Rapport de stage de fin de première année : exemples de groupes, leur traitement par MAGMA, et applications en cryptographie

Encadré par Guénaël Renault

Tristan Vaccon

juin 2009-juillet 2009

Table des matières

Préambule

Mon stage a eu lieu du 09/06 au 23/06 et du 06/07 au 17/07, parmi l'équipe SALSA (Solvers for ALgebraic Systems and Applications), au Laboratoire d'Informatique de Paris 6, encadré par Guénaël Renault. J'avais un poste de travail dans une salle réservée aux stagiaires. La première semaine de mon stage a constitué en une découverte de la théorie de Galois par le livre de J.P. Escofier [1], la seconde en un apprentissage de MAGMA, et les deux dernières en la découverte de quelques notions de cryptographie, de théorie des courbes elliptiques, et la mise en place d'un cryptosystème utilisant des courbes elliptiques [2], en MAGMA. Le but de ce stage était, outre la découverte de la théorie de Galois et de celle des courbes elliptiques, de voir comment ceux-ci, et en particulier les groupes qui intéressent ces théories, peuvent être traités par un logiciel de calcul formel, MAGMA, et appliqués en cryptographie.

Introduction

La première structure intéressante en algèbre est la structure de groupe. Nous allons étudier deux familles de groupes importantes en mathématiques et voir comment un logiciel de calcul formel peut les traiter : d'abord ceux de la théorie de Galois, puis un groupe plus "géométrique", celui des courbes elliptiques. Enfin, en dernière partie, on verra l'intérêt de ces derniers en cryptographie par l'étude d'un cryptosystème fondé sur les couplages dans les courbes elliptiques.

1 Présentation de MAGMA, éléments de théorie de Galois, et leur traitement par MAGMA

1.1 MAGMA

1.1.1 Première approche de MAGMA, types

MAGMA est un logiciel de calcul formel développé par l'université de Sydney, depuis 1993. Il tire son nom de la structure la plus simple de l'algèbre, celle de magma (un ensemble avec une loi de composition interne). Sa puissance permet de réaliser une très grande partie des opérations de l'algèbre, et la syntaxe de son langage de programmation permet de plus de le faire assez facilement.

MAGMA est un langage fortement typé : chaque objet dans MAGMA est d'un certain type ; mathématiquement, c'est en fait un ensemble dans lequel il vit. Une fonction, Parent, permet de donner l'ensemble dans lequel MAGMA considère qu'un élément vit.

Par exemple, 42 est naturellement considéré comme un entier naturel.

```
>a:=42;
>Parent(a);
Integer Ring
```

C'est cependant aussi un réel. Pour être considéré comme tel, il faut effectuer une coertion : on écrit A !b l'image de b par l'inclusion canonique (si elle existe) de Parent(b) dans A, et A !b est donc un élément de A.

```
>a:=RealField()!42;
>Parent(a);
Real Field of precision 30
```

1.1.2 le type set

On a parlé plus haut d'ensembles, mais MAGMA possède un type set, qui correspond aux ensembles mathématiques. Non seulement les opérations usuelles sur les ensembles sont possibles avec les set de MAGMA, mais l'on peut définir un set par une formule mathématiquement naturelle de la forme $\{x : x \in A \, et \, Pred(x)\}$, où A est un ensemble.

Ainsi, pour calculer $\sum_{x \in A} x$ avec $A = \{p/ \, p \, premier \, et \, 1 \leq p \leq 10\}$, on écrit simplement :

```
>&*{x : x in [1..10] | IsPrime(x)};
210
```

1.1.3 Puissance de MAGMA sur les groupes, et plus généralement en algèbre

Maintenant, on peut voir comment MAGMA travaille sur les groupes : on peut considérer le célèbre S_4 et dresser la liste de ses sous-groupes distingués.

```
> S4:=SymmetricGroup(4);
> S4;
Symmetric group S4 acting on a set of cardinality 4
Order = 24 = 2^3 * 3
> NormalSubgroups(S4);
Conjugacy classes of subgroups
------------------------------

[1]     Order 1            Length 1
        Permutation group acting on a set of cardinality 4
```

```
        Order = 1
[2]     Order 4            Length 1
        Permutation group acting on a set of cardinality 4
        Order = 4 = 2^2
            (1, 4)(2, 3)
            (1, 3)(2, 4)
[3]     Order 12           Length 1
        Permutation group acting on a set of cardinality 4
        Order = 12 = 2^2 * 3
            (2, 3, 4)
            (1, 4)(2, 3)
            (1, 3)(2, 4)
[4]     Order 24           Length 1
        Permutation group acting on a set of cardinality 4
        Order = 24 = 2^3 * 3
            (3, 4)
            (2, 3, 4)
            (1, 4)(2, 3)
            (1, 3)(2, 4)
```

Mais plus impressionnant : on voit qu'on peut, étant donné un groupe et un de ses sous-groupes distingués (ici S_5 et A_5), calculer leur groupe quotient.

```
> S5:=SymmetricGroup(5);
> A5:=AlternatingGroup(5);
> A5;
Permutation group A5 acting on a set of cardinality 5
Order = 60 = 2^2 * 3 * 5
    (3, 4, 5)
    (1, 2, 3)
> S5/A5;
Permutation group acting on a set of cardinality 2
    Id($)
    (1, 2)
```

En fait, les notions, fondamentales en algèbre, de quotient par des sous-groupe distingués ou des idéaux sont possibles en MAGMA.

Par exemple, voici une construction en MAGMA de $\mathbb{C} \simeq \mathbb{R}[Z]/(Z^2+1)$.

```
> K;
Univariate Polynomial Ring in X over Rational Field
> I:=ideal<K|X^2+1>;
```

```
> I;
Ideal of Univariate Polynomial Ring in X over Rational Field generated by X^2 +
    1
> A<Z>:=K/I;
> A;
Univariate Quotient Polynomial Algebra in Z over Rational Field
with modulus Z^2 + 1
> Z*Z;
-1
```

1.1.4 Comment je me suis initié à MAGMA

Pour m'initier à MAGMA, j'ai écrit quelques programmes, plus ou moins classiques, que l'on pourra trouver dans le même ordre dans l'annexe, comme les `Algorithm` 1 à 4. Voici donc quelques exemples de programmes classiques que j'ai écrit en MAGMA :

- Un algorithme d'exponentiation rapide (qui montre au passage les capacités de programmation récursive de MAGMA) ;
- l'algorithme de multiplication rapide de matrices de Strassen (qui montre que MAGMA permet aussi de manipuler des matrices avec aisance) ;
- un algorithme Hensel's lift : la remontée de Hensel ;

Ce dernier consiste à prendre une factorisation d'un polynôme (pris au départ à coefficient dans $\mathbb{Z}$) obtenu dans un $\mathbb{Z}/p\mathbb{Z}$ et de le remonter, pour un k demandé, dans un $\mathbb{Z}/p^k\mathbb{Z}$. La factorisation initiale pourrait s'obtenir par un algorithme utilisant des réseaux que je n'ai pas essayé de comprendre et d'implémenter, et ensuite, cet algorithme peut s'inscrire dans une méthode de factorisation des polynômes de $\mathbb{Z}[X]$

1.2 éléments de théorie de Galois, corps finis

La première partie de mon stage m'a amené à m'initier à la théorie de Galois. Si j'ai pu considérer certains problèmes importants de la théorie de Galois et de la théorie de Galois effective, comme celui du calcul effectif du groupe de Galois associé à un polynôme, de l'éventualité d'un polynôme associé à un groupe donné, etc ... ce qui suit, fortement inspiré de [1], va permettre une rapide présentation de cette théorie et de ses objets, jusqu'à voir que le problème des extensions finies de corps finis est reglé.

1.2.1 première approche des fondements de la théorie de Galois

Présentons tout d'abord les objets fondamentaux de la théorie de Galois, jusqu'à pouvoir définir la notion de groupe de Galois.

Définition 1. Soit $\mathbb{K}$ un corps et a un nombre algébrique sur K, de polynôme minimal P sur $\mathbb{K}$. Toute racine de P dans une cloture algébrique de $\mathbb{K}$ est appelée nombre conjugué de a sur $\mathbb{K}$. Deux racines d'un polynôme irréductible de $\mathbb{K}[X]$ sont dites conjuguées sur $\mathbb{K}$.

Les morphismes que l'on considérera seront maintenant tous des morphismes d'anneaux unitaires.

