 & 487 & -351 & 169 & -655 & -297 & -413 & 86 & 52 & 82 & 757 & -7 & 38 & 105 \\
886 & 62 & -212 & -157 & 155 & 972 & 192 & -1114 & -158 & 878 & 24 & 1036 & 20 & 49 & -99 & 6 & 72 \\
-443 & -706 & 368 & -231 & 287 & -561 & -31 & 608 & 9 & 394 & 131 & 464 & -86 & -70 & -24 & -1508 & -56
\end{bmatrix}$$

```
Mmx] R[0,0,2,s]
```

$$\begin{bmatrix}
-1 & 1 & -1 & 1 & -1 & -1 \\
0 & -1 & 0 & -1 & 0 & 0
\end{bmatrix}$$

```
Mmx] Hp == facts_p[1] * facts_p[3]
```

$$(1+O(p^{10}))\,x^4+O(p^{10})\,x^3+(9963+9972\,p+9972\,p^2+9972\,p^3+9972\,p^4+9972\,p^5+9972\,p^6+9972\,p^7+9972\,p^8+9972\,p^9+O(p^{10}))\,x^2+O(p^{10})\,x+1+O(p^{10})$$

```
Mmx] reconstruct u == {
       x == integer u[0,3];
       if x > 9973^3 div 2 then x - 9973^3 else x; };
Mmx] g == polynomial (@map (reconstruct, [@Hp]))
```

$$x^4-10\,x^2+1$$

```
Mmx] f rem g
```

$$0$$

```
Mmx] f div g
```

$$x^8-40\,x^6+352\,x^4-960\,x^2+576$$

Défi 4.1. L'algorithme PSLQ conçu par Ferguson et Bailey au début des années 90 [28] permet de calculer une relation entière non triviale de petite taille entre plusieurs nombres (entiers ou décimaux en fixant la précision souhaitée) ou bien de prouver qu'il n'en existe pas d'une norme inférieure à une valeur donnée en entrée de l'algorithme. Pour une description précise de l'algorithme nous renvoyons le lecteur à [13, Appendix B]. Des résultats récents sur la complexité de PSLQ et son lien avec l'algorithme de réduction de réseau se trouvent dans l'article [18].

```
Mmx] use "lattiz"
Mmx] pslq ([100, 150, 200], 5)
```

$[1, -2, 1]$

Sachant que les relations μ_i qui interviennent dans le problème de factorisation ne font intervenir que des 0 et des 1, est-il possible de tirer parti de PSLQ pour obtenir une meilleure borne de complexité, ou bien de meilleures performances pratiques ?

4.10 Extension algébrique

Dans cette section nous supposons que nous disposons d'un algorithme de factorisation dans $\mathbb{K}[x]$ et nous expliquons comment déduire un algorithme de factorisation dans $\mathbb{K}(\alpha)[x]$, lorsque α est algébrique sur $\mathbb{K}$. Si $\mathbb{K}$ est un corps fini, alors $\mathbb{K}(\alpha)$ est aussi fini et les méthodes du chapitre précédent s'appliquent. Nous pouvons donc supposer que $\mathbb{K}$ est infini.

Supposons d'abord que α est séparable, de polynôme minimal q sur $\mathbb{K}$. Soit $f \in \mathbb{K}[x, y]$ tel que $f(x,\ \alpha)$ est séparable. La factorisation de $f(x,\ \alpha)$ dans $\mathbb{K}(\alpha)[x]$ correspond à la décomposition en irréductibles de l'idéal $I := (q(y),\ f(x,\ y))$ de $\mathbb{K}[x,\ y]$. Comme $\mathbb{K}$ est infini, nous pouvons trouver $\lambda \in \mathbb{K}$ et construire un isomorphisme entre $\mathbb{K}[x,\ y]/I$ et $\mathbb{K}[z]/(Q(z))$, où Q est le polynôme minimal de $u := x + \lambda\ y$ dans $\mathbb{K}[x,\ y]/I$. Notons $v_x \in \mathbb{K}[z]$ et $v_y \in \mathbb{K}[z]$ les polynômes de degré strictement inférieur au degré de Q et tels que $x = v_x(u)$ et $y = v_y(u)$ dans $\mathbb{K}[x,\ y]/I$. Notons $Q_1, ..., Q_r$ les facteurs irréductibles de Q. Les facteurs irréductibles $f_1(x, \alpha), ..., f_r(x, \alpha)$ de $f(x, \alpha)$ peuvent être calculés comme $f_i(x, \alpha) = \gcd\left(f(x, \alpha), Q_i(x + \lambda\ \alpha)\right)$. Cette technique est détaillée dans de livre [144]. Sa mise en pratique en calcul formel semble remonter au travaux de Trager [140, 141].

Si α est inséparable sur $\mathbb{K}$, sans perte de généralité, nous pouvons nous ramener au cas où $\alpha^p = a \in \mathbb{K}$ et $\alpha \notin \mathbb{K}$. Une première stratégie consiste à réorganiser la présentation de $\mathbb{K}(\alpha)$ sur son sous-corps premier de sorte à éviter les extensions inséparables. Cette technique est détaillée par exemple dans le livre [84, Chapter VIII, Section 4]. Une autre approche consiste à factoriser $g(x^p) := f^p(x) \in \mathbb{K}[x^p]$: pour chaque facteur g_i de g, soit $g_i(x^p)$ est la puissance p-ième d'un facteur irréductible de f dans $\mathbb{K}(\alpha)[x]$, soit $g_i(x^p)$ est un facteur irréductible de f dans $\mathbb{K}(\alpha)[x]$. Cette approche est simple à mettre en œuvre mais requiert que l'extraction de racines p-ièmes dans n'importe quelle extension algébrique de $\mathbb{K}$ soit disponible.

5

Polynômes à deux variables

Dans ce chapitre nous nous intéressons à la factorisation d'un polynôme $F \in \mathbb{K}[x, y]$, où $\mathbb{K}$ est un corps commutatif quelconque en supposant que nous disposons déjà d'algorithmes pour factoriser dans $\mathbb{K}[y]$. Notons d_x le degré de F en la variable x et d_y celui en la variable y. Nous supposons que F est primitif et séparable vu comme polynôme de $\mathbb{K}[x][y]$. Cette dernière hypothèse revient à réaliser un certain nombre de calculs de p.g.c.d. dans $\mathbb{K}[x]$, ce qui peut être réalisé au moyen d'algorithmes rapides sans hypothèse particulière sur $\mathbb{K}$, comme expliqué dans le deuxième chapitre.

5.1 Notations

Les facteurs irréductibles de F sont notés $F_1, ..., F_r$. Le coefficient de tête $\mathrm{lc}_y(F)$ de F vu comme polynôme de $\mathbb{K}[x][y]$ est noté c. Les coefficients de têtes $\mathrm{lc}_y(F_i)$ sont notés c_i. Afin de simplifier la présentation nous supposons aussi que les coefficients de têtes de c et des c_i sont tous 1.

Supposons maintenant que l'on peut trouver un point $a \in \mathbb{K}$ tel que $F(a, y)$ reste de degré d_y et séparable. Quitte à appliquer une translation à la variable x, nous pouvons alors calculer les facteurs irréductibles de F dans $\mathbb{K}[[x]][y]$, que nous notons $\mathfrak{F}_1, ..., \mathfrak{F}_s$ et appelons *facteurs analytiques* par opposition aux *facteurs rationnels* F_i. Pour chaque $i \in \{1, ..., r\}$, nous associons l'unique vecteur $\mu_i \in \{0, 1\}^s$ défini par $F_i = c_i \prod_{j=1}^{s} \mathfrak{F}_j^{\mu_{i,j}}$. Nous examinerons plus tard la situation où $\mathbb{K}$ est trop petit pour garantir de trouver un tel point a, mais en attendant nous supposerons les hypothèses suivantes satisfaites :

$$\text{(N)} \quad \begin{cases} \deg\left(F(0, y)\right) = d_y, \\ \mathrm{Res}\left(F(0, y), \frac{\partial F}{\partial y}(0, y)\right) \neq 0, \\ F \text{ est primitif vu dans } \mathbb{K}[x][y]. \end{cases}$$

Si $A = \sum_{i,j \geqslant 0} a_{i,j}\, x^i\, y^j$ représente un polynôme de $\mathbb{K}[[x]][y]$ alors nous notons $[A]_k^l$ la projection sur $\mathbb{K}[x, y]$ suivante :

$$[A]_k^l := \sum_{k \leqslant i \leqslant l-1, j \geqslant 0} a_{i,j}\, x^i\, y^j.$$

Nous introduisons aussi les produits partiels suivants :

$$\hat{F}_i := \prod_{j=1, j \neq i}^{r} F_j = \frac{F}{F_i}, \qquad \hat{\mathfrak{F}}_i := c \prod_{j=1, j \neq i}^{s} \mathfrak{F}_j = \frac{F}{\mathfrak{F}_i}.$$

Le calcul des μ_i à partir des $\mathfrak{F}_i$ est appelé le *problème de recombinaison* des facteurs analytiques. Les $\mathfrak{F}_i$ peuvent être calculés à partir de la factorisation de $F(0,y)$ et de la remontée de Newton-Hensel présentée dans le chapitre précédent.

Le premier algorithme (en temps exponentiel) semble remonter aux travaux de Kronecker. L'idée était de factoriser $F\big(y^{d_y+1},y\big)$ puis de chercher parmi toutes les recombinaisons possibles celle donnant $F_i\big(y^{d_y+1},\ y\big)$. Dans ce chapitre nous nous concentrerons principalement sur les techniques utilisant la remontée de Newton-Hensel. Pour des détails historiques concernant la factorisation des polynômes à plusieurs variables nous renvoyons le lecteur aux livres [44, 97, 155], mais aussi vers les articles [21, 35, 41, 58, 67, 68, 69, 70].

5.2 Recherche exhaustive

Tout comme dans le cas de la factorisation dans $\mathbb{Q}[x]$ vue dans le chapitre précédent, une première solution pour trouver comment les facteurs analytiques se recombinent en les facteurs rationnels consiste à énumérer tous les sous-ensembles de $\{1,\ldots,s\}$ par cardinal croissant de 1 à $\lceil s/2\rceil$. La situation est en fait plus simple qu'avec $\mathbb{Q}[x]$ puisque le degré partiel des facteurs de F en x ne dépasse pas d_x. Au lieu de décrire à nouveau la méthode, nous l'illustrons sur l'exemple suivant.

Exemple 5.1.

```
Mmx] use "factorix";
     p == modulus 101; x == polynomial (0, 1);
     F == polynomial ((-x^2 - x^4) mod p, (2 + 2*x^2) mod p,
                      polynomial (-1) mod p, polynomial (-2) mod p,
                      polynomial (1) mod p)
```

$$y^4+99\,y^3+100\,y^2+(2\,x^2+2)\,y+100\,x^4+100\,x^2$$

```
Mmx] set_variable_name (series (@(x mod p)), 'x);
     set_variable_name (polynomial series (@(x mod p)), 'y);
     Fs == polynomial (series (@z) | z in @F)
```

$$(1+O(x^{10}))\,y^4+(99+O(x^{10}))\,y^3+(100+O(x^{10}))\,y^2+(2+2\,x^2+O(x^{10}))\,y+100\,x^2+100\,x^4+O(x^{10})$$

```
Mmx] fs == irreducible_factorization (Fs)
```

$$((1+O(x^{10}))\,y+50\,x^2+63\,x^4+82\,x^6+26\,x^8+O(x^{10}))\,((1+O(x^{10}))\,y+99+51\,x^2+38\,x^4+19\,x^6+75\,x^8+O(x^{10}))\,((1+O(x^{10}))\,y+100+50\,x^2+38\,x^4+82\,x^6+75\,x^8+O(x^{10}))\,((1+O(x^{10}))\,y+1+51\,x^2+63\,x^4+19\,x^6+26\,x^8+O(x^{10}))$$

```
Mmx] Gs == factor (fs[0]) * factor (fs[1])
```

$$(1+O(x^{10}))\,y^2+(99+O(x^{10}))\,y+x^2+O(x^{10})$$

```
Mmx] G == polynomial (Gs[i][0,4] | i in 0..#Gs)
```

$$y^2+99\,y+x^2$$

Remarque 5.2. Cette méthode par recherche exhaustive peut être améliorée avec des techniques similaires au cas de $\mathbb{Q}[x]$. Par exemple pour vérifier qu'un produit partiel de facteurs analytiques ne conduit pas à un facteur rationnel nous pouvons nous contenter de n'en calculer qu'une partie. D'autre part nous pouvons essayer plusieurs points a et choisir celui qui implique le moins de facteurs analytiques.

Rappelons que l'utilisation de la remontée de Newton-Hensel en calcul formel pour factoriser les polynômes est apparue à la fin des années soixante dans les travaux de Zassenhaus [151]. L'application à plusieurs variables a commencé dans les travaux [100, 145, 146]. Pendant longtemps l'étape de recombinaison a été effectuée par la méthode de recherche exhaustive que nous venons d'illustrer. Son coût est exponentiel dans le pire des cas. Néanmoins, si $\mathbb{K}$ est un corps fini, en moyenne sur l'ensemble des polynômes, son coût est essentiellement quadratique [36].

5.3 Temps polynomial

Les premiers algorithmes en temps polynomiaux sont dus à Kaltofen au début des années 80. Plusieurs autres auteurs ont contribué à ce problème pour couvrir les corps de coefficient usuels : Chistov, von zur Gathen, Grigoriev et Lenstra. Tout comme dans le cas de la factorisation dans $\mathbb{Q}[x]$, ces algorithmes calculent un facteur rationnel à partir d'un seul facteur analytique, sans utiliser la totalité de la décomposition analytique. Les exposants des bornes de complexité sont plutôt élevés et l'efficacité pratique n'est pas très spectaculaire par rapport à la recherche exhaustive en moyenne.

La première réduction de la factorisation de deux à une variable en temps quasi-quadratique, pour la caractéristique nulle ou suffisamment grande, est due à Shuhong Gao [34], en utilisant la *factorisation absolue*. Dans le présent chapitre nous nous concentrons sur les méthodes utilisant la remontée de Newton-Hensel, qui conduisent aux meilleures bornes de complexité connues à l'heure actuelle. Ces méthodes ont été étudiées par Musser [100], Wang et Rothschild [146], puis Wang [145], von zur Gathen [38, 39], Bernardin et Monagan [9].

Dans les articles [124, 125, 126], Sasaki et ses collaborateurs avaient étudié une méthode pour ramener le problème de recombinaison des facteurs analytiques à de l'algèbre linéaire, mais malheureusement sans prouver de borne de coût polynomiale (*cf.* exemple dans l'introduction de l'article [15]). Une variante de leur algorithme a néanmoins conduit à une borne polynomiale grâce aux travaux de Belabas, van Hoeij, Klüners et Steel [6]. L'idée est de construire un système linéaire dont les solutions fournissent les vecteurs μ_i, à partir d'une factorisation analytique à une précision $\sigma \geqslant d\,(d-1)+1$ où d est le degré total de F. Cette borne inférieure de précision se trouve être essentiellement nécessaire lorsque $\mathbb{K}$ est de petite caractéristique positive. Dans les paragraphes suivants nous présentons cette méthode, mais en exprimant la borne en fonction des degrés partiels d_x et d_y de F.

