Autour des extensions de corps p-adiques

Alexandre Guillemot, Julien Soumier

4 décembre 2023

Table des matières

Introduction

Dans ce rapport, nous nous penchons sur l'arithmétique effective dans les extensions non ramifiées de $\mathbb{Q}_p$. Ayant une forme toute particulière, il est possible de les représenter par différents paramètres, afin de faire des opérations sur leurs éléments. Elles disposent de plus de propriétés théoriques intéressantes qui emmènent plusieurs questions, telles que la résolution de l'équation généralisée d'Artin-Schreier.

Ainsi nous avons décidé d'implémenter un nouveau module en `C`, `zqadic`, qui se base sur la librairie `flint`, et qui l'étend en rajoutant plusieurs procédures et types permettant de faire des calculs effectifs dans de telles extensions. La présentation théorique et pratique de ce module sera articulée en deux chapitres, ayant comme objectif final de présenter une implémentation de la résolution de l'équation d'Artin-Schreier.

Le premier chapitre aura donc pour but de poser les bases théoriques ainsi que les fondements du module : sont au programme extensions non ramifiées de $\mathbb{Q}_p$ et leur représentation informatique, substitution de Frobenius et l'algorithme pour la calculer.

Le deuxième chapitre portera quant à lui sur le coeur du sujet. Une première section plus théorique apportera le bagage nécessaire à la preuve de l'existence de solutions d'Artin-Schreier, pour ensuite présenter la mise en pratique consistant en la réalisation informatique de l'algorithme de Harley.

Chapitre 1

Arithmétique de $\mathbb{Z}_q$

Dans ce premier chapitre, nous nous penchons sur les méthodes de calculs dans les extensions non ramifiées de $\mathbb{Q}_p$. A l'instar des corps finis, on peut les écrire comme des quotients par un polynôme irréductible, ce qui nous permet de produire des algorithmes pour calculer effectivement dans de telles structures. Ainsi le côté théorique des différents aspects de ces extensions sera agrémenté des implémentations associées.

1.1 Quelques résultats sur les nombres p-adiques

1.1.1 Extensions non ramifiées de $\mathbb{Q}_p$

Nous exposons ici des résultats classiques sur l'existence et les propriétés des extensions non ramifiées du corps p-adique. Le lecteur curieux pourra trouver les démonstrations d'énoncés plus généraux, dont les théorèmes suivant sont des cas particuliers, dans l'excellent [Ser2]. (Chapitres 1,2 et 3).

Dans toute la suite $p \in \mathbb{N}$ désigne un nombre premier, et $\mathbb{Q}_p$ le corps des nombres p-adiques correspondant. Commençons par définir les extensions auxquelles nous allons nous intéresser. Pour cela on a besoin d'un premier lemme, qui permet d'étendre la norme (et donc la valuation) sur $\mathbb{Q}_p$ à toute extension finie.

Lemme 1.1.1. *Soit $K/\mathbb{Q}_p$ finie de degré $n \in \mathbb{N}$ une extension finie de $\mathbb{Q}_p$. Alors il existe une unique norme notée $|.|_K$ qui prolonge la norme p-adique.*

On rappelle avant toute chose, la correspondance entre valuation et norme de corps :

Rq 1.1.1. Si $|.|$ est une valeur absolue , alors on a une valuation associée :

$$v(x) = -\log(|x|)$$

Ainsi avec les mêmes notation, on muni K d'une structure de corps valué, et on peut définir son anneau de valuation :

$$\mathcal{R} = \{x \in K \mid |x|_K \leq 1\} = \{x \in K \mid v_K(x) \geq 0\}$$

son idéal maximal :

$$\mathcal{M} = \{x \in K \mid |x|_K < 1\} = \{x \in K \mid v_K(x) > 0\}$$

et son corps résiduel :

$$\mathcal{K} = \mathcal{R}/\mathcal{M}$$

On peut montrer que dans ce cas, le corps résiduel $\mathcal{K}$ est toujours isomorphe à une extension algébrique de $\mathbb{F}_q$, de degré noté f. On est donc en mesure de donner la définition cohérente suivante :

Définition 1.1.1. (Extension non ramifiée) Considérons une extension $K/\mathbb{Q}_p$ finie de degré $n \in \mathbb{N}$. On dira que K est une extension non ramifiée quand son corps résiduel est isomorphe à $\mathbb{F}_q$ avec $q{=}p^n$.

Rq 1.1.2. On peut plus généralement définir l'indice de ramification absolu de $K/\mathbb{Q}_p$ par $e = v_K(p \times 1_{\mathbb{Z}_p})$, et si n est le degré de $K/\mathbb{Q}_p$, alors on a l'équation :

$$n = ef$$

qui montre que K est non ramifiée si et seulement si $e = 1$.

D'autre part, la propriété d'être non ramifiée est assez rigide pour permettre une classification à isomorphisme près.

Proposition 1.1.1. *Notons $K = \mathbb{Q}_p[X]/(m)$ avec $m \in \mathbb{Z}_p[X]$ irréductible. Si on étend linéairement $\mathcal{R} \to \mathcal{K}$ la projection canonique, en $\pi : \mathcal{R}[X] \to \mathcal{K}[X]$ et on note $\overline{m} = \pi(m)$ on obtient l'équivalence suivante :*

$$K/\mathbb{Q}_p \text{ est non ramifiée} \iff deg(m) = deg(\overline{m})$$

Ainsi on a $\mathcal{K} = \mathbb{F}_q$ où $q = p^d$ avec $d = deg(\overline{m})$. Et l'unicité :
Si K_1 et K_2 sont deux extensions non ramifiées définies par respectivement m_1 et m_2, alors :

$$K_1 \cong K_2 \iff deg(m_1) = deg(m_2)$$

Ainsi pour tout $n \geq 1$, il existe une unique extension de $\mathbb{Q}_p$ non ramifiée de degré n, que l'on notera $\mathbb{Q}_q$, et dont l'anneau de valuation sera noté $\mathbb{Z}_q$.

1.1.2 Représentation effective de $\mathbb{Z}_q$

Dans l'objectif de faire des calculs effectifs dans $\mathbb{Z}_q$, nous avons dans un premier temps besoin de définir ce qui représentera un tel objet. Nous avons vu qu'il peut s'écrire $\mathbb{Z}_p[X]/(M)$, pour $M \in \mathbb{Z}_p[X]$ un relèvement irréductible unitaire d'un polynôme irréductible $m \in \mathbb{F}_p[X]$, et tel que $\deg m = \deg M$. Ainsi nous aurons besoin principalement de cette donnée pour représenter notre extension, mais d'autres éléments vont s'y ajouter pour des raisons pratiques et d'efficacité. Nous avons donc décidé de créer deux types `zqadic_t` et `zqadic_ctx_t`, que nous présentons dans cette partie :

```
typedef padic_poly_t zqadic_t;

typedef struct _zqadic_ctx_t
{
    slong prec;
    slong deg;
    enum rep_type type;
    fmpz_t p;
    zqadic_t* C;
    padic_ctx_t pctx;
    padic_poly_t M;
} zqadic_ctx_t[1];
```

Nous représenterons donc les éléments de $\mathbb{Z}_q$ comme des polynômes à coefficients dans $\mathbb{Z}_p$, et nous les réduirons modulo M au fur et à mesure des calculs. Et pour `zqadic_ctx_t`, les différents champs sont :

- `M` représente le polynôme M, de type `padic_poly_t` défini dans le modulo `padic` de `flint`,
- `pctx` est le contexte p-adique associé à $\mathbb{Z}_p$, regroupant donc les données associé à cette structure,
- `p` est le nombre premier tel que $q = p^r$,
- `deg` est le degré de l'extension associée à $\mathbb{Z}_q$, donc égal au r tel que $q = p^r$,
- `C` est un tableau d'éléments précalculés de $\mathbb{Z}_q$, dont nous auront utilité dans la suite du rapport,
- `prec` est la précision maximale à laquelle les calculs peuvent être réalisés dans l'extension. En effet, un nombre p-adique contient potentiellement une infinité de donnée, et donc nous regardons en permanance de tels nombres modulo une puissance de p. Et alors, `prec` représente la puissance maximale à laquelle on peut regarder un élément dans cette extension, sachant que la précision d'un polynôme à coefficients dans $\mathbb{Z}_p$ correspond à la précision de ses coefficients.

- `type` est un `enum` qui peut valoir `SPARSE` ou `TEICHMULLER`. Il permet d'adapter les algorithmes utilisés aux propriétés qu'ont le représentant M de l'extension.

1.2 Philosophie de Hensel, relèvement de solutions approchées

Dans cette section on s'intéresse à une propriété fondamentale des nombres p-adiques, le raffinement de solutions approchées modulo p. L'idée (que l'on retrouve en parcourant la démonstration) est la même que la méthode de Newton, mais dans notre cas, la distance est *ultramétrique*, car associée à une valuation p-adique, et non *archimédienne* comme la valeur absolue.

Le premier résultat, historiquement du à Hensel est le suivant. Le lecteur curieux pourra trouver la démonstration en annexe.

Théorème 1.2.1. *(Lemme de Hensel)*
Soient $f \in \mathbb{Z}_p[X_1...X_m]$, $x \in (\mathbb{Z}_p)^m$, $n, k \in \mathbb{Z}$ et $j \in [\![1, m]\!]$ tels que :

$$0 \leq 2k < n, \;\; f(x) \equiv 0 \; [p^n], \;\; v_p(\tfrac{\partial f}{\partial X_j}(x)) = k.$$

Alors il existe un zéro y de f dans $(\mathbb{Z}_p)^m$ tel que $x \equiv y \, [p^{n-k}]$.

De cette manière on peut construire des solutions dans $\mathbb{Z}_p$ à partir de solutions dans $\mathbb{Z}/p^n\mathbb{Z}$. Une application de ce principe nous amène à un nouveau procédé pour factoriser les polynômes. On remarque que la précision augmente de manière quadratique, ce qui rappelle encore une fois l'itération de Newton.

