



HAL
open science

On the p -adic stability of the FGLM algorithm

Guénaél Renault, Tristan Vaccon

► **To cite this version:**

Guénaél Renault, Tristan Vaccon. On the p -adic stability of the FGLM algorithm. 2016. hal-01266071

HAL Id: hal-01266071

<https://hal.science/hal-01266071v1>

Preprint submitted on 2 Feb 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

On the p -adic stability of the FGLM algorithm

Guénaël Renault
PoISys project INRIA Paris-Rocquencourt,
UPMC Univ. Paris 06, CNRS, UMR 7606, LIP6,
Paris, France
guenael.renault@lip6.fr

Tristan Vaccon
JSPS–Rikkyo University
Tokyo, Japan
vaccon@rikkyo.ac.jp

ABSTRACT

Nowadays, many strategies to solve polynomial systems use the computation of a Gröbner basis for the graded reverse lexicographical ordering, followed by a change of ordering algorithm to obtain a Gröbner basis for the lexicographical ordering. The change of ordering algorithm is crucial for these strategies. We study the p -adic stability of the main change of ordering algorithm, FGLM.

We show that FGLM is stable and give explicit upper bound on the loss of precision occurring in its execution. The variant of FGLM designed to pass from the grevlex ordering to a Gröbner basis in shape position is also stable.

Our study relies on the application of Smith Normal Form computations for linear algebra.

Categories and Subject Descriptors

I.1.2 [Computing Methodologies]: Symbolic and Algebraic Manipulations—*Algebraic Algorithms*

General Terms

Algorithms, Theory

Keywords

Gröbner bases, FGLM algorithm, p -adic precision, p -adic algorithm, Smith Normal Form

1. INTRODUCTION

The advent of arithmetic geometry has seen the emergence of questions that are purely local (*i.e.* where a prime p is fixed at the very beginning and one can not vary it). As an example, one can cite the work of Caruso and Lubicz [CL14] who gave an algorithm to compute lattices in some p -adic Galois representations. A related question is the study of p -adic deformation spaces of Galois representations. Since the work of Taylor and Wiles [TW95], one knows that these spaces play a crucial role in many questions in number theory. Being able to compute such spaces appears then as

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or to publish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ISSAC'16, July, 2016, London, Canada.

Copyright is held by the owner/author(s). Publication rights licensed to ACM.

ACM 9 ...\$15.00.

DOI: <http://dx.doi.org/>.

an interesting question of experimental mathematics and require the use of purely p -adic Gröbner bases and more generally p -adic polynomial system solving.

Since [Vac14], it is possible to compute a Gröbner basis, under some genericness assumptions, for a monomial ordering ω of an ideal generated by a polynomial sequence $F = (f_1, \dots, f_s) \subset \mathbb{Q}_p[X_1, \dots, X_n]$ if the coefficients of the f_i 's are given with enough initial precision. Unfortunately, one of the genericness assumptions (namely, the sequence (f_1, \dots, f_i) has to be weakly- ω) is at most generic for the graduate reverse lexicographical (denoted grevlex in the sequel) ordering (conjecture of Moreno-Socias). Moreover, in the case of the lexicographical ordering (denoted lex in the sequel) this statement is proved to be generically not satisfied for some choices of degrees of the entry polynomials. In the context of polynomial system solving, where the lex plays an important role, this fact becomes a challenging problem that is essential to overcome.

Thus, in this paper, we focus on the fundamental problem of change of ordering for a given p -adic Gröbner basis. In particular we provide precise results in the case where the input basis has a grevlex ordering and one wants to compute the lex basis corresponding. We will use the following notations.

1.1 Notations

Throughout this paper, K is a field with a discrete valuation val such that K is complete with respect to the norm defined by val . We denote by $R = O_K$ its ring of integers, m_K its maximal ideal and $k = O_K/m_K$ its fraction field. We denote by CDVF (complete discrete-valuation field) such a field. We refer to Serre's Local Fields [Ser79] for an introduction to such fields. Let $\pi \in R$ be a uniformizer for K and let $S_K \subset R$ be a system of representatives of $k = O_K/m_K$. All numbers of K can be written uniquely under its π -adic power series development form: $\sum_{k \geq l} a_k \pi^l$ for some $l \in \mathbb{Z}$, $a_k \in S_K$.

The case that we are interested in is when K might not be an effective field, but k is (*i.e.* there are constructive procedures for performing rational operations in k and for deciding whether or not two elements in k are equal). Symbolic computation can then be performed on truncation of π -adic power series development. We will denote by finite-precision CDVF such a field, and finite-precision CDVR for its ring of integers. Classical examples of such CDVF are $K = \mathbb{Q}_p$, with p -adic valuation, and $\mathbb{Q}[[X]]$ or $\mathbb{F}_q[[X]]$ with X -adic valuation. We assume that K is such a finite-precision CDVF.

The polynomial ring $K[X_1, \dots, X_n]$ will be denoted A , and $u = (u_1, \dots, u_n) \in \mathbb{Z}_{\geq 0}^n$, we write x^u for $X_1^{u_1} \dots X_n^{u_n}$.

1.2 Mains results

In the context of p -adic algorithmic, one of the most important behavior to study is the stability of computation: how the quality of the result, in terms of p -adic precision, evolves from the input. To quantify such a quality, it is usual to use an invariant, called condition number, related to the computation under study. Thus, we define such an invariant for the change of ordering.

Definition 1.1. Let $I \subset A$ be a zero-dimensional ideal. Let \leq_1 and \leq_2 be two monomial orderings on A . Let B_{\leq_1} and B_{\leq_2} be the canonical bases of A/I for \leq_1 and \leq_2 . Let M be the matrix whose columns are the $NF_{\leq_1}(x^\beta)$ for $x^\beta \in B_{\leq_2}$. We define the condition number of I for \leq_1 to \leq_2 , with notation $cond_{\leq_1, \leq_2}(I)$ (or $cond_{\leq_1, \leq_2}$ when there is no ambiguity) as the biggest valuation of an invariant factor of M .

We can now state our main result on change of ordering of p -adic Gröbner basis.