À partir de la dérivée logarithmique de la formule $F_i = c_i \prod_{j=1}^{s} \mathfrak{F}_j^{\mu_{i,j}}$ nous avons :

$$\frac{\frac{\partial F_i}{\partial y}}{F_i} = \sum_{j=1}^{s} \mu_{i,j} \frac{\frac{\partial \mathfrak{F}_j}{\partial y}}{\mathfrak{F}_j}, \text{ et donc } \hat{F}_i \frac{\partial F_i}{\partial y} = \sum_{j=1}^{s} \mu_{i,j} \hat{\mathfrak{F}}_j \frac{\partial \mathfrak{F}_j}{\partial y}. \tag{5.1}$$

Comme le degré partiel en x de $\hat{F}_i \frac{\partial F_i}{\partial y}$ est au plus d_x, la dernière inégalité conduit au système linéaire suivant, en les inconnues $\ell_1, \ldots, \ell_s$, et dont les vecteurs μ_i sont solutions :

$$\left[\sum_{j=1}^{s} \ell_j \hat{\mathfrak{F}}_j \frac{\partial \mathfrak{F}_j}{\partial y} \right]_{d_x+1}^{\sigma} = 0. \tag{5.2}$$

Proposition 5.3. *Si $\sigma \geqslant d_x\,(2\,d_y - 1) + 1$, alors $\mu_1, \ldots, \mu_r$ forment la base échelonnée réduite (à une permutation près) de l'espace des solutions de (5.2).*

Démonstration. Soit $\ell_1, ..., \ell_s$ des valeurs solution du système, et posons

$$G := \left[\sum_{j=1}^{s} \ell_j \, \hat{\mathfrak{F}}_j \, \frac{\partial \mathfrak{F}_j}{\partial y} \right]_0^{d_x+1} \in \mathbb{K}[x, y].$$

Considérons le résultant $R(x) := \mathrm{Res}_y\Big(F(x, y), \ell_1 \, \frac{\partial F}{\partial y}(x, y) - G(x, y) \Big) \in \mathbb{K}[x]$. Comme le degré partiel de G en y est au plus $d_y - 1$, le degré de R est au plus $d_x \, (2 \, d_y - 1)$. Par ailleurs nous avons

$$\begin{aligned} R(x) &= c^{d_y - 1} \prod_{j=1}^{s} \mathrm{Res}_y\Big(\mathfrak{F}_j, \ell_1 \frac{\partial F}{\partial y}(x, y) - G(x, y) \Big) \\ &= c^{d_y - 1} \prod_{j=1}^{s} \mathrm{Res}_y\Big(\mathfrak{F}_j, (\ell_1 - \ell_j) \, \hat{\mathfrak{F}}_j \frac{\partial \mathfrak{F}_j}{\partial y}(x, y) \Big) + O(x^\sigma) \\ &= 0 + O(x^\sigma). \end{aligned}$$

Donc, si $\sigma > \deg(R)$ alors $R = 0$. Comme $F(0, y)$ est séparable, nous déduisons aussi que $\prod_{j, \ell_j = \ell_1} \mathfrak{F}_j$ est le p.g.c.d. de F et $\ell_1 \, \frac{\partial F}{\partial y}(x, y) - G(x, y)$ et donc un produit de facteurs irréductibles de F. En d'autres termes, en répétant cette opération pour chaque ℓ_j (à la place de ℓ_1), nous obtenons que $(\ell_1, ..., \ell_s)$ est bien une combinaison linéaire des μ_i. □

Dans l'article [15], en notant d le degré total de F, il a été prouvé que la borne en précision $d_x \, (2 \, d_y - 1) + 1$ peut être ramenée à $3 \, d - 2$ si la caractéristique de $\mathbb{K}$ est nulle ou au moins $d \, (d - 1) + 1$. Dans l'article [85], un autre système linéaire nécessitant une précision $2 \, d$ a été proposé sous les même hypothèses. Nous ne présentons pas ces variantes ici. Dans les prochaines sections nous montrons une approche sensiblement différente, issue de l'article [88] utilisant une précision $d_x + 1$, sans hypothèse sur la caractéristique.

5.4 Espace des résidus

À partir de maintenant, les objets centraux que nous allons étudier sont les polynômes suivants :

$$\mathfrak{G}_i := \left[\hat{\mathfrak{F}}_i \, \frac{\partial \mathfrak{F}_i}{\partial y} \right]_0^{d_x+1}, \text{ pour tout } i \in \{1, ..., s\}.$$

La méthode que nous présentons ici se réduit presque essentiellement à résoudre un problème linéaire construit à partir des polynômes $\mathfrak{G}_i$. Le calcul de ces derniers nécessite la factorisation irréductible de $F(0, y)$, et une remontée de Hensel de ces facteurs à la précision $O(x^{2 d_x + 1})$, qui correspond essentiellement à la précision à atteindre dans le pire des cas par la méthode de recherche exhaustive des recombinaisons, et qui peut se faire en temps quasi-linéaire.

Le $\mathbb{K}$-espace vectoriel des polynômes de degré au plus k en x et au plus l en y est noté $\mathbb{K}[x, y]_{k,l}$. Soit $\mathbb{F}$ un sous-corps de $\mathbb{K}$, introduisons :

$$\mathcal{L}_{\mathbb{F}} := \left\{ (\ell_1, ..., \ell_s) \in \mathbb{F}^s \Big| \sum_{i=1}^{s} \ell_i \, \mathfrak{G}_i \in \left\langle \hat{F}_1 \, \frac{\partial F_1}{\partial y}, ..., \hat{F}_r \, \frac{\partial F_r}{\partial y} \right\rangle_{\mathbb{F}} \right\},$$

où $\left\langle \hat{F}_1 \frac{\partial F_1}{\partial y}, \ldots, \hat{F}_r \frac{\partial F_r}{\partial y} \right\rangle_{\mathbb{F}}$ représente le $\mathbb{F}$-espace vectoriel engendré par les polynômes $\hat{F}_1 \frac{\partial F_1}{\partial y}, \ldots, \hat{F}_r \frac{\partial F_r}{\partial y}$. La connaissance de $\mathcal{L}_{\mathbb{F}}$ résout le problème de recombinaison :

Lemme 5.4. *Les vecteurs $\mu_1, \ldots, \mu_r$ forment la base échelonnée réduite de $\mathcal{L}_{\mathbb{F}}$ (à une permutation près).*

Démonstration. Soit $i \in \{1, \ldots, r\}$. Puisque $\deg_y(F_1) + \cdots + \deg_y(F_r) = d_y$, $\deg_y\left(\frac{\partial F_i}{\partial y}\right) \leqslant \deg_y(F_i) - 1$, $\deg_x(F_1) + \cdots + \deg_x(F_r) = d_x$, et $\deg_x\left(\frac{\partial F_i}{\partial y}\right) \leqslant \deg_x(F_i)$, nous obtenons $\hat{F}_i \frac{\partial F_i}{\partial y} \in \mathbb{K}[x, y]_{d_x, d_y - 1}$. À partir de l'équation (5.1), nous déduisons :

$$\hat{F}_i \frac{\partial F_i}{\partial y} = \left[\hat{F}_i \frac{\partial F_i}{\partial y} \right]_0^{d_x+1} = \left[\sum_{j=1}^{s} \mu_{i,j} \hat{\mathfrak{F}}_j \frac{\partial \mathfrak{F}_i}{\partial y} \right]_0^{d_x+1} = \sum_{j=1}^{s} \mu_{i,j} \mathfrak{G}_j. \tag{5.3}$$

Puisque $\mu_i \in \{0, 1\}^s \subseteq \mathbb{F}^s$, nous avons $\mu_i \in \mathcal{L}_{\mathbb{F}}$. Par l'hypothèse (N), les polynômes $\mathfrak{G}_1(0, y), \ldots, \mathfrak{G}_s(0, y)$ sont linéairement indépendants sur $\mathbb{F}$, il en est donc de même pour $\mathfrak{G}_1, \ldots, \mathfrak{G}_s$. Par le même argument, $\hat{F}_1 \frac{\partial F_1}{\partial y}, \ldots, \hat{F}_r \frac{\partial F_r}{\partial y}$ sont aussi linéairement indépendants sur $\mathbb{F}$. Par conséquent $\mathcal{L}_{\mathbb{F}}$ a bien pour dimension r. □

Considérons un s-uplet $(\ell_1, \ldots, \ell_s) \in \mathbb{F}^s$ et posons $G := \sum_{i=1}^{s} \ell_i \mathfrak{G}_i$. Notons $\rho_1, \ldots, \rho_{d_y}$ les résidus de G/F définis par :

$$\frac{G}{F} = \sum_{i=0}^{d_y} \frac{\rho_i}{(y - \varphi_i)}, \tag{5.4}$$

où $\varphi_1, \ldots, \varphi_{d_y}$ représentent les racines de F dans une clôture algébrique de $\mathbb{K}(x)$.

Lemme 5.5. *Si $(\ell_1, \ldots, \ell_s) \in \mathcal{L}_{\mathbb{F}}$, alors $\rho_1, \ldots, \rho_{d_y}$ appartiennent tous à $\mathbb{F}$. Réciproquement, si $\rho_1, \ldots, \rho_{d_y}$ appartiennent tous à $\bar{\mathbb{K}}$, alors $(\ell_1, \ldots, \ell_s) \in \mathcal{L}_{\mathbb{F}}$.*

Démonstration. Par définition, si $(\ell_1, \ldots, \ell_s) \in \mathcal{L}_{\mathbb{F}}$ alors G est une combinaison linéaire sur $\mathbb{F}$ des $\hat{F}_i \frac{\partial F_i}{\partial y}$. Puisque, pour tout i, les résidus de $\hat{F}_i \frac{\partial F_i}{\partial y}/F = \frac{\partial F_i}{\partial y}/F_i$ sont tous dans $\{0, 1\}$, tous les ρ_i appartiennent à $\mathbb{F}$. Réciproquement, supposons que tous les ρ_i soient dans $\bar{\mathbb{K}}$. En remplaçant x par 0 dans l'égalité (5.4) nous obtenons

$$\frac{G(0, y)}{F(0, y)} = \sum_{i=1}^{d_y} \frac{\rho_i(0)}{y - \phi_i(0)} = \sum_{j=1}^{s} \ell_j \frac{\frac{\partial \mathfrak{F}_j}{\partial y}(0, y)}{\mathfrak{F}_j(0, y)},$$

de sorte que l'hypothèse (N) implique que $\rho_i = \rho_i(0) = \ell_j$ dès que $\mathfrak{F}_j(0, \phi_i(0)) = 0$, d'où

$$G = \sum_{j=1}^{s} \ell_j \hat{\mathfrak{F}}_i \frac{\partial \mathfrak{F}_i}{\partial y}.$$

Soient i et j dans $\{1, \ldots, s\}$ tels que $\mathfrak{F}_i$ et $\mathfrak{F}_j$ divisent le même facteur irréductible F_k de F pour un certain $k \in \{1, \ldots, r\}$. Par construction, $\mathfrak{F}_i$ et $\mathfrak{F}_j$ divisent respectivement $\ell_i \frac{\partial F}{\partial y} - G$ et $\ell_j \frac{\partial F}{\partial y} - G$ dans $\mathbb{K}[[x]][y]$. Puisque ces deux derniers polynômes sont dans $\mathbb{K}[x, y]$, ils sont multiples de F_k. Par conséquent, $\mathfrak{F}_i$ divise $\ell_j \frac{\partial F}{\partial y} - G$ dans $\mathbb{K}[[x]][y]$, d'où $\ell_i = \ell_j$ puisque $\frac{\partial F}{\partial y}$ est inversible modulo $\mathfrak{F}_i$ par l'hypothèse (N). Au final, $(\ell_1, \ldots, \ell_s)$ est une combinaison linéaire sur $\mathbb{F}$ des μ_i. Le conclusion vient alors du lemme 5.4. □

5.5 Caractéristique zéro

En caractéristique 0, pour assurer que tous les ρ_i sont dans $\bar{\mathbb{K}}$, il suffit de garantir que tous les ρ_i' sont identiquement nuls. En posant

$$\begin{aligned}\mathsf{D}\colon\quad & \mathbb{K}[x,y]_{d_x,d_y-1}\to\mathbb{K}[x,y]_{3d_x-1,3d_y-3}\\ & G\mapsto\left(\frac{\partial G}{\partial x}\frac{\partial F}{\partial y}-\frac{\partial G}{\partial y}\frac{\partial F}{\partial x}\right)\frac{\partial F}{\partial y}-\left(\frac{\partial^2 F}{\partial x\,y}\frac{\partial F}{\partial y}-\frac{\partial^2 F}{\partial y^2}\frac{\partial F}{\partial x}\right)G,\end{aligned}$$

les ρ_i' peuvent être calculés comme suit :

$$\begin{aligned}\rho_i'=&\frac{d}{dx}\left(\frac{G(x,\phi_i(x))}{\frac{\partial F}{\partial y}(x,\phi_i(x))}\right)\\ =&\frac{\frac{\partial G}{\partial x}(x,\phi_i(x))+\frac{\partial G}{\partial y}(x,\phi_i(x))\,\phi_i'(x)}{\frac{\partial F}{\partial y}(x,\phi_i(x))}\\ &-\frac{\frac{\partial^2 F}{\partial x\,y}(x,\phi_i(x))+\frac{\partial^2 F}{\partial y^2}(x,\phi_i(x))\,\phi_i'(x)}{\frac{\partial F}{\partial y}(x,\phi_i(x))^2}G(x,\phi_i(x))\\ =&\frac{\mathsf{D}(G)(x,\phi_i(x))}{\frac{\partial F}{\partial y}(x,\phi_i(x))^3},\end{aligned}$$

puisque $\phi_i'(x)=-\frac{\partial F}{\partial x}(x,\phi_i(x))/\frac{\partial F}{\partial y}(x,\phi_i(x))$. Vérifier que tous les ρ_i' sont identiquement nuls revient à tester que F divise $\mathsf{D}(G)$ dans $\mathbb{K}[[x]][y]$. Rappelons que cette division est bien définie puisque c est supposé inversible dans $\mathbb{K}[[x]]$ par l'hypothèse (N). Le lemme suivant nous fournit un algorithme pour tester cette dernière divisibilité.

Lemme 5.6. *Soient Q et R le quotient et le reste de la division de $\mathsf{D}(G)$ par F dans $\mathbb{K}[[x]][y]$, de sorte que nous ayons $\mathsf{D}(G)=Q\,F+R$, avec $\deg_y(R)\leqslant d_y-1$. Alors, sous l'hypothèse (N), F divise $\mathsf{D}(G)$ dans $\mathbb{K}[[x]][y]$ si, et seulement si, $[Q]_{2d_x}^{3d_x}=[R]_0^{3d_x}=0$.*

Démonstration. Supposons que F divise $\mathsf{D}(G)$ dans $\mathbb{K}[[x]][y]$. Alors $R=0$ et $\mathsf{D}(G)=Q\,F$ dans $\mathbb{K}[[x]][y]$. Puisque $\mathsf{D}(G)$ et F sont des polynômes alors $Q\in\mathbb{K}(x)[y]$ et $\mathsf{D}(G)=Q\,F$ dans $\mathbb{K}(x)[y]$. Soit $a\in\mathbb{K}[x]$ et $A\in\mathbb{K}[x,y]$ tels que $Q=A/a$. Nous pouvons écrire $a\,\mathsf{D}(G)=A\,F$ dans $\mathbb{K}[x][y]$. Puisque F est primitif par l'hypothèse (N), le lemme de Gauss [84, Chapter IV, Theorem 2.1] implique que a divise le contenu de A. Par conséquent Q appartient à $\mathbb{K}[x,y]$, d'où $\deg_x(Q)\leqslant 2\,d_x-1$. Réciproquement, supposons $[Q]_{2d_x}^{3d_x}=[R]_0^{3d_x}=0$. Alors nous avons $\mathsf{D}(G)=[Q]_0^{2d_x}F$ dans $\mathbb{K}[[x]]/(x^{3d_x})[y]$. Puisque $\deg_x([Q]_0^{2d_x}F)\leqslant 3\,d_x-1$, nous en déduisons que $\mathsf{D}(G)=[Q]_0^{2d_x}F$ dans $\mathbb{K}[x,y]$, et donc que F divise $\mathsf{D}(G)$ dans $\mathbb{K}[[x]][y]$. □

Lorsque $p>0$ nous écrivons $\bar{\mathbb{K}}[[x^p]]$ pour l'anneau des séries formelles en x^p à coefficients dans $\bar{\mathbb{K}}$. Lorsque $p=0$, par convention, nous posons $\bar{\mathbb{K}}[[x^0]]:=\bar{\mathbb{K}}$. Le lemme 5.6 nous incite à examiner les applications suivantes :

$$\begin{aligned}\mathsf{D}\colon\quad & \mathbb{K}[x,y]_{d_x,d_y-1}\to\mathbb{K}[x,y]_{d_x-1,2d_y-3}\times\mathbb{K}[x,y]_{3d_x-1,d_y-1}\\ & G\mapsto([Q]_{2d_x}^{3d_x}/x^{2d_x},[R]_0^{3d_x}),\\ \mathsf{D}_{\mathbb{F}}\colon\quad & \mathbb{F}^s\to\mathbb{K}[x,y]_{d_x-1,2d_y-3}\times\mathbb{K}[x,y]_{3d_x-1,d_y-1}\\ & (\ell_1,\ldots,\ell_s)\mapsto\mathsf{D}\left(\sum_{i=1}^{s}\ell_i\,\mathfrak{G}_i\right).\end{aligned}$$

Nous introduisons l'hypothèse suivante, qui nous permet de distinguer par la suite le cas de la caractéristique nulle ou suffisamment grande du cas de la petite caractéristique positive :

(C) $\mathbb{K}$ est de caractéristique 0 ou au moins $d_x\,(2\,d_y-1)+1$.