Théorème 1. *Soient $\mathbb{K}$ un sous-corps de $\mathbb{C}$, a un élément algébrique de degré n et de polynôme minimal P sur $\mathbb{K}$.*

- *Alors si L est une extension de $\mathbb{K}$, si $a \in L$ et si $\sigma : L \to \mathbb{C}$ est un $\mathbb{K}$-morphisme, alors $\sigma(a)$ est un conjugué de a sur $\mathbb{K}$.*
- *Si L contient toutes les racines d'un polynôme S de $\mathbb{K}[X]$, σ induit une permutation de l'ensemble des racines de S.*
- *Si b est un conjugué de a sur $\mathbb{K}$, il existe un $\mathbb{K}$-isomorphisme unique $\sigma : \mathbb{K}[a] \to \mathbb{K}[b]$ tel que $\sigma(a) = b$.*
- *Le nombre de $\mathbb{K}$-morphismes distincts de $\mathbb{K}[a]$ dans $\mathbb{C}$ est n.*

Démonstration. On a $P(\sigma(a)) = \sigma(P(a)) = 0$, d'où $\sigma(a)$ est un conjugué de a. En tant que morphisme d'anneaux unitaires entre corps, σ est injectif, ce qui donne le résultat pour S en le décomposant en produit de facteurs irréductibles.

Si Φ et Ψ sont les isomorphismes canoniques de $\frac{\mathbb{K}[X]}{(P)}$ dans, respectivement, $\mathbb{K}[a]$ et $\mathbb{K}[b]$, alors $\sigma = \Psi\Phi^{-1}$ convient. Si σ et σ' conviennent, alors soit f la surjection canonique de $\mathbb{K}[X]$ dans $\mathbb{K}[a]$. On σf et $\sigma' f$ qui sont l'identité sur $\mathbb{K}$, et tel que $\sigma f(X) = b = \sigma' f(X)$, donc $\mathbb{K} = \sigma' f$, et f étant surjective, on en déduit l'unicité.

P étant irréductible de degré n, a a, lui-même compris, n conjugués, ce qui précéde montre alors le résultat. □

Définition 2. Si P est un polynôme de degré n de $\mathbb{K}[X]$ et $x_1, ..., x_n$ sont ses racines, non nécessairement distinctes, dans une cloture algébrique de $\mathbb{K}$, alors le corps $\mathbb{K}[x_1, ..., x_n]$ est le corps de décomposition de P sur $\mathbb{K}$.

Définition 3. On appelle extension normale (ou encore quasi-galoisienne) d'un corps $\mathbb{K}$ une extension algébrique N de $\mathbb{K}$ tel que les conjugués des éléments de N soient dans N (autrement dit, tel que tout polynôme irréductible de $\mathbb{K}[X]$ ayant une racine dans N les ait toutes).

1.2.2 Groupe de Galois

On peut maintenant définir ce qu'est un group de Galois, et énoncer le théorème de correspondance de Galois.

Définition 4. Soient $\mathbb{K}$ un corps, L une extension de $\mathbb{K}$. Le groupe des $\mathbb{K}$-automorphismes de L est appelé groupe de Galois de l'extension L sur $\mathbb{K}$, et sera noté $Gal(L|\mathbb{K})$.

Définition 5. Soient $\mathbb{K}$ un corps, L une extension de $\mathbb{K}$ et H un sous-groupe de $Gal(L|\mathbb{K})$. Alors l'ensemble $I(H)$ des éléments de L invariants par H (id est l'ensemble des x tels que $\forall \sigma \in H, \sigma(x) = x$) est un sous-corps de L, appelé le corps des invariants de H.

Théorème 2 (Théorème de correspondance de Galois)**.** *Soient $\mathbb{K}$ un corps, N une extension normale de degré fini de $\mathbb{K}$, E l'ensemble des extensions intermédiaires entre $\mathbb{K}$ et N, G l'ensemble des sous-groupes de $Gal(N|\mathbb{K})$. Soient $I : G \to E$ qui à $H \in G$ associe $I(H)$ et $J : E \to G$ qui à $L \in E$ associe $Gal(L|\mathbb{K})$.*

Alors I et J définissent des bijections réciproques, décroissantes pour l'inclusion, et leurs restrictions $I' : G' \to E'$ et $J' : E' \to G'$, où E' est l'ensemble des extensions normales de $\mathbb{K}$ contenues dans N et G' est l'ensemble des sous-groupes distingués de $Gal(N|\mathbb{K})$, sont bien définies et sont bijections réciproques.

Si L et L' sont dans E, alors L' est une extension normale de L si et seulement si $Gal(N|L')$ est un sous-groupe distingué de $Gal(N|L)$, et alors $Gal(L'|L) = \frac{Gal(N|L)}{Gal(N|L')}$. Par ailleurs, si on a seulement $L \subset L'$, alors $[L' : L] = \frac{\sharp Gal(N|L)}{\sharp Gal(N|L')}$.

Exemple. On a $\mathbb{C} \simeq \frac{\mathbb{R}[Z]}{Z^2+1}$, on peut en déduire que $Gal(\mathbb{C}|\mathbb{R})$ est le groupe a deux éléments, et est plus précisément composé de l'identité et de la conjugaison (qui envoie i sur $-i$).

Proposition 1. *Soient n un entier, $2 \leq n$ et ζ une racine primitive n-ième de l'unité dans $\mathbb{C}$. Alors $\mathbb{Q}[\zeta]$ est une extension normale de $\mathbb{Q}$, $[\mathbb{Q}[\zeta] : \mathbb{Q}] = \phi(n)$ (indicatrice d'Euler), et $Gal(\mathbb{Q}[\zeta]|\mathbb{Q}) \simeq U(n)$ (groupe des racines n-ième de l'unité).*

1.2.3 Cas des extensions finies de corps finis

Maintenant qu'on a vu ce que pouvait donner la théorie de Galois pour des corps quelconque, on va regarder plus particulièrement les corps finis, et voir que l'on peut régler le cas des extensions finies de corps finis.

Définition 6. Soit K un corps de cardinal $q \in \mathbb{N}$, de caractéristique p. On appelle homomorphisme de Froebenius, noté F ou F_q et défini de K dans K par $F_q(x) = x^q$.

Remarque. Comme K un corps de cardinal $q \in \mathbb{N}$ et de caractéristique p, on a $q = p^r$ ($r \in \mathbb{N}^*$), et dans K, $(a+b)^p = a^p + b^p$, donc $(a+b)^q = a^q + b^q$, ce qui permet facilement de voir que F_q est bien un homomorphisme de corps.

On rappelle le théorème suivant :

Théorème 3. *Le groupe multiplicatif d'un corps fini K, de cardinal q, est cyclique d'ordre $q-1$.*

Il va nous permettre de montrer ce dernier théorème, donnant le groupe de Galois d'une extension finie d'un corps fini :

Théorème 4. *Soient K et L deux corps finis, $K \subset L$, de caractéristique p, K de cardinal $q = p^r$, L de cardinal q^s (nécessairement une puissance de q du fait de sa structure de K-espace vectoriel), $r, s \geq 1$, alors $Gal(L|K)$ est cyclique d'ordre s, engendré par F_q.*

Démonstration. F est un K-automorphisme de L, et par composition, ses puissances aussi. Il existe un élément x d'ordre $q^s - 1$ d'après le théorème précédent, donc $\forall k, 1 \leq k \leq s-1, x^{q^k} \neq 1$, donc $\forall k, 1 \leq k \leq s-1, F^k \neq Id(L)$, et $F^s = Id(L)$. De plus, les $F^i(x)$ sont distinctes (x engendre L^*), donc les F^i sont bien distincts, et forment s K-automorphismes de L. Comme $[L : k] = s$, x ne peut avoir d'autres conjugués que les $F^i(x)$, donc il n'existe pas d'autres K-automorphismes de L. On en déduit finalement le résultat. □

1.3 théorie de Galois avec MAGMA, corps finis en MAGMA

1.3.1 calcul pour la théorie de Galois

MAGMA est capable d'effectuer beaucoup de calculs pour la théorie de Galois :

– Le groupe de Galois de polynômes :

```
> B<W>:=PolynomialRing(RationalField());
> GaloisGroup(W^5+W^2+7);
Symmetric group S5 acting on a set of cardinality 5
Order = 120 = 2^3 * 3 * 5
    (1, 2, 3, 4, 5)
    (1, 2)
[ -9077329*$.1^5 + 7860960*$.1^4 + 5875063*$.1^3 + 6505581*$.1^2 +
4246756*$.1 + 5824792 + O(29^5), 2741830*$.1^5 + 9157920*$.1^4 +
5730188*$.1^3 - 8187114*$.1^2 + 2042132*$.1 + 8002579 + O(29^5),
6335499*$.1^5 + 3492269*$.1^4 + 8905898*$.1^3 + 1681533*$.1^2 -
6288888*$.1 - 6248050 + O(29^5), -1655143*$.1^5 - 1888793*$.1^4 -
6603330*$.1^3 + 6311579*$.1^2 - 4136205*$.1 - 2989598 + O(29^5),
```

```
1655143*$.1^5 + 1888793*$.1^4 + 6603330*$.1^3 - 6311579*$.1^2 +
4136205*$.1 - 4589723 + O(29^5) ]
GaloisData over Z_29
```

– Le calcul de polynômes minimaux

```
> Q := RationalField();
> A<x, y> := AffineAlgebra<Q, x, y | x^2 - 2, y^3 - 5>;
> UP<z> := PolynomialRing(Q);
> MinimalPolynomial(x + y);
z^6 - 6*z^4 - 10*z^3 + 12*z^2 - 60*z + 17
```

– Le calcul de degrés d'extension sur $\mathbb{Q}$.