Théorème 1.2.2. *(Factorisation de Hensel) Soit $k \in \mathbb{N}$, $f, g_k, h_k \in \mathbb{Z}_p[X]$, h_k unitaire, tels que :*

$$f = g_k h_k \mod p^k \quad \text{et} \quad g_k, \, h_k \text{ premiers entre eux}$$

Alors il existe g_{2k}, $h_{2k} \in \mathbb{Z}_p[X]$ tels que :

$$\begin{cases} f = g_{2k} h_{2k} \mod p^{2k} \\ g_{2k} = g_k \mod p^k \\ h_{2k} = h_k \mod p^k \end{cases}$$

1.3 Substitution de Frobenius et modulo de Teichmüller

1.3.1 Groupe de galois d'une extension non ramifiée de $\mathbb{Q}_p$

Soit $q = p^n$, on note toujours $\mathbb{Q}_q$ une extension non ramifiée de $\mathbb{Q}_p$ et $\mathbb{Z}_p$ l'anneau de valuation associé. Cette partie est consacrée à une propriété fondamentale de ces exten-

sions qui en fait tout son intêret : elles respectent les propriétés galoisiennes de leurs corps résiduels. En clair on a :

Théorème 1.3.1. *(Caractérisation du groupe de Galois)* $\mathbb{Q}_q/\mathbb{Q}_p$ *est une extension galoisienne, et on a*

$$\mathrm{Gal}(\mathbb{Q}_q/\mathbb{Q}_p) \cong \mathrm{Gal}(\mathbb{F}_q/\mathbb{F}_p)$$

Une conséquence directe de ce théorème est que le groupe de Galois $\mathrm{Gal}(\mathbb{Q}_q/\mathbb{Q}_p)$ est cyclique, engendré par un élément, que l'on notera Σ, appelé *substitution de Frobenius*. Une propriété fondamentale de la substitution, est l'égalité suivante qui justifiera beaucoup de calculs par la suite :

$$\Sigma(x) \equiv x^p \mod p \quad \text{pour } x \in \mathbb{Z}_q \tag{1.1}$$

1.3.2 Modulo de Teichmüller

En conservant les notations de la section précédente, on sait que $\mathbb{Q}_q$ peut être représenté comme un $\mathbb{Q}_p[X]/(M)$ pour un certain $M \in \mathbb{Z}_p[X]$ relèvement unitaire irréductible d'un $m \in \mathbb{F}_p[X]$ unitaire irréductible, tel que $\deg M = \deg m = n$. L'idée de la représentation de Teichmüller est de choisir M de façon à conserver les bonnes propriétés que le morphisme de Frobenius possède sur $\mathbb{F}_q/\mathbb{F}_p$, pour pouvoir les transposer sur $\mathbb{Z}_q$. Remarquons alors dans un premier temps que $m \mid X^q - X$, car $\mathbb{F}_q = \mathbb{F}_p[X]/(m)$ et $\mathbb{F}_q$ est le corps de décomposition (et de rupture) de $X^q - X$. Ainsi, la factorisation de Hensel assure le résultat suivant :

Proposition 1.3.1. *(Polynôme de Teichmüller) Il existe un unique* $M \in \mathbb{Z}_p[X]$ *vérifiant les conditions suivantes :*

1. $M \mid X^{q-1} - 1$
2. $M(X) \equiv m(X) \mod p$

On pourra consulter l'annexe pour avoir le détail de la construction. On remarque alors plusieurs choses, qui justifient le choix de M. Si θ est une racine de M vu dans une clôture algébrique, on identifie $\mathbb{Q}_q$ et $\mathbb{Q}_p(\theta)$.

$$0 = M(\theta) = \theta^{q-1} - 1$$

et donc θ est une racine $(q-1)^{ieme}$ de l'unité dans $\mathbb{Z}_q$. En outre comme Σ est un automorphisme de Galois, $\Sigma(\theta)$ est encore une racine $(q-1)^{ieme}$ de l'unité. Ceci, combiné au fait que $\Sigma(\theta) \equiv \theta^p \mod p$, nous assure que :

$$\Sigma(\theta) = \theta^p \tag{1.2}$$

1.3.3 Calcul effectif d'un modulo de Teichmüller

Le calcul du polynôme de Teichmüller

Pour le calcul du polynôme de Teichmüller, nous avons suivi une idée de Harley, reposant sur le fait que l'on peu doubler la précision de solutions approchées, pour arriver à une précision convenable. Nous commencerons par le cas p premier quelconque, puis nous nous restreindrons au cas $p = 2$ qui nous intéresse particulièrement.

On se place dans le cadre suivant. $\mathbb{F}_q = \mathbb{F}_p[\overline{\theta}]$, et on note θ l'unique élément de $\mathbb{Z}_q$ vérifiant :

$$\begin{aligned} \theta^{q-1} &= 1 \\ \theta &\equiv \overline{\theta} \mod p \end{aligned}$$

Un tel θ existe d'après le lemme d'Hensel appliqué à $f(X) = X^{q-1} - 1$. On conserve les notations de la section précédente, avec $\mathbb{Z}_q = \mathbb{Z}_p[\theta] = \mathbb{Z}_p[X]/(M)$, M de degré d. Comme θ est racine de M qui est son polynôme minimal, on obtient les autres racines par action de Σ qui engendre $Gal(\mathbb{Q}_q/\mathbb{Q}_p)$. On factorise alors :

$$M(X) = \prod_{i=0}^{d-1} (X - \theta^{p^i})$$

Notons ζ_p une racine primitive de l'unité dans une extension. Sachant que pour $i \in [\![0, d-1]\!]$ on a :

$$X^p - \theta^{p^i} = (-1)^{p-1} \prod_{k=0}^{p-1} (\zeta_p^k X - \theta^{p^{i-1}})$$

et donc

$$M(X^p) = (-1)^{(p-1)d} \prod_{i=0}^{p-1} M(\zeta_p^i X)$$

Or si on écrit $M(X) = \sum_{i=0}^{p-1} M_i(X^p) X^i$ avec les $M_i \in \mathbb{Z}_q[X]$, on obtient en réinjectant :

$$\begin{aligned} M(X^p)(-1)^{(p-1)d} &= \prod_{i=0}^{p-1} \Big(\sum_{i=0}^{p-1} \zeta_p^{ji} X^i M_i(X^p) \Big) \\ &= \sum_{k=0}^{p-1} h_k(M_0(X^p), \ldots, M_{p-1}(X^p)) X^{pk} \end{aligned}$$

Où un calcul pénible montre que les $h_k \in \mathbb{Z}_q[Y_0, \ldots, Y_{p-1}]$ sont homogènes de degré p. Et donc

$$M(X) = (-1)^{(p-1)d} \sum_{k=0}^{p-1} h_k(M_0(X), \ldots, M_{p-1}(X)) X^k \tag{1.3}$$

On s'est ramené à la recherche des polynômes h_k (qui sont fixes à p fixés) et des M_i qui nécessiteront plus de travail pour être calculés.

Restriction au cas $p = 2$ et heuristique

Calculons les h_k dans notre cas. On a :

$$\begin{aligned}(-1)^d M(X^2) =& (M_0(X^2) + M_1(X^2)X)(M_0(X^2) - M_1(X^2)X) \\ =& M_0(X^2) - M_1(X^2)^2 X^2\end{aligned}$$

Ainsi on a $h_0 = Y_0^2$ et $h_1 = -Y_1^2$. La prochaine étape de l'algorithme consiste à voir comment on peut retrouver M à partir d'une approximation M_t à l'ordre $t \in \mathbb{N}^*$ (par exemple pour $t = 1$ on a m). Supposons donc que l'on ai :

$$M_t(X) \equiv M(X) \mod 2^t$$

On écrit

$$M(X) = M_t(X) + 2^t \delta_t(X)$$

et donc en substituant dans (1.3) réduite modulo 2^t, on obtient la relation :

$$\delta_t - 2(M_{t,0}\delta_{t,0} - XM_{t,1}\delta_{t,1}) + V_t \equiv 0 \mod 2^t \tag{1.4}$$

où on a :

$$V_t \equiv \frac{1}{2^t}(M_t - M_{t,0}^2 + XM_{t,1}^2) \mod 2^t$$

On à réduit notre problème à trouver δ_t. Et pour cela nous allons utiliser de la récursivité, pour doubler la précision d'une potentielle solution approchée. Notons alors $t' = \lceil \frac{t}{2} \rceil$, et supposons que l'on ai un algorithme qui trouve $\delta_t \mod 2^{t'}$ grâce à l'équation (1.4). On écrit alors $\delta_t = \delta_{t'} + 2^{t'}\Delta_{t'}$ et on l'injecte dans (1.4). On obtient dans ce cas une équation tout à fait similaire :

$$\Delta_{t'} - 2(M_{t,0}\Delta_{t',0} - XM_{t,1}\Delta_{t',1}) + W_t \equiv 0 \mod 2^{t-t'}$$

avec

$$W_t \equiv \frac{1}{2^{t'}}\Big(V_t + \delta_{t'} - 2(M_{t,0}\delta_{t',0} - XM_{t,1}\delta_{t',1})\Big) \mod 2^{t-t'}$$

Or $t - t' \leq t'$, et donc on peut réutiliser l'algorithme hypothétique pour calculer $\Delta_{t'}$. Et donc on retrouve δ_t. Et enfin M.