Proposition 5.7. *Nous avons* $\langle \mu_1, \ldots, \mu_r\rangle_{\mathbb{F}} \subseteq \ker(\mathsf{D}_{\mathbb{F}})$. *Réciproquement, si* $(\ell_1, \ldots, \ell_s)$ *appartient à* $\ker(\mathsf{D}_{\mathbb{F}})$ *alors* $\rho_1, \ldots, \rho_{d_y}$ *appartiennent à* $\bar{\mathbb{K}}[[x^p]]$. *De plus, si (C) est satisfaite alors* $\mu_1, \ldots, \mu_r$ *est la base échelonnée réduite de* $\ker(\mathsf{D}_{\mathbb{F}})$ *(à une permutation près).*

Démonstration. Le lemme 5.6 se reformule comme suit : $(\ell_1, \ldots, \ell_s)$ appartient à $\ker(\mathsf{D}_{\mathbb{F}})$ si, et seulement si, $\rho_1, \ldots, \rho_{d_y}$ appartiennent tous à $\bar{\mathbb{K}}[[x^p]]$. Si $(\ell_1, \ldots, \ell_s) = \mu_j$ alors on a $G = \hat{F}_j\,\frac{\partial F_j}{\partial y}$ par (5.3), et donc tous les ρ_j sont dans $\{0, 1\}$, d'où l'inclusion $\langle \mu_1, \ldots, \mu_r\rangle_{\mathbb{F}} \subseteq \ker(\mathsf{D}_{\mathbb{F}})$. Soit $(\ell_1, \ldots, \ell_s) \in \ker(\mathsf{D}_{\mathbb{F}})$ et supposons que (C) soit satisfaite. Pour prouver que $(\ell_1, \ldots, \ell_s) \in \langle \mu_1, \ldots, \mu_r\rangle_{\mathbb{F}}$, il suffit de prouver que tous les ρ_i sont dans $\bar{\mathbb{K}}$, grâce aux lemmes 5.4 et 5.5. Le cas où $p=0$ est clair. Nous pouvons supposer maintenant que $p \geqslant d_x\,(2\,d_y-1)+1$. Soit $i \in \{1, \ldots, d_y\}$ et soit $A \in \bar{\mathbb{K}}[x, y]$ l'unique facteur irréductible de F dans $\bar{\mathbb{K}}[x, y]$ tel que $A(x, \phi_i)=0$. Le résultant

$$B := \mathrm{Res}_y\left(A,\, \rho_i(0)\,\frac{\partial F}{\partial y} - G\right) \in \bar{\mathbb{K}}[x]$$

a pour degré au plus $d_x\,(2\,d_y-1)$ et appartient à (x^p). Il est donc nul puisque $p \geqslant \deg(B)+1$. Par conséquent A divise $\rho_i(0)\,\frac{\partial F}{\partial y} - G$ dans $\bar{\mathbb{K}}[[x]][y]$, et donc $\rho_i(0) = G(x,\, \phi_i)/\frac{\partial F}{\partial y}(x, \phi_i) = \rho_i$. □

Remarque 5.8. Si F est séparable en y, si $G \in \mathbb{K}[x, y]_{d_x, d_y-1}$ et si $H \in \mathbb{K}[x, y]_{d_x-1, d_y}$ vérifient

$$\frac{\partial}{\partial x}\left(\frac{G}{F}\right) = \frac{\partial}{\partial y}\left(\frac{H}{F}\right), \tag{5.5}$$

alors, en écrivant $\frac{G}{F} = \sum_{i=0}^{d_y} \frac{\rho_i}{(y-\varphi_i)}$ et $\frac{H}{F} = \sum_{i=0}^{d_y} \frac{\sigma_i}{(y-\varphi_i)}$, nous obtenons

$$\sum_{i=0}^{d_y}\left(\frac{\rho_i'}{y-\varphi_i} - \frac{\rho_i\,\varphi_i'}{(y-\varphi_i)^2}\right) = -\sum_{i=0}^{d_y}\frac{\sigma_i}{(y-\varphi_i)^2},$$

ce qui implique que tous les ρ_i' sont nuls et donc que $\rho_i \in \bar{\mathbb{K}}$. Ceci fournit une autre méthode pour résoudre le problème de recombinaison comme expliqué dans l'article [85].

Par ailleurs la formule (5.5) définit un système linéaire dont les inconnues sont les coefficients de G et H. En caractéristique 0 ou suffisamment grande, une base des solutions correspond à une base du premier groupe de cohomologie de De Rham du complémentaire de l'hypersurface définie par $F = 0$. Dans l'article [35], Shuhong Gao a montré comment calculer une telle base efficacement et a proposé un algorithme pour en déduire la décomposition irréductible de F dans $\bar{\mathbb{K}}[x, y]$, qui est appelée la *décomposition en facteurs absolument irréductibles* de F. L'algorithme de Shuhong Gao est dans la veine des travaux de Ruppert [122, 123] sur les tests d'irréductibilité. Mentionnons que les résultats de Ruppert sont aussi présentés dans le livre de Schinzel [127, Chapter V]. Pour un bref historique sur le sujet nous renvoyons aussi le lecteur à l'article [23], qui contient par ailleurs un autre algorithme de factorisation absolue inspiré de la méthode de remontée de Newton-Hensel. D'autres méthodes pour la factorisation absolue ont été proposées dans [19, 20, 22].

5.6 Caractéristique positive

La proposition 5.7 réduit le problème de recombinaison à de l'algèbre linéaire sous l'hypothèse (C). Dans cette section nous examinons le cas où (C) n'est pas satisfaite. Nous prenons pour $\mathbb{F}$ le sous-corps premier $\mathbb{F}_p$ de $\mathbb{K}$, et utiliserons les applications $\mathbb{F}$-linéaires suivantes, construites à partir des opérateurs de Berlekamp et Niederreiter introduits au chapitre 3:

$$\begin{aligned} \mathsf{B}: \quad & \mathbb{K}[x,y]_{d_x,d_y-1} \to \mathbb{K}(x)[y]/(F) \\ & G \mapsto G^p - \left(\frac{\partial F}{\partial y}\right)^{p-1} G \end{aligned} \qquad \begin{aligned} \mathsf{N}: \quad & \mathbb{K}[x,y]_{d_x,d_y-1} \to \mathbb{K}[x,y^p]_{pd_x,d_y-1} \\ & G \mapsto G^p + \frac{\partial^{p-1}}{\partial y^{p-1}}(F^{p-1}G). \end{aligned}$$

Ici $\mathbb{K}[x,y^p]_{pd_x,d_y-1}$ doit être lu comme l'espace des polynômes en x et y^p de degré au plus $p\,d_x$ en x et au plus d_y-1 en y^p. Soulignons que N est bien défini puisque $\frac{\partial}{\partial y}\mathsf{N}(G)=0$. Comme pour $\mathsf{D}_{\mathbb{F}}$, nous nous intéressons aux noyaux des applications $\mathbb{F}$-linéaires suivantes :

$$\begin{aligned} \mathsf{B}_{\mathbb{F}}: \quad & \mathbb{F}^s \to \mathbb{K}(x)[y]/(F) \\ & (\ell_1,\ldots,\ell_s) \mapsto \mathsf{B}\left(\sum_{i=1}^{s} \ell_i\,\mathfrak{G}_i\right) \end{aligned} \qquad \begin{aligned} \mathsf{N}_{\mathbb{F}}: \quad & \mathbb{F}^s \to \mathbb{K}[x,y^p]_{pd_x,d_y-1} \\ & (\ell_1,\ldots,\ell_s) \mapsto \mathsf{N}\left(\sum_{i=1}^{s} \ell_i\,\mathfrak{G}_i\right). \end{aligned}$$

Soit $(\ell_1,\ldots,\ell_s)\in\mathbb{F}^s$, $G:=\sum_{i=1}^s \ell_i\,\mathfrak{G}_i$, et continuons de noter $\rho_1,\ldots,\rho_{d_y}$ les résidus de G/F.

Proposition 5.9. *$\mu_1,\ldots,\mu_r$ est la base échelonnée réduite de* $\ker(\mathsf{B}_{\mathbb{F}})=\ker(\mathsf{N}_{\mathbb{F}})$.

Démonstration. La proposition 3.2 appliquée à F vu comme polynôme de $\mathbb{K}(x)[y]$ nous donne que $\ker(\mathsf{B}_{\mathbb{F}})=\ker(\mathsf{N}_{\mathbb{F}})$, et que $(\ell_1,\ldots,\ell_s)$ appartient à ces noyaux si, et seulement si, tous les ρ_i appartiennent à $\mathbb{F}$. Le résultat est donc une conséquence des lemmes 5.4 et 5.5. □

La proposition 5.9 ne fournit pas directement une façon satisfaisante pour résoudre le problème de recombinaison car elle induit un système linéaire de taille croissant linéairement en p. Le lemme suivant explique comment réduire la taille de ce système.

Lemme 5.10. *Si $(\ell_1,\ldots,\ell_s)\in\ker(\mathsf{D}_{\mathbb{F}})$ alors $\mathsf{N}_{\mathbb{F}}(\ell_1,\ldots,\ell_s)$ appartient à $\mathbb{K}[x^p,y^p]_{d_x,d_y-1}$.*

Démonstration. Si $(\ell_1,\ldots,\ell_s)\in\ker(\mathsf{D}_{\mathbb{F}})$ alors $\rho_1,\ldots,\rho_{d_y}$ appartiennent tous à $\bar{\mathbb{K}}[[x^p]]$ par la proposition 5.7. Par conséquent l'égalité (3.1) se réécrit en

$$\mathsf{N}(G)=F^p\left(\sum_{i=1}^{d_y}\frac{\rho_i^p-\rho_i}{(y-\phi_i)^p}\right),$$

ce qui donne $\mathsf{N}_{\mathbb{F}}(\ell_1,\ldots,\ell_s)=\mathsf{N}(G)\in\mathbb{K}[x^p,y^p]_{d_x,d_y-1}$. □

Soit $\Delta_y(x)$ le discriminant de F en y, c'est-à-dire $\mathrm{Res}_y\Big(F(x,y),\frac{\partial F}{\partial y}(x,y)\Big)$.

Proposition 5.11. *Soit S un sous-ensemble de $\bar{\mathbb{K}}$ contenant d_x+1 points, et supposons $(\ell_1,\ldots,\ell_s)\in\ker(\mathsf{D}_{\mathbb{F}})$. Alors nous avons l'équivalence suivante :*

$$\mathsf{N}_{\mathbb{F}}(\ell_1,\ldots,\ell_s)=0 \Longleftrightarrow \forall a\in S,\ \mathsf{N}_{\mathbb{F}}(\ell_1,\ldots,\ell_s)(a,y)=0.$$

De plus, si $c(a)\,\Delta_y(a) \neq 0$ *pour tout* $a \in S$*, alors cette équivalence reste vraie en remplaçant* $\mathsf{N}_{\mathbb{F}}$ *par* $\mathsf{B}_{\mathbb{F}}$*.*

Démonstration. Comme l'application $z \mapsto z^p$ est injective dans $\bar{\mathbb{K}}$, l'annulation de $\mathsf{N}(G)$ peut être testée avec seulement $d_x + 1$ valeurs de spécialisation pour x par le lemme 5.10. Par la proposition 3.2, l'égalité $\mathsf{B}(G) = 0$ est équivalente à $\mathsf{N}(G) = 0$. En appliquant à nouveau la même proposition 3.2 à $F(a, y)$, l'égalité $\mathsf{B}(G)(a, y) = 0$ est équivalente à $\mathsf{N}(G)(a, y) = 0$, pour n'importe quelle valeur de $a \in \bar{\mathbb{K}}$ qui satisfait $c(a)\,\Delta_y(a) \neq 0$. □

5.7 Algorithme de recombinaison

Les propositions 5.7, 5.9 et 5.11 conduisent à l'algorithme de recombinaison suivant :

Algorithme 5.1

Entrée : $F \in \mathbb{K}[x, y]$ satisfaisant (N), et $\mathfrak{F}_1, \ldots, \mathfrak{F}_s$ à la précision (x^{d_x+1}).
Sortie : $\mu_1, \ldots, \mu_r$.

1. Pour chaque $i \in \{1, \ldots, s\}$, calculer $\hat{\mathfrak{F}}_i$ comme le quotient de F par $\mathfrak{F}_i$ à précision (x^{d_x+1}).
2. Calculer $\hat{\mathfrak{F}}_1 \frac{\partial \mathfrak{F}_1}{\partial y}, \ldots, \hat{\mathfrak{F}}_s \frac{\partial \mathfrak{F}_s}{\partial y}$ à précision (x^{d_x+1}) et déduire $\mathfrak{G}_1, \ldots, \mathfrak{G}_s$.
3. Calculer $\mathsf{D}(\mathfrak{G}_1), \ldots, \mathsf{D}(\mathfrak{G}_s)$.
4. Calculer la base échelon réduite $\tilde{\mu}_1, \ldots, \tilde{\mu}_t$ du système linéaire suivant dans les inconnues $(\ell_1, \ldots, \ell_s) \in \mathbb{F}^s$:
$$\sum_{i=1}^{s} \ell_i\, \mathsf{D}(\mathfrak{G}_i) = 0. \tag{5.6}$$
Si (C) est vraie alors retourner $\hat{\mu}_1, \ldots, \hat{\mu}_t$.
5. Pour tout $i \in \{1, \ldots, t\}$, calculer $\tilde{\mathfrak{G}}_i = \sum_{j=1}^{s} \tilde{\mu}_{i,j}\, \mathfrak{G}_j$.
6. Si p divise $d_x + 1$ alors poser $e := d_x + 2$, sinon poser $e := d_x + 1$. Soit ζ la classe de z dans $\mathbb{K}[z]/(z^e - 1)$. Calculer $\mathsf{N}(\tilde{\mathfrak{G}}_1)(\zeta, y), \ldots, \mathsf{N}(\tilde{\mathfrak{G}}_t)(\zeta, y)$.
7. Calculer une base $\nu_1, \ldots, \nu_r$ des solutions du système linéaire suivant dont les inconnues sont $(\ell_1, \ldots, \ell_t) \in \mathbb{F}^t$:
$$\sum_{i=1}^{t} \ell_i\, \mathsf{N}(\tilde{\mathfrak{G}}_i)(\zeta, y) = 0. \tag{5.7}$$
8. Calculer et retourner la base échelonnée réduite de $\langle \sum_{j=1}^{t} \nu_{i,j} \tilde{\mu}_j \mid i \in \{1, \ldots, r\} \rangle_{\mathbb{F}}$.