Theorem 1.2. *Let \leq_1 and \leq_2 be two monomial orderings. Let $G = (g_1, \dots, g_t) \in K[X_1, \dots, X_n]^t$ be an approximate reduced Gröbner basis for \leq_1 of the ideal I it generates, with $\dim I = 0$ and $\deg I = \delta$, and with coefficients known up to precision $O(\pi^N)$. Let β be the smallest valuation of a coefficient in G . Then, if $N > cond_{\leq_1, \leq_2}(I)$, the stabilized FGLM Algorithm, Algorithm 3, computes a Gröbner basis G_2 of I for \leq_2 . The coefficients of the polynomials of G_2 are known up to precision $N + n^2(\delta + 1)^2\beta - 2cond_{\leq_1, \leq_2}$. The time-complexity is in $O(n\delta^3)$.*

In the case of a change of ordering from grevlex to lex, we provide a more precise complexity result:

Theorem 1.3. *With the same notations and hypothesis as in Theorem 1.2. If \leq_1, \leq_2 are respectively instantiated to grevlex and lex, and if we assume that the ideal I is in shape position. Then, the adapted FGLM Algorithm for general position, Algorithm 6, computes a Gröbner basis G_2 of I for lex, in shape position. The coefficients of the polynomials of G_2 are known up to precision $N + \beta\delta - 2cond_{\leq_1, \leq_2}$. The time-complexity is in $O(n\delta^2) + O(\delta^3)$.*

In order to obtain these results, one has to tackle technical problems related to the core of the FGLM algorithm. Thus, we first present a summary of some important facts on this algorithm. Then we present more precisely the underlying problems in the p -adic situation.

1.3 The FGLM algorithm

For a given zero-dimensional I in a polynomial ring A , the FGLM algorithm [FGLM93] is mainly based on computational linear algebra in A/I . It allows to compute a Gröbner basis G_2 of I for a monomial ordering \leq_2 starting from a Gröbner basis G_1 of I for a first monomial ordering \leq_1 . To solve polynomial systems, one possible method is the computation of a Gröbner basis for lex. However, computing a Gröbner basis for lex by a direct approach is usually very time-consuming. The main application of the FGLM algorithm is to allow the computation of a Gröbner basis for lex by computing a Gröbner basis for grevlex then by applying a change of ordering to lex. The superiority of this approach is mainly due to the fact that the degrees of the intermediate objects are well controlled during

the computation of the grevlex Gröbner basis. The second step of this general method for polynomial system solving, is what we call the FGLM algorithm. Many variants and improvements (in special cases) of the FGLM algorithm have been published, *e.g.* Faugère and Mou in [FM11, FM13, Mou13] and Faugère, Gaudry, Huot and Renault [FGHR13, FGHR14, Huo13] takes advantages of sparse linear algebra and fast algorithm in linear algebra to obtain efficient algorithms. In this paper, as a first study of the problem of loss of precision in a change of ordering algorithm, we follow the original algorithm. This study already brings to light some problems for the loss of precision and propose solutions to overcome them. Thus, the FGLM algorithm we consider can be sketched as follows:

1. Order the images in A/I of the monomials of A according to \leq_2 .
2. Starting from the first monomial, test the linear independence in A/I of a monomial x^α with the x^β smaller than it for \leq_2 .
3. In case of independence, x^α is added to the canonical (for \leq_2) basis A/I in construction.
4. Otherwise, $x^\alpha \in LM(I)$ and the linear relation with the x^β smaller than it for \leq_2 give rise to a polynomial in I whose leading term is x^α .

Precision problems arise in step 2 and 4. The first one is the issue of testing the independence of a vector from a linear subspace. While it is possible to prove independence when the precision is enough, it is usually not possible to prove directly dependence. It is however possible to prove some dependence when there is more vectors in a vector space than the dimension of this vector space. It is indeed some dimension argument that permits to prove the stability of FGLM. We show (see Section 3) that it is enough to treat approximate linearly dependence (up to some precision) in the same way as in the non-approximate case and to check at the end of the execution of the algorithm that the number of independent monomials found is the same as the degree of the ideal. The second issue corresponds to adding the computation of an approximate relation. We show that the same idea of taking approximate linear dependence as non-approximate and check at the end of the computation is enough.

1.4 Linear algebra and Smith Normal Form

As we have seen, the FGLM algorithm relies mainly on computational linear algebra: testing linear independence and solving linear systems.

The framework of differential precision of [CRV14] has been applied to linear algebra in [CRV15] for some optimal results on the behavior of the precision for basic operations in linear algebra (matrix multiplication, LU factorization). From this analysis it seems clear, and this idea is well accepted by the community of computation over p -adics, that using the Smith Normal Form (SNF) to compute the inverse of a matrix or to solve a well-posed linear system is highly efficient and easy to handle. Moreover, it always achieves a better behavior than classical Gaussian elimination, even allowing gain in precision in some cases. Its optimality re-

mains to be proved but in comparison with classical Gaussian elimination, the loss of precision is far fewer.¹

This is the reason why we use the SNF in the p -adic version of FGLM we propose in this paper. In Section 2 we briefly recall some properties of the SNF and its computation. We also provide a dedicated version of SNF computation for the FGLM algorithm. More precisely, to apply the SNF computation to iterative tests of linear independence (as in step 2), we adapt SNF computation into some iterative SNF in Algorithm 4. This allows us to preserve an overall complexity in $O(nd^3)$.

2. SNF AND LINEAR SYSTEMS

2.1 SNF and approximate SNF

We begin by presenting our main tool, the SNF of a matrix in $M_{n,m}(K)$:

Proposition 2.1. *Let $M \in M_{n,m}(K)$. There exist some $P \in GL_n(O_K)$, $\text{val}(\det P) = 0$, $Q \in GL_m(O_K)$, $\det Q = \pm 1$ and $\Delta \in M_{n,m}(K)$ such that $M = P\Delta Q$ and Δ is diagonal, with diagonal coefficients being $\pi^{a_1}, \dots, \pi^{a_s}, 0, \dots, 0$ with $a_1 \leq \dots \leq a_s$ in \mathbb{Z} . Δ is unique and called the Smith Normal Form of M , and we say that P, Δ, Q realize the SNF of M . The a_i are called the invariant factors of M .*

In a finite-precision context, we introduce the following variant of the notion of SNF:

Definition 2.2. Let $M \in M_{n,m}(K)$, known up to precision $O(\pi^l)$. We define an **approximate SNF** of M as a factorization

$$M = P\Delta Q$$

with $P \in M_n(R)$, $\text{val}(\det P) = 0$, $Q \in M_m(R)$ with $\det Q = \pm 1$ known up to precision $O(\pi^l)$ and $\Delta \in M_{n,m}(K)$ such that $\Delta = \Delta_0 + O(\pi^l)$, where $\Delta_0 \in M_{n,m}(K)$ is a diagonal matrix, with diagonal coefficients of the form $\Delta_0[1, 1] = \pi^{\alpha_1}, \dots, \Delta_0[\min(n, m), \min(n, m)] = \pi^{\alpha_{\min(n, m)}}$ with $\alpha_1 \leq \dots \leq \alpha_{\min(n, m)}$. $\alpha_i = +\infty$ is allowed. (P, Δ, Q) are said to realize an approximate SNF of M .