1.3.2 représentation des corps finis en MAGMA

$\mathbb{F}_{p^k}$ (p premier, k entier) peut être représenté de plusieurs façons, avec diverses efficacités pour les calculs.

De base, les éléments non nuls de $\mathbb{F}_{p^k}$ sont représentés comme puissances d'un élément générateur de son groupe multiplicatif :

```
> Set(GF(3^4));
{ 1, $.1, $.1^2, $.1^3, $.1^4, $.1^5, $.1^6, $.1^7, $.1^8, $.1^9, $.1^10,
$.1^11, $.1^12, $.1^13, $.1^14, $.1^15, $.1^16, $.1^17, $.1^18, $.1^19, $.1^20,
$.1^21, $.1^22, $.1^23, $.1^24, $.1^25, $.1^26, $.1^27, $.1^28, $.1^29, $.1^30,
$.1^31, $.1^32, $.1^33, $.1^34, $.1^35, $.1^36, $.1^37, $.1^38, $.1^39, 2,
$.1^41, $.1^42, $.1^43, $.1^44, $.1^45, $.1^46, $.1^47, $.1^48, $.1^49, $.1^50,
$.1^51, $.1^52, $.1^53, $.1^54, $.1^55, $.1^56, $.1^57, $.1^58, $.1^59, $.1^60,
$.1^61, $.1^62, $.1^63, $.1^64, $.1^65, $.1^66, $.1^67, $.1^68, $.1^69, $.1^70,
$.1^71, $.1^72, $.1^73, $.1^74, $.1^75, $.1^76, $.1^77, $.1^78, $.1^79, 0 }
```

Cette représentation rend les multiplications aisées.

Une autre représentation utilise cette fois-ci un élément parmi les générateurs du groupe multiplicatif de $\mathbb{F}_{p^k}$ qui est dit normal :

Définition 7. Un élément $\alpha \in \mathbb{F}_{q^k}$ est dit normal sur $\mathbb{F}_q$ si $(\alpha, \alpha^q, ..., \alpha^{q^{k-1}})$ est une $\mathbb{F}_q$-base de $\mathbb{F}_{q^k}$. Celle-ci est appelée une base normale de $\mathbb{F}_{q^k}$ sur $\mathbb{F}_q$.

Un avantage d'une telle représentation est le fait que l'homomorphisme de Froebenius consiste alors simplement à un décalage cyclique des coordonnées dans cette base.

Il y a des algorithmes pour calculer un tel élément normal (comme un théorème dit qu'il en existe toujours un), par exemple le cinquième `Algorithm` de l'annexe.

```
> eltnormal(3,4);
$.1^7
```

Il utilise la propriété suivante : α est normal si et seulement si $\gcd(X^k - 1, \alpha X^{k-1} + \Phi_q(\alpha)X^{k-2} + ... + \Phi_q^{k-2}(\alpha)X + \Phi_q^{k-1}(\alpha)) = 1$ [3].

2 Courbes elliptiques

2.1 un groupe "géométrique"

2.1.1 courbes elliptiques

Définition 8. Une courbe elliptique $E(\mathbb{K})$ (E s'il n'y a pas d'ambiguïtés) sur un corps $\mathbb{K}$ est l'ensemble des solutions (X, Y) dans $\mathbb{K}^2$ à une équation de la forme $Y^2 = X^3 + AX + B$, avec $(A, B) \in \mathbb{K}^2$, tel que $4A^3 + 27B^2 \neq 0$.

Remarque. Cette dernière condition correspond au fait que la courbe soit non-singulière (par exemple, n'ait pas de point double).

Remarque. On notera parfois EC pour Elliptic Curve et courbe elliptique

Les figures 1 à 3 qui suivent sont des exemples de courbes elliptiques, ou de courbes singulières.

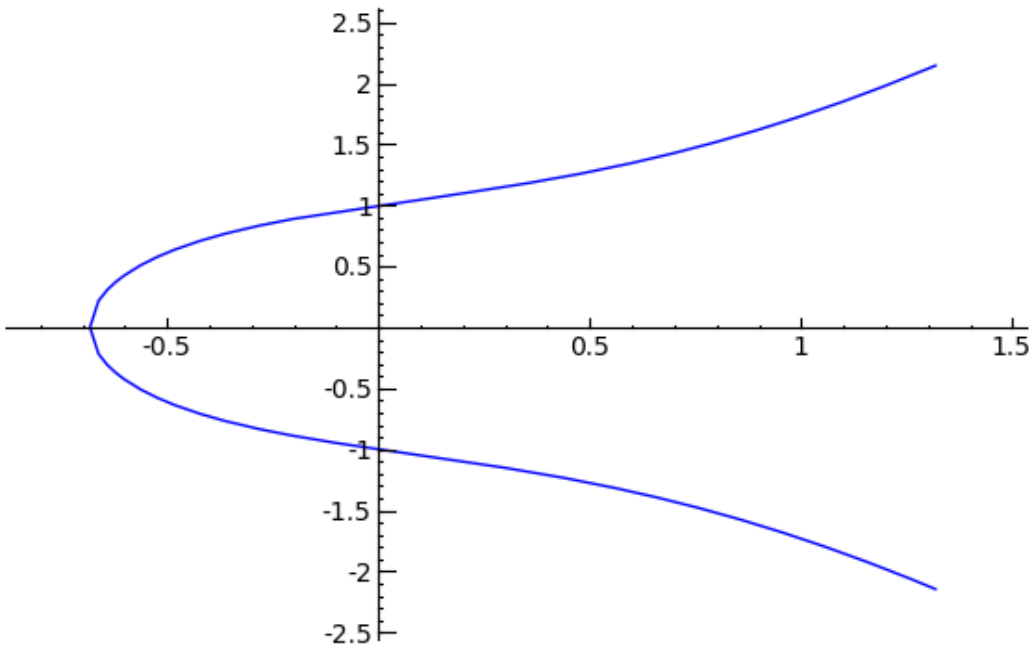

FIGURE 1 – courbe elliptique avec $A = 1$ et $B = 1$

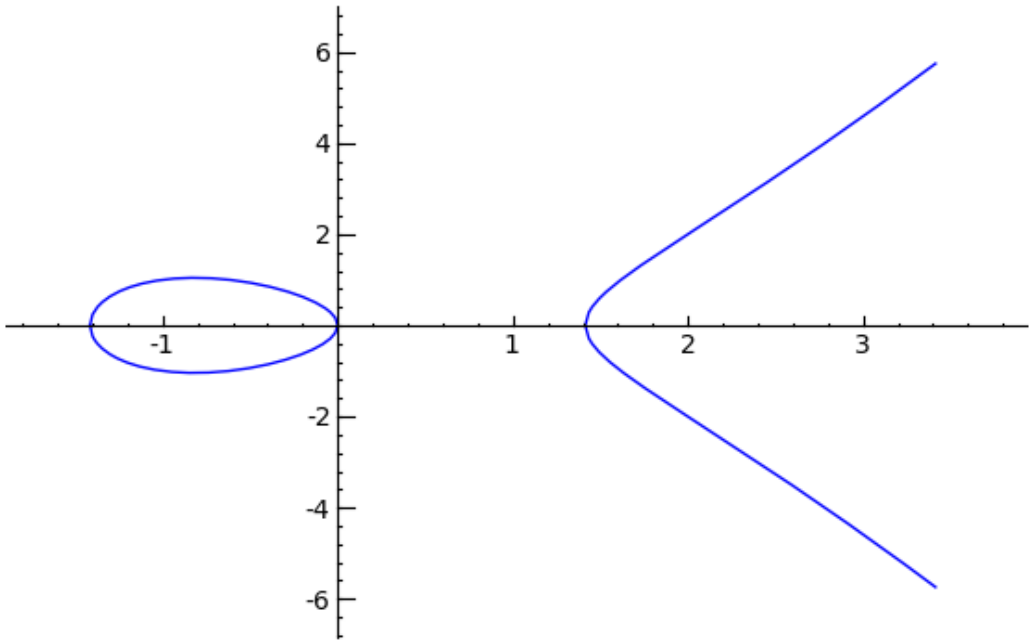

FIGURE 2 – courbe elliptique avec $A = -2$ et $B = 0$

2.1.2 un groupe sur une courbe ?