Proposition 5.12. *Sous l'hypothèse (N) l'algorithme 5.1 est correct.*

a) Si (C) est satisfaite alors l'algorithme 5.1 effectue $O(d_x\ d_y\ s^{\omega-1} + s\ \mathsf{M}(d_x)\ \mathsf{M}(d_y))$ *opérations dans* $\mathbb{K}$*.*

b) Si (C) n'est pas satisfaite alors l'algorithme 5.1 effectue

$$O(\mathsf{M}(d_x)\,(d_y^2\, s^{\omega-2} + \mathsf{M}(d_y)\,(s \log s + d_y + \log(d_x\, d_y)) + s\, d_y \log(d_x\, d_y)))$$

opérations dans $\mathbb{K}$ *plus* $O(\max(e_{\mathsf{D}}, e_{\mathsf{N}}, s)\, s^{\omega-1})$ *opérations dans* $\mathbb{F}_p$*, où* e_{D} *and* e_{N} *représentent le nombre d'équations constituant respectivement les systèmes (5.6) et (5.7).*

Démonstration. Sous l'hypothèse (C) la proposition 5.7 nous donne $\langle \tilde{\mu}_1, ..., \tilde{\mu}_t \rangle = \langle \mu_1, ..., \mu_r \rangle$. Sinon, sans l'hypothèse (C) la proposition 5.7 nous donne seulement $\langle \mu_1, ..., \mu_r \rangle_{\mathbb{F}} \subseteq \langle \tilde{\mu}_1, ..., \tilde{\mu}_t \rangle_{\mathbb{F}}$, ce qui implique que $\langle \mu_1, ..., \mu_r \rangle_{\mathbb{F}}$ est le noyau de la restriction de $\mathsf{N}_{\mathbb{F}}$ à $\langle \tilde{\mu}_1, ..., \tilde{\mu}_t \rangle_{\mathbb{F}}$ par la proposition 5.9. Puisque e est choisi de sorte que $z^e - 1$ admette $e \geqslant d_x + 1$ racines distinctes de $\bar{\mathbb{K}}$, calculer dans $\bar{\mathbb{K}}[z]/(z^e - 1)$ revient à calculer avec e valeurs distinctes de x en parallèle dans l'étape 6. Par conséquent la proposition 5.11 implique que $\nu_1, ..., \nu_r$ est une base du noyau de cette restriction exprimée dans les coordonnées $\bar{\mu}_1$, ..., $\bar{\mu}_t$. La base échelonnée réduite de l'étape 8 est donc bien $\mu_1, ..., \mu_r$.

Analysons maintenant le coût de l'algorithme. Tester si (C) est satisfaite ou non, et obtenir la caractéristique p de $\mathbb{K}$ le cas échéant nécessite $O(d_x\, d_y)$ opérations dans $\mathbb{K}$. Les étapes de 1 à 3 prennent $O(s\, \mathsf{M}(d_x)\, \mathsf{M}(d_y))$ opérations dans $\mathbb{K}$. Si (C) est satisfaite alors le système linéaire (5.6) implique s inconnues et $O(d_x\, d_y)$ équations. Sa résolution nécessite donc $O(d_x\, d_y\, s^{\omega-1})$ opérations dans $\mathbb{K}$ par la proposition 1.30. Ceci achève la preuve de la partie (a).

Supposons maintenant que (C) ne soit pas satisfaite. La résolution du système linéaire (5.7) prend maintenant $O(\max(e_{\mathsf{D}}, s)\, s^{\omega-1})$ opérations dans $\mathbb{F}$ par le corollaire 1.30. Dans l'étape 5, le calcul de chaque $\tilde{\mathfrak{G}}_i$ peut-être réalisé au moyen de la technique de l'arbre des sous-produits utilisant $O(\mathsf{M}(d_x)\, \mathsf{M}(d_y) \log s)$ opérations dans $\mathbb{K}$.

Par la proposition 3.9 le coût de l'étape 6 tombe dans

$$O(\mathsf{M}(d_x)\, (\mathsf{M}(d_y)\, (d_y + \log p) + d_y^2\, t^{\omega-2} + t\, d_y\, (\log d_x + \log p))).$$

La résolution du système linéaire (5.7) coûte $O(\max(e_{\mathsf{N}}, t)\, t^{\omega-1})$ opérations dans $\mathbb{F}$ par le corollaire 1.30. Dans l'étape 8 le produit de matrices et le calcul de la forme échelonnée réduite consomme $O(s^\omega)$ opérations dans $\mathbb{F}$ grâce au lemme 1.29. Finalement la partie (b) s'obtient en prenant la somme des coûts de toutes les étapes en prenant en compte $\log(p) \in O(\log(d_x\, d_y))$. □

Corollaire 5.13. *Si $\mathbb{K} := \mathbb{F}_{p^k}$ alors l'algorithme 5.1 nécessite $\tilde{O}(k\, d_x\, d_y^\omega)$ opérations dans $\mathbb{F}_p$.*

Démonstration. Ce résultat est une conséquence de la proposition précédente puisque $e_{\mathsf{D}} \in O(k\, d_x\, d_y)$ et $e_{\mathsf{N}} \in (k\, d_x\, d_y)$. □

5.8 Algorithme de factorisation

Dans l'algorithme suivant nous résumons les différentes étapes nécessaires à la factorisation de polynômes primitifs et séparables en y.

Algorithme 5.2

Entrée : $F \in \mathbb{K}[x, y]$ primitif et séparable en y.
Sortie : les facteurs irréductibles $F_1, ..., F_r$ de F.

1. Trouver $b \in \mathbb{K}$ tel que $F(b, y)$ est séparable de degré d_y. Remplacer F par $F(x + b, y)$.
2. Calculer la factorisation irréductible de $F(0, y)$.
3. Calculer $\mathfrak{F}_1, ..., \mathfrak{F}_s$ à la précision (x^{2d_x+1}) par remontée de Newton-Hensel.
4. Recombiner les facteurs analytiques au moyen de l'algorithme 5.1 de sorte à obtenir les facteurs irréductibles $F_1, ..., F_r$ de F.
5. Pour tout $i \in \{1, ..., r\}$ remplacer F_i par $F_i(x - b, y)$.

Proposition 5.14. *Si F est primitif et séparable en y, et que $\mathbb{K}$ contient au moins $2\,d_x\,d_y+1$ éléments, alors l'algorithme 5.2 est correct et utilise une factorisation irréductible dans $\mathbb{K}[y]$ en degré d_y, plus*

a) si (C) est satisfaite, $O(d_x\,d_y^\omega)$ opérations dans $\mathbb{K}$;
b) sinon, si $\mathbb{K}:=\mathbb{F}_{p^k}$, $\tilde{O}(k\,d_x\,d_y^\omega)$ operations dans $\mathbb{F}_p$.

Démonstration. Puisque $\deg(c\,\Delta_y)\leqslant 2\,d_x\,d_y$, l'hypothèse sur le cardinal de $\mathbb{K}$ nous assure de trouver une valeur b convenable. Cette recherche utilise $\tilde{O}(d_x\ d_y^2)$ opérations *via* les algorithmes rapides d'évaluation en plusieurs points. La translation de la variable x peut être réalisée en temps quasi-linéaire. Dans les étapes de 2 à 4 l'hypothèse (N) est satisfaite. Le calcul de $\mathfrak{F}_1,\ldots,\mathfrak{F}_s$ à la précision (x^{2d_x+1}) à partir des facteurs irréductibles de $F(0,y)$ coûte $\tilde{O}(d_x\,d_y)$ opérations dans $\mathbb{K}$ grâce à la proposition 4.11. La partie (a) est donc une conséquence de la proposition 5.12. La partie (b) est obtenue grâce au corollaire 5.13.

Soit $i\in\{1,\ldots,r\}$, le calcul de F_i à partir de μ_i est réalisé en calculant la partie primitive de $c\,\mathfrak{F}_1^{\mu_{i,1}}\cdots\mathfrak{F}_s^{\mu_{i,s}}$ calculé à la précision (x^{2d_x+1}), puis vu comme un polynôme de $\mathbb{K}[x][y]$. □

5.9 Algorithme complet de factorisation

En utilisant les résultats du chapitre 2, nous obtenons l'algorithme complet suivant :

Algorithme 5.3

Entrée : $F\in\mathbb{K}[x,y]$.
Sortie : les facteurs irréductibles $F_1,\ldots,F_r$ de F et leur multiplicité respective $e_1,\ldots,e_r$.

1. Calculer le contenu et la partie primitive de F en y. Remplacer F par sa partie primitive et initialiser la liste L avec la factorisation irréductible du contenu.
2. Calculer la décomposition séparable $(G_1,q_1,m_1),\ldots,(G_s,q_s,m_s)$ de F.
3. Calculer la décomposition irréductible de $G_1,\ldots,G_s$.
4. Pour tout $i\in\{1,\ldots,s\}$, et pour tout facteur irréductible E de G_i faire :
   a) Si $p=0$ alors ajouter (E,m_i) à la liste L,
   b) sinon calculer $(H,h)\in\mathbb{K}[x,y]\times\mathcal{B}$ tel que $H^h(y^{q_i/h})=E(y^{q_i})$, avec $h\leqslant q_i$ et h aussi grand que possible, et ajouter $(H(y^{q_i/h}),h\,m_i)$ à L.
5. Retourner L.

Proposition 5.15. *Si $\mathbb{K}$ contient au moins $2\,d_x\,d_y+d_x+1$ éléments, alors l'algorithme 5.3 calcule la factorisation irréductible de polynômes à une variable sur $\mathbb{K}$ dont la somme des degrés est bornée par d_x+d_y, plus*

a) si (C) est satisfaite, $O(d_x\,d_y^\omega)$ opérations arithmétiques dans $\mathbb{K}$;
b) si $\mathbb{K}:=\mathbb{F}_{p^k}$, $\tilde{O}(k\,d_x\,d_y^\omega)$ opérations arithmétiques dans $\mathbb{F}_p$.

Démonstration. Le calcul de la partie primitive et du contenu peut être fait en temps quasi-linéaire. L'étape 2 prend $\tilde{O}(d_x\,d_y^2)$ par la proposition 2.26 grâce à l'hypothèse sur le cardinal de $\mathbb{K}$. L'étape 3 peut être réalisée par l'algorithme 5.2 : la proposition 5.14(a) donne un coût total dans $O(d_x\,d_y^\omega)$, ce qui prouve la partie (a).

Pour la partie (b), le calcul de chaque paire (H,h) coûte $O(\deg_x(E)\deg_y(E))$ extractions de racines p-ièmes dans $\mathbb{F}_{p^k}$. Chacune de ces extractions dans $\mathbb{F}_{p^k}$ requiert $O((k-1)\log p)$ opérations arithmétiques dans $\mathbb{F}_{p^k}$. La preuve découle enfin de la proposition 5.14(b). □

Remarque 5.16. Remarquons que, sans perte de généralité, nous pouvons supposer que $d_y \leqslant d_x$, ce qui conduit à un coût $\tilde{O}(d_x\, d_y^{\omega}) \subseteq \tilde{O}((d_x\, d_y)^{(\omega+1)/2})$. De cette façon nous obtenons un algorithme pour réduire la factorisation de deux à une variable en temps moins que quadratique.

Le problème linéaire intervenant dans l'algorithme 5.1 est très largement surdéterminé. En appliquant des techniques classiques probabilistes il est possible d'accélérer la résolution du système. Comme il est aussi possible de tester rapidement si les facteurs candidats divisent bien le polynôme de départ, alors nous obtenons le résultat suivant, dont nous omettons la preuve :

Théorème 5.17. *Si* $\mathbb{K}$ *contient au moins* $10\, d_x\, d_y$ *éléments, alors le calcul de la décomposition irréductible* $(F_1, e_1), \ldots, (F_r, e_r)$ *de* F *peut-être obtenu au moyen d'un algorithme qui nécessite de factoriser les polynômes à une variable à coefficients dans* $\mathbb{K}$ *dont la somme des degrés n'excède pas* $d_x + d_y$*, plus*

a) si (C) est satisfaite, $O((d_x\, d_y)^{1,5})$ *opérations dans* $\mathbb{K}$ *;*
b) si $\mathbb{K} := \mathbb{F}_{p^k}$*,* $\tilde{O}(k\,(d_x\, d_y)^{1,5})$ *opérations dans* $\mathbb{F}_p$*.*

L'algorithme retourne soit rien soit un résultat correct avec une probabilité au moins $1/2$*.*

Problème ouvert 5.1. Est-il possible d'améliorer les exposants 1,5 dans l'énoncé du théorème précédent ?

5.10 Petite cardinalité

Si $\mathbb{K}$ est un corps fini $\mathbb{F}_q$ de cardinal trop petit pour pouvoir appliquer les résultats de la section précédente, alors nous pouvons procéder comme suit. Tout d'abord nous construisons une extension algébrique $\mathbb{F}_{q^l}$ de $\mathbb{F}_q$ de degré l sous la forme $\mathbb{F}_q[z]/(\mu(z))$, avec μ irréductible de degré l. Notons α une racine de μ dans $\mathbb{F}_{q^l}$. En prenant $l \in O(\log_q(d_x\, d_y))$ suffisamment grand pour pouvoir factoriser F dans $\mathbb{F}_{q^l}$ avec les algorithmes de la section précédente, nous pouvons obtenir les facteurs irréductibles $\mathcal{F}_1, \ldots, \mathcal{F}_t$ de F dans $\mathbb{F}_{q^l}$. En prenant les préimages des coefficients des $\mathcal{F}_i$ dans $\mathbb{F}_q[z]$, nous construisons des polynômes $\mathsf{F}_i(x, y, z)$ de degré au plus $l-1$ en z, tels que $\mathcal{F}_i(x, y) = \mathsf{F}_i(x, y, \alpha)$.

Pour chaque $i \in \{1, \ldots, t\}$, nous pouvons calculer $F_i(x, y) := \mathrm{Res}_z(\mathsf{F}_i(x, y, z), \mu(z))$ qui est un facteur irréductible de F. Chaque F_i peut être obtenu par évaluation/interpolation rapide en temps $\tilde{O}(\deg_x(F_i)\deg_y(F_i)\, l)$ opérations dans $\mathbb{F}_{q^l}$. Au total tous les F_i sont obtenus en temps quasi-linéaire. Chaque facteur irréductible de F apparaît exactement l fois dans la liste des F_i ainsi calculés. Les redondances sont éliminées simplement en triant la liste des F_i. Au final le surcoût total reste un facteur borné par $\log^{O(1)}(d_x\, d_y)$.