To compute an approximate SNF, with use Algorithm 1.

Algorithm 1: SNFApproximate: Computation of an approximate SNF

Input : $M \in M_{n \times m}(K)$, known up to precision $O(\pi^l)$, with $l > \text{cond}(M)$
Output: P, Δ, Q realizing an approximate SNF of M .
 Find i, j such that the coefficient $M_{i,j}$ realize $\min_{k,t} \text{val}(M_{k,t})$;
 Track the following operations to obtain P and Q ;
 Swap rows 1 and i and columns 1 and j ;
 Normalize $M_{1,1}$ to the form $\pi^{\alpha_1} + O(\pi^l)$;
 By pivoting reduce coefficients $M_{i,1}$ ($i > 1$) and $M_{1,j}$ ($j > 1$) to $O(\pi^l)$. ;
Recursively, proceed with $M_{i \geq 2, j \geq 2}$;
 Return P, M, Q ;

Behaviour of Algorithm 1 is given by the following proposition:

Proposition 2.3. *Given an input matrix M , of size $n \times m$, with precision $(O(\pi^l))$ on its coefficients, Algorithm 1 terminates and returns U, Δ, V realising an approximate SNF of*

¹See Chapter 1 of [Vac15] for more details on the comparison between these two strategies.

M . Coefficients of U, Δ and V are known up to precision $O(\pi^l)$. Time complexity is in $O(\min(n, m) \max(n, m)^2)$ operations in K at precision $O(\pi^l)$.

Now, it is possible to compute the SNF of M , along with an approximation of a realization, from some approximate SNF of M with Algorithm 2.

Algorithm 2: SNFPrecised : from approximate SNF to SNF

Input : (U, Δ, V) (precision $O(\pi^l)$) realizing an approximate SNF of $M \in M_{n \times m}(K)$, of full rank. We assume $\text{cond}(M) < l$.
Output: Δ_0 the SNF of M , and U', V' known with precision $O(\pi^{l - \text{cond}(M)})$ such that $M = U'\Delta_0V'$, $\text{val}(\det U') = 0$ and $\det V = \pm 1$.

$\Delta_0 \leftarrow \Delta$;
 Track the following operations to obtain P and Q ;
 $t := \min(n, m)$;
for i from 1 to t **do**
 Normalize $\Delta_0[i, i]$;
 if $t = m$ **then**
 By pivoting with $\Delta_0[i, i]$, eliminate the coefficients $\Delta_0[j, i]$;
 else
 By pivoting with $\Delta_0[i, i]$, eliminate the coefficients $\Delta_0[i, j]$;
 Return Δ_0, P, Q ;

Proposition 2.4. *Given an input matrix M , of size $n \times m$, with precision $(O(\pi^l))$ on its coefficients ($l > \text{cond}(M)$), and (U, Δ, V) , known at precision $O(\pi^l)$, realizing an approximate SNF of M , Algorithm 2 computes the SNF of M , with U' and V' known up to precision $O(\pi^{l - \text{cond}(M)})$. Time complexity is in $O(\max(n, m)^2)$.*

We refer to [Vac14, Vac15] for more details on how to prove this result. We can then conclude on the computation of the SNF:

Theorem 2.5. *Given an input matrix M , of size $n \times m$, with precision $O(\pi^l)$ on its coefficients ($l > \text{cond}(M)$), then by applying Algorithms 1 and 2, we compute P, Q, Δ with $M = P\Delta Q$ and Δ the SNF of M . Coefficients of P and Q are known at precision $O(\pi^{l - \text{cond}(M)})$. Time complexity is in $O(\max(n, m)^2 \min(n, m))$ operations at precision $O(\pi^l)$.*

2.2 Solving linear systems

Computation of P and Q in the previous algorithms can be slightly modified to obtain (approximation of) P^{-1} and Q^{-1} , and thus M^{-1} .

Proposition 2.6. *Using the same context as the previous theorem, by modifying Algorithms 1 and 2 using the inverse operations of the one to compute P and Q , we can obtain P^{-1} and Q^{-1} with precision $O(\pi^{l - \text{cond}(M)})$. When $M \in GL_n(K)$, using $M^{-1} = Q^{-1}\Delta^{-1}P^{-1}$, we get M^{-1} with precision $O(\pi^{l - 2\text{cond}(M)})$. Time complexity is in $O(n^3)$ operations at precision $O(\pi^l)$.*

We can then estimate the loss in precision in solving a linear system:

Theorem 2.7. *Let $M \in GL_n(K)$ be a matrix with coefficients known up to precision $O(\pi^l)$ with $l > 2\text{cond}(M)$. Let $Y \in K^n$ be known up to precision $O(\pi^l)$. Then one can solve $Y = MX$ in $O(n^3)$ operations at precision $O(\pi^l)$. X is known at precision $O(\pi^{l - 2\text{cond}(M)})$.*

When the system is not square but we can ensure that $Y \in \text{Im}(M)$, then we have the following variant:

Proof. The proof of the correction of the classical FGLM algorithm shows that, if \mathbf{v} is a matrix whose columns are the $NF_{\leq}(x^\alpha)$ with $x^\alpha \in B_{\leq_2}$ and $x^\alpha < x^\beta$ (written in the basis B_{\leq}), then $NF_{\leq}(x^\beta) \in \text{Im}(\mathbf{v})$.

By applying the proof of the Proposition 2.8, we obtain that $NF_{\leq}(x^\beta) \in \pi^{-\text{cond}(\mathbf{v})} \text{Vect}_R(\{NF_{\leq}(x^\alpha) | x^\alpha \in B_{\leq_2}, x^\alpha < x^\beta\})$.

Finally, Lemma 3.2 implies that $\text{cond}(\mathbf{v}) \leq \text{cond}_{\leq, \leq_2}(I)$. The result is then clear. \square

3.2.2 Correction and termination

We can now prove the correction and termination of Algorithm 3 under the assumption that the initial precision is enough. Which precision is indeed enough is addressed in the following Subsubsection.