L'implémentation du calcul de polynômes de Teichmüller

En se basant sur l'approche proposée dans le paragraphe précédent, deux procédures `_zqadic_teichmuller_modulus` et `_zqadic_teichmuller_modulus_increment` ont été implémentées pour calculer un modulo de Teichmüller. Le deuxième permet de calculer δ_t, en s'appelant récursivement pour le calcul de $\delta_{t'}$ et le calcul de $\Delta_{t'}$, puis `_zqadic_teichmuller_modulus` s'appelle récursivement pour calculer M_{2t} à partir de M_t et δ_t calculé avec `_zqadic_teichmuller_modulus_increment`. A chaque fois, la précision à laquelle le résultat est calculé est donné par la précision à laquelle le résultat (M et δ) est initialisé.

```
void _zqadic_teichmuller_modulus(padic_poly_t M, padic_poly_t m, padic_ctx_t C)
{
    slong N = padic_poly_prec(M);

    if (N == 1) padic_poly_set(M, m, C);
    else
    {
        slong Nr = (N >> 1) + (N & 1); // partie entiere supérieure de N/2

        padic_poly_t P1, P2; // variables cache
        padic_poly_t Mr, M0, M1, V, delta; // M', M_0, M_1, V, \delta

        padic_poly_init2(Mr, 0, Nr);

        _zqadic_teichmuller_modulus(Mr, m, C);

        // calcul de M0
        padic_poly_init2(M0, 0, N);
        padic_poly_init2(P1, 0, N + 1);
        padic_poly_init2(P2, 0, N + 1);

        padic_poly_set(P1, Mr, C);
        zqadic_comp_moins_x(P2, P1, C);
        padic_poly_add(P1, P1, P2, C);
        zqadic_mul_pn(M0, P1, -1, C);
        zqadic_precomp_x2(M0, M0, C);

        // calcul de M1
        padic_poly_init2(M1, 0, N);

        padic_poly_set(P1, Mr, C);
        zqadic_comp_moins_x(P2, P1, C);
        padic_poly_sub(P1, P1, P2, C);
        zqadic_mul_pn(M1, P1, -1, C);
        padic_poly_shift_right(M1, M1, 1, C); // division par X
        zqadic_precomp_x2(M1, M1, C);

        padic_poly_clear(P1);
```

```
        padic_poly_clear(P2);

        // calcul de V
        padic_poly_init2(V, 0, N - Nr);
        padic_poly_init2(P1, 0, N);
        padic_poly_init2(P2, 0, N);

        padic_poly_mul(P1, M1, M1, C);
        padic_poly_shift_left(P1, P1, 1, C);
        padic_poly_mul(P2, M0, M0, C);
        padic_poly_add(P1, P1, Mr, C);
        padic_poly_sub(P1, P1, P2, C);
        zqadic_mul_pn(V, P1, -Nr, C);

        padic_poly_clear(P1);
        padic_poly_clear(P2);

        // calcul de delta
        padic_poly_init2(delta, 0, N - Nr);

        _zqadic_teichmuller_modulus_increment(delta, M0, M1, V, C);

        // calcul de M
        padic_poly_init2(P1, 0, N);

        padic_poly_set(P1, delta, C);
        zqadic_mul_pn(P1, P1, Nr, C);
        padic_poly_add(M, Mr, P1, C);

        padic_poly_clear(P1);

        // clears
        padic_poly_clear(Mr);
        padic_poly_clear(M0);
        padic_poly_clear(M1);
        padic_poly_clear(V);
        padic_poly_clear(delta);
    }
}

void _zqadic_teichmuller_modulus_increment(padic_poly_t delta, padic_poly_t M0,
↪  padic_poly_t M1, padic_poly_t V, padic_ctx_t C)
{
    slong N = padic_poly_prec(delta);

    if (N == 1) padic_poly_neg(delta, V, C);
    else
    {
        slong Nr = (N >> 1) + (N & 1); // partie entiere supérieureerieure de N/2
```

```
        padic_poly_t P1, P2; // Variables cache
        padic_poly_t Delta, deltar, delta0, delta1, Vr; // \Delta, \delta', \delta_0,
↪  \delta_1, V'

        padic_poly_init2(deltar, 0, Nr);
        _zqadic_teichmuller_modulus_increment(deltar, M0, M1, V, C);

        // calcul de delta0
        padic_poly_init2(delta0, 0, N);
        padic_poly_init2(P1, 0, N + 1);
        padic_poly_init2(P2, 0, N + 1);

        padic_poly_set(P1, deltar, C);
        zqadic_comp_moins_x(P2, P1, C);
        padic_poly_add(P1, P1, P2, C);
        zqadic_mul_pn(delta0, P1, -1, C);
        zqadic_precomp_x2(delta0, delta0, C);

        // calcul de delta1
        padic_poly_init2(delta1, 0, N);

        padic_poly_set(P1, deltar, C);
        zqadic_comp_moins_x(P2, P1, C);
        padic_poly_sub(P1, P1, P2, C);
        zqadic_mul_pn(delta1, P1, -1, C);
        padic_poly_shift_right(delta1, delta1, 1, C); // division par X
        zqadic_precomp_x2(delta1, delta1, C);

        padic_poly_clear(P1);
        padic_poly_clear(P2);

        // calcul de Vr
        padic_poly_init2(Vr, 0, N - Nr);
        padic_poly_init2(P1, 0, N);
        padic_poly_init2(P2, 0, N);

        padic_poly_mul(P1, M0, delta0, C);
        padic_poly_mul(P2, M1, delta1, C);
        padic_poly_shift_left(P2, P2, 1, C);
        padic_poly_sub(P1, P1, P2, C);
        zqadic_mul_pn(P1, P1, 1, C);
        padic_poly_sub(P1, deltar, P1, C);
        padic_poly_add(P1, V, P1, C);
        zqadic_mul_pn(Vr, P1, -Nr, C);

        padic_poly_clear(P1);
        padic_poly_clear(P2);

        // calcul de Delta
        padic_poly_init2(Delta, 0, N - Nr);
```

```
        _zqadic_teichmuller_modulus_increment(Delta, M0, M1, Vr, C);

        // calcul de delta
        padic_poly_init2(P1, 0, N);

        padic_poly_set(P1, Delta, C);
        zqadic_mul_pn(P1, P1, Nr, C);
        padic_poly_add(delta, deltar, P1, C);

        padic_poly_clear(P1);

        // clears
        padic_poly_clear(delta0);
        padic_poly_clear(delta1);
        padic_poly_clear(Vr);
        padic_poly_clear(Delta);
        padic_poly_clear(deltar);
    }
}
```

Ces deux procédures sont alors situées dans le script `ctx_init.c`, qui regroupe les procédures utiles à l'initialisation d'un contexte `zqadic_ctx_t`, représentant $\mathbb{Z}_q$ pour un certain $q = p^n$. Ainsi nous pouvons désormais initialiser un tel contexte avec `zqadic_ctx_init_teichmuller` qui calcule le modulo de Teichmüller d'un polynôme irréductible aléatoire de $\mathbb{F}_p[X]$. Nous avons aussi implémenté une procédure `zqadic_ctx_init` qui permet d'initialiser un contexte en prenant le relèvement naturel d'un polynôme irréductible creux $m \in \mathbb{F}_p[X]$ en voyant ses coefficients comme des éléments de $[\![0, p-1]\!]$ et en utilisant l'inclusion canonique $\mathbb{Z} \to \mathbb{Z}_p$. Dans le cas où `zqadic_ctx_init_teichmuller` est appelée, le `type` d'un contexte vaut `TEICHMULLER`, et il vaudra `SPARSE` si c'est `zqadic_ctx_init` qui est utilisée. Les deux représentations ont leurs avantages et leurs inconvénients : le modulo de Teichmüller permet un calcul plus efficace de la substitution de Frobenius, mais changer la précision maximale d'un contexte défini par un modulo de Teichmüller requiert de recalculer le modulo à cette nouvelle précision, alors qu'un modulo creux est déjà défini à précision infinie, donc les calculs dans une extension définie par un modulo creux peuvent être faits à n'importe quelle précision. Aussi certaines opérations telles que la multiplication par le modulo (qui intervient notamment dans la réduction modulo M) peuvent s'avérer peu coûteuses si la plupart des coefficients du modulo sont nuls.

1.3.4 Implémentation du Frobenius et de son inverse

Dans cette section, nous passons en revue l'implémentation que nous avons réalisée pour calculer la substitution de Frobenius, dans le cas où le contexte est défini par un modulo de Teichmüller. Nous avons aussi implémenté la substitution de Frobenius pour un modulo creux, mais son intêret est moindre vu que son calcul est beaucoup moins efficace.

Le Frobenius

On considère donc à partir de maintenant que $\mathbb{Z}_q$ est représenté par $\mathbb{Z}_p[X]/(M) = Z_p[\theta]$ où M est un polynôme de Teichüller. Au vu du comportement de Σ par rapport à θ (1.2), on a directement l'expression du Frobenius par le reste de la division par M : si on note $a = \sum_{i=0}^{d-1} a_i X_i \in \mathbb{Z}_p[X]$,

$$\Sigma(a) = \Sigma\left(\sum_{i=0}^{d-1} a_i X^i\right) = \sum_{i=0}^{d-1} a_i X^{ip} \mod M(X)$$

En terme d'implémentation, nous avons créé la procédure `_zqadic_frobenius_substitution` qui permet de calculer la substitution de Frobenius d'un élément de $\mathbb{Z}_q$. Nous ne donnons ici que la partie qui la calcule dans le cas où le contexte est défini par un modulo de Teichmüller :

```
void _zqadic_frobenius_substitution(zqadic_t rop, zqadic_t op, zqadic_ctx_t
↪  ctx)
{
    zqadic_t cache;

    zqadic_init2(cache, zqadic_prec(op), ctx);

    padic_poly_compose_pow(cache, op, fmpz_get_si(ctx -> p), ctx -> pctx);
    zqadic_reduce(cache, ctx);
    zqadic_set(rop, cache, ctx);

    zqadic_clear(cache);
}
```

Ici `rop` et `op` sont supposés définis à précision égale (une procédure intermédiaire `zqadic_frobenius_substitution` s'occupe de monter tous les paramètres à la précision de `rop` avant d'appeler `_zqadic_frobenius_substitution`), puis `padic_poly_compose_pow` calcule la composition de `op` avec X^p, qui est ensuite réduite modulo M par `zqadic_reduce`.