6

Polynômes à plusieurs variables

L'algorithme de factorisation des polynômes à deux variables du chapitre précédent peut être utilisé directement pour traiter le cas à plusieurs variables. En effet, si F est un polynôme de $\mathbb{K}[z_1, ..., z_n, y]$, alors il peut être vu comme un polynôme de $\mathbb{K}[z_1, ..., z_{n-1}][z_n, y]$, dont nous pouvons factoriser son contenu en y par récurrence sur le nombre de variables, puis factoriser sa partie primitive en y dans $\mathbb{K}(z_1, ..., z_{n-1})[z_n, y]$. L'inconvénient de cette approche est double. Premièrement les calculs dans $\mathbb{K}(z_1, ..., z_{n-1})$ sont coûteux car chaque opération sur les fractions nécessite un calcul de p.g.c.d. pour s'assurer que les numérateurs et dénominateurs sont premiers entre eux. Deuxièmement, lors de la phase de remontée de Newton-Hensel des facteurs dans $\mathbb{K}(z_1, ..., z_{n-1})[[z_n]][y]$, la taille des fractions rationnelles croît rapidement. Il est alors préférable de procéder plutôt ainsi :

1. Factoriser le polynôme $F(0, ..., 0, y)$.
2. Remonter les facteurs de sorte à obtenir la factorisation de F dans $\mathbb{K}[[z_1, ..., z_n]][y]$ à la precision $(z_1, ..., z_n)^{2d_z+1}$, où $\mathbb{K}[[z_1, ..., z_n]]$ représente l'algèbre des séries à n variables, et d_z est le degré total de F en les seules variables $z_1, ..., z_n$.
3. Résoudre le problème de recombinaison pour trouver les facteurs rationnels de F.

En fait, dans ce chapitre, nous présentons une autre technique attribuée à Hilbert, assurant que, sous certaines hypothèses, les facteurs irréductibles de F sont en correspondance avec ceux de $F(\alpha_1 x, ..., \alpha_n x, y)$ pour presque toutes les valeurs de $\alpha_1, ..., \alpha_n$. La factorisation se ramène alors au cas de deux variables, mais il reste à préciser la probabilité de faire des bons choix pour les α_i.

6.1 Réduction à deux variables

Nous commençons par examiner le problème de réduction à deux variables sous l'hypothèse suivante, généralisant l'hypothèse (N) du chapitre précédent :

$$\text{(N)} \quad \begin{cases} \deg\left(F(0, ..., 0, y)\right) = d_y, \\ \operatorname{Res}\Big(F(0, ..., 0, y), \frac{\partial F}{\partial y}(0, ..., 0, y)\Big) \neq 0, \\ F \text{ est primitif vu dans } \mathbb{K}[z_1, ..., z_n][y]. \end{cases}$$

Définition 6.1. *Sous l'hypothèse (N), un point* $(\alpha_1, ..., \alpha_n) \in \mathbb{K}^n$ *est dit de* Hilbert *si, pour tout facteur irréductible* F_i *de* F*, le polynôme* $F_i\,(\alpha_1\,x,\,...,\,\alpha_n\,x,\,y)$ *est irréductible. En d'autres termes, les facteurs irréductibles de* F *sont en bijection avec ceux de* $H(x, y) := F\,(\alpha_1\,x, ..., \alpha_n\,x, y) \in \mathbb{K}[x, y]$.

D'un point de vue pratique, la donnée d'un point de Hilbert conduit naturellement à l'algorithme suivant :

1. Factoriser $H(x, y)$ dans $\mathbb{K}[x, y]$.
2. Remonter la factorisation pour retrouver les facteurs $F_1, ..., F_r$ de F.

Le complémentaire de l'ensemble des points de Hilbert de $\mathbb{K}^n$ est noté $\mathcal{H}(F)$. Commençons avec un exemple nous fournissant une borne inférieure sur la densité de $\mathcal{H}(F)$.

Exemple 6.2. Soit $n \geqslant 2$, $\mathbb{K} := \mathbb{C}$, $F := y^d + z_1^{d-1}\,y - z_2^{d-1} - 1$. Un critère d'irréductibilité attribué à Stepanov et Schmidt implique que $F(0,\,z_2,\,0,\,...,\,0,\,y) = y^d - z_2^{d-1} - 1$ est irréductible, et donc que F lui-même est irréductible [34, p. 503]. Soit S l'ensemble des racines de $z^{d(d-1)} - 1$. Pour n'importe quel point $(\alpha_1,\,...,\,\alpha_n) \in S^n$, le polynôme $y - (\alpha_2/\alpha_1)^{d-1}$ divise $H = y^d - 1 + x^{d-1}\,(\alpha_1^{d-1}\,y - \alpha_2^{d-1})$. Par conséquent, aucun des points de S^n n'est un point de Hilbert pour F, ce qui donne les égalités suivantes :

$$\frac{|\mathcal{H}(F) \cap S^n|}{|S|^n} = \frac{d\,(d-1)}{|S|} = 1.$$

Par le lemme classique de Schwartz-Zippel [130, 152] : un polynôme non nul A en n variables ne peut avoir plus de $\deg(A)|S|^{n-1}$ racines dans S^n. Nous en déduisons qu'il n'existe aucun polynôme A de degré au plus $d\,(d-1) - 1$ qui s'annule sur tout les points de $\mathcal{H}(F)$. Le théorème suivant donne une borne supérieure pour le degré d'un tel polynôme A s'annulant sur tout $\mathcal{H}(F)$.

Lemme 6.3. *Sous l'hypothèse (N), si* F *est irréductible, alors* $F(a_1\,x,\,...,\,a_n\,x,\,y)$ *est irréductible dans* $\mathbb{K}(a_1, ..., a_n)[x, y]$.

Démonstration. Posons $G(a_1,\,...,\,a_n,\,x,\,y) := F(a_1\,x,\,...,\,a_n\,x,\,y)$. Comme G est primitif en y et appartient à $\mathbb{K}[a_1,\,...,\,a_n,\,x,\,y]$, il suffit de prouver qu'il est irréductible dans $\mathbb{K}[a_1, ..., a_n, x, y]$. Comme $F(0, ..., 0, y) = G(0, ..., 0, 0, y)$, les facteurs irréductibles $\mathfrak{F}_1, ..., \mathfrak{F}_s$ de F dans $\mathbb{K}[[z_1, ..., z_n]][y]$ sont en bijection avec ceux de G dans $\mathbb{K}(a_1, ..., a_n)[[x]][y]$ notés $\mathfrak{G}_1$, ..., $\mathfrak{G}_s$, précisément nous avons $\mathfrak{G}_i = \mathfrak{F}_i(a_1\,x,\,...,\,a_n\,x,\,y)$. Par conséquent tout facteur irréductible de G dans $\mathbb{K}[a_1, ..., a_n, x, y]$ est nécessairement de la forme $G(a_1\,x, ..., a_n\,x, y)$ avec $G \in \mathbb{K}[z_1, ..., z_n, y]$. En remplaçant x par 1, le polynôme $G(a_1, ..., a_n, y)$ est un facteur de F. Il en découle que G est bien irréductible. □

Théorème 6.4. *Sous l'hypothèse (N), il existe un polynôme de* $\mathbb{K}[a_1, ..., a_n] \setminus \{0\}$ *de degré au plus* $d_z\,(d_y - 1)\,(2\,d_y - 1)$ *qui s'annule sur tout* $\mathcal{H}(F)$.

Démonstration. Puisque la fonction $\delta : d \mapsto (d-1)\,(2\,d-1)$ satisfait $\delta(d_1) + \delta(d_2) \leqslant \delta(d_1 + d_2)$, pour n'importe quels entiers strictement positifs d_1 et d_2, il suffit de prouver le résultat en supposant F irréductible. Supposons d'abord que $\alpha_1, ..., \alpha_n$ soient des indéterminées. Alors $H(x, y) := F(\alpha_1\,x, ..., \alpha_n\,x, y) \in \mathbb{K}(\alpha_1, ..., \alpha_n)[x, y]$ est irréductible par le lemme précédent.

Notons $\mathfrak{H}_1(x,y),...,\mathfrak{H}_s(x,y)$ les facteurs analytiques de H dans $\mathbb{K}(\alpha_1,...,\alpha_n)[[x]][y]$. En posant $\hat{\mathfrak{H}}_i := H/\mathfrak{H}_i$, et $\sigma := d_x\,(2\,d_y-1)+1$, la proposition 5.3 affirme que le système linéaire suivant en les inconnues $\ell_1,...,\ell_s$ a pour rang $s-1$:

$$\left[\sum_{j=1}^{s} \ell_j\,\hat{\mathfrak{H}}_j\,\frac{\partial\mathfrak{H}_j}{\partial y}\right]_{d_x+1}^{\sigma} = 0.$$

Il existe par conséquent un mineur non nul $A\in\mathbb{K}[a_1,...,a_n]$ de taille $s-1$. Si maintenant $\alpha_1,...,\alpha_n$ sont des valeurs dans $\mathbb{K}$ telles que $A(\alpha_1,...,\alpha_n)\neq 0$ alors les mêmes calculs restent valables et le système linéaire ainsi spécialisé reste de rang $s-1$.

Pour chaque $j\geqslant 0$ et $k\geqslant 0$, le coefficient de $x^j\,y^k$ dans $\hat{\mathfrak{H}}_l\,\frac{\partial\mathfrak{H}_l}{\partial y}$ est un polynôme de degré au plus j. Le degré total de A est donc au plus $(s-1)\,d_z\,(2\,d_y-1)$. □

Dans les paragraphes suivants nous améliorons ce résultat en caractéristique suffisamment grande ou suffisamment petite.

Théorème 6.5. *Sous l'hypothèse (N), si la caractéristique de $\mathbb{K}$ est nulle ou bien au moins $d_z\,(2\,d_y-1)+1$, alors il existe un polynôme de $\mathbb{K}[a_1,...,a_n]\setminus\{0\}$ de degré au plus $(3\,d_z-1)\,(d_y-1)$ qui s'annule sur tout $\mathcal{H}(F)$.*

Démonstration. Il suffit de prouver le résultat en supposant F irréductible. Supposons d'abord que α_1, ..., α_n soient des indéterminées. Alors $H(x,\,y) := F(\alpha_1\,x,\,...,\,\alpha_n\,x,\,y)\in\mathbb{K}(\alpha_1,...,\alpha_n)[x,y]$ est irréductible par le lemme précédent.

Notons $\mathfrak{H}_1(x,y),...,\mathfrak{H}_s(x,y)$ les facteurs analytiques de H dans $\mathbb{K}(\alpha_1,...,\alpha_n)[[x]][y]$. En posant $\hat{\mathfrak{H}}_i := H/\mathfrak{H}_i$, et $\mathfrak{G}_i := \left[\hat{\mathfrak{H}}_i\,\frac{\partial\mathfrak{H}_i}{\partial y}\right]_0^{d_z+1}$, la proposition 5.7 affirme que le système linéaire suivant en les inconnues $\ell_1,...,\ell_s$ a pour rang $s-1$:

$$\sum_{j=1}^{s} \ell_j\,\mathsf{D}(\mathfrak{H}_i) = 0.$$

Il existe par conséquent un mineur non nul $A\in\mathbb{K}[a_1,...,a_n]$ de taille $s-1$. Si maintenant $\alpha_1,...,\alpha_n$ sont des valeurs dans $\mathbb{K}$ telles que $A(\alpha_1,...,\alpha_n)\neq 0$ alors les mêmes calculs restent valables et le système linéaire ainsi spécialisé reste de rang $s-1$.

Pour chaque $j\geqslant 0$ et $k\geqslant 0$, le coefficient de $x^j\,y^k$ dans $\mathsf{D}(\mathfrak{H}_l)$ est un polynôme de degré au plus j. Le degré total de A est donc au plus $(s-1)\,(3\,d_z-1)$. □

Théorème 6.6. *Sous l'hypothèse (N), si la caractéristique p de $\mathbb{K}$ est strictement positive, alors il existe un polynôme de $\mathbb{K}[a_1,...,a_n]\setminus\{0\}$ de degré au plus $(p\,d_z-1)\,(d_y-1)$ qui s'annule sur tout $\mathcal{H}(F)$.*

Démonstration. La preuve est similaire à celle du théorème précédent en utilisant l'opérateur N et la proposition 5.9. □

Corollaire 6.7. *Sous l'hypothèse (N), en notant p la caractéristique de $\mathbb{K}$, pour n'importe quel sous-ensemble non vide S de $\mathbb{K}$, nous avons les inégalités suivantes :*

– $\frac{|\mathcal{H}(F)\cap S^n|}{|S|^n} \leqslant \frac{3\,d_z\,d_y}{|S|}$ *si* $p=0$ *ou* $p\geqslant d_z\,(2\,d_y-1)+1$,

– $\frac{|\mathcal{H}(F)\cap S^n|}{|S|^n} \leqslant \frac{d_z\, d_y \min(2\, d_y, p)}{|S|}$ *si* $p > 0$.

Démonstration. Il s'agit d'une conséquence des théorèmes précédents en utilisant le lemme de Schwartz-Zippel. □

En d'autres termes, ce corollaire assure qu'il est nécessaire et suffisant de choisir S de taille suffisamment grande devant $d_z\, d_y^2$ pour s'assurer de trouver un point de Hilbert avec une forte probabilité.

Problème ouvert 6.1. Est-il possible d'améliorer le corollaire précédent pour obtenir une borne en $O(d_z\, d_y)/|S|$ même dans le cas de la caractéristique positive ?

Formellement, les résultats de cette section sont nouveaux, mais il s'agit juste de variantes de ceux de l'article [86].

6.2 Algorithme de factorisation

En supposant que nous disposons d'algorithmes pour effectuer les opérations élémentaires sur les séries à une précision donnée, alors nous obtenons l'algorithme suivant :

Algorithme 6.1

Entrée : F satisfaisant l'hypothèse (N) et $(\alpha_1, ..., \alpha_n) \in \mathbb{K}^n$ un point de Hilbert pour F.
Sortie : les facteurs irréductibles $F_1, ..., F_r$ de F.

1. Calculer les facteurs irréductibles $H_1(x, y), ..., H_r(x, y)$ de $H(x, y) := F(\alpha_1\, x, ..., \alpha_n\, x, y)$.
2. Utiliser la remontée de Newton-Hensel pour calculer la factorisation de F.

Supposons que, pour toutes les valeurs d et n, nous disposons d'arbres de calcul effectuant les additions et soustractions dans $\mathbb{K}[[z_1, ..., z_n]]/(z_1, ..., z_n)^d$ en temps linéaire, ainsi que le produit en temps au plus $\mathsf{S}(d, n)$. Pour des raisons techniques nous supposons que S est super-additive en d, c'est à dire $\mathsf{S}(d_1, n) + \mathsf{S}(d_2, n) \leqslant \mathsf{S}(d_1 + d_2, n)$. Sous l'hypothèse (N), si $(\alpha_1, ..., \alpha_n)$ est un point de Hilbert pour F, alors l'algorithme 6.1 effectue la factorisation d'un polynôme à deux variables de degrés partiels d_z et d_y. Puis la remontée de Newton-Hensel peut se faire jusqu'en précision $d_z + \deg_z(\mathrm{lc}_y(F)) + 1$ au moyen de

$$O(\mathsf{S}(d_z + \deg_z(\mathrm{lc}_y(F) + 1), n)\, \mathsf{M}(d_y) \log d_y)$$

opérations dans $\mathbb{K}$ grâce à l'algorithme 4.1. Si $\mathcal{F}$ est l'un des facteurs unitaires remontés, alors $\mathrm{lc}_y(F)\, \mathcal{F}$ est l'image d'un polynôme dont il suffit de prendre la partie primitive pour obtenir un facteur irréductible de F.

Remarque 6.8. Lorsque F est unitaire en y et lorsque le produit de séries a un coût quasi-linéaire, alors, dès que $n \geqslant 2$, l'algorithme 6.1 permet de réduire le problème de factorisation de 3 à 1 variable en temps quasi-linéaire en moyenne grâce au théorème 5.17.