Aussi étonnant que cela puisse paraitre de prime abord, on peut munir les points d'une courbe elliptique E sur $\mathbb{R}$ d'une loi de groupe :

Définition 9. Soit $(P, Q) \in E^2$. Soit Δ la droite liant P et Q, ou la tangente à la courbe en P si $P = Q$, alors soit Δ coupe E en un unique point R distinct de P et Q, et on pose $P + Q = R'$ où R' est le symétrique de R par rapport à l'axe des abscisses, soit Δ est verticale ne coupant E qu'en P, et dans ce cas, on considère un point "à l'infini", qu'on considérera maintenant comme élément de E, qui sera présent sur tout les Δ verticales, qui jouera le rôle de zéro, et ici, $P + Q = 0$.

Remarque. On utilisera indifférement les notations 0 et ∞ pour ce point à l'infini.

Les figures 4 à 6 illustrent cette définition.

Proposition 2. *Ce qui précéde définit une loi de groupe, commutative, sur E.*

Démonstration. Si le fait que cette "addition" définisse bien une loi de composition interne, que tout élément ait un inverse (son symétrique par rappport à l'axe des abscisses), qu'il y ait bien un élément neutre (ce zéro "à l'infini"), et que cette loi soit commutative sont assez naturels, l'associativité n'est pas triviale. Par

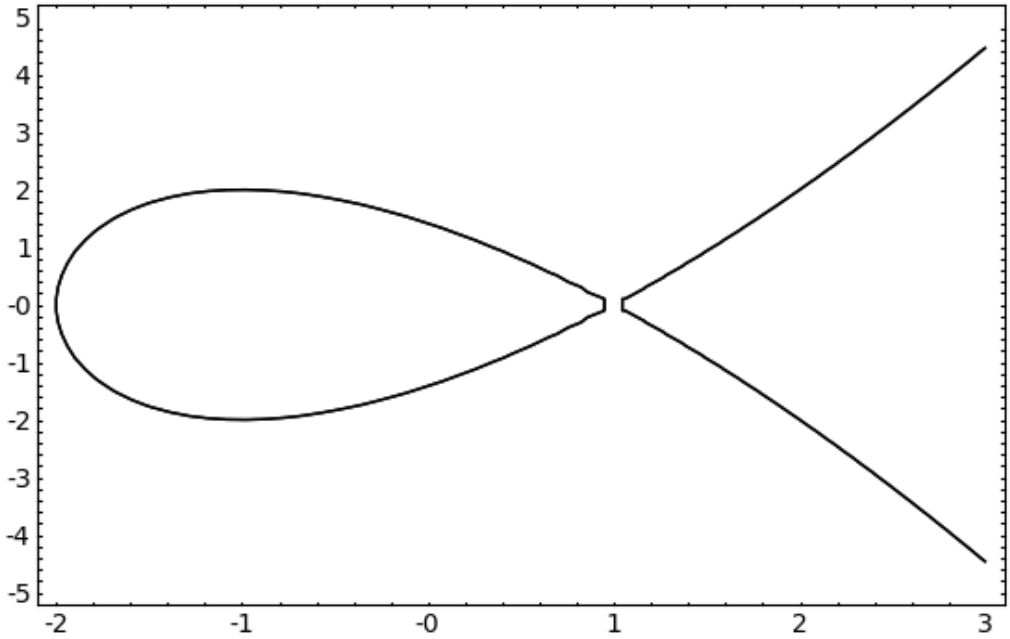

FIGURE 3 – Avec $A = -3$ et $B = 2$, on n'a pas une courbe elliptique (du fait du point double)

quelques laborieux calculs (écriture de l'équation de Δ et recherche des points d'intersections entre Δ et E, en connaissant déjà deux), on peut explicitement exprimer les coordonnées de $P+Q$ en fonction de celles de P et Q, et ensuite voir qu'effectivement, en utilisant ces formules, la loi est bien associative. Cependant, ces calculs sont trop long (et foncièrement, sans grand intérêt) pour figurer ici (au point que je ne les ai même jamais vus explicités dans les livres de référence que j'ai pu consulter sur les courbes elliptiques). Ces formules sont tout de même données ci-après. □

Proposition 3. *Soit $(P_1, P_2) \in E^2$, alors on peut calculer $P_1 + P_2$ par l'algorithme suivant :*

1 *Soit $P_1 = 0$, et alors $P_1 + P_2 = P_2$;*
2 *Soit $P_2 = 0$, et alors $P_1 + P_2 = P_1$;*
3 *Sinon, $P_1 = (x_1, y_1)$ et $P_2 = (x_2, y_2)$;*
4 *Soit $x_1 = x_2$ et $y_1 = -y_2$, et alors $P_1 = -P_2$ et $P_1 + P_2 = 0$;*
5 *Sinon, si $P_1 \neq P_2$, soit $\lambda = \frac{y_2 - y_1}{x_2 - x_1}$ et soit $\lambda = \frac{3x_1^2 + A}{2y_1}$ sinon. Alors soit $x_3 = \lambda^2 - x_1 - x_2$ et $y_3 = \lambda(x_1 - x_3) - y_1$, on a $P_1 + P_2 = (x_3, y_3)$.*

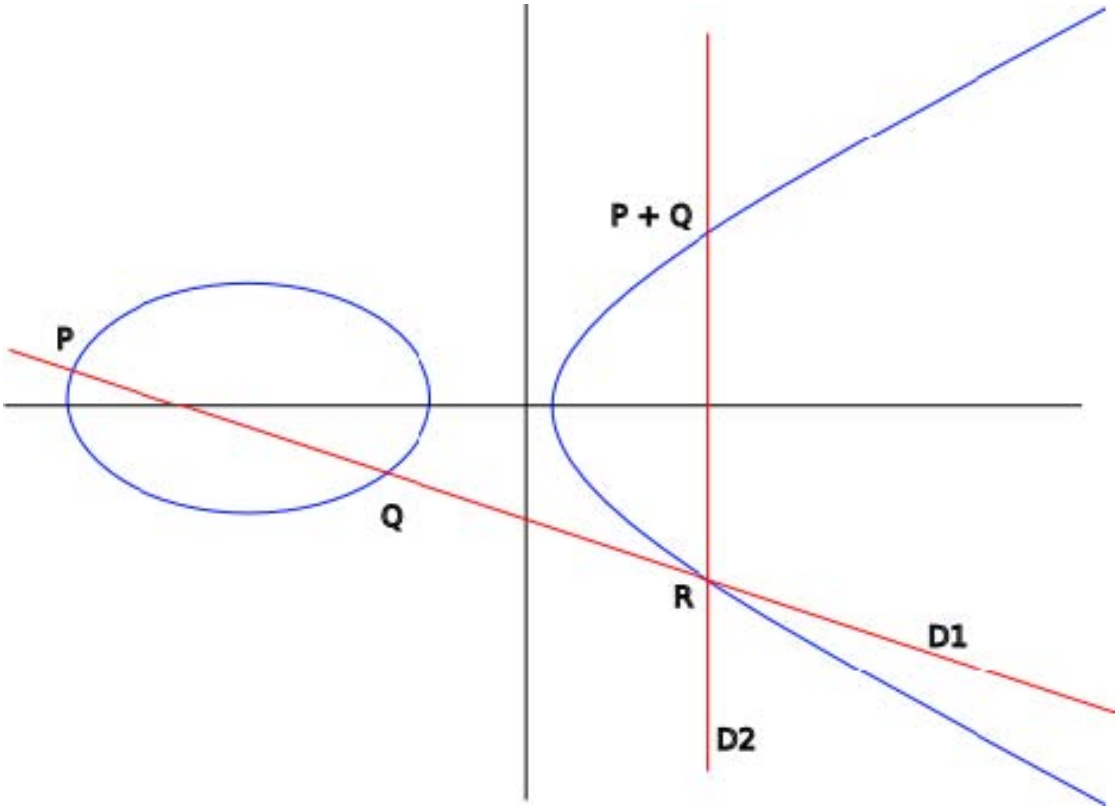

FIGURE 4 – addition de deux points distincts

2.1.3 cas où $\mathbb{K} = \mathbb{Q}$ ou $\mathbb{C}$

On remarque que, avec les notations de cet algorithme, si x_1, x_2, y_1 et y_2 sont rationnels, alors λ, x_3 et y_3 le seront aussi. Ainsi, cette loi de groupe peut se restreindre aux points rationnels de $E(\mathbb{R})$, pour donner $E(\mathbb{Q})$, et y définir une loi de groupe. De même, ces formules s'étendent bien à $\mathbb{C}$ et $E(\mathbb{C})$

2.1.4 théorèmes de structure

On connait des résultats assez puissants sur la structure du groupe d'une courbe elliptique, ou sur son nombre d'élément. Par exemple dans le cas rationnel :