Proposition 3.5. *Let $G_1, \leq, \leq_2, B_{\leq}, B_{\leq_2}, I$ be as in Theorem 1.2. Then, assuming that the coefficients of the polynomials of G_1 are all known up to a precision $O(\pi^N)$ for some $N \in \mathbb{Z}_{>0}$ big enough, the stabilized FGLM algorithm 3 terminates and returns a Gröbner basis G_2 of I for \leq_2 .*

Proof. The computation of the multiplication matrices only involves multiplication and addition and the operation performed do not depend on the precision. This is similar for the computation of the $NF_{\leq}(x^\alpha)$ processed in the algorithm and obtained as product of T_i 's and $\mathbf{1}$. We may assume that all those $NF_{\leq}(x^\alpha)$, for $|x^\alpha|$, are obtained up to some precision $O(\pi^N)$ for some $N \in \mathbb{Z}_{>0}$ big enough. Subsubsection 3.2.3 gives a precise estimation on such an N and when it is big enough.

Let M be the matrix whose columns are the $NF_{\leq}(x^\beta)$ for $x^\beta \in B_{\leq_2}$. Let $\text{cond}_{\leq, \leq_2}$ be as in Definition 1.1.

To show the result, we use the following loop invariant: at the beginning of each time in the **while** loop of Algorithm 3, we have (i), $B_2 \subset B_{\leq_2}$ and (ii) if $x^\beta = B_2[j]x_i$ (where $(j, i) = m$, m taken at the beginning of the loop), then every monomial $x^\alpha <_2 x^\beta$ satisfies $x^\alpha \in B_{\leq_2}$ or $NF_{\leq}(x^\alpha) \in \pi^{N-\text{cond}_{\leq, \leq_2}} \text{Vect}_R(NF_{\leq}(B_{\leq_2})) + O(\pi^{N-\text{cond}_{\leq, \leq_2}})$. Here, $O(\pi^{N-\text{cond}_{\leq, \leq_2}})$ is the R -module generated by the $\pi^{N-\text{cond}_{\leq, \leq_2}} \epsilon$'s for $\epsilon \in B_{\leq}$.

We begin by first proving that this proposition is a loop invariant. It is indeed true when entering the first loop since $1 \in B_{\leq_2}$, for I is zero-dimensional.

We then show that this proposition is stable when passing through a loop. Let $x^\beta = B_2[j]x_i$ with $(j, i) = m$. By the way we defined it, x^β is in the border of B_2 (i.e. non-trivial multiple of a monomial of B_2). Since $B_2 \subset B_{\leq_2}$, we deduce that x^β is also in either in B_{\leq_2} , or in the border of B_{\leq_2} , also denoted by $\mathcal{B}_{\leq_2}(I)$.

We begin by the second case. We then have, thanks to Lemma 3.4, $NF_{\leq}(x^\beta) \in \pi^{-\text{cond}_{\leq, \leq_2}} \text{Vect}_R(\{NF_{\leq}(x^\alpha) | x^\alpha \in B_{\leq_2}, x^\alpha < x^\beta\})$. Precision being finite, it tells us that $\lambda = P_1 v = P_1 NF_{\leq}(x^\beta)$ only appears with coefficients of the form $O(\pi^{l'})$ for its coefficients or row of index $i > s$. This corresponds to being in the image of Δ .

Hence, the **if** test succeeds, and x^β is not added to B_2 . Points (i) and (ii) are still satisfied.

We now consider the first case, where $x^\beta \in B_{\leq_2}$. Once again, two cases are possible. The first one is the following: we have enough precision for, when computing $\lambda = P_1 v$ where $v = NF_{\leq}(x^\beta)$, we can prove that v is not in $\text{Vect}(NF_{\leq}(B_{\leq_2}))$. In other words, we are in the *else* case,

and x^β is rightfully added to B_2 . The points (i) and (ii) remain satisfied. In the other case: we do not have enough precision for, when computing $\lambda = P_1 v$ with $v = NF_{\leq}(x^\beta)$, we can prove that v is not in $\text{Vect}(B_{\leq_2})$. In other words, numerically, we get $NF_{\leq}(x^\beta) \in \pi^{-\text{cond}(\mathbf{v})} \text{Vect}_R(\{NF_{\leq}(x^\alpha) | x^\alpha \in B_{\leq_2}, x^\alpha < x^\beta\}) + O(\pi^{N-\text{cond}(\mathbf{v})})$. In that case, the **if** condition is successfully passed and, since $\text{cond}(\mathbf{v}) \leq \text{cond}_{\leq, \leq_2}$, the points (i) and (ii) remain satisfied.

This loop invariant is now enough to conclude this demonstration. Indeed, since $B_2 \subset B_{\leq_2}$ is always satisfied, we can deduce that L is always included in $B_{\leq_2} \cup \mathcal{B}_{\leq_2}(I)$, and since a monomial can not be considered more than once inside the **while** loop, there is at most $n\delta$ loops. Hence the termination.

Regarding correction, if the **if** test with $\text{card}(B_2) = \delta = \text{card}(B_{\leq_2})$ is passed, then, because of the inclusion we have proved, we have $B_2 = B_{\leq_2}$. In that case, the leading monomials which passed the **if** are necessarily inside the border of $\mathcal{B}_{\leq_2}(I)$, and can indeed be written in the quotient A/I in terms of the monomials of B_2 smaller than them. In other words, the linear system solving with the assumption of membership indeed builds a polynomial in I . In fine, G_2 is indeed a Gröbner basis of I for \leq_2 .

In the second case, where the **if** test is failed, with $\text{card}(B_2) \neq \delta$, then precision was not enough. \square

Algorithm 4: Update, iterated approximate SNF

Input : $s \in \mathbb{Z}_{\geq 0}$. A matrix \mathbf{v} of size $\delta \times s$, P_1, Q_1, Δ some matrices such that $P_1 \mathbf{v}' Q_1 = \Delta$ is an **approximate SNF** of \mathbf{v}' with \mathbf{v}' the sub-matrix of \mathbf{v} corresponding to its $s - 1$ first columns. P_2, Q_2 are the inverses of P_1, Q_1 .

Output : $P_1, P_2, Q_1, Q_2, \Delta$ updated such that $P_1 \mathbf{v}' Q_1 = \Delta$ is an approximate SNF of \mathbf{v} , and P_2, Q_2 are inverses of P_1, Q_1 .