Pour obtenir un algorithme complet de factorisation, il faut considérer, comme dans le chapitre précédent pour deux variables, un algorithme pour calculer le contenu et la partie primitive de F en y, un algorithme pour la factorisation séparable en y et un algorithme pour se ramener à l'hypothèse (N). Nous omettons ici l'analyse détaillée des coûts de ces opérations qui dépendent largement des opérations élémentaires sur les polynômes et séries à plusieurs variables. Par exemple si $\mathbb{K}$ est de caractéristique nulle, alors l'algorithme présenté dans l'article [89] fournit un produit de séries en *représentation dense* en temps quasi-linéaire. Ici la représentation dense signifie que chaque série en n variables et précision d est représentée par le vecteur de tous ses coefficients dans la base des monômes jusqu'en degré total $d-1$. En caractéristique positive plusieurs algorithmes rapides sont présentés dans l'article [55]. Cet article contient aussi la description des algorithmes classiques pour les représentations dense et creuse des polynômes et séries, avec une bibliographie récente sur le sujet.

6.3 Théorème de Bertini-Hilbert

Dans cette section nous abordons une autre façon pour se ramener à l'hypothèse (N). Nous utilisons un changement de variables le plus général possible pour ramener le problème à deux variables. D'un point de vue géométrique cela revient à construire l'équation de l'intersection de l'hypersurface définie par $F=0$ avec un plan générique. Pour éviter d'éventuelles confusions avec les notations, nous considérons dans cette section un polynôme P de degré $d\geqslant 1$ en les variables $v_1,...,v_n$ sur $\mathbb{K}$. Pour tout triplet de points $(\alpha_1,...,\alpha_n)$, $(\beta_1,...,\beta_n)$ et $(\gamma_1,...,\gamma_n)$ de $\mathbb{K}^n$, nous définissons le polynôme à deux variables x et y suivant :

$$P_{\alpha,\beta,\gamma} := P(\alpha_1\,x+\beta_1\,y+\gamma_1,...,\alpha_n\,x+\beta_n\,y+\gamma_n). \tag{6.1}$$

Selon un résultat classique souvent appelé théorème d'irréductibilité de Bertini [134, Chapter II, Section 6.1], si P est irréductible, alors il existe un ouvert de Zariski de $(\mathbb{K}^n)^3$ tel que $P_{\alpha,\beta,\gamma}$ est irréductible pour chaque triplet $(\alpha_1,\ ...,\ \alpha_n)$, $(\beta_1,\ ...,\ \beta_n)$, $(\gamma_1,\ ...,\ \gamma_n)$ dans cet ouvert. Nous dirons qu'un tel triplet est un *point de Bertini* pour P si, pour tout facteur irréductible Q de P, le polynôme $Q(\alpha_1\ x\ +\ \beta_1\ y\ +\ \gamma_1,\ ...,\ \alpha_n\ x\ +\ \beta_n\ y\ +\ \gamma_n)$ est irréductible de même degré total que Q. En d'autres termes les facteurs irréductibles de P sont en bijection avec ceux de $P_{\alpha,\beta,\gamma}$.

D'un point de vue pratique les coefficients des vecteurs $(\alpha_1,...,\alpha_n)$, $(\beta_1,...,\beta_n)$ et $(\gamma_1,...,\gamma_n)$ doivent être choisis dans un sous-ensemble fini S de $\mathbb{K}$, et il nous faut donner une estimation de la probabilité qu'un triplet pris au hasard dans $(S^n)^3$ soit un point de Bertini pour P.

Théorème 6.9. *Soit S un sous-ensemble fini de $\mathbb{K}$. Si $P\in\mathbb{K}[v_1,...,v_n]$ est séparable en au moins une variable et est de degré total $d\geqslant 1$, alors la proportion $\mathsf{B}(P,S)$ de triplets à coordonnées dans S qui ne sont pas des points de Bertini pour P est au plus*

- $5\,d^2/|S|$ *si la caractéristique p de $\mathbb{K}$ est nulle ou au moins $d\,(2\,d-1)+1$,*
- $d^2\min(2\,d,p+1)/|S|$ *si $p>0$.*

Démonstration. Nous pouvons supposer que P est irréductible. Soient $w_1,...,w_n$, $z_1,...,z_n$ de nouvelles variables. Pour tout $(\beta_1,...,\beta_n)$ et $(\gamma_1,...,\gamma_n)$ posons

$$P_\beta := P(w_1+\beta_1\,y,...,w_n+\beta_n\,y)\in\mathbb{K}[w_1,...,w_n,y],$$
$$P_{\beta,\gamma} := P_\beta(z_1+\gamma_1,...,z_n+\gamma_n,y)\in\mathbb{K}[z_1,...,z_n,y].$$

Les vecteurs $(\beta_1, ..., \beta_n) \in \mathbb{K}^n$ tels que P_β n'est pas séparable en y satisfont

$$\frac{\partial P_\beta}{\partial y}(w_1, ..., w_n, y) = \sum_{i=1}^{n} \beta_i \frac{\partial P}{\partial z_i}(w_1 + \beta_1 y, ..., w_n + \beta_n y) = 0.$$

Comme P est irréductible et séparable en au moins une variable, disons z_1, alors $\frac{\partial P}{\partial z_1}(w_1 + \beta_1 y, ..., w_n + \beta_n y)$ est non nul quelles que soient les valeurs des β_i. Il existe donc un monôme de $\frac{\partial P_\beta}{\partial y}$ dont le coefficient est un polynôme B non nul en les β_i de degré total au plus d. Fixons des valeurs pour les β_i telles que $B(\beta_1, ..., \beta_n) \neq 0$. Le polynôme P_β est irréductible, primitif en y et séparable en y.

Le discriminant C_β de P_β en y n'est pas nul et est de degré au plus $d\,(d-1)$. Pour tout $(\gamma_1, ..., \gamma_n) \in \mathbb{K}^n$ tel que $C_\beta(\gamma_1, ..., \gamma_n) \neq 0$, le polynôme $F := P_{\beta,\gamma}$ satisfait l'hypothèse (N). Une fois un tel $(\gamma_1, ..., \gamma_n)$ fixé, les théorèmes 6.4, 6.5 et 6.6 assurent l'existence d'un polynôme non nul $A_{\beta,\gamma} \in \mathbb{K}[a_1, ..., a_n]$ de degré au plus $(3\,d-1)\,(d-1)$ si $p = 0$ ou $p \geqslant d\,(2\,d-1)+1$, ou au plus $(d-1)\min(d\,(2\,d-1), p\,d-1)$ sinon, et tel que tout $(\alpha_1, ..., \alpha_n) \in \mathbb{K}^n$ satisfaisant $A_{\beta,\gamma}(\alpha_1, ..., \alpha_n) \neq 0$ conduise à un polynôme $P_{\beta,\gamma}(\alpha_1 x, ..., \alpha_n x, y)$ qui est irréductible. Les triplets qui ne sont pas de Bertini pour P sont par conséquent contenus dans l'ensemble suivant :

$$\begin{aligned}
&\mathbb{K}^n \times \{(\beta_1, ..., \beta_n) |\, B(\beta_1, ..., \beta_n) = 0\} \times \mathbb{K}^n \\
\cup\, &\mathbb{K}^n \times \{(\beta_1, ..., \beta_n), (\gamma_1, ..., \gamma_n) |\, B(\beta_1, ..., \beta_n) \neq 0, C_\beta(\gamma_1, ..., \gamma_n) = 0\} \\
\cup\, &\{(\alpha_1, ..., \alpha_n), (\beta_1, ..., \beta_n), (\gamma_1, ..., \gamma_n) | \\
&\qquad B(\beta_1, ..., \beta_n) \neq 0, C_\beta(\gamma_1, ..., \gamma_n) \neq 0, A_{\beta,\gamma}(\alpha_1, ..., \alpha_n) = 0\}.
\end{aligned}$$

Finalement, en utilisant le lemme de Schwartz-Zippel avec B, C_β et $A_{\beta,\gamma}$, le nombre de triplets à coordonnées dans S qui ne sont pas de Bertini est borné par

$$d\,|S|^{3n-1} + d\,(d-1)\,|S|^{3n-1} + (3\,d-1)\,(d-1)\,|S|^{3n-1} \leqslant 5\,d^2\,|S|^{3n-1}$$

si $p = 0$ ou $p \geqslant d\,(2\,d-1)+1$, ou bien

$$d\,|S|^{3n-1} + d\,(d-1)\,|S|^{3n-1} + (d-1)\min(d\,(2\,d-1), p\,d-1)\,|S|^{3n-1}$$

si $p > 0$. Cette dernière quantité est au plus $d^2 \min(2\,d, p+1)\,|S|^{3n-1}$. □

Ce que nous appelons ici théorème de Bertini est plutôt un cas particulier du résultat plus général présenté par exemple dans le livre [134, Chapter II, Section 6.1]. Pour des renseignements historiques et techniques nous renvoyons le lecteur à [57, 81]. En fait, ce cas particulier semble dû à Hilbert [51, p. 117], comme souligné dans l'article [69]. Pour cette raison il est fréquent de trouver les terminologies « théorème de Hilbert » ou « théorème de Bertini » dans la littérature sur la factorisation des polynômes.

L'utilisation du théorème de Bertini en calcul formel a été mise en avant pour la première fois dans les articles [48, 59]. Son utilisation s'est rapidement imposée dans de nombreux algorithmes et études de complexité [38, 42, 60, 61, 62, 63]. En caractéristique quelconque, et sous une hypothèse de normalisation similaire à notre hypothèse (N), von zur Gathen avait montré que les points qui ne sont pas de Hilbert pour un polynôme donné de degré total d sont inclus dans une hypersurface de degré au plus 9^{d^2} [42].

Lorsque $\mathbb{K}$ est le corps des nombres complexes, l'article [5] obtient la borne $\mathsf{B}(P,S) \leqslant (d^4 - 2\,d^3 + d^2 + d + 1)/|S|$ en suivant la preuve du théorème de Bertini donnée dans [98, Theorem 4.17]. Pour un corps parfait, Kaltofen avait obtenu la borne $\mathsf{B}(P,S) \leqslant 2\,d^4/|S|$ dans son article [69]. Lorsque $p = 0$ ou $p \geqslant 2\,d^2$, Shuhong Gao avait prouvé la borne $\mathsf{B}(P, S) \leqslant 2\,d^3/|S|$ [35], qui a été ensuite améliorée en $\mathsf{B}(P, S) \leqslant d\,(d^2 - 1)/|S|$ par Chèze [20, Chapitre 1]. Les résultats de cette section sont dérivés de et complètent ceux de l'article [86].

6.4 Représentation creuse

Lorsque les polynômes à manipuler n'ont que peu de termes non nuls dans leur représentation dense il est souvent préférable d'utiliser une autre représentation. Une *représentation creuse* des polynômes de $\mathbb{K}[x_1, ..., x_n]$ consiste à stocker ceux-ci sous la forme d'une liste de termes non nuls. Chaque terme est la donnée d'un coefficient et d'un monôme de la forme $x_1^{e_1} \cdots x_n^{e_n}$. Dans la suite nous identifierons librement un monôme $x_1^{e_1} \cdots x_n^{e_n}$ au vecteur de ses exposants $(e_1, ..., e_n) \in \mathbb{N}^n$.

Définition 6.10. *Le support d'un polynôme $F \in \mathbb{K}[x_1, ..., x_n]$, noté* $\mathrm{supp}(F)$*, est l'ensemble des exposants de ses monômes non nuls.*

Si $F = \sum_{e \in \mathbb{N}^n} F_e\, x_1^{e_1} \cdots x_n^{e_n}$, alors $\mathrm{supp}(F) = \{e \in \mathbb{N}^n \mid F_e \neq 0\}$.

Définition 6.11. *La* somme de Minkowski *de deux sous-ensembles Q et R de $\mathbb{R}^n$, notée $Q + R$, est définie par $Q + R := \{e + f \mid (e, f) \in Q \times R\}$.*

Définition 6.12. *Le* polytope de Newton *de $F \in \mathbb{K}[x_1, ..., x_n]$, noté $N(f)$, est l'enveloppe convexe dans $\mathbb{R}^n$ de* $\mathrm{supp}(F)$. L'enveloppe convexe entière *de F est l'ensemble des points de $\mathbb{Z}^n$ contenus dans $N(F)$.*

Théorème 6.13. (Ostrowski) *Si F se factorise en $G\,H$ alors $N(F) = N(G) + N(H)$.*

Démonstration. Nous renvoyons le lecteur à la preuve originale [110] ou sa traduction [111]. □

L'analyse du support d'un polynôme donne donc des informations sur les supports possibles de ses facteurs. Un cas particulier important concerne les polynômes dont le support ne peut justement pas se décomposer sous une forme $N(F) = Q + R$, où Q et R sont des polytope à sommets dans $\mathbb{Z}^n$ contenant au moins deux points. On dit alors que $N(F)$ est *indécomposable*. Le théorème précédent assure que si F n'est multiple d'aucune variable et si $N(F)$ est indécomposable, alors F est irréductible. Notons que cette conclusion ne dépend pas du corps des coefficients de F. Par conséquent, F est irréductible dans $\bar{\mathbb{K}}[x_1, ..., x_n]$. Pour une utilisation plus avancée de ces techniques, nous renvoyons le lecteur à [34, 37].

6.5 Support dense dans le polytope de Newton

Dans le pire des cas, la taille des facteurs irréductibles est exponentielle dans la taille du polynôme en représentation creuse à factoriser.

Exemple 6.14. Si p est premier alors $F := x^p - 1 \in \mathbb{Q}[x]$ est composé de seulement deux monômes mais ses facteurs irréductibles sont $x - 1$ et $(x^p - 1)/x - 1$. Ce dernier est composé de p monômes.

Définition 6.15. *La taille convexe de $F \in \mathbb{K}[x_1, \ldots, x_n]$ est le nombre de points de l'enveloppe convexe entière de $N(F)$.*

Plusieurs algorithmes de factorisation ont un coût qui dépend de la taille convexe du polynôme à factoriser dans le cas à deux variables. Nous allons brièvement présenter l'un d'entre eux, qui permet de se ramener directement à la situation étudiée dans le chapitre précédent en terme des degrés partiels.

Définition 6.16. *Le groupe affine de $\mathbb{Z}^2$, noté $\mathrm{Aff}(\mathbb{Z}^2)$, est l'ensemble des applications*

$$U : (i, j) \mapsto \begin{pmatrix} \alpha & \beta \\ \alpha' & \beta' \end{pmatrix} \begin{pmatrix} i \\ j \end{pmatrix} + \begin{pmatrix} \gamma \\ \gamma' \end{pmatrix}, \tag{6.2}$$

avec α, β, γ, α', β', et γ' dans $\mathbb{Z}$, tels que $\alpha\,\beta' - \alpha'\,\beta = \pm 1$.

Définition 6.17. *Un sous-ensemble fini S de $\mathbb{Z}^2$ est dit normalisé s'il appartient à $\mathbb{N}^2$ et s'il contient au moins un point dans $\{0\} \times \mathbb{N}$ et au moins un point dans $\mathbb{N} \times \{0\}$.*

Théorème 6.18. *[11, Theorem 1.2] Si S est un sous-ensemble fini normalisé de $\mathbb{Z}^2$ de cardinal σ, de taille convexe π, et inclus dans $[0, d_x] \times [0, d_y]$, alors nous pouvons calculer une application $U \in \mathrm{Aff}(\mathbb{Z}^2)$ comme dans (6.2), ainsi que $U(S)$, tels que $U(S)$ est normalisé et contenu dans $[0, d'_x] \times [0, d'_y]$ avec $(d'_x + 1)\,(d'_y + 1) \leqslant 9\,\pi$, en utilisant $O(\sigma \log^2((d_x + 1)\,(d_y + 1)))$ opérations binaires.*

L'utilisation de ce théorème pour la factorisation des polynômes repose sur le lemme suivant :

Lemme 6.19. *Si $F \in \mathbb{K}[x, y]$ n'est divisible ni par x ni par y, et si $U \in \mathrm{Aff}(\mathbb{Z}^2)$, alors le polynôme*

$$U(f) := \sum_{(e_x, e_y) \in \mathrm{supp}(F)} F_{(e_x, e_y)}\, x^{\alpha e_x + \beta e_y + \gamma}\, y^{\alpha' e_x + \beta' e_y + \gamma'}$$

est irréductible dans $\mathbb{K}[x, y, x^{-1}, y^{-1}]$ si, est seulement si, F est irréductible.