Théorème 5 (Mordell-Weil)**.** *Si $E(\mathbb{Q})$ est une courbe elliptique sur $\mathbb{Q}$, alors son groupe est un groupe abélien de type fini.*

Théorème 6 (Mazur)**.** *Le sous-groupe de torsion de $E(\mathbb{Q})$ est soit $\mathbb{Z}/N\mathbb{Z}$ avec $1 \leq N \leq 10$ ou $N = 12$, soit $\mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/2N\mathbb{Z}$ avec $1 \leq N \leq 4$*

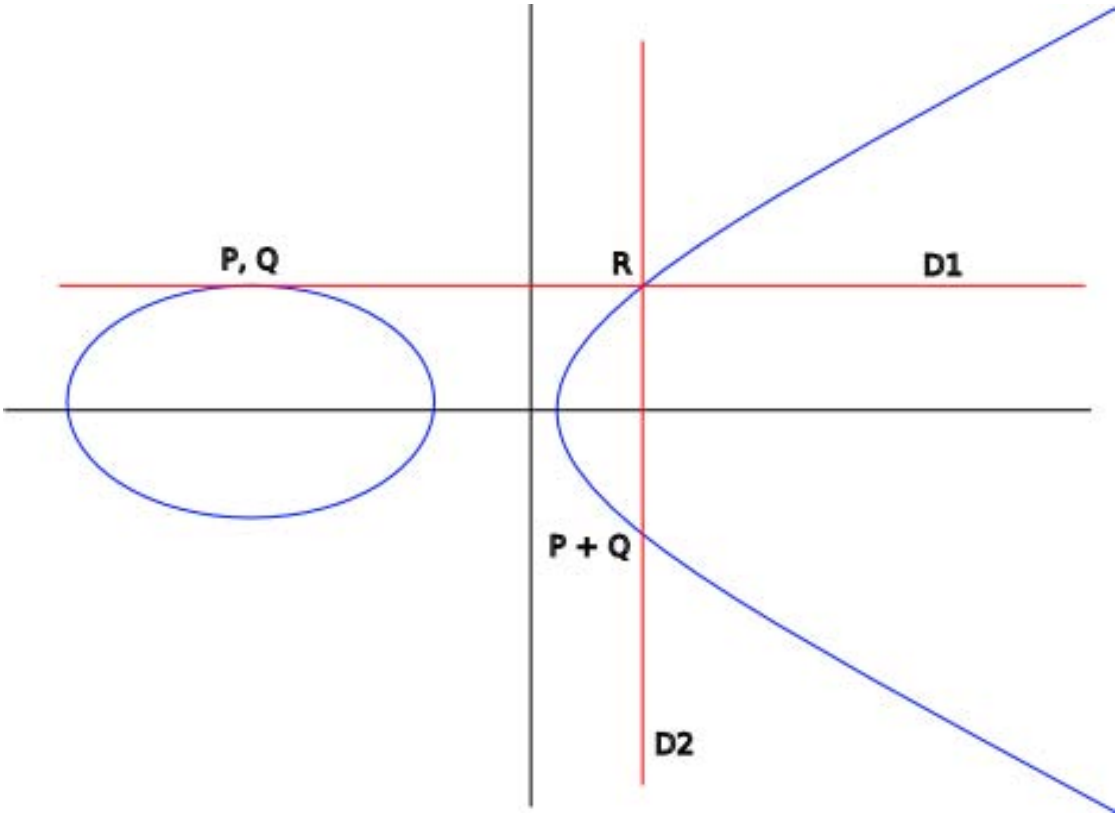

FIGURE 5 – addition d'un point à lui-même

Cependant, la démonstration de ces théorèmes est bien trop compliquée pour pouvoir figurer ici.

2.2 le cas des courbes sur un corps fini, de caractéristique strictement supérieure à 2

Soit p un nombre primaire qui ne soit pas une puissance de 2, alors de même que plus haut, les formules d'additions associent bien à deux points de $E(\mathbb{F}_p)$ (définis donc sur $\mathbb{F}_p$) un point de $\mathbb{F}_p^2$, qui se trouve être aussi dans $E(\mathbb{F}_p)$.

Remarque. Les courbes elliptiques sur $\mathbb{F}_{2^k}$ existent et sont utilisées, mais nous n'en verrons pas ici.

2.2.1 structure du groupe des points de m-torsion

On connait la structure du groupe des points de m-torsion d'une courbe elliptique sur un corps fini :

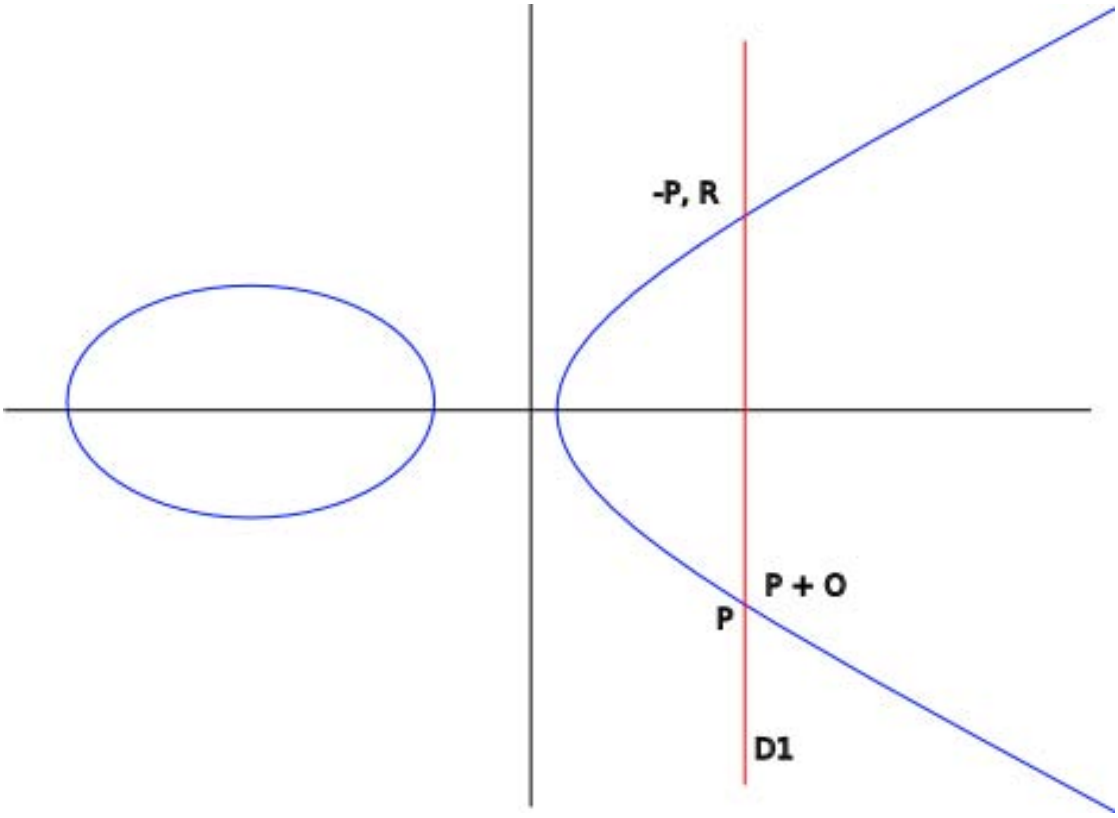

FIGURE 6 – ajouter l'élément neutre

Théorème 7. *Si char*$(\mathbb{K})$ *est premier avec* m*, alors*

$$E(\mathbb{K})[m] \cong (\mathbb{Z}/m\mathbb{Z} \times \mathbb{Z}/m\mathbb{Z})$$

Il existe plusieurs manière de montrer ce résultat (avec une étude des isogenies (qui ressemblent à des morphismes de courbes elliptiques), de l'analyse complexe et des réseaux, ou une étude des polynômes qui permettent d'exprimer les n-ième puissances d'un point de la courbe ...), mais aucune n'est assez élémentaire pour pouvoir figurer ici (malgré mes efforts pour tenter d'en produire une qui puissent convenir).