Son inverse

Grâce à cette forme élégante du Frobenius, il est aussi aisé de calculer son inverse de manière efficace. En clair, on a $\Sigma^{-1}(X) = X^{p^{d-1}}$, et alors en écrivant $a = \sum_{i=0}^{d-1} a_i X^i$ comme

$$a = \sum_{j=0}^{p-1}\left(\sum_{0\leq pk+j<d} a_{pk+j} X^{pk+j}\right) = \sum_{j=0}^{p-1}\left(\sum_{0\leq pk+j<d} a_{pk+j}\Sigma(X^k)\right) X^j$$

on obtiens que

$$\Sigma^{-1}(a) = \sum_{j=0}^{p-1} \left(\sum_{0 \leq pk+j < d} a_{pk+j} X^k \right) C_j(X) \tag{1.5}$$

Avec $C_j(X) = \Sigma^{-1}(X^j) = X^{jp^{d-1}} \mod M$ des polynômes fixes. Ainsi on obtiens une méthode efficace pour calculer $\Sigma^{-1}(a)$, en supposant précalculés les C_j. En pratique, les C_j sont précalculés à l'initialisation d'un contexte `zqadic_ctx_t`, grâce à la méthode `_cj_precomputation` de `ctx_init.c`, et stockés dans le champ `C` d'un contexte. Finalement, on évalue simplement l'inverse de la substitution de Frobenius grâce à la procédure suivante :

```
void zqadic_inv_frobenius_substitution(zqadic_t rop, zqadic_t op,
↪  zqadic_ctx_t ctx)

    {
        slong N = zqadic_prec(rop);
        slong p = fmpz_get_si(ctx -> p);

        zqadic_t op_auxi; // Pour emmener op à la précision de rop.
        zqadic_init2(op_auxi, N, ctx);
        zqadic_set(op_auxi, op, ctx);

        zqadic_t cache1;
        zqadic_t cache2;
        zqadic_t sum;
        zqadic_init2(cache1, N, ctx);
        zqadic_init2(cache2, N, ctx);
        zqadic_init2(sum, N, ctx);

        zqadic_zero(sum);

        for (slong j = 0; j < p; j++)
        {
            zqadic_set(cache1, (ctx -> C)[j], ctx);
            _zqadic_choisi_coeff(cache2, op_auxi, j, ctx);
            zqadic_mul(cache1, cache1, cache2, ctx);
            zqadic_add(sum, sum, cache1, ctx);
        }

        zqadic_set(rop, sum, ctx);
```

```
        zqadic_clear(op_auxi);
        zqadic_clear(cache1);
        zqadic_clear(cache2);
        zqadic_clear(sum);
    }
```

Où l'algorithme ligne 23, `choisi_coeff`, permet de construire le polynôme $\sum_{0 \leq pk+j<d} a_{pk+j} X^k$, connaissant $\sum_{i=0}^{d-1} a_i X^i$, par simple permutation des coefficients.

Analyse de la complexité

On note $T_{d,N}$ le temps de calcul du produit de deux polynômes de degré $\leq d$, dans $\mathbb{Z}/p^N\mathbb{Z}[X]$. Dans notre code, nous utilisons un algorithme qui permet la réduction modulo M en au plus p multiplications. Ainsi le calcul d'une substitution de Frobenius est en $\mathcal{O}(pT_{d,N})$. En outre sachant que les C_j sont précalculés, le calcul de l'inverse de Frobenius ne nécessite que $p-1$ multiplications. On trouve alors également une complexité en $\mathcal{O}(pT_{d,N})$.

Chapitre 2

Hilbert 90 et équation d'Artin-Schreier

2.1 Hlibert 90, formulation calssique et cohomologique

2.1.1 Version multiplicative

Afin de prouver l'existence de solutions pour l'équation d'Artin-Schreier, nous énonçons dans cette section le théorème 90 de Hilbert. Nous commençons par sa forme la plus générale, basée sur une approche cohomologique (détaillée en annexe), et en déduisons une version explicite qui nous servira à prouver l'existence de telles solutions

Fixons K/k une extension galoisienne finie. Comme vu précédemment, nous notons $G = \mathrm{Gal}(K/k)$. Si $GL_n(K)$ est le groupe des matrices à coefficients dans K, on fait agir G sur le groupe linéaire coefficients par coefficients. On peut alors définir le groupe de cohomologie : $H^1(G, GL_n(K))$. Voici le théorème en question. Nous en faisons la démonstration en annexe.

Théorème 2.1.1. *(*90 de Hilbert*).*

$$H^1(G, GL_n(K)) = \{0\}$$

On en déduit alors dans le cas $n = 1$ le corollaire suivant :

Corollaire 2.1.1.

$$H^1(G, K^*) = \{0\}$$

Dans ce paragraphe nous allons voir comment ce résultat cohomologique nous donne la formulation historique du théorème de Hilbert. Nous allons bien sûr à contre courant de la chronologie. Voici ce que dit l'énoncé d'origine :

Théorème 2.1.2. *Soit K/k une extension galoisienne cyclique finie, de groupe de galois $G =< \sigma >$. Soit $\alpha \in K$,*

$$\mathrm{N}(\alpha) = 1 \Longleftrightarrow \exists \beta \in K \ \alpha = \frac{\beta}{\sigma(\beta)}$$

Où N *désigne la norme usuelle* $\mathrm{N}_{K/k}$.

Démonstration. On considère admis les résultats sur le calcul cohomologique au moyen de cochaînes. On se permet d'abréger 1-cocycle (resp. 1-cobord) en cocycle (resp. cobord). L'implication réciproque "$\Leftarrow$" est claire, montrons le sens direct. Si G est de cardinal n, comme il est cyclique, tout cocycle est déterminé par son image sur le générateur σ. Construisons alors un cocycle f tel que $f(\sigma) = \alpha$. Sachant que $\mathrm{N}(\alpha) = 1$, on a :

$$\alpha\sigma(\alpha)\dots\sigma(\alpha)^{n-1} = 1$$

Et donc on peut définir un cocycle par :

$$f(\sigma^i) = \alpha\sigma(\alpha)\dots\sigma(\alpha)^{i-1}$$

qui vérifie bien $f(\sigma) = \alpha$. Or le corollaire au théorème de Hilbert (2.1.1) nous dis que tout cocycle est un cobord, et donc qu'il existe $\gamma \in K^*$ tel que :

$$f(\sigma) = \frac{\sigma(\gamma)}{\gamma} = \alpha$$

Ainsi en posant $\beta = \gamma^{-1}$ on a bien :

$$f(\sigma) = \alpha = \frac{\beta}{\sigma(\beta)}$$

□

2.1.2 Version additive

Le théorème 90 de Hilbert peut aussi s'exprimer dans une forme additive : c'est cette version dont nous auront besoin. On considère alors non pas le G module $(K^*, \times)$, mais bien $(K, +)$. On peut démontrer par un argument d'existence de bases normales (voir [Ser2] paragraphe 1 chapitre X) un théorème d'annulation cohomologique plus fort, en ce sens qu'il est valable en dimension supérieure !

Théorème 2.1.3. *Pour tout $n \in \mathbb{Z}$, on a $H^n(G, K) = 0$.*

En sachant cela, on peut en suivant la même preuve que dans le cas multiplicatif arriver à la proposition souhaitée :

Proposition 2.1.1. *Soit K/k une extension galoisienne cyclique, de groupe de galois $G =< \sigma >$, et un $\alpha \in K$. Alors on a l'équivalence :*

$$\mathrm{Tr}(\alpha) = 0 \Longleftrightarrow \exists\, \beta \in K \ \alpha = \beta - \sigma(\beta)$$

2.2 Equation d'Artin-Schreier

2.2.1 En caractéristique p>0

Formulons rapidement la théorie d'Artin-Schreier en caractéristique positive. On note $\mathbb{F}_q$ un corps fini avec $q = p^n$ pour un certain p premier. On sait que :

$$\mathrm{Gal}(\mathbb{F}_q/\mathbb{F}_p) =< \sigma >$$

Où σ désigne l'endomorphisme de Frobenius usuel. En particulier on peut appliquer le théorème 90 de Hilbert dans sa forme additive (2.1.1) :

$$X^p - X - a = 0 \,\text{admet une solution dans}\, \mathbb{F}_q \iff \mathrm{Tr}(a) = 0$$

On sait même que ce polynôme $P = X^p - X - a$ entretien des liens étroits avec les extensions cycliques du corps sur lequel il est considéré, voir [Lan].