Afin de calculer la factorisation irréductible de F, nous pouvons par conséquent calculer une application U *via* le théorème précédent. Ensuite nous calculons la décomposition irréductible de $U(F)$ et appliquons U^{-1} sur chacun des facteurs. De cette façon le coût total dépend de la taille convexe de F au lieu de sa taille dense $(d_x + 1)\,(d_y + 1)$.

Au lieu d'utiliser le théorème précédent il est possible d'étendre plusieurs des techniques du chapitre 5 pour prendre en compte non seulement le polytope de Newton de F mais aussi sa taille creuse. Nous renvoyons le lecteur vers les articles [2, 3, 38, 40, 63, 147, 148, 153, 154].

6.6 Polynômes lacunaires et représentation fonctionnelle

Le calcul de facteurs de « petits degrés » fixés de polynômes à coefficients dans $\mathbb{Q}$ peut se faire en temps polynomial [72, 91] dans la taille binaire creuse (c'est à dire le total des tailles des coefficients et des exposants). En revanche, lorsque les coefficients sont dans un corps fini, le problème devient nettement plus difficile [78]. Par exemple, le résultat suivant est dû à Kaltofen et Koiran :

Théorème 6.20. *[71, Corollary 1] Supposons que nous disposions d'un algorithme de type Monte Carlo pour tester l'irréductibilité des polynômes de $\mathbb{F}_{2^m}[x, y]$ en temps polynomial en la taille binaire creuse, pour m au voisinage de l'infini. Alors il serait possible de factoriser les entiers en temps polynomial avec un algorithme de type Las Vegas.*

Pour d'avantage de résultats sur ce sujet nous renvoyons le lecteur vers [4, 72, 92, 119].

Une *représentation fonctionnelle* de polynômes consiste en une structure de donnée permettant d'évaluer un polynôme en un point. La taille d'une telle structure représente le nombre d'opérations arithmétiques à effectuer pour évaluer le polynôme. Les principales structures utilisées sont des graphes acycliques orientés ou bien des programmes sans branchement où chaque instruction représente une opération arithmétique élémentaire.

Un polynôme de petite taille creuse peut être évalué rapidement, ce qui justifie l'intérêt de cette représentation. Un autre intérêt concerne le calcul symbolique où l'utilisateur a souvent des expressions algébriques naturellement données par une structure en évaluation. Nous n'entrerons pas d'avantage dans les détails et renvoyons le lecteur vers les articles suivants traitant de ce problème : [42, 64, 66, 74, 75].

Références

1 J. Abbott, V. Shoup et P. Zimmermann : Factorization in $\mathbb{Z}[x]$: the searching phase. *In ISSAC '00: Proceedings of the 2000 international symposium on Symbolic and algebraic computation*, pages 1–7, New York, NY, USA, 2000. ACM Press.

2 F. K. Abu Salem : An efficient sparse adaptation of the polytope method over $\mathbb{F}_p$ and a record-high binary bivariate factorisation. *J. Symbolic Comput.*, 43(5):311–341, 2008.

3 F. K. Abu Salem, Shuhong Gao et A. G. B. Lauder : Factoring polynomials via polytopes. *In ISSAC '04: Proceedings of the 2004 International Symposium on Symbolic and Algebraic Computation*, pages 4–11, New York, 2004. ACM Press.

4 M. Avendaño, T. Krick et M. Sombra : Factoring bivariate sparse (lacunary) polynomials. *J. Complexity*, 23(2):193–216, 2007.

5 C. Bajaj, J. Canny, T. Garrity et J. Warren : Factoring rational polynomials over the complex numbers. *SIAM J. Comput.*, 22(2):318–331, 1993.

6 K. Belabas, M. van Hoeij, M., J. Klüners et A. Steel : Factoring polynomials over global fields. *J. Théor. Nombres Bordeaux*, 21(1):15–39, 2009.

7 E. R. Berlekamp : Factoring polynomials over finite fields. *Bell System Tech. J.*, 46:1853–1859, 1967.

8 E. R. Berlekamp : Factoring polynomials over large finite fields. *Math. Comp.*, 24:713–735, 1970.

9 L. Bernardin et M. B. Monagan : Efficient multivariate factorization over finite fields. *In Applied algebra, algebraic algorithms and error-correcting codes (Toulouse, 1997)*, volume 1255 de *Lecture Notes in Comput. Sci.*, pages 15–28. Springer-Verlag, 1997.

10 J. Berthomieu, J. van der Hoeven et G. Lecerf : Relaxed algorithms for p-adic numbers. *Journal de Théorie des Nombres de Bordeaux*, 23(3):541–577, 2011.

11 J. Berthomieu et G. Lecerf : Reduction of bivariate polynomials from convex-dense to dense, with application to factorizations. *Math. Comp.*, 81(279):1799–1821, 2012.

12 D. A. Bini et G. Fiorentino : Design, analysis, and implementation of a multiprecision polynomial rootfinder. *Numer. Algorithms*, 23(2-3):127–173, 2000.

13 P. B. Borwein : *Computational Excursions in Analysis and Number Theory*. CMS Books in Mathematics. Springer-Verlag, 2002.

14 A. Bostan : *Algorithmes rapides pour les polynômes, séries formelles et matrices*, volume 1 de *Les cours du CIRM*, pages 75–262. cedram.org, 2010. Disponible depuis http://dx.doi.org/10.5802/ccirm.9.

15 A. Bostan, G. Lecerf, B. Salvy, É. Schost et B. Wiebelt : Complexity issues in bivariate polynomial factorization. *In ISSAC '04: Proceedings of the 2004 international symposium on Symbolic and algebraic computation*, pages 42–49. ACM Press, 2004.

16 P. Bürgisser, M. Clausen et M. A. Shokrollahi : *Algebraic complexity theory*. Springer-Verlag, 1997.

17 D. G. Cantor et H. Zassenhaus : A new algorithm for factoring polynomials over finite fields. *Math. Comp.*, 36(154):587–592, 1981.

18 Jingwei Chen, D. Stehlé et G. Villard : A new view on HJLS and PSLQ: sums and projections of lattices. À paraître dans les actes de la conférence ISSAC'13, 2013.

19 G. Chèze : Absolute polynomial factorization in two variables and the knapsack problem. *In ISSAC '04: Proceedings of the 2004 international symposium on Symbolic and algebraic computation*, pages 87–94. ACM Press, 2004.

20 G. Chèze : *Des méthodes symboliques-numériques et exactes pour la factorisation absolue des polynômes en deux variables*. Thèse de doctorat, Université de Nice-Sophia Antipolis (France), 2004.

21 G. Chèze et A. Galligo : Four lectures on polynomial absolute factorization. *In* A. Dickenstein et I. Z. Emiris, éditeurs : *Solving polynomial equations: foundations, algorithms, and applications*, volume 14 de *Algorithms Comput. Math.*, pages 339–392. Springer-Verlag, 2005.

22 G. Chèze et A. Galligo : From an approximate to an exact absolute polynomial factorization. *J. Symbolic Comput.*, 41(6):682–696, 2006.

23 G. Chèze et G. Lecerf : Lifting and recombination techniques for absolute factorization. *J. Complexity*, 23(3):380–420, 2007.

24 D. Coppersmith et C. A. Neff : Roots of a polynomial and its derivatives. *In Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms (Arlington, VA, 1994)*, pages 271–279. ACM Press, 1994.

25 J. H. Davenport et B. M. Trager : Factorization over finitely generated fields. *In SYMSAC'81: Proceedings of the fourth ACM symposium on Symbolic and algebraic computation*, pages 200–205. ACM Press, 1981.

26 W. Decker, G.-M. Greuel, G. Pfister et H. Schönemann : Singular 3-1-6 — A computer algebra system for polynomial computations, 2012. http://www.singular.uni-kl.de.

27 I. Z. Emiris, B. Mourrain et E. P. Tsigaridas : Real algebraic numbers: Complexity analysis and experimentation. *In* P. Hertling, C. M. Hoffmann, W. Luther et N. Revol, éditeurs : *Reliable Implementation of Real Number Algorithms: Theory and Practice*, volume 5045 de *Lecture Notes in Computer Science*, pages 57–82. Springer Berlin Heidelberg, 2008.

28 H. R. P. Ferguson, D. H. Bailey et S. Arno : Analysis of PSLQ, an integer relation finding algorithm. *Math. Comp.*, 68:351–369, 1999.

29 Ph. Flajolet et J.-M. Steyaert : A branching process arising in dynamic hashing, trie searching and polynomial factorization. *In* M. Nielsen et E. Schmidt, éditeurs : *Automata, Languages and Programming*, volume 140 de *Lecture Notes in Computer Science*, pages 239–251. Springer Berlin Heidelberg, 1982.

30 A. Fröhlich et J. C. Shepherdson : On the factorisation of polynomials in a finite number of steps. *Math. Z.*, 62:331–334, 1955.

31 A. Fröhlich et J. C. Shepherdson : Effective procedures in field theory. *Philos. Trans. Roy. Soc. London. Ser. A.*, 248:407–432, 1956.

32 M. Fürer : Faster integer multiplication. *In Proceedings of the Thirty-Ninth ACM Symposium on Theory of Computing (STOC 2007)*, pages 57–66. ACM Press, 2007.

33 M. Fürer : Faster integer multiplication. *SIAM J. Comput.*, 39(3):979–1005, 2009.

34 Shuhong Gao : Absolute irreducibility of polynomials via Newton polytopes. *J. Algebra*, 237(2):501–520, 2001.

35 Shuhong Gao : Factoring multivariate polynomials via partial differential equations. *Math. Comp.*, 72(242):801–822, 2003.

36 Shuhong Gao et A. G. B. Lauder : Hensel lifting and bivariate polynomial factorisation over finite fields. *Math. Comp.*, 71(240):1663–1676, 2002.

37 Shuhong Gao et V. M. Rodrigues : Irreducibility of polynomials modulo p via Newton polytopes. *J. Number Theory*, 101(1):32–47, 2003.

38 J. von zur Gathen et E. Kaltofen : Factoring sparse multivariate polynomials. *J. Comput. System Sci.*, 31(2):265–287, 1985. Special issue: Twenty-fourth annual symposium on the foundations of computer science (Tucson, Ariz., 1983).

39 J. von zur Gathen et E. Kaltofen : Factorization of multivariate polynomials over finite fields. *Math. Comp.*, 45(171):251–261, 1985.

40 J. von zur Gathen : Factoring sparse multivariate polynomials. *In 24th Annual IEEE Symposium on Foundations of Computer Science*, pages 172–179, Los Alamitos, CA, USA, 1983. IEEE Computer Society.

41 J. von zur Gathen : Hensel and Newton methods in valuation rings. *Math. Comp.*, 42(166):637–661, 1984.

42 J. von zur Gathen : Irreducibility of multivariate polynomials. *J. Comput. System Sci.*, 31(2):225–264, 1985. Special issue: Twenty-fourth annual symposium on the foundations of computer science (Tucson, Ariz., 1983).

43 J. von zur Gathen : Who was who in polynomial factorization. *In Proceedings of the 2006 international symposium on Symbolic and algebraic computation*, pages 1–2, New York, NY, USA, 2006. ACM Press.

44 J. von zur Gathen et J. Gerhard : *Modern computer algebra*. Cambridge University Press, New York, 2^e édition, 2003.

45 J. Gerhard : Fast modular algorithms for squarefree factorization and Hermite integration. *Appl. Algebra Engrg. Comm. Comput.*, 11(3):203–226, 2001.

46 P. Gianni et B. Trager : Square-free algorithms in positive characteristic. *Appl. Algebra Engrg. Comm. Comput.*, 7(1):1–14, 1996.

47 W. Hart, M. van Hoeij et A. Novocin : Practical polynomial factoring in polynomial time. *In ISSAC '11: Proceedings of the 36th international symposium on Symbolic and algebraic computation*, pages 163–170, New York, NY, USA, 2011. ACM Press.

48 J. Heintz et M. Sieveking : Absolute primality of polynomials is decidable in random polynomial time in the number of variables. *In Automata, languages and programming (Akko, 1981)*, volume 115 de *Lecture Notes in Comput. Sci.*, pages 16–28. Springer-Verlag, 1981.

49 P. Henrici : *Applied and Computational Complex Analysis, Volume 1, Power Series Integration Conformal Mapping Location of Zero*. Wiley Classics Library. Wiley, 1988.

50 G. Hermann : Die Frage der endlich vielen Schritte in der Theorie der Polynomideale. *Math. Ann.*, 95(1):736–788, 1926.

51 D. Hilbert : Ueber die Irreducibilität ganzer rationaler Functionen mit ganzzahligen Coefficienten. *J. Reine Angew. Math.*, 110, 1892.

52 M. van Hoeij : Factoring polynomials and the knapsack problem. *J. Number Theory*, 95(2):167–189, 2002.

53 J. van der Hoeven : New algorithms for relaxed multiplication. *J. Symbolic Comput.*, 42(8):792–802, 2007.

54 J. van der Hoeven : *Calcul analytique*, volume 2 de *Les cours du CIRM*, pages 1–85. cedram.org, 2011. Cours no. IV, disponible depuis http://ccirm.cedram.org/item?id=CCIRM_2011__2_1_A4_0.

55 J. van der Hoeven et G. Lecerf : On the bit-complexity of sparse polynomial and series multiplication. *J. Symbolic Comput.*, 50:227–254, 2013.

56 J. van der Hoeven, G. Lecerf, B. Mourrain *et al.* : Mathemagix, 2002. http://www.mathemagix.org.

57 J.-P. Jouanolou : *Théorèmes de Bertini et applications*, volume 42 de *Progress in Mathematics*. Birkhäuser Boston, 1983.

58 E. Kaltofen : Polynomial factorization. *In* B. Buchberger, G. Collins et R. Loos, éditeurs : *Computer algebra*, pages 95–113. Springer-Verlag, 1982.

59 E. Kaltofen : A polynomial reduction from multivariate to bivariate integral polynomial factorization. *In Proceedings of the 14th Symposium on Theory of Computing*, pages 261–266. ACM Press, 1982.

60 E. Kaltofen : Effective Hilbert irreducibility. *Inform. and Control*, 66(3):123–137, 1985.

61 E. Kaltofen : Fast parallel absolute irreducibility testing. *J. Symbolic Comput.*, 1(1):57–67, 1985.

62 E. Kaltofen : Polynomial-time reductions from multivariate to bi- and univariate integral polynomial factorization. *SIAM J. Comput.*, 14(2):469–489, 1985.