Augment trivially the matrices Q_1, Q_2 into square invertible matrices with one more row and one more column ;
 Compute U_1, V_1 and Δ' realizing an approximate SNF of $P_1 \mathbf{v}' Q_1$, as well as U_2, V_2 the inverses of U_1, V_1 for Algorithm 1 ;
 $P_1 := U_1 \times P_1$;
 $Q_1 := Q_1 \times V_1$;
 $P_2 := P_2 \times U_2$;
 $Q_2 := V_2 \times Q_2$;
 $\Delta := \Delta'$;

3.2.3 Analysis of the loss in precision

We can now analyse the behaviour of the loss in precision during the execution of the stabilized FGLM algorithm 3, and thus estimate what initial precision is big enough for the execution to be without error. To that intent, we analyse the precision on the computation of the multiplication matrices and we use the notion of condition number of Definition 1.1 to show that it can handle the behaviour of precision inside the execution the stabilized FGLM algorithm 3. This is what is shown in the following propositions.

Lemma 3.6. *Let $I, G_1, \leq, B_{\leq}, \delta, \beta$ be as defined when announcing Theorem 1.2. Then the coefficients of the multiplication matrices for I are of valuation at least $n\delta\beta$.*

Proof. G_1 is a reduced Gröbner basis of a zero-dimensional ideal. Hence, it is possible to build a Macaulay matrix Mac with columns indexed by the monomials of $\text{mon} := \{X_i \times \epsilon, i \in [1, n], \epsilon \in B_{\leq}\}$, in decreasing order for \leq , and rows of the form $x^\alpha g$, with x^α a monomial and $g \in G$, such that this matrix is under row-echelon form, (left)-injective and all monomials in $\text{mon} \cap LM_{\leq}(I)$ are leading monomial of exactly one row of Mac . Since G_1 is a reduced Gröbner

Algorithm 5: Computation of the multiplication matrices

Input : The reduced Gröbner basis G of the zero-dimensional ideal $I \subset A$ for a monomial ordering \leq , $\deg I = \delta$.
 $B_{\leq} = (1 = \epsilon_1 \leq \epsilon_2 \leq \dots \leq \epsilon_\delta)$ the canonical basis of A/I for \leq .

Output : The multiplication matrices T_i 's for I and \leq .

for $i \in \llbracket 1, n \rrbracket$ **do**
 $T_i := 0_{\delta \times \delta}$;

$L := [x_i \epsilon_k | i \in \llbracket 1, n \rrbracket \text{ et } \epsilon_k \in B_{\leq}]$, ordered increasingly for \leq with no repetition ;

for $u \in L$ **do**
 if $u \in \mathcal{E}_{\leq}(I)$ **then**
 $T_i[u, u/x_i] = 1$ for all i such that $x_i | u$;
 /* The column indexed by u is zero, except on its coefficient indexed by u/x_i */
 else if $u = LM(g)$ for a certain $g \in G$ **then**
 Write g as $u + \sum_{k=1}^{\delta} a_k \epsilon_k$;
 $T_i[\cdot, u/x_i] := -{}^t(a_1, \dots, a_\delta)$ for all i such that $x_i | u$;
 else
 Find the smallest x_j for \leq such that $x_j | u$;
 Let $v = u/x_j$;
 Find ϵ and l such that $v = x_l \epsilon$;
 $V := T_l[\cdot, v]$ (this column contains $NF_{\leq}(v)$) ;
 $W := T_j V$ (W is the vector corresponding to the normal form $NF_{\leq}(x_j v) = NF_{\leq}(u)$) ;
 $T_i[\cdot, u/x_i] := W$ for all i such that $x_i | u$;
 Return T_1, \dots, T_n ;

basis, the first non-zero coefficients of the rows are 1 and all other coefficients are of valuation at least β . Mac has at most $n\delta$ columns and rows.

The computation of the reduced row-echelon form of Mac yields a matrix whose coefficients are of valuation at least $n\delta\beta$, except the first non-zero coefficient of each row which is equal to 1.

$NF_{\leq}(x^\alpha)$ for $x^\alpha \in \text{mon} \setminus B_{\leq}$ can then be read on the row of Mac of leading monomial x^α . It proves that the coefficients of such a $NF_{\leq}(x^\alpha)$ are of valuation at least $n\delta\beta$. The result is then clear. \square

Proposition 3.7. *Let $I, G_1, \leq, \leq_2, B_{\leq}, B_{\leq_2}$ be as defined when announcing Theorem 1.2. Let M be the matrix whose columns are the $NF_{\leq}(x^\beta)$ for $x^\beta \in B_{\leq_2}$. Then, if the coefficients of the polynomials of G_1 are all known up to some precision $O(\pi^N)$ with $N \in \mathbb{Z}_{>0}$, $N > \text{cond}_{\leq, \leq_2}(M) + n^2(\delta + 1)^2\beta$, the stabilized FGLM algorithm 3 terminates and returns an approximate Gröbner basis G_2 of I for \leq_2 . The coefficients of the polynomials of G_2 are known up to precision $N - n^2(\delta + 1)^2\beta - 2\text{cond}_{\leq, \leq_2}(M)$.*

Proof. We first analyse the behaviour of precision for the computation of the multiplication matrices. There are at most $n\delta$ matrix-vector multiplication in the execution of Algorithm 5. The coefficients involved in those multiplication are of valuation at least $n\delta\beta$ thanks to Lemma 3.6. Hence, the coefficients of the T_i are known up to precision $O(\pi^{N - (n\delta)^2\beta})$.

We now analyse the execution of Algorithm 3. The computation of v involves the multiplication of $\deg(v)$ T_i 's and $\mathbf{1}$. Hence, v is known up to precision $O(\pi^{N - (n\delta)^2\beta - \deg(v)n\delta\beta})$, which can be lower-bounded by $O(\pi^{N - (\delta+1)^2n^2\beta})$.

As a consequence, all coefficients of M are known up to precision $O(\pi^{N - (\delta+1)^2n^2\beta})$ and this is the same for its approximate SNF.