2.2.2 couplages

Définition 10. Un pairing, ou couplage, en cryptographie est une application $G_1 \times G_1 \to G_2$, où G_1 est un groupe additif et G_2 un groupe multiplicatif (G_1 est un sous-groupe du groupe d'une courbe elliptique), tout deux d'ordre p premier, vérifiant :

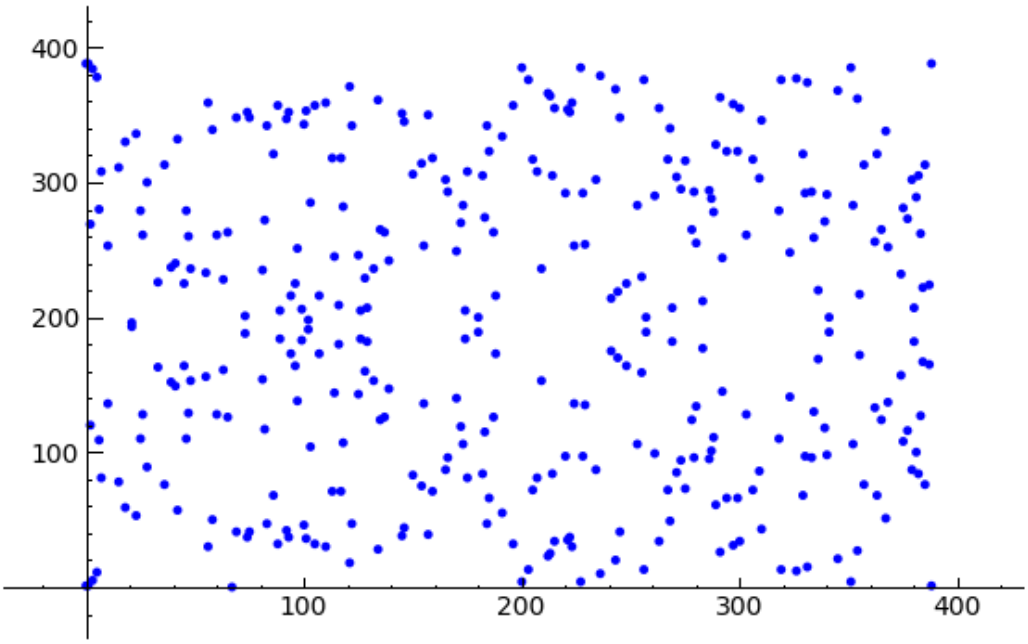

FIGURE 7 – Courbes elliptiques sur $\mathbb{F}p$, $p = 389$, $y^2 = x^2 - x + 1$

- La bilinéarité : $\forall (P, Q) \in G_1^2, (a, b) \in \mathbb{F}_p^*, e(aP, bQ) = e(P, Q)^{ab}$
- La non-dégenerescence : $\forall P \in G_1, P \neq \infty, e(P, P) \neq 1$

Il existe différents types de pairing, plus ou moins facilement calculables, et permettant des applications plus ou moins utiles ... Nous allons considérer le couplage de Tate modifié, ou modified Tate pairing :

Définition 11. Soit q un nombre primaire, E une courbe elliptique sur $\mathbb{F}_q$ et l tel que $q \equiv 1 \mod l$, alors le couplage de Tate modifié est un couplage : $\hat{\tau}$: $E(\mathbb{F}_q)[l] \times E(\mathbb{F}_q)[l] \to \mathbb{F}_q^*$ ayant la propriété de non-dégénerescence particulière, si $P \neq 0$, alors $\hat{\tau}(P, P)$ est une racine primitive l-ième de l'unité.

Il nous servira en fin de troisième partie, pour la mise en place d'un cryptosystème sur des courbes elliptiques.

Remarque. Le modified Tate pairing peut être construit explicitement, et se calcule assez aisément. Cependant, sa construction demande quelques éléments théoriques qu'il serait trop long de développer ici.

2.3 traitement en MAGMA

2.3.1 comment MAGMA considère les courbes elliptiques : l'espace projectif

MAGMA permet de travailler aisément dans sur des courbes elliptiques sur $\mathbb{Q}$, $\mathbb{R}$, des corps finis ... Les éléments d'une courbe elliptiques sont vus écrits comme élément de l'espace projectif $\mathbb{P}^3$. On en rappelle la définition :

Définition 12. On appelle $\mathbb{P}^n$, espace projectif de dimension n ($n \in \mathbb{N}$), le quotient de $\mathbb{K}^{n+1} \setminus \{0\}$, $\mathbb{K}$-espace vectoriel de dimension $n+1$, par la relation d'équivalence définie par $x \sim y$ si il existe $k \in \mathbb{K}^*$ tel que $y = kx$.

Il y a une injection naturelle de $\mathbb{K}^2$ dans $\mathbb{P}^2$: $(x, y) \mapsto (x, y, 1)$, et dans le sens contraire on a l'application $\mathbb{P}^2$ dans $\mathbb{K}^2 \cup \{\infty\}$, par $(x, y, z) \mapsto (\frac{x}{z}, \frac{y}{z})$, avec la convention que pour $z = 0$, l'image est ce point ∞.

Cette dernière application est bijective en la restreignant de A dans $\mathbb{K}^2$, avec A l'image de $\mathbb{K}^2 \times \mathbb{K}^*$ par la surjection canonique de $\mathbb{K}^3$ dans $\mathbb{P}^2$.

L'équation de $E(\mathbb{K})$, $Y^2 = X^3 + aX + b$ peut alors s'écrire, en voyant les points de $\mathbb{K}^2$ comme des points de A, $\frac{Y}{Z}^2 = \frac{X}{Z}^3 + a\frac{X}{Z} + b$, ou encore ($Z \in \mathbb{K}^*$)

$$Y^2Z = X^3 + aXZ^2 + bZ^3.$$

Si l'on regarde cette équation dans $\mathbb{P}^2$, on remarque que $Z = 0$ amène $X = 0$ et comme 0 n'est pas dans $\mathbb{P}^2$, il n'y a qu'une seule solution dans $\mathbb{P}^2$ à cette équation, avec $Z = 0$, $(0, 1, 0)$ (on rappelle que ces points sont vues à une multiplication par un élément de $\mathbb{K}^*$ près).

L'espace projectif peut alors servir à enlever "l'infini" des points à l'infini : l'application de $E(\mathbb{K})$ dans $\mathbb{P}^2$, définie par $(X, Y) \mapsto (X, Y, 1)$ et $\infty \mapsto (0, 1, 0)$ est injective. MAGMA ne va alors travailler que sur les points de l'image de cette application, qui d'après ce qui précède, sont les seules solutions dans $\mathbb{P}^2$ à $Y^2Z = X^3 + aXZ^2 + bZ^3$ (du fait de la bijection entre A et $\mathbb{K}^2$ et de l'unique solution dans $\mathbb{P}^2$ ayant $Z = 0$).

On peut écrire pour ceux-ci des formules d'addition qui n'auront plus à considérer le point à l'infini à part (devenu simplement $(0, 1, 0)$).

2.3.2 exemples de traitement de courbes elliptiques par MAGMA

Voici un premier exemple de la manière dont MAGMA traite les courbes elliptiques, avec une courbe elliptique sur $\mathbb{F}_7$:

```
> E:=EllipticCurve([GF(7)!(-1),0]);
> E;
```

```
Elliptic Curve defined by y^2 = x^3 + 6*x over GF(7)
> Set(E);
{ (4 : 5 : 1), (6 : 0 : 1), (1 : 0 : 1), (5 : 1 : 1), (5 : 6 : 1), (4 : 2 : 1),
(0 : 0 : 1), (0 : 1 : 0) }
> A:=Random(E);
> A;
(6 : 0 : 1)
> B:=Random(E);
> B;
(4 : 2 : 1)
> A+B;
(5 : 6 : 1)
```

Les calculs dans une courbe elliptique sont relativement compliqués, on peut donc penser et écrire un programme naïf qui, similairement à l'exponentiation rapide, permet une multiplication rapide d'un point par un scalaire. On appelle cet algorithme (qui suit) *double and add* :

Algorithm 1 programme *double and add* : mult(P,k)= kP

```
Q :=P ; R :=Id(Parent(P)) ;
n :=k ;
while n gt 0 do
  if ((n mod 2) eq 1) then
    R := R+Q ;
  end if
  Q :=Q+Q ;
  n :=(n div 2) ;
end while
return R ;
```

```
> mult(A+B,3);
(5 : 1 : 1)
```

3 applications en cryptographie

Enfin, nous allons nous intéresser aux applications des courbes elliptiques et de MAGMA à la cryptographie. Pour cela, nous allons d'abord présenter dans les deux premières sous-parties suivantes quelques éléments classiques de cryptographies. Celles-ci trouvent leur inspiration principale dans [4].

Ensuite, nous nous intéresserons à un cryptosystème utilisant des courbes elliptiques, que l'on doit à [2].

3.1 DLP : the Discrete Logarithm Problem

3.1.1 le principe du DLP

Le problème du logarithme discret est une des bases de la cryptographie moderne. Si g est un élément générateur du groupe mutiplicatif de $\mathbb{F}_q$, où q est un nombre primaire, et si h est un élément non nul de $\mathbb{F}_q$, le DLP consiste à trouver un n entier tel que $g^n = h$.