63 E. Kaltofen : Sparse Hensel lifting. *In Proceedings of EUROCAL '85, Vol. 2 (Linz, 1985)*, volume 204 de *Lecture Notes in Comput. Sci.*, pages 4–17. Springer-Verlag, 1985.
64 E. Kaltofen : Uniform closure properties of p-computable functions. *In Proc. 18th Annual ACM Symp. Theory Comput.*, pages 330–337. ACM Press, 1986. Also published as part of [65] and [66].
65 E. Kaltofen : Greatest common divisors of polynomials given by straight-line programs. *J. ACM*, 35(1):231–264, 1988.
66 E. Kaltofen : Factorization of polynomials given by straight-line programs. *In* S. Micali, éditeur : *Randomness and Computation*, volume 5 de *Advances in Computing Research*, pages 375–412. JAI Press Inc., Greenwhich, Connecticut, 1989.
67 E. Kaltofen : Polynomial factorization 1982–1986. *In Computers in mathematics (Stanford, CA, 1986)*, volume 125 de *Lecture Notes in Pure and Appl. Math.*, pages 285–309. Dekker, 1990.
68 E. Kaltofen : Polynomial factorization 1987–1991. *In LATIN '92 (São Paulo, 1992)*, volume 583 de *Lecture Notes in Comput. Sci.*, pages 294–313. Springer-Verlag, 1992.
69 E. Kaltofen : Effective Noether irreducibility forms and applications. *J. Comput. System Sci.*, 50(2):274–295, 1995.
70 E. Kaltofen : Polynomial factorization: a success story. *In ISSAC '03: Proceedings of the 2003 international symposium on Symbolic and algebraic computation*, pages 3–4. ACM Press, 2003.
71 E. Kaltofen et P. Koiran : On the complexity of factoring bivariate supersparse (lacunary) polynomials. *In ISSAC '05: Proceedings of the 2005 International Symposium on Symbolic and Algebraic Computation*, pages 208–215, 2005.
72 E. Kaltofen et P. Koiran : Finding small degree factors of multivariate supersparse (lacunary) polynomials over algebraic number fields. *In ISSAC '06: Proceedings of the 2006 International Symposium on Symbolic and Algebraic Computation*, pages 162–168, 2006.
73 E. Kaltofen et V. Shoup : Subquadratic-time factoring of polynomials over finite fields. *Math. Comp.*, 67(223):1179–1197, 1998.
74 E. Kaltofen et B. Trager : Computing with polynomials given by black boxes for their evaluations: Greatest common divisors, factorization, separation of numerators and denominators. *In Proc. 29th Annual Symp. Foundations of Comp. Sci.*, pages 296–305. IEEE, 1988.
75 E. Kaltofen et B. Trager : Computing with polynomials given by black boxes for their evaluations: Greatest common divisors, factorization, separation of numerators and denominators. *J. Symbolic Comput.*, 9(3):301–320, 1990.
76 R. Kannan, A. K. Lenstra et L. Lovász : Polynomial factorization and nonrandomness of bits of algebraic and some transcendental numbers. *In Proceedings of the sixteenth annual ACM symposium on Theory of computing*, STOC '84, pages 191–200, New York, NY, USA, 1984. ACM Press.
77 K. S. Kedlaya et C. Umans : Fast modular composition in any characteristic. *In 49th Annual IEEE Symposium on Foundations of Computer Science 2008 (FOCS '08)*, pages 146–155, 2008.
78 A. Kipnis et A. Shamir : Cryptanalysis of the HFE public key cryptosystem by relinearization. *In* M. J. Wiener, éditeur : *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, volume 1666 de *Lecture Notes in Computer Science*, pages 19–30. Springer-Verlag, 1999.
79 P. Kirrinnis : Partial fraction decomposition in $\mathbb{C}(z)$ and simultaneous Newton iteration for factorization in $\mathbb{C}[z]$. *J. Complexity*, 14(3):378–444, 1998.
80 S. C. Kleene : General recursive functions of natural numbers. *Mathematische Annalen*, 112:727–742, 1936.
81 S. L. Kleiman : Bertini and his two fundamental theorems. *Rend. Circ. Mat. Palermo (2) Suppl.*, 55:9–37, 1998. Studies in the history of modern mathematics, III.
82 D. E. Knuth : *The Art of Computer Programming, Volume II: Seminumerical Algorithms*. Addison Wesley Longman, 3^e édition, 1998.
83 L. Kronecker : Grundzüge einer arithmetischen Theorie der algebraischen Grössen. *J. reine angew. Math.*, 92:1–122, 1882.

84 S. Lang : *Algebra*, volume 211 de *Graduate Texts in Mathematics*. Springer-Verlag, 3^e édition, 2002.

85 G. Lecerf : Sharp precision in Hensel lifting for bivariate polynomial factorization. *Math. Comp.*, 75:921–933, 2006.

86 G. Lecerf : Improved dense multivariate polynomial factorization algorithms. *J. Symbolic Comput.*, 42(4):477–494, 2007.

87 G. Lecerf : Fast separable factorization and applications. *Appl. Alg. Eng. Comm. Comp.*, 19(2), 2008.

88 G. Lecerf : New recombination algorithms for bivariate polynomial factorization based on Hensel lifting. *Appl. Alg. Eng. Comm. Comp.*, 21(2):151–176, 2010.

89 G. Lecerf et É. Schost : Fast multivariate power series multiplication in characteristic zero. *SADIO Electronic Journal on Informatics and Operations Research*, 5(1):1–10, 2003.

90 A. K. Lenstra, H. W. Lenstra, Jr. et L. Lovász : Factoring polynomials with rational coefficients. *Math. Ann.*, 261(4):515–534, 1982.

91 H. W. Lenstra, Jr. : Finding small degree factors of lacunary polynomials. *In* Kálmán G., Henryk I. et Jerzy U., éditeurs : *Number Theory in Progress*, volume 1 Diophantine Problems and Polynomials, pages 267–276. Stefan Banach Internat. Center, Walter de Gruyter Berlin/New York, 1999. Proc. Internat. Conf. Number Theory in Honor of the 60th Birthday of Andrzej Schinzel, Zakopane, Poland June 30–July 9, 1997.

92 L. Leroux : *Algorithmes pour les polynômes lacunaires*. Thèse de doctorat, Université de Caen, 2011. http://tel.archives-ouvertes.fr/docs/00/58/06/56/PDF/these.pdf.

93 M. Marden : *Geometry of polynomials*. Second edition. Mathematical Surveys, No. 3. American Mathematical Society, 1966.

94 M. Mignotte : An inequality about factors of polynomials. *Math. Comp.*, 28:1153–1157, 1974.

95 R. Mines et F. Richman : Separability and factoring polynomials. *Rocky Mountain J. Math.*, 12(1):43–54, 1982.

96 R. Mines, F. Richman et W. Ruitenburg : *A course in constructive algebra*. Universitext. Springer-Verlag, 1988.

97 G. L. Mullen et D. Panario : *Handbook of Finite Fields*. Discrete Mathematics and Its Applications. Chapman and Hall/CRC, 2013.

98 D. Mumford : *Algebraic geometry. I Complex projective varieties*. Classics in Mathematics. Springer-Verlag, 1995. Reprint of the 1976 edition.

99 D. R. Musser : *Algorithms for Polynomial Factorization*. Thèse de doctorat, C.S. Department, Univ. of Wisconsin, 1971.

100 D. R. Musser : Multivariate polynomial factorization. *J. Assoc. Comput. Mach.*, 22:291–308, 1975.

101 A. Neff et J. H. Reif : An $O\left(n^{1+\epsilon}\log b\right)$ algorithm for the complex roots problem. *In Proceedings of the 35th Annual IEEE Conference on Foundations of Computer Science (FOCS '94)*, pages 540–547. IEEE Computer Society Press, 1994.

102 C. A. Neff et J. H. Reif : An efficient algorithm for the complex roots problem. *J. Complexity*, 12(2):81–115, 1996.

103 Phong Q. Nguyen et B. Vallée, éditeurs. *The LLL Algorithm. Survey and Applications*. Information Security and Cryptography. Springer, 2010.

104 H. Niederreiter : A new efficient factorization algorithm for polynomials over small finite fields. *Appl. Algebra Engrg. Comm. Comput.*, 4(2):81–87, 1993.

105 H. Niederreiter : Factoring polynomials over finite fields using differential equations and normal bases. *Math. Comp.*, 62(206):819–830, 1994.

106 A. Nijenhuis et H. Wilf : *Combinatorial Algorithms for Computers and Calculators*. Academic Press, 1978.

107 A. Novocin : *Factoring Univariate Polynomials over the Rationals*. Thèse de doctorat, Department of Mathematics Florida State University Tallahassee, 2008.

108 A. Novocin et M. van Hoeij : Factoring univariate polynomials over the rationals. *ACM Commun. Comput. Algebra*, 42(3):157–157, 2009.

109 A. Novocin, D. Stehlé et G. Villard : An LLL-reduction algorithm with quasi-linear time complexity: extended abstract. *In Proceedings of the 43rd annual ACM symposium on Theory of computing*, STOC '11, pages 403–412, New York, NY, USA, 2011. ACM Press.
110 A. M. Ostrowski : Über die Bedeutung der Theorie der konvexen Polyeder für die formale Algebra. *Jahresber. Deutsch. Math.-Verein.*, 30(2):98–99, 1921. Talk given at *Der Deutsche Mathematikertag vom 18–24 September 1921 in Jena*.
111 A. M. Ostrowski : On the significance of the theory of convex polyhedra for formal algebra. *ACM SIGSAM Bull.*, 33(1):5, 1999. Translated from [110].
112 V. Y. Pan : Optimal (up to polylog factors) sequential and parallel algorithms for approximating complex polynomial zeros. *In STOC '95: Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*, pages 741–750. ACM Press, 1995.
113 V. Y. Pan : Optimal and nearly optimal algorithms for approximating polynomial zeros. *Comput. Math. Appl.*, 31(12):97–138, 1996.
114 V. Y. Pan : Solving a polynomial equation: some history and recent progress. *SIAM Rev.*, 39(2):187–220, 1997.
115 V. Y. Pan : Approximating complex polynomial zeros: modified Weyl's quadtree construction and improved Newton's iteration. *J. Complexity*, 16(1):213–264, 2000. Real computation and complexity (Schloss Dagstuhl, 1998).
116 V. Y. Pan : Univariate polynomials: nearly optimal algorithms for numerical factorization and root-finding. *J. Symbolic Comput.*, 33(5):701–733, 2002.
117 The PARI Group, Bordeaux. *PARI/GP*, 2012. Software available from http://pari.math.u-bordeaux.fr.
118 Ph. Saux Picart : The schur–cohn algorithm revisited. *J. Symbolic Comput.*, 26(4):387–408, 1998.
119 D. A. Plaisted : New NP-hard and NP-complete polynomial and integer divisibility problems. *Theoret. Comput. Sci.*, 13:125–138, 1984.
120 J. Renegar : On the worst-case arithmetic complexity of approximating zeros of polynomials. *J. Complexity*, 3(2):90–113, 1987.
121 F. Richman : Seidenberg's condition *P*. *In Constructive mathematics (Las Cruces, N.M., 1980)*, volume 873 de *Lecture Notes in Math.*, pages 1–11. Springer-Verlag, 1981.
122 W. M. Ruppert : Reduzibilität ebener Kurven. *J. Reine Angew. Math.*, 369:167–191, 1986.
123 W. M. Ruppert : Reducibility of polynomials $f(x, y)$ modulo p. *J. Number Theory*, 77(1):62–70, 1999.
124 T. Sasaki, T. Saito et T. Hilano : Analysis of approximate factorization algorithm. I. *Japan J. Indust. Appl. Math.*, 9(3):351–368, 1992.
125 T. Sasaki et M. Sasaki : A unified method for multivariate polynomial factorizations. *Japan J. Indust. Appl. Math.*, 10(1):21–39, 1993.
126 T. Sasaki, M. Suzuki, M. Kolář et M. Sasaki : Approximate factorization of multivariate polynomials and absolute irreducibility testing. *Japan J. Indust. Appl. Math.*, 8(3):357–375, 1991.
127 A. Schinzel : *Polynomials with special regard to reducibility*, volume 77 de *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, 2000.
128 A. Schönhage : Schnelle Berechnung von Kettenbruchentwicklugen. *Acta Inform.*, 1:139–144, 1971.
129 A. Schönhage : The fundamental theorem of algebra in terms of computational complexity. Rapport technique, Preliminary Report of Mathematisches Institut der Universität Tübingen, Germany, 1982.
130 J. T. Schwartz : Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM*, 27(4):701–717, 1980.
131 A. Seidenberg : Construction of the integral closure of a finite integral domain. *Rend. Sem. Mat. Fis. Milano*, 40:100–120, 1970.
132 A. Seidenberg : Constructions in algebra. *Trans. Amer. Math. Soc.*, 197:273–313, 1974.

133 A. Seidenberg : Constructions in a polynomial ring over the ring of integers. *Amer. J. Math.*, 100(4):685–703, 1978.

134 I. R. Shafarevich : *Basic algebraic geometry. 1 Varieties in projective space*. Springer-Verlag, 2^e édition, 1994.

135 V. Shoup : NTL: A library for doing number theory. http://www.shoup.net.

136 A. Steel : Conquering inseparability: primary decomposition and multivariate factorization over algebraic function fields of positive characteristic. *J. Symbolic Comput.*, 40(3):1053–1075, 2005.

137 W. A. Stein *et al.* : *Sage Mathematics Software*. The Sage Development Team, 2004. http://www.sagemath.org.

138 J. Stern : *Fondements mathématiques de l'informatique*. Ediscience international, 1994.

139 A. Storjohann : *Algorithms for matrix canonical forms*. Thèse de doctorat, ETH, Zürich, Switzerland, 2000.

140 B. M. Trager : Algebraic factoring and rational function integration. *In Proceedings of the third ACM symposium on symbolic and algebraic computation*, pages 219–226. ACM Press, 1976.

141 B. M. Trager : *Integration of algebraic functions*. Thèse de doctorat, M.I.T. (USA), 1984.

142 E. P. Tsigaridas et I. Z. Emiris : On the complexity of real root isolation using continued fractions. *Theoretical Computer Science*, 392(1–3):158–173, 2008.

143 B. L. van der Waerden : Eine Bemerkung über die Unzerlegbarkeit von Polynomen. *Math. Ann.*, 102(1):738–739, 1930.

144 B. L. van der Waerden : *Modern Algebra. Vol. I*. Frederick Ungar Publishing Co., New York, N. Y., 1949.

145 P. S. Wang : An improved multivariate polynomial factoring algorithm. *Math. Comp.*, 32(144):1215–1231, 1978.

146 P. S. Wang et L. P. Rothschild : Factoring multivariate polynomials over the integers. *Math. Comp.*, 29:935–950, 1975.

147 M. Weimann : A lifting and recombination algorithm for rational factorization of sparse polynomials. *J. Complexity*, 26(6):608–628, 2010.

148 M. Weimann : Algebraic osculation and application to factorization of sparse polynomials. *Found. Comput. Math.*, 12(2):173–201, 2012.

149 J.-C. Yakoubsohn : Finding zeros of analytic functions: α theory for secant type methods. *J. Complexity*, 15(2):239–281, 1999.

150 J.-C. Yakoubsohn : Numerical analysis of a bisection-exclusion method to find zeros of univariate analytic functions. *J. Complexity*, 21(5):652–690, 2005.

151 H. Zassenhaus : On Hensel factorization I. *J. Number Theory*, 1(1):291–311, 1969.

152 R. Zippel : Probabilistic algorithms for sparse polynomials. *In Proceedings EUROSAM' 79*, numéro 72 de Lecture Notes in Computer Science, pages 216–226. Springer-Verlag, 1979.

153 R. Zippel : Probabilistic algorithms for sparse polynomials. *In EUROSAM '79: Proceedings of the International Symposium on Symbolic and Algebraic Computation*, numéro 72 de Lecture Notes in Comput. Sci., pages 216–226. Springer-Verlag, 1979.

154 R. Zippel : Newton's iteration and the sparse Hensel algorithm (Extended Abstract). *In SYMSAC '81: Proceedings of the fourth ACM Symposium on Symbolic and Algebraic Computation*, pages 68–72, New York, 1981. ACM Press.

155 R. Zippel : *Effective Polynomial Computation*. Kluwer Academic Publishers, 1993.