Now, we can address the loss in precision for the linear system solving. Thanks to Proposition 2.8, and with the membership assumption of v to $\text{Im}(\mathbf{v})$, a precision $O(\pi^N)$ with N strictly bigger than $(\delta+1)^2n^2\beta$ plus the biggest valuation c of an invariant factor of \mathbf{v} is enough to solve the linear system $\mathbf{v}W = v$, and the coefficients of W are determined up to precision $O(\pi^{N - n^2(\delta+1)^2\beta - 2c})$. The Lemma 3.2 then allows us to conclude that at any time, $c \leq \text{cond}_{\leq, \leq_2}(M)$, hence the result. \square

3.2.4 Complexity

To conclude the proof of Theorem 1.2, what remains is to give an estimation of the complexity of Algorithm 3. Regarding to the computation of multiplication matrices, there is no modification concerning complexity, and what we have to study is only the complexity of the iterated SNF computation. This is done in the following lemma:

Lemma 3.8. *Let $1 \leq s \leq \delta$ and prec be integers, $k \in \llbracket 1, s \rrbracket$ and $M, C^{(k)}$ be two matrices in $M_{\delta \times s}(K)$. We assume that the coefficients of M satisfies $M_{i,j} = m_{i,j}\delta_{i,j} + O(\pi^{\text{prec}})$ for some $m_{i,j} \in K$ and the coefficients of $C^{(k)}$ satisfies $C_{i,j}^{(k)} = c_{i,j}\delta_{j,k} + O(\pi^{\text{prec}})$ for some $c_{i,j} \in K$. Let $C_{\text{SNF}}(M + C^{(k)})$ be the number of operations in K (at precision $O(\pi^{\text{prec}})$) applied on rows and columns to compute an approximate SNF for $M + C^{(k)}$ at precision $O(\pi^{\text{prec}})$. Then $C_{\text{SNF}}(M + C^{(k)}) \leq s\delta$.*

Proof. We show this result by induction on s . For $s = 1$, for any $\delta, \text{prec}, k, M$ and $C^{(k)}$, the result is clear.

Let us assume that for some $s \in \mathbb{Z}_{>0}$, we have for any $\delta, \text{prec}, k, M$ and $C^{(k)} \in M_{\delta \times (s-1)}(K)$ as in the lemma, $C_{\text{SNF}}(M + C^{(k)}) \leq (s-1)\delta$.

Then, let us take some $\delta \geq s, k \in \llbracket 1, s \rrbracket$ and $\text{rec} \in \mathbb{Z}_{\geq 0}$. Let $M, C^{(k)}$ be two matrices in $M_{\delta \times s}(K)$ such that their coefficients satisfies $M_{i,j} = m_{i,j}\delta_{i,j} + O(\pi^{\text{prec}})$, for some $m_{i,j} \in K$, and $C_{i,j}^{(k)} = c_{i,j}\delta_{j,k} + O(\pi^{\text{prec}})$ for some $c_{i,j} \in K$. Let $N = M + C^{(k)}$.

We apply Algorithm 1 until the recursive call. Let us assume that the coefficient used as pivot, that is, one $N_{i,j}$ which attains the minimum of the $\text{val}(N_{i,j})$'s, is $N_{1,1}$. Then 1 operation on the columns is done when going through the two consecutives **for** loops in Algorithm 1. The only other case is that of pivot being some $N_{i,k}$ for some i . Then $\delta - 1$ operations on the rows and 1 operation on the columns are done.

The matrix $N' = \tilde{N}_{i \geq 2, j \geq 2}$ can be written $N' = M' + C'^{(k)}$ with M' and $C'^{(k)}$ in $M_{(\delta-1) \times (s-1)}(K)$ of the desired form, for $k = s - 1$ if the pivot $N_{i,j}$ is $N_{1,1}$ and $k = i$ if it is $N_{i,s}$. By applying the induction hypothesis on N' , we obtain that $C_{\text{SNF}}(M + C^{(k)}) \leq \delta + (\delta - 1) \times (s - 1) \leq \delta s$.

The result is then proved by induction. \square

We then have the following result regarding the complexity of Algorithm 3 :

Proposition 3.9. *Let G_1 be an approximate reduced Gröbner basis, for some monomial ordering \leq , of some zero-dimensional $I \subset A$ of degree δ , and let \leq_2 be some monomial ordering. We assume that the coefficients of G_1 are known up to precision $O(\pi^N)$ for some $N > \text{cond}_{\leq, \leq_2}$. Then, the complexity of the execution of Algorithm 3 is in $O(n\delta^3)$ operations in K at absolute precision $O(\pi^N)$.*

Proof. Firstly, we remark that the computation of the matrices of multiplication is in $O(n\delta^3)$ operations at precision $O(\pi^N)$. Now, we consider what happens inside the **while** loop in Algorithm 3. The computation of approximate SNF through Algorithm 4 are in $O(\delta^2)$ operations at precision $O(\pi^N)$ thanks to Lemma 3.8. The solving of linear systems thanks to Proposition 2.8 are also in $O(\delta^2)$ operations at precision $O(\pi^N)$. There is at most $n\delta$ entrance in this loop thanks to the proof of termination in Proposition 3.5. The result is then proved. \square

We can recall that the complexity of the classical FGLM algorithm is also in $O(n\delta^3)$ operations over the base field.

4. SHAPE POSITION

In this Section, we analyse the special variant of FGLM to compute a shape position Gröbner basis. We show that the gain in complexity observed in the classical case is still satisfied in our setting. We can combine this result with that of [Vac14] to express the loss in precision to compute a shape position Gröbner basis starting from a regular sequence.

4.1 Grevlex to shape

To fasten the computation of the multiplication matrices, we use the following notion.

Definition 4.1. I is said to be semi-stable for x_n if for all x^α such that $x^\alpha \in LM(I)$ and $x_n \mid x^\alpha$ we have for all $k \in [1, n-1]$ $\frac{x_k}{x_n} x^\alpha \in LM(I)$.

Semi-stability's application is then explained in Proposition 4.15, Theorem 4.16 and Corollary 4.19 of [Huo13] (see also [FGHR13]) that we recall here:

Proposition 4.2. *Applying FGLM for a zero-dimensional ideal I starting from a Gröbner basis G of I for grevlex:*

1. $T_i 1$ ($i < n$) can be read from G and requires no arithmetic operation;
2. If I is semi-stable for x_n , T_n can be read from G and requires no arithmetic operation;
3. After a generic change of variable, I is semi-stable for x_n .

The FGLM algorithm can then be adapted to this setting in the special case of the computation of a Gröbner basis of an ideal in shape position, with Algorithm 6.

Remark 4.3. If the ideal I is weakly grevlex (or the initial polynomials satisfy the more restrictive **H2** of [Vac14]), then I is semi-stable for x_n .