Remarque. Si n résout le DLP, alors tout les $n + l * (q - 1)$, avec l entier, le feront aussi, du fait que le groupe multiplicatif de $\mathbb{F}_q$ soit d'ordre $q - 1$. Le plus petit entier positif n tel que $g^n = h$ sera noté $\log_g(h)$.

Remarque. La condition que g soit un élément générateur n'est pas nécessaire : seule l'existence d'un tel n l'est.

Même s'il existe ainsi une infinité de solutions, et si l'équivalent réel est "relativement" aisé, cela n'est pas le cas du DLP : q est souvent, dans les faits, considéré très grand, ce qui rend une première approche naïve et directe, tester les puissances successives de g, inutilisable car étant en $O(q)$ (à un facteur négligé dépendant de l'efficacité avec laquelle les exponentiations sont faites). Il existe cependant quelques méthodes plus efficaces : les meilleures méthodes connues sont sous-exponentielles (en $O(p^\epsilon)$ pour tout $\epsilon > 0$), comme celle dite d'*index calculus*, mais sont un peu trop compliquées pour être traitées ici. Les deux méthodes que nous allons exposer dans ce qui suit sont respectivement en $O(\sqrt{p}\log(p))$ et $O(\sqrt{p})$.

3.1.2 Une première technique de résolution : l'algorithme Babystep-Giantstep de Shanks

Il fait partie des techniques de résolution dites de collision, qui consistent à chercher un élément commun dans plusieurs tableaux différents (qui réalise la "collision") permettant de résoudre le problème.

Soit N l'ordre de g (ou sinon, un de ses multiples, par exemple $q - 1$). Alors l'algorithme Babystep-Giantstep va résoudre le DLP en $O(\sqrt{N}\log N)$:

1 Soit $n = 1 + E(\sqrt{N})$ où E désigne la fonction partie entière ;
2 On crée les listes $e, g, ..., g^n$ et $h, h * g^{-n}, h * g^{-2*n}, ..., h * g^{-n^2}$;
3 On cherche i et j tel $g^i = h * g^{-j*n}$;
4 Alors soit $x = i + j * n$, on a $g^x = h$.

Démonstration. Montrons d'abord qu'une telle collision existe bien dès qu'existe x une solution au DLP que nous allons prendre minimale (positive). Comme $n = 1 + E(\sqrt{N}) > \sqrt{N}$ et que comme N est l'ordre de g, on a $x < N$, et si l'on écrit $x = q * n + r$ (division euclidienne de x par n), alors $q = \frac{x-r}{n} \leq \frac{x}{n} < \frac{N}{n} < \sqrt{N} < n$

et par définition, $r \leq n$, donc si l'on écrit $g^r = h * g^{-q*n}$, on voit que $i = r$ et $j = q$ existent et conviennent. Ceci démontre la correction de l'algorithme.

Le temps de calcul va fortement dépendre de la manière dont la recherche de i et j sera faite. Des algorithmes de tri et de recherche dans des tableaux triés, appliqué successivement aux éléments d'un tableau, permettent (après une relative réflexion sur la manière dont les éléments sont représentés que j'ai epargné à mon implémantation) de réaliser cela en $O(n \log n)$ soit en $O(\sqrt{N} \log N)$. □

Voici une implémentation naïve de cet algorithme en MAGMA (utilisant les fonctions d'intersection de deux ensemble, et de recherche d'éléments dans un tableau).

Algorithm 2 algorithme Babystep-Giantstep de Shanks : babygiant(g,h)

```
N :=Order(g) ;
n :=Ceiling((RealField() !N)^(1/2)) ;
L1 :=[g^i : i in [0..n]] ;
L2 :=[h*g^(-i*n) : i in [0..n]] ;
l1 :=g^i : i in [0..n] ;
l2 :=h*g^(-i*n) : i in [0..n] ;
r :=(l1 meet l2) ;
if #r ne 0 then
   a :=Random(r) ;
   i :=Index(L1,a)-1 ; {Index(L1,a) calcule l'indice du premier élément de L1
   valant a}
   j :=Index(L2,a)-1 ;
   return (i+j*n) ;
else
   print "pas de solutions" ;
   return -1 ;
end if
```

```
> g:=(GF(11^4)!6);
> h:=g^17;
> g,h;
6 8
> babygiant(g,h);
7
> g^7;
8
```

3.1.3 Une deuxième méthode, la méthode ρ de Pollard

La méthode suivante va se réveler un peu plus efficace :

Principe : Le principe général de cette méthode se ramène à détecter une période dans la suite $x_n = f^n(x)$, $(x_n) \in S^{\mathbb{N}}$, où S est un ensemble fini, et f une fonction de cet ensemble dans lui-même. Connaître le début du premier cycle et la longueur du cycle va permettre, pour une fonction f correctement choisie, de résoudre le DLP. Pour cela, on va chercher un i tel que $x_i = x_{2i}$ (la suite y_i se calculant simplement par $y_{i+1} = f^2(y_i)$. Par de laborieux calculs on peut montrer que si f est "suffisament aléatoire" (soit, qu'on puisse considérer les tirages $x_0, ..., x_n, ...$ comme indépendants), on peut obtenir une telle collision en $O(\sqrt{N})$ (notons que celle-ci existe toujours, de même que l'existence d'une période, du fait qu'il n'existe pas d'injection de $\mathbb{N}$ dans S). Cependant, il n'est pas encore connu de fonction qui soit à la fois "suffisament aléatoire", et qui permette, étant donné la collision, d'en déduire une solution au DLP : il n'a jamais été prouvé que la fonction que nous allons utiliser vérifie cette hypothèse, mais, d'après [4] elle a montré une bonne efficacité.

Mise en place : Pour mettre en place cette méthode, on pose

$$\begin{aligned} f(x) &= gx \; si \; 0 \leq x < p/3 \\ &= x^2 \; si \; p/3 \leq x < 2p/3 \\ &= ax \; sinon \end{aligned}$$

On a alors, en prenant $x_0 = 1$,

$$x_i = g^{\alpha_i} a^{\beta_i},$$

avec $\alpha_0 = \beta_0 = 0$ et

$$\begin{aligned} \alpha_{i+1} &= \alpha_i + 1 \; si \; 0 \leq x < p/3 \\ &= 2\alpha_i \; si \; p/3 \leq x < 2p/3 \\ &= \alpha_i \; sinon \end{aligned}$$

$$\begin{aligned} \beta_{i+1} &= \beta_i \; si \; 0 \leq x < p/3 \\ &= 2\beta_i \; si \; p/3 \leq x < 2p/3 \\ &= \beta_i + 1 \; sinon, \end{aligned}$$

en comptant les α_i et β_i modulo $p - 1$. De même, $y_i = x_{2i} = g^{\gamma_i} a^{\delta_i}$, avec $\gamma_i = \alpha_{2i}$ et $\delta_i = \beta_2 i$.

comment déduire d'une collision une solution au DLP Supposons que l'on a réussi à trouver un i tel que $x_i = y_i$, soit $g^{\alpha_i} a^{\beta_i} = g^{\gamma_i} a^{\delta_i}$.

Posons $u = \alpha_i - \gamma_i \mod p-1$ et $v = \delta_i - \beta_i \mod p-1$. On a alors $g^u = a^v$ dans $\mathbb{F}_p$, ce qui se ramène au fait que l'on a $v \log_g(a) = u \mod p-1$. Il y a alors deux cas à traiter : soit $d = pgcd(v, p-1) = 1$ et alors on peut directement calculer $\log_g(a) = u * v^{-1}$, soit ce n'est pas le cas, et alors, il existe s tel que (égalité de Bezout) $sv = d \mod p-1$, et alors, en écrivant $w = su$, on a

$$d \log_g(a) = w \mod p-1.$$

De cette égalité, comme par définition $d|p-1$, on a nécessairement $d|w$ et on peut écrire

$$\log_g(a) = \frac{w}{d} \mod \frac{p-1}{d},$$

il ne reste donc plus qu'à trouver un élément qui convienne parmi $\{\frac{w}{d} + k\frac{p-1}{d}, 0 \leq k < d\}$.

Mon algorithme : Cette dernière partie de déduction est effectuée par la fonction loga. La fonction rho combine une fonction phi qui correspond à la fonction f utilisée plus haut, et qui calcule aussi les α_i et β_i, et cette fonction loga pour réaliser la méthode ρ de Pollard. Le code en MAGMA constitue les `Algorithm` 6 à 8 de l'annexe.