2.2.2 Sur les q-adiques

On construit ici une analogie de l'équation d'Artin-Schreier en caractéristique 0, dans le monde q-adique. Comme précédement on est encore dans le cas d'une extension galoisienne cyclique. En effet on remplace juste $\mathrm{Gal}(\mathbb{F}_q/\mathbb{F}_p)$ par $G := \mathrm{Gal}(\mathbb{Q}_q/\mathbb{Q}_p)$. Un automorphisme générateur de G étant donné par la substitution de Frobenius, on retrouve par le théorème 90 de Hilbert additive (2.1.1) une équation du genre :

$$\begin{aligned} a &= x - \Sigma(x) \\ \text{i.e} \quad 0 &= \Sigma(x) - x + a \end{aligned}$$

On sait que l'on a de bonnes chances de trouver des solutions à cette équation, cependant nous allons la tordre un peu en autorisant des coefficients dans l'anneau $\mathbb{Z}_q$. Enfin, sous certaines conditions sur ces derniers, nous pouvons tout de même expliciter des solutions et les construire algorithmiquement. Nous allons proposer deux méthodes. Pour la première nous n'en donnerons que l'heuristique, tandis que pour la seconde nous en donnerons l'implémentation. On considère ainsi l'équation suivante :

$$\alpha\Sigma(x) + \beta x + \gamma = 0 \tag{2.1}$$

où l'on cherche $x \in \mathbb{Z}_q$ et les coefficients β, γ sont fixés dans $\mathbb{Z}_q$, tandis que α est fixé dans $\mathbb{Z}_q^*$. La première chose à remarquer est que les hypothèses nous permettent de nous ramener à la forme plus pratique

$$\Sigma(x) = ax + c \tag{2.2}$$

en posant $a = -\beta/\alpha$ et $b = -\gamma/\alpha$. On peut alors appliquer successivement Σ à (2.2)pour obtenir :

$$\begin{aligned}
&\Sigma^1(x) = a_1 x + c_1 \\
&\Sigma^2(x) = \Sigma^1(a_1)\ (a_1 x + b_1) + \Sigma^1(b_1) = a_2 x + b_2,\ a_2 = a_1\Sigma(a_1),\ b_2 = \Sigma(a_1)b_1 + \Sigma(b_1) \\
&\qquad \vdots \\
&\Sigma^d(x) = x = a_d x + b_d
\end{aligned}$$

Donc sous réserve que $a_d \neq 1$,

$$x = \frac{b_d}{1 - a_d}$$

Est une solution de l'équation. Avec les formules explicites, cela donne

$$\begin{aligned}
a_d =& a_1\Sigma^1(a_1)...\Sigma^{d-1}(a_1) = \mathrm{N}(a_1) \neq 1 \\
b_d =& \sum_{i=0}^{d-1} \Sigma^i(b_1) \times \prod_{j=i+1}^{d-1} \Sigma^j(a_1)
\end{aligned}$$

Ainsi, un algorithme naturel naît de cette heuristique ; c'est l'algorithme de Lercier-Lubicz. On utilise une approche *square and multiply* pour calculer les a_k et b_k. Cette approche effective prouve donc l'existence et l'unicité d'une solution à l'équation d'Artin-Schreier, mais nous n'utiliserons pas cet algorithme pour la résoudre. Nous présentons dans la section suivante l'algorithme de Harley, qui résout l'équation par une approche différente.

2.2.3 Résolution algorithmique

L'idée est similaire à celle pour calculer le relevé de Teichmüller ; on double la précision de solutions approchées. Imaginons que l'on possède un algorithme qui trouve une solution $x_{N'}$ de (2.1) à la précision $N' = \lceil \frac{N}{2} \rceil$. Cherchons alors une solution de la forme :

$$x_N = x_{N'} + p^{N'}\Delta_N$$

En remplaçant dans (2.1) on obtient :

$$\begin{aligned}
\alpha\Sigma(x_{N'} + p^{N'}\Delta_N) + \beta x_{N'} + \beta p^{N'}\Delta_N + \gamma \equiv 0 \mod p^N \\
p^{N'}\Big(\alpha\Sigma(\Delta_N) + \beta\Delta_N\Big) + \alpha\Sigma(x_{N'}) + \beta x_{N'} + \gamma \equiv 0 \mod p^N
\end{aligned}$$

D'où en divisant par $p^{N'} < p^N$:

$$\alpha\Sigma(\Delta_N) + \beta\Delta_N + \frac{\alpha\Sigma(x_{N'}) + \beta x_{N'} + \gamma}{p^{N'}} \equiv 0 \mod p^{N-N'}$$

On reconnaît une équation d'Artin-Schreier à la précision $N - N' \leq N'$, et donc on peut ré-appliquer l'algorithme dont on a supposé l'existence pour trouver Δ_N. En connaissant $x_{N'}$ et Δ_N on trouve alors une solution x_N et on a doublé notre précision. Il reste maintenant à trouver une solution initiale, on en cherche donc à précision 1. C'est là qu'interviennent les hypothèses $v_p(\beta) > 0$ et $\alpha \in \mathbb{Z}_p^*$, en effet on a alors :

$$\begin{aligned} & \alpha\Sigma(x) + \gamma \equiv 0 \mod p \\ \iff & \alpha x^p + \gamma \equiv 0 \mod p \quad \text{d'après(1.1)} \\ \iff & x \equiv (-\frac{\gamma}{\alpha})^{\frac{1}{p}} \mod p \end{aligned}$$

Remarquons que nous sommes dans un contexte où il est peu cher de calculer une racine p-ieme. En effet, le liens entre la substitution de Frobenius *modulo* p et la puissance p-ieme sur $\mathbb{F}_p$ (1.1) nous donne l'équation suivante calculé grâce à l'inverse de la substitution de Frobenius (1.5) :

$$\left(\sum_{i=0}^{d-1} \overline{a}_i \overline{\theta}^i\right)^{1/p} = \sum_{j=0}^{p-1} \left(\sum_{0 \leq pk+j < d} \overline{a}_{pk+j} \overline{\theta}^k\right) C_j(\overline{\theta})$$

Ainsi le code suivant calcule une solution approchée, obtenue en augmentant la précision d'une solution initiale. Les paramètres sont supposés à précision égale (une procédure intermédiaire `zqadic_artin_schreier_root` s'occupe de tout mettre à la précision de `x`).

```
void _zqadic_artin_schreier_root(zqadic_t x, zqadic_t alpha, zqadic_t beta, zqadic_t gamma,
↪  zqadic_ctx_t ctx)
{
    slong N = padic_poly_prec(x);

    if (N == 1)
    {
        zqadic_t cache;

        zqadic_init2(cache, 1, ctx);

        zqadic_inv(cache, alpha, ctx);
        zqadic_mul(cache, gamma, cache, ctx);
        zqadic_neg(cache, cache, ctx);
        zqadic_inv_frobenius_substitution(cache, cache, ctx);
        zqadic_set(x, cache, ctx);

        zqadic_clear(cache);
    }
```

```
    else
    {
        slong Nr = (N >> 1) + (N & 1); // partie entière supérieure de N

        zqadic_t cache1, cache2; // Varialbes cache
        zqadic_t xr, gammar, Deltar; // x', gamma', \Delta'
        padic_t p, p_pow_Nr, p_pow_mNr; // p, P^{N'}, P^{-N'}

        zqadic_init2(xr, Nr, ctx);
        zqadic_init2(gammar, N - Nr, ctx);
        zqadic_init2(Deltar, N - Nr, ctx);
        zqadic_init2(cache1, N, ctx);
        zqadic_init2(cache2, N, ctx);
        padic_init2(p, N);
        padic_init2(p_pow_mNr, N);
        padic_init2(p_pow_Nr, N);

        padic_set_fmpz(p, ctx -> p, ctx -> pctx);
        padic_pow_si(p_pow_Nr, p, Nr, ctx -> pctx);
        padic_pow_si(p_pow_mNr, p, -Nr, ctx -> pctx);

        _zqadic_artin_schreier_root(xr, alpha, beta, gamma, ctx);
        zqadic_set(cache1, xr, ctx);
        zqadic_frobenius_substitution(cache2, cache1, ctx);
        zqadic_mul(cache2, alpha, cache2, ctx);
        zqadic_mul(cache1, beta, cache1, ctx);
        zqadic_add(cache2, cache2, gamma, ctx);
        padic_poly_scalar_mul_padic(cache2, cache2, p_pow_mNr, ctx -> pctx); // cache2
↪   contient gamma'
        zqadic_set(gammar, cache2, ctx);
        _zqadic_artin_schreier_root(Deltar, alpha, beta, gammar, ctx);
        zqadic_set(cache1, Deltar, ctx);
        padic_poly_scalar_mul_padic(cache1, cache1, p_pow_Nr, ctx -> pctx); // cache1
↪   contient p^{N'}\Delta'
        zqadic_set(cache2, xr, ctx);
        zqadic_add(x, cache1, cache2, ctx);

        zqadic_clear(xr);
        zqadic_clear(gammar);
        zqadic_clear(Deltar);
        zqadic_clear(cache1);
        zqadic_clear(cache2);
        padic_clear(p);
        padic_clear(p_pow_Nr);
        padic_clear(p_pow_mNr);
    }
}
```

2.2.4 Complexité algorithmique et expérimentale

Intéressons nous à la complexité de cet algorithme. On fait deux appels récursifs aux précisions $N' = \lceil \frac{N}{2} \rceil$ et $N - N' \cong N'$. Et dans chaque appel récursif sont réalisées des substitutions de Frobenius (et inverse à l'initialisation), qui coûtent $\mathcal{O}(T_{d,N})$ à p fixé. Donc si p est fixé, on arrive à la majoration :

$$T(N) \leq 2T(\lceil \frac{N}{2} \rceil) + kT_{d,N}$$

Où $T(N)$ représente le temps nécessaire à notre algorithme pour trouver des solutions à la précision N, et k une constante. Ainsi on a une complexité totale de l'ordre :

$$T(N) = \mathcal{O}(T_{d,N} \log N)$$

En pratique, si résout mille fois l'équation, avec $p = 2$ et en faisant varier la précision de la solution et le degré de l'extension, ainsi que le type de représentant, on obtient

TEICH *deg* / *prec*	10	20	30
10	$1.1s$	$1.8s$	$2.5s$
20	$2.6s$	$3.5s$	$4.8s$
30	$3.4s$	$4.6s$	$6.4s$

SPARSE *deg* / *prec*	10	20	30
10	$11.7s$	$14.6s$	$19.2s$
20	$18.5s$	$28.5s$	$40.2s$
30	$25.7s$	$36.8s$	$55.8s$

On observe ainsi un gros gain d'efficacité, selon si le contexte est initialisé par un modulo creux ou un modulo de Teichmüller.

Chapitre 3

Annexes

3.1 Sur Hensel

On commence par un lemme qui forme le rouage de base du relèvement de Hensel. Nous nous sommes appuyé sur l'excellent *Cours d'arithmétique* de Serre, [Ser1].

Lemme 3.1.1. *Soit $f \in \mathbb{Z}_p[X]$, et soit f' sa dérivée. Soient $x \in \mathbb{Z}_p$, $n, k \in \mathbb{Z}$ tels que :* $0 \leq 2k < n$, $f(x) \equiv 0 \mod p^n$, $v_p(f'(x)) = k$.