The remaining of this Section is then devoted to the proof of Theorem 1.3.

4.2 Correction, termination and precision

We begin by proving correction and termination of this algorithm.

Proposition 4.4. *We assume that the coefficients of the polynomials of the reduced Gröbner basis G_1 for grevlex are known up to a big enough precision, and that the ideal $I = \langle G_1 \rangle$ is in general position and semi-stable for x_n . Then Algorithm 6 terminates and returns a Gröbner basis for lex of I , yielding an univariate representation. Time complexity is in $O(\delta^3) + O(n\delta^2)$.*

Proof. As soon as one can certify that the rank of M is δ , the dimension of A/I , then we can certify that I possesses an univariate representation. Correction, termination are then clear. Computing T_n and the $T_i 1$ is free, computing the SNF is in $O(\delta^3)$ and solving the linear systems is in $O(n\delta^2)$, hence the complexity is clear. \square

What remains to be analysed is the loss in precision. To that intent, we use again the condition number of I (from grevlex to lex) and the smallest valuation of a coefficient of G_1 .

Proposition 4.5. *Let G_1 be the reduced Gröbner basis for grevlex of some zero-dimensional ideal $I \subset A$ of degree δ . We assume that the coefficients of the polynomials of G_1 are known up to precision $O(\pi^N)$ for some $N \in \mathbb{Z}_{>0}$, except the leading coefficients, which are exactly equal to 1. Let β be the smallest valuation of a coefficient of G_1 . Let $m = \text{cond}_{\text{grevlex,lex}}(I)$. We assume that $m - \delta\beta < N$, that I is in shape position and semi-stable for x_n . Then Algorithm 6 computes a Gröbner basis $(x_1 - h_1, \dots, x_{n-1} - h_{n-1}, h_n)$ of I for lex which is in shape position. Its coefficients are known up to precision $O(\pi^{N-2m+\delta\beta})$. The valuation of the coefficients of h_n is at least $\beta\delta - m$, and those of the h_i 's is at least $\beta - m$.*

Proof. There is no loss in precision for the computation concerning the multiplication matrices since it only involves reading coefficients on G_1 . Their coefficients are of valuation at least β . The columns of $M := \text{Mat}_{B_{\text{grevlex}}}(NF_{\leq}(1), \dots, NF_{\leq}(x_n^{\delta-1}))$ are obtained using T_n . Their coefficients are known up to precision $O(\pi^{N+(\delta-1)\beta})$ and are of valuation at least $(\delta-1)\beta$. For $\mathbf{z}[\delta]$, it is $O(\pi^{N+\delta\beta})$ and $\delta\beta$. The only remaining step to analyse is then the solving of linear systems, which is clear thanks to Theorem 2.7. \square

4.3 Summary on shape position

Thanks to the results of [Vac14] and [Vac15], we can express the loss in precision to compute a Gröbner basis in shape position under some genericity assumptions. Let $F = (f_1, \dots, f_n) \in R[X_1, \dots, X_n]$ be a sequence of polynomials satisfying the hypotheses **H1** and **H2** of [Vac14] for grevlex. Let D be the Macaulay bound of F and $I = \langle F \rangle$. We assume that I is strongly stable for x_n . Let $\delta = \deg(I)$. Let $\beta = -\text{prec}_{MF5}(F, D, \text{grevlex})$ be the bound on loss in precision to compute an approximate grevlex Gröbner basis of [Vac14]. Let $\gamma = -\delta\beta + 2\text{cond}_{\text{grevlex,lex}}(I)$.

Theorem 4.6. *If the coefficients of the f_i 's are known up to precision $N > \gamma$, then one can compute a shape position Gröbner basis for I with precision $N - \gamma$ on its coefficients.*

Proof. An approximate reduced Gröbner basis of I for grevlex is determined up to precision $N + 2\beta$ and its coefficients are of valuation at least β . Thanks to Proposition 4.5, the lexicographical Gröbner basis of I is of the form $x_1 - h_1(x_n), \dots, x_{n-1} - h_{n-1}(x_n), h_n(x_n)$. Moreover, the coefficients of h_n are of valuation at least $\delta\beta - \text{cond}_{\text{grevlex,lex}}(I)$ and known at precision $N - \delta\beta - 2\text{cond}_{\text{grevlex,lex}}(I)$. For the other h_i 's, the coefficients are of valuation at least $\beta - \text{cond}_{\text{grevlex,lex}}(I)$ and precision $N - \gamma$. \square

Remark 4.7. As a corollary, if $x_n \in R$ is such that $\text{val}(f'_n(x)) = 0$, then x_n lifts to $x \in V(I)$, known at precision $N - 2\gamma$.

Algorithm 6: Stabilized FGLM algorithm for an ideal in shape position starting from grevlex

Input : An approximate reduced Gröbner basis G_1 for grevlex of some ideal $I \subset A$ of dimension zero and degree δ . I is semi-stable for x_n and in in shape position.

Output : An approximate Gröbner basis G_2 of I for \leq_{lex} , in shape position, or **Error** if the precision is not enough.

Read the multiplication matrix T_n for I and grevlex using G ;
 $G_2 := \emptyset$;
 Read the $\mathbf{y}[i] := T_i \mathbf{1}$'s from G ($1 \leq i < n$);
 $\mathbf{z}[0] := \mathbf{1}$;
for i from 1 to δ **do**
 \lfloor Compute $\mathbf{z}[i] = T_n \mathbf{z}[i-1]$;
 $M := \text{Mat}_{B_{\leq}}(\mathbf{z}[0], \dots, \mathbf{z}[\delta-1])$;
 Compute Δ the SNF of M with $\Delta = PMQ$;
if $\text{rank}(M) == \delta$ **then**
 for i from 1 to $n-1$ **do**
 Let U s.t. $\mathbf{y}[i] = -M \cdot U$ thanks to P, Q, Δ and Thm 2.7;
 $h_i(T) := \sum_{i=1}^{\delta-1} U[i]T^i$;
 Let U s.t. $\mathbf{z}[\delta] = -M \cdot U$ thanks to P, Q, Δ and Thm 2.7;
 $h_n(T) := T^\delta + \sum_{i=1}^{\delta-1} U[i]T^i$;
 Return $x_1 - h_1(x_n), \dots, x_{n-1} - h_{n-1}(x_n), h_n(x_n)$;
else
 \lfloor Return "Error, not enough precision"

5. EXPERIMENTAL RESULTS

An implementation in Sage [S⁺11] of the previous algorithms is available at <http://www2.rikkyo.ac.jp/web/vaccon/fglm.sage>. Since the main goal of this implementation is the study of precision, it has not been optimized regarding to time-complexity. We have applied the main Matrix-F5 algorithm of [Vac14] to homogeneous polynomials of given degrees, with coefficients taken randomly in \mathbb{Z}_p (using the natural Haar measure): f_1, \dots, f_s , of degree d_1, \dots, d_s in $\mathbb{Z}_p[X_1, \dots, X_s]$, known at precision $O(p^{150})$, for grevlex, using the Macaulay bound D . We also used the extension to the affine case of [Vac14] to handle affine polynomials with the same setting (we specify this property in the column aff.). We have then applied our p-adic variant of FGLM algorithm, specialized for grevlex to lex or not, on the obtained Gröbner bases to get Gröbner bases for the lex order.

$d =$	nb_{test}	aff.	fast	D	p	max	mean	fail
[3,3,3]	20	no	no	7	2	21	3	(0,0)
[3,3,4]	20	no	no	8	2	21	3	(0,0)
[4,4,4]	20	no	no	10	2	28	5.2	(0,0)
[3,3,3]	20	yes	no	7	2	150	78	(0,0)
[3,3,4]	20	yes	no	8	2	149	92	(0,5)
[4,4,4]	20	yes	no	10	2	150	118	(0,11)
[3,3,3]	20	yes	yes	7	2	145	65	(0,1)
[3,3,4]	20	yes	yes	8	2	150	89	(0,7)
[4,4,4]	20	yes	yes	10	2	156	124	(0,15)
[3,3,3]	20	no	no	7	65519	0	0	(0,0)
[4,4,4]	20	no	no	10	65519	0	0	(0,0)
[3,3,3]	20	yes	no	7	65519	0	0	(0,0)
[4,4,4]	20	yes	no	10	65519	0	0	(0,0)
[3,3,3]	20	yes	yes	7	65519	0	0	(0,0)
[4,4,4]	20	yes	yes	10	65519	0	0	(0,0)

This experiment has been realized nb_{test} times for each given choice of parameters. We have reported in the previous array the maximal (column max), resp. mean (column mean), loss in precision (in successful computations), and the number of failures. This last quantity is given as a couple: the first part is the number of failure for the Matrix-F5 part and the second for the FGLM part.

We remark that these results suggest a difference of order in the loss in precision between the affine and the homogeneous case. Qualitatively, we remark that, for some given initial degrees, more computation (particularly computation involving loss in precision) are done in the affine case, because of the inter-reduction step. Also, it seems clear that loss in precision decreases when p increases, in particular, on small instances like here, loss in precision when $p = 65519$ are very unlikely.

6. FUTURE WORKS

Following this work, it would be interesting to investigate whether the sub-cubics algorithms of [FM11, FM13, Mou13, FGHR13, FGHR14, Huo13] could be adapted to the p -adic setting with reasonable loss in precision. Another possibility of interest for p -adic computation would be the extension of FGLM to tropical Gröbner bases.

7. REFERENCES

- [CL14] Xavier Caruso and David Lubicz. Linear algebra over $\mathbb{Z}_p[[u]]$ and related rings. *LMS J. Comput. Math.*, 17(1):302–344, 2014.
- [CRV14] Xavier Caruso, David Roe, and Tristan Vaccon. Tracking p -adic precision. *LMS J. Comput. Math.*, 17(suppl. A):274–294, 2014.
- [CRV15] Xavier Caruso, David Roe, and Tristan Vaccon. p -Adic Stability In Linear Algebra. pages 101–108, 2015.
- [FGHR13] Jean-Charles Faugère, Pierrick Gaudry, Louise Huot, and Guénaél Renault. Polynomial Systems Solving by Fast Linear Algebra. preprint, 2013. 23 pages.
- [FGHR14] Jean-Charles Faugère, Pierrick Gaudry, Louise Huot, and Guénaél Renault. Sub-cubic Change of Ordering for Gröbner Basis: A Probabilistic Approach. In *Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation*, pages 170–177, Kobe, Japon, July 2014. ACM.
- [FGLM93] Jean-Charles Faugère, Patrizia Gianni, Daniel Lazard, and Teo Mora. Efficient computation of zero-dimensional Gröbner bases by change of ordering. *Journal of Symbolic Computation*, 16(4):329–344, 1993.
- [FM11] Jean-Charles Faugère and Chenqi Mou. Fast Algorithm for Change of Ordering of Zero-dimensional Gröbner Bases with Sparse Multiplication Matrices. In *Proceedings of the 36th international symposium on Symbolic and algebraic computation*, ISSAC '11, pages 115–122, New York, NY, USA, 2011. ACM.
- [FM13] Jean-Charles Faugère and Chenqi Mou. Sparse FGLM algorithms. *CoRR*, abs/1304.1238, 2013.
- [Huo13] Louise Huot. *Résolution de systèmes polynomiaux et cryptologie sur les courbes elliptiques*. PhD thesis, Université Pierre et Marie Curie (Paris VI), December 2013. <http://tel.archives-ouvertes.fr/tel-00925271>.
- [Mou13] Chenqi Mou. *Solving Polynomial Systems over Finite Fields: Algorithms, Implementation and Applications*. Theses, Université Pierre et Marie Curie, May 2013.
- [S⁺11] W. A. Stein et al. *Sage Mathematics Software (Version 4.7.2)*. The Sage Development Team, 2011. <http://www.sagemath.org>.
- [Ser79] Jean-Pierre Serre. *Local fields*, volume 67 of *Graduate Texts in Mathematics*. Springer-Verlag, New York-Berlin, 1979. Translated from the French by Marvin Jay Greenberg.
- [TW95] Richard Taylor and Andrew Wiles. Ring-theoretic properties of certain Hecke algebras. *Ann. of Math. (2)*, 141(3):553–572, 1995.
- [Vac14] Tristan Vaccon. Matrix-F5 algorithms over finite-precision complete discrete valuation fields. In *Proceedings of the 2014 ACM on International Symposium on Symbolic and Algebraic Computation*, ISSAC '14, Kobe, Japan, July 23-25, 2014, pages 397–404, 2014.
- [Vac15] Tristan Vaccon. *p -adic precision*. Theses, Université Rennes 1, July 2015.