Alors il existe $y \in \mathbb{Z}_p$ tel que :

$$\begin{array}{ccc} f(y) \equiv 0 \mod p^{n+1} & & \\ v_p(f'(y)) = k & et & y \equiv x \mod p^{n-k} \end{array}$$

Démonstration. Commençons par prendre y de la forme $x + p^{n-k}z$ avec $z \in \mathbb{Z}_p$. D'après la formule de Taylor appliquée à f en x on a :

$$f(y) = f(x) + p^{n-k}zf'(x) + p^{2n-2k}a, \quad \text{pour un} \quad a \in \mathbb{Q}_p$$

Montrons que $a \in \mathbb{Z}_p$. On suppose

$$f(X) = a_m X^m + ... + a_1 X + a0, a_i \in \mathbb{Z}_p, \forall i \in \{1, ..., m\},$$

ainsi

$$a = z^2.\frac{f^{(2)}(x)}{2!} + ... + (p^{n-k})^{m-2}.z^m.\frac{f^{(m)}(x)}{m!}.$$

Pour voir que a est un entier, il suffit de montrer que, pour tout $l \in \{1, ..., m\}$,

$$\frac{f^{(l)}(x)}{l!} \in \mathbb{Z}_p;$$

Par linéarité, on se ramène à montrer, pour tout $j \geq l$, que :

$$\frac{(a_j X^j)^{(l)}}{l!}(x) = a_j \frac{j(j-1)...(j-l+1)}{l!} x^{j-l} \in \mathbb{Z}_p,$$

et on conclut en utilisant le fait que

$$\frac{j(j-1)...(j-l+1)}{l!} = \frac{l!\binom{j}{l}}{l!} \in \mathbb{Z}.$$

D'autre part, il existe $b \in \mathbb{Z}_p$, $c \in \mathbb{U}$ tels que $f(x) = p^n b$ et $f'(x) = p^k c$. Comme c est inversible, en choisissant $z = -bc^{-1} \in \mathbb{Z}_p$, on a : $b + zc = 0$. Par hypothèse, $2n - 2k > n$, donc on a :

$$f(y) = p^n(b + zc) + p^{2n-2k}a \equiv 0 \mod p^{n+1}$$

En appliquant la formule de Taylor à f' en y, on obtient :

$$f'(y) = p^k c + p^{n-k} z f''(y) + p^{2n-2k} a' \qquad \text{avec } a' \in \mathbb{Z}_p$$

(On applique ici le même raisonnement que ci-dessus pour voir que $a' \in \mathbb{Z}_p$.) Comme $n - k > k$, on a bien $v_p(f'(y)) = k$, et donc y convient. □

Une fois qu'on a ce lemme technique, il reste à prouver (1.2.1).

Théorème 3.1.1. *(Lemme de Hensel)*
Soient $f \in \mathbb{Z}_p[X_1...X_m]$, $x \in (\mathbb{Z}_p)^m$, $n, k \in \mathbb{Z}$ et $j \in [\![1, m]\!]$ tels que :

$$0 \leq 2k < n, \quad f(x) \equiv 0 \ [p^n], \quad v_p(\tfrac{\partial f}{\partial X_j}(x)) = k.$$

Alors il existe un zéro y de f dans $(\mathbb{Z}_p)^m$ tel que $x \equiv y \ [p^{n-k}]$.

Démonstration. On va d'abord travailler dans le cas $m = 1$, d'un polynôme à une indéterminée. L'idée est d'appliquer récursivement le lemme précèdent afin de créer, grâce aux dérivées, une suite dont la limite sera notre solution. On reconnaît le schéma de Newton dans le cas réel.
On note $x_0 = x$, (le premier terme de notre suite) et le lemme nous donne x_1 tel que :

$$\begin{aligned} f(x_1) &\equiv 0 \mod p^{n+1} \\ v_p(f'(x_1)) = k \qquad &\text{et} \qquad x_1 \equiv x_0 \mod p^{n-k}. \end{aligned}$$

En remplaçant n par $n+1$ on constate que x_1 satisfait à son tour les hypothèses du lemme précèdent. On construit alors x_2 de même. Ainsi de proche en proche on a construit la suite $(x_n)_n$ telle que pour tout $i \in \mathbb{N}$:

$$(1)\ f(x_i) \equiv 0 \mod p^{n+i} \qquad \text{et} \qquad (2)\ x_{i+1} \equiv x_i \mod p^{n+i-k}.$$

.

La deuxième assertion nous donne $|x_{i+1} - x_i|_p \leq \frac{1}{n+i-k}$, donc que $(x_n)_n$ est de Cauchy dans $\mathbb{Z}_p$ (car $|.|_p$ est ultramétrique), donc converge vers une limite noté y. Par construction on a bien $y \equiv x \mod p^{n-k}$ et par passage à la limite dans (1) on a $f(y) = 0$.

Maintenant si $m > 1$ on va se ramener au cas où $m = 1$. Soit un f comme dans l'énoncer, avec $v_p(\frac{\partial f}{\partial X_j}(x)) = k$ et $x = (x_i)_{1 \leq i \leq m}$. Alors on introduit la fonction polynomiale d'une variable,
$g : a \mapsto f(x_1, ..., x_{j-1}, a, x_{j+1}, ..., x_m)$. Alors appliquer ce que l'on a fait dans le cas m=1 à g et x_j. D'où il existe $y_j \in \mathbb{Z}_p$ tel que $x_j \equiv y_j \, [p^{n-k}]$ et $g(y_j) = 0$. On construit finalement :

$$y = (x_1, ..., x_{j-1}, y_j, x_{j+1}, ..., x_m) \in (\mathbb{Z}_p)^m$$

Ce y répond à la question. □

On a maintenant les cartes en main pour justifier la définition suivante :

Définition 3.1.1. (Polynôme de Teichmüller)
Premièrement on constate que $m \,|\, X^q - X$, car $\mathbb{F}_q = \mathbb{F}_p[X]/(m)$ et $\mathbb{F}_q$ est le corps de décomposition (et de rupture) de $X^q - X$. Le lemme de Hensel assure l'existence et l'unicité d'un polynôme $M \in \mathbb{Z}_p[X]$ vérifiant les conditions suivantes :

- $M \,|\, X^{q-1} - 1$
- $M(X) \equiv m(X) \mod p$

Expliquons rapidement l'existence de M. Il suffit d'écrire :

$$m(X)h(X) = X^{q-1} - 1 \mod p \quad \text{pour } h \in \mathbb{F}_p[X]$$

Car $m \,|\, X^q - X$ et donc $m \,|\, X^{q-1} - 1$ par primalité. Comme $X^{q-1} - 1$ est scindé à racine simple dans $\mathbb{F}_q$, m et h sont premiers entre eux. On utilise le théorème de factorisation de Hensel (1.2.2), pour obtenir :

$$\begin{aligned} X^{q-1} - 1 &= m_2 h_2 \mod p^2 \\ m_2 &= m \mod p \\ h_2 &= h \mod p \end{aligned}$$

Et donc on construit récursivement une suite de polynômes pour $k \in \mathbb{N}$, $m_{2^k} \in \mathbb{Z}_p[X]$ tels que :

$$\begin{aligned} m_1 &= m \\ m_{2^k} &= m_{2^{k-1}} \mod p^{2^k} \end{aligned}$$

En voyant $\mathbb{Z}_p$ comme limite projective, on construit la limite de la suite (m_{2^k}) notée $M \in \mathbb{Z}_p[X]$. On fait de même pour $H \in \mathbb{Z}_p[X]$ la limte de la suite (h_{2^k}). On obtient alors par construction :

$$\begin{aligned} X^{q-1} - 1 &= MH \\ M &= m \mod p \\ H &= h \mod p \end{aligned}$$

3.2 Sur la cohomologie

L'objectif de cette section est de jeter les bases de la cohomologie des groupes en supposant connue les notions de catégorie abélienne, foncteur exact et foncteur additif. Le lecteur intéressé trouvera des réponses à ses question dans le [Wei].

3.2.1 Cohomologie d'un complexe de chaîne

En mathématiques, il n'est pas rare de rencontrer des suites exactes, plus encore, un bon nombre de théorèmes peuvent s'énoncer en terme d'exactitude d'une certaine suite (courte ou non). Ainsi il est naturel de chercher des outils permettant de *mesurer* le défaut d'exactitude d'une suite. La théorie cohomologique présentée ci-dessous vise à répondre à ce problème.

Définition 3.2.1. Complexe de chaîne, pour $\mathcal{C}$ une catégorie abélienne.
Soit $(E_i)_{i\in\mathbb{Z}}$ une famille d'objets de $\mathcal{C}$ et $(d^i : E^i \to E^{i+1})_{i\in\mathbb{Z}}$ une famille de morphismes. On dit alors que $((E^i)_i, (d^i)_i)$ est un *complexe de chaîne* quand pour tout $i \in \mathbb{Z}$, $d^{i+1} \circ d^i = 0$.

$$\ldots \xrightarrow{d^{i-1}} E^i \xrightarrow{d^i} E^{i+1} \xrightarrow{d^{i+1}} E^{i+2} \xrightarrow{d^{i+2}} \ldots$$
$$d^{i+1}\circ d^i = 0$$

On abregera "complexe de chaîne" en "complexe", et on notera (E, d) un complexe, avec $E = (E^i)_i$ et $d = (d^i)_i$. On remarque que l'on peut tout aussi bien considérer des complexes indexés par $\mathbb{N}$, c'est à dire des :

$$E^0 \xrightarrow{d^0} E^1 \xrightarrow{d^1} E^2 \xrightarrow{d^2} E^3 \xrightarrow{d^3} \ldots$$

avec bien sur $i \in \mathbb{N}$, $d^{i+1} \circ d^i = 0$.

Définition 3.2.2. Soit (E, d) un complexe.

- Les d^i sont appelés les *différentielles* du complexe
- Les noyaux $Ker(d^{i+1})$ sont appelés les *cycles* du complexe et notés Z^i (ou $Z^i(E)$ si il y a ambiguïté).
- Les images $Im(d^i)$ sont appelés les *bords* du complexe et notés B^i (ou $B^i(E)$ si il y a ambiguïté).
- Et l'objet qui nous intéresse, le *i-ème groupe de cohomologie* $H^i(E) = Ker(d^{i+1})/Im(d^i) = Z^i/B^i$

Ex 3.2.1. Pour $i \in \mathbb{N}$, $E^i = \mathbb{Z}/8\mathbb{Z}$. On pose $E^{-1} = 0 = \{e\}$. On note $d^i : x \mapsto 4x$ (ou simplement d). On a alors la suite :

$$0 \longrightarrow \mathbb{Z}/8\mathbb{Z} \xrightarrow{d} \mathbb{Z}/8\mathbb{Z} \xrightarrow{d} \mathbb{Z}/8\mathbb{Z} \xrightarrow{d} \dots$$

Or on remarque que $d \circ d = 0$, et que d et un morphisme, donc on a bien défini un complexe de groupe. On voit que pour $i \geq 1$ on a

$$H^i(E) = Ker(d_{i+1})/Im(d_i) = 2E^i/4E^i = \frac{2(\mathbb{Z}/8\mathbb{Z})}{4(\mathbb{Z}/8\mathbb{Z})} = \mathbb{Z}/2\mathbb{Z}$$

3.2.2 La notion de G-module

On introduit ici une nouvelle structure algébrique, les G-modules pour G un groupe. La définition n'a rien de choquant si l'on connaît déja la notion de module sur un anneau :

Définition 3.2.3. *G-module*
Le triplet $(A, +, .)$ est un G-module quand on a :

- $(A, +)$ est un groupe abélien
- $. : (g, x) \mapsto g.x$ est une action de G sur A
- Pour tout $g \in G$, l'application $\phi_g : x \mapsto g.x$ est un automorphisme du groupe abélien A.

Ex 3.2.2. L'exemple qui nous intéresse pour ce travail est finalement un cas très particulier de G-module. Considérons K/k une extension de corps galoisienne, et notons $G = Gal(K/k)$ son groupe de Galois. Alors le groupe $(K, +)$ (ou $(K^*, \times)$) est muni d'une structure de G-module canonique par :

$$\forall \sigma \in G, \forall a \in K,\ \sigma.a = \sigma(a)$$

L'étude cohomologique de ces derniers cas particuliers forme ce que l'on appelle la théorie de la *cohomologie galoisienne*.

Il reste à définir les morphismes de cette catégorie. Dans notre cas il s'agira bien sûr de morphismes de groupes, mais qui respecterons en plus l'action de G :

Définition 3.2.4. *G-morphisme*
Soit A, A' deux G-modules. Une application $f : A \to A'$ est un *morphisme de G-module* (ou un *G-morphisme*), quand c'est un morphisme de groupe entre A et A' et que :

$$\forall (g, x) \in G \times A,\ f(g.x) = g.f(x)$$

Rq 3.2.1. On définit ainsi la catégorie des G-module notée $\mathbf{Mod}_G$, et l'ensemble des G-morphismes de A dans B sera noté $Hom_G(A, B)$. Une vérification rapide montre que c'est une catégorie abélienne.

On construit alors ici le foncteur qui servira de base à la construction de la cohomologie des groupes. Sachant que l'on a une action de G sur notre G-module A, il est naturel de regarder ses invariant, c'est à dire les éléments $a \in A$ tels que $g.a = a$ pour tout $g \in G$.

Théorème 3.2.1. *Soit A un G-module.*
On note A^G l'ensemble $\{a \in A | \forall g \in G\, g.a = a\}$ sous-groupe de A. Alors le foncteur :

$$F : \left\{ \begin{array}{rcl} \boldsymbol{Mod}_G & \to & \boldsymbol{Ab} \\ A & \mapsto & A^G \end{array} \right. .$$

est additif exact à gauche.

Où **Ab** est la catégorie des groupes abéliens, muni des morphismes classiques.

Démonstration. Premièrement il convient de vérifier rapidement que F est un foncteur. Soit A, C, B trois G-modules, et $f : A \to C$ et $g : C \to B$ deux G-morphismes. Le morphisme associé à f par F est :

$$\begin{array}{rcl} F(f) : A^G & \to & C^G \\ a & \mapsto & f(a) \end{array}$$

$F(f)$ est bien à valeur dans C^G car pour $a \in A^G$ on a $g.f(a) = f(g.a) = f(a)$, et c'est bien un morphisme de groupe abélien car f en est un. Il est alors clair que $F(id_A) = Id_{C^G} = Id_{F(A)}$ et que $F(g \circ f) = F(g) \circ F(f)$.

Deuxièmement, montrons que ce foncteur est exact à gauche (il est clairement additif), donnons nous une suite exacte de G-module :

$$0 \longrightarrow A \xrightarrow{f} C \xrightarrow{g} B$$

Comme f est injective, $F(f)$ l'est car c'est la restriction de f à A^G. Montrons que $Ker(F(g)) = Im(F(f))$.

On a la suite d'équivalence :

$$\begin{aligned} c \in Ker(F(g)) &\Longleftrightarrow c \in C^G \ and \ F(g)(c) = 0 \\ &\Longleftrightarrow c \in C^G \ and \ g(c) = 0 \\ &\Longleftrightarrow c \in C^G \ and \ c \in Ker(g) \\ &\Longleftrightarrow c \in C^G \ and \ c \in Im(f) \\ &\Longleftrightarrow c \in C^G \ and \ \exists a \in A, \ f(a) = c \end{aligned}$$

Si on suppose que $c \in C^G$ et l'existence de $a \in A$ tel que $f(a) = c$.Montrons que $a \in A^G$. Soit $g \in G$. Alors

$$g.c = g.f(a) = f(g.a) = c = f(a)$$

Ainsi $f(g.a) = f(a)$ et par injectivité de f, on a bien $a = g.a$, donc $a \in A^G$. Et donc on a $c = f(a)$ avec $a \in A^G$ donc $c \in Im(F(f))$.

Inversement si $c \in Im(F(f))$ alors on a directement l'existence de $a \in A$ tel que $c = f(a)$ et donc on a par ce qui précède $c \in Ker(F(g))$.

Ainsi on a bien $F(f)$ injective, et $Ker(F(g)) = Im(F(f))$, donc on a la suite exacte suivante, et F est bien exact à gauche :

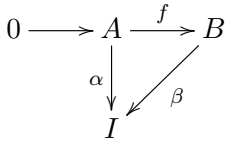
$$0 \longrightarrow F(A) \xrightarrow{F(f)} F(C) \xrightarrow{F(g)} F(B)$$

□

3.2.3 Résolutions injectives, foncteur dérivé droit et cohomologie des groupes

Nous avons présenté quelques pages plus haut la théorie cohomologique d'un complexe en général. La théorie cohomologique des groupes va justement être l'étude cohomologique d'un complexe bien choisi. La construction de ce complexe est fascinante, nous ne détaillerons cependant pas les démonstrations, et passerons sous silence certaines subtilités. Nous nous inspirons entre autres du [Wei].

Définition 3.2.5. Un objet I de $\mathbf{Mod}_G$ est dit *injectif* quand pour tout morphisme injectif $f : A \to B$ et morphisme $\alpha : A \to I$ il existe $\beta : B \to I$ tel que $\alpha = \beta \circ f$, en terme de diagramme commutatif on a :



Intuitivement, cela signifie que l'on peut toujours prolonger correctement les morphismes à valeur dans I. Il vient maintenant une propriété que $\mathbf{Mod}_G$ partage avec la catégorie des modules sur un anneau par exemple. Ces dernières possèdent *suffisamment d'objets injectifs.* En particulier on a la proposition suivante :

Proposition 3.2.1. *Existence de résolutions injectives.*
Soit M un G-module. Alors il existe des G-modules injectifs notés I^i tels que l'on ai une suite exacte :

$$0 \to M \to I^0 \to I^2 \to I^2 \to \dots$$

Une telle suite exacte est appelé résolution injective de M

Passons à la construction du foncteur dérivé droit. On se donne donc M un G-module et des $(I^i)_i$ formant une résolution injective. En outre on se donne F un foncteur exacte à gauche et additif de $\mathbf{Mod}_G$ dans une autre catégorie abélienne. En appliquant F à la résolution injective de M on obtient un complexe de chaîne (**attention**, ce n'est **pas** une suite exacte en général) :

$$0 \to F(M) \to F(I^0) \to F(I^1) \to F(I^2) \to \dots$$

Ce complexe donne alors lieu à des groupes de cohomologie, les $H^n(F(I))$. Le résultat surprenant est le suivant :

Proposition 3.2.2. *Si on se donne deux résolutions injectives d'un G-module M :*

$$0 \to M \to I^0 \to I^1 \to I^2 \to \dots$$

et

$$0 \to M \to E^0 \to E^1 \to E^2 \to \dots$$

alors pour tout $n \in \mathbb{N}$ on a un isomorphisme entre $H^n(F(I))$ et $H^n(F(E))$

On voit donc que ces groupes ne dépendent pas de la résolution injective choisie. On peut donc donner la définition suivante :

Définition 3.2.6. Avec les notations ci-dessus, on appelle n^{ieme} *foncteur dérivé droit* de F à l'objet M le groupe de cohomologie :

$$R^n F(M) = H^n F(I)$$

qui est bien unique à isomorphisme près.

On note M un G-module. Maintenant, il reste à savoir quel foncteur étudier. Cependant nous avons vu au (3.2.1) que le foncteur associé $F : A \mapsto A^G$ est additif, exact à gauche. On a donc toutes les cartes en main pour donner la définition des groupes de cohomologie de G à valeur dans M.

Définition 3.2.7. Cohomologie des groupes
Pour $n \in \mathbb{N}$ on note

$$H^n(G, M) = R^n F(M)$$

le n^{ieme} groupe de cohomologie de G à coefficient dans M, pour le foncteur $F : A \mapsto A^G$.

Bien que cela soit inutile dans le cadre de ce projet, je souhaite présenter ce qui me semble être une des propriétés principales de cette cohomologie. Ce prochain théorème donne finalement la *bonne* propriété que l'on cherchait pour étudier des complexe de chaîne. Le lecteur curieux pourra suivre les idées de Grothendieck, à travers l'étude des δ-*foncteurs* (voir par exemple [Wei]).

Théorème 3.2.2. *Pour toute suite exacte* ***courte*** $0 \to A \to B \to C \to 0$ *de G-module, et tout $n \geq 0$ on a l'existence d'un morphisme naturel :*

$$\delta^n : R^n F(C) \to R^{n+1} F(A)$$

qui induit une suite exacte ***longue*** *:*

$$\ldots \longrightarrow R^nF(A) \longrightarrow R^nF(B) \longrightarrow R^nF(C) \xrightarrow{\delta^n} R^{n+1}F(A) \longrightarrow \ldots$$

3.2.4 Calcul de la cohomologie au moyen de cochaîne

La cohomologie du fait de sa puissance et de sa généralité, est une science qui peut sembler très abstraite. Nous avons construit jusqu'ici les groupe de cohomologie $H^i(G, A)$, au moyen des foncteurs dérivés, mais ce ne sont par les objets les plus faciles à manipuler. On cherche alors un moyen plus pratique de se les représenter. On admet le théorème suivant qui va nous permettre de calculer ces groupes, en les obtenant comme les groupes de cohomologie d'un autre complexe. Pour les démonstrations, nous renvoyons à [Har].

Pour $i \in \mathbb{N}$ on note E_i l'ensemble des $(i+1)$-uplets $(g_0, \ldots, g_i)$ d'éléments de G. On considère alors L_i le $\mathbb{Z}$-module libre de base E_i. On fait agir G sur L_i par translation :

$$s.(g_0, \ldots, g_i) = (sg_0, \ldots, sg_i)$$

Dans la suite, on fixe A un G-module.

Définition 3.2.8. Définition du complexe de cochaîne
On pose pour $i \in \mathbb{N}$:

$$K^i = Hom_G(L_i, A)$$

Et les application $d^i : K^i \to K^{i+1}$ définies par :

$$\begin{aligned} d^i(f)(g_0, \ldots, g_{i+1}) =& g_i f(g_1, \ldots, g_{i+1}) \\ &+ \sum_{j=1}^{i} (-1)^j f(g_0, \ldots, g_{j-1},\, g_j g_{j+1}\,, g_{j+2}, \ldots, g_{i+1}) + (-1)^{i+1} f(g_0, \ldots, g_i) \end{aligned}$$

Rq 3.2.2. On vérifie par le calcul que (K^i, d^i) forme un complexe, c'est à dire que $d^{i+1} \circ d^i = 0$

On en vient alors au théorème qui nous donne un moyen de calculer explicitement les groupes de cohomologie en petite dimension.

Théorème 3.2.3. *Les groupes $H^i(G, A)$ pour $i \geq 1$ s'obtiennent comme les groupes de cohomologie du complexe :*

$$K^0 \xrightarrow{d^0} K^1 \xrightarrow{d^1} K^2 \xrightarrow{d^2} \ldots$$

De prime abord il n'est pas clair de voir que l'on a amélioré les choses. Cependant regardons en petite dimension les définitions que l'on a.

Rq 3.2.3. Ainsi un 1-cocycle peut être vu comme une application $f : G \to A$ telle que :

$$0 = g_1 f(g_2) - f(g_1 g_2) + f(g_1)$$

Un 1-cobord est un 1-cocycle dans l'image de d^0. Donc si f est un cocycle, c'est un cobord quand f vérifie :

$$f(g) = g.a - a \quad \text{pour un } \ a \in A.$$

De même, un 2-cocycle est une application $f : G \times G \to A$ telle que

$$0 = g_1 f(g_2, g_3) - f(g_1 g_2, g_3) + f(g_1, g_2 g_3) - f(g_1, g_2)$$

3.3 Démonstration du théorème 90 de Hilbert

L'objectif est alors de démontrer le théorème (2.1.1), que l'on rappelle :

$$H^1(G, GL_n(K)) = \{0\}$$

Pour y parvenir, énonçons d'abord un résultat général sur l'indépendance des morphismes :

Proposition 3.3.1. *Soit $(M, *)$ un monoïde et K un corps. Alors toute famille de morphismes $(M, *) \to (K^*, \times)$ deux à deux distincts est libre dans le K-espace $\mathcal{F}(M, E)$*

Démonstration. Il suffit de traiter le cas des familles finies. Procédons par récurrence sur le cardinal de la famille $n \in \mathbb{N}^*$. Si $n = 1$ c'est clair, car 0_M est envoyé sur 1_K par tout morphisme.

Si on note $(\phi_i)_{i \leq n+1}$ la famille en question, et si on prend $\lambda_1, \ldots, \lambda_{n+1} \in K$ tels que :

$$\sum_{i=1}^{n+1} \lambda_i \phi_i = 0$$

Soit $x \in M$, alors pour tout $y \in M$ on a :

$$0 = \sum_{i=1}^{n+1} \lambda_i \phi_i(x * y) \tag{3.1}$$

$$= \sum_{i=1}^{n+1} \lambda_i \phi_i(x) \phi_i(y) \tag{3.2}$$

et

$$0 = \sum_{i=1}^{n+1} \lambda_i \phi_i(y) \tag{3.3}$$

Ainsi en faisant l'opération (3.2)$-\phi_{n+1}(x)\times$(3.3), on élimine le dernier terme, on obtient donc :

$$\forall y \in M \quad \sum_{i=1}^{n} \lambda_i \phi_i(y) \left(\phi_i(x) - \phi_{n+1}(x)\right) = 0$$

Or par hypothèse de récurrence les $\phi_1, \ldots, \phi_n$ sont libres, on a donc pour $i \in [\![1, n]\!]$:

$$\lambda_i \left(\phi_i(x) - \phi_{n+1}(x)\right) = 0$$

Ceci étant vrai pour tout $x \in K$, et les ϕ_i étant deux à deux distincts on a pour tout $i \in [\![1, n]\!],\ \lambda_i = 0$. Ainsi il reste $\lambda_{n+1}\phi_{n+1} = 0$ qui implique $\lambda_{n+1} = 0$ comme dans le cas $n = 1$. □

Passons à la démonstration du théorème 90 de Hilbert.

Démonstration. (On considère admis les résultats sur le calcul cohomologique au moyen de cochaînes, *cf* l'annexe (3.2.4)). Soit $a : s \mapsto a_s$ un 1-cocycle, montrons que c'est également un cobord, c'est à dire qu'il existe $b \in GL_n(K)$ tel que :

$$\forall s \in G \quad a_s = s(b)b^{-1}$$

Soit $c \in M_n(K)$ quelconque, on pose : $b_0 = \sum_{t\in G} a_t t(c)$. On a pour tous $s \in G$:

$$\begin{aligned} s(b_0) =& \sum_{t\in G} s(a_t)st(c) \\ =& \sum_{t\in G} a_s^{-1}a_{st}st(c) \qquad \text{car } a \text{ est un 1-cocycle} \\ =& a_s^{-1}b_0 \end{aligned}$$

Il suffit donc de bien choisir la matrice c, pour faire de b_0 un élément de $GL_n(K)$, pour l'inverser. Soit $x \in K^n$ fixé.

Notons $b(x) = \sum_{s\in G} a_s(s(x))$. Montrons que les $b(x)$ engendre K^n, par dualité. Si u est une forme linéaire nulle sur tous les $b(x)$, alors :

$$\forall k \in K \quad 0 = u(b(kx)) = \sum_{s\in G} a_s u(s(k)s(x)) = \sum_{s\in G} s(k)a_s u(s(x))$$

Ceci nous donne une relation linéaire sur les $s \in G$, en considérant la variable k. Or par la proposition précédente, ces derniers forment une famille libre, ainsi les $a_s u(s(x))$ sont tous nuls. Or les a_s sont inversibles par définition donc on a $\forall x \in K^n\ u(s(x)) = 0$, et donc $u = 0$.

On peut alors considérer $x_1, \ldots, x_n$ dans K^n tels que les $b(x_i)$ soient K-indépendants. On prend alors pour c la matrice de l'application qui envoie la base canonique (e_i) de K^n sur la famille (x_i). Dans ce cas on a pour $i \in [\![1, n]\!]$:

$$\begin{aligned} b_0 e_i =& \sum_{t \in G} a_t t(c) e_i = \sum_{t \in G} a_t t(ce_i) \quad \text{car } e_i \text{ est à coefficients dans } k, \text{ donc invariant par } G \\ =& \sum_{t \in G} a_t t(x_i) \\ =& b(x_i) = y_i \end{aligned}$$

Ainsi b_0 envoie une base sur une base, elle est donc inversible, ce qu'il fallait démontrer. □

Bibliographie

[CFA] Henri Cohen, Gerhard Frey, Roberto M. Avanzi. *Handbook of elliptic and hyperelliptic curve cryptography*. Taylor and Francis, 2006.

[Har] David Harari. *Cohomologie galoisienne et théorie du corps de classes*. CNRS, 2017.

[Lan] Serge Lang. *Algebra*. Springer, New York, NY, 2002.

[Ser1] Jean-Pierre Serre. *Cours d'arithmétique*. Puf, 1994.

[Ser2] Jean-Pierre Serre. *Corps locaux*. Hermann, 1997.

[Wei] Charles A. Weibel. *An Introduction to Homological Algebra*. Cambridge Studies in Advanced Mathematics. Cambridge University Press, 1994.