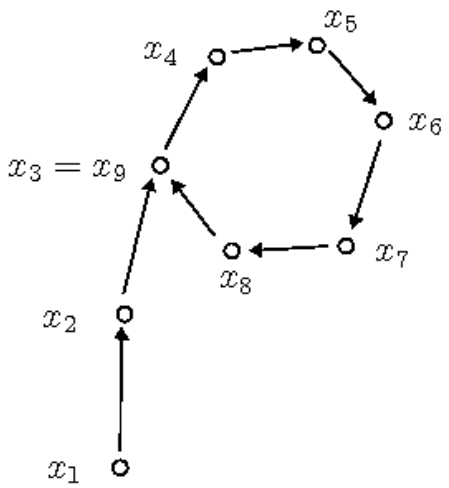

FIGURE 8 – le nom de la méthode, ρ vient de cette forme qu'on peut donner à la suite considérée

```
> IsPrime(659);
true
```

```
> g;
6
> GF(659)!123;
123
> (GF(659)!123)^45;
377
> rho(659,GF(659)!123,GF(659)!377);
45
```

3.2 ... et avec des courbes elliptiques

Le concept de DLP se transpose facilement au groupe d'une EC, avec deux avantages. Tout d'abord, les calculs seront plus longs, donc cela rendra la résolution du problème plus difficile, mais de plus, si les deux méthodes vues plus haut, Babystep-Giantstep et ρ de Pollard, peuvent relativement aisément s'adapter aux EC, ce n'est pas le cas des méthodes sous-exponentielles qui existent pour le DLP (par exemple, l'*index calculus* repose sur des concepts comme ceux de nombres *B-smooth* (friables) ou du symbole de Jacobi, qui ne se transposent pas au groupe d'une EC). C'est donc l'intérêt de la cryptographie sur des EC : on n'y connait pas d'algorithmes aussi rapides que ceux que l'on connait pour résoudre le DLP. Néanmoins, elle a l'inconvéniant du fait que choisir des points sur une courbe pour transcrire un message n'est pas aussi naturel que considérer les nombres binaires de ses lettres.

J'ai par exemple adapté la méthode Babystep-Giantstep de Shanks aux courbes elliptiques.

```
> E:=EllipticCurve([0,GF(19)!(-1)]);
> E;
Elliptic Curve defined by y^2 = x^3 + 18 over GF(19)
> g:=Random(E);
> 47*g;
(3 : 11 : 1)
> g;
(15 : 12 : 1)
> ecbaby(g,47*g);
5
> 5*g;
(3 : 11 : 1)
```

On pourrait aussi adapter la méthode ρ de Pollard, dont il se trouve qu'elle est la méthode la plus rapide connue pour résoudre le DLP des EC.

3.3 Knapsack Cryptosystem on Elliptic Curves

Nous allons développer dans ce qui suit un cryptosystème sur des courbes elliptiques, d'abord en expliquant son principe, puis son implémentation. Ce cryptosystème à clé publique et clé privé, va se fonder sur l'usage du modified Tate pairing, dont nous avons parlé plus haut, pour générer ses clés. Il est dû à Koichiro Noro et Kunikatsu Kobayashi et est exposé dans [2].

3.3.1 principe : génération des clés

Soit p et n deux nombres premiers ($p \neq n$), nous allons regarder $E(\mathbb{F}_p)[n]$, le groupe des points de n-torsion de la courbe elliptique $E(\mathbb{F}_p)$ définie par $y^2 = x^3 + ax + b$ sur $\mathbb{F}_p$.

p est pris comme étant un grand nombre premier (plus de 1024 bits), et n aussi (plus de 160 bits), tel que n divise l'ordre du groupe de $E(\mathbb{F}_p)$, et que $E(\mathbb{F}_p)[n]$ soit d'ordre n.

On va prendre P dans $E(\mathbb{F}_p)[n]$ et Q dans $E(\mathbb{F}_{p^2})$. On choisit $k \in \mathbb{N}$ au hasard tel que $\sum_{i=1}^{ur}(k2^{i-1}) < \frac{n-1}{2}$ (id est $k < \frac{n-1}{2(2^{ur}-1)}$), et on calcule les a_iP, avec $a_i = k2^{i-1}$ (avec $ur > 100$ pour résister à des attaques "force brute").

Soit $R \in E(\mathbb{F}_p)[n]$, tel que $\forall i, 1 < i < ur, R \neq a_iP$, soit $d \in \mathbb{Z}/n\mathbb{Z}$, et soit $S = dR$. On calcule les $b_{i,j} = \hat{\tau}(P,Q)^{k*2^{((i-1)*r)*j)}}$, avec $1 < j < 2^{r-1}$ et $1 < i < u$, puis les polynômes $f_i(x) = (x-1) \times \prod_{k=1}^{2^{r-1}}(x - b_{ik})$, pour i de 1 à u.

On a alors généré les clés :

Clé Publique : $a_1P, ..., a_{ur}P, E(\mathbb{F}_p), R, S, r$

Clé Privée : $d, f_1, ..., f_u, \hat{\tau}(P,Q), a_1, ..., a_{ur}$

3.3.2 principe : encryption

Le texte à encrypter est $M = (m_1, ..., m_{ur}) \in 0, 1^{ur}$. On pose

$$C = m_1(a_1P) + ... + m_{ur}(a_{ur}P),$$

puis, pour i de 1 à $u-1$

$$C_i = m_{(i-1)r+1}(a_{(i-1)r+1}P) + ... + m_{ir}(a_{ir}P).$$

On prend $t_1, ..., t_{u-1}$ au hasard dans $\mathbb{Z}/n\mathbb{Z}$, et on calcule, pour i de 1 à $u-1$:

$$C_{i,1} = t_iR,$$

$$C_{i,2} = C_i + t_iS.$$

Enfin, le texte transmis et crypté est : $C, C_{1,1}, C_{1,2}, ..., C_{u-1,1}, C_{u-1,2}$.

3.3.3 principe : décryption

Grâce à la clé secrète, nous allons décrypter le texte transmis.

Tout d'abord, nous récupérons les $C_1, ..., C_{u-1}$ à partir des $C_{1,1}, C_{1,2}, ..., C_{u-1,1}, C_{u-1,2}$, avec le fait que $S = dR$, pour i de 1 à $u-1$:

$$\begin{aligned} C_{i,2} - dC_{i,1} &= C_i + t_i S - dt_i R \\ &= C_i + t_i dR - dt_i R \\ &= C_i \end{aligned}$$

Maintenant que l'on connaît les C_i, on peut décrypter les m_i. Soit i, $1 \leq i < u$. Soit $Y = \frac{\hat{\tau}(C_i,Q)}{\hat{\tau}(P,Q)^{a_{ir}}}$. Les propriétés de bilinéairité du couplage et les définition de C_i et a_i ($a_i = k2^{i-1}$) donnent :

$$\begin{aligned} Y &= \hat{\tau}(C_i, Q)/\hat{\tau}(P,Q)^{a_{ir}} \\ &= \hat{\tau}(m_{(i-1)r+1}(a_{(i-1)r+1}P) + ... + m_{ir}(a_{ir}P), Q)/\hat{\tau}(a_{ir}P, Q) \\ &= \hat{\tau}(m_{(i-1)r+1}(a_{(i-1)r+1}P) + ... + m_{ir}(a_{ir}P) - a_{ir}P, Q) \\ &= \hat{\tau}((m_{(i-1)r+1}a_{(i-1)r+1} + ... + m_{ir}a_{ir} - a_{ir})P, Q) \\ &= \hat{\tau}(k(m_{(i-1)r+1}a_{(i-1)r+1} + ... + m_{ir}a_{ir} - a_{ir})P, Q) \\ &= \hat{\tau}(k(m_{(i-1)r+1}2^{(i-1)r} + ... + m_{ir}2^{ir-1} - 2^{ir-1})P, Q) \\ &= \hat{\tau}(k2^{(i-1)r}(m_{(i-1)r+1} + ... + m_{ir}2^{r-1} - 2^{r-1})P, Q) \\ &= (\hat{\tau}(P,Q)^k)^{2^{(i-1)r}(m_{(i-1)r+1}+...+m_{ir}2^{r-1}-2^{r-1})}. \end{aligned}$$

On rappelle maintenant que $\hat{\tau}(P,Q)$ est, par définition du couplage de Tate modifié, une racine n-ème de l'unité.

Or, $b_{i,j} = \hat{\tau}(P,Q)^{k*2^{((i-1)*r)*j)}}$, avec $1 < j < 2^{(r-1)}$ et $1 < i < u$.

Comme on a $\sum_{i=1}^{ur}(k2^{i-1}) < \frac{n-1}{2} < n$, alors les $b_{i,j}$ sont parmi les $\frac{n-1}{2}$ premières puissances de $\hat{\tau}(P,Q)$ (et sont bien sûr tous distincts).

Ainsi, si $m_{ir} = 0$, $(\hat{\tau}(P,Q)^k)^{2^{(i-1)r}(m_{(i-1)r+1}+...+m_{ir}2^{r-1}-2^{r-1})}$, est $\hat{\tau}(P,Q)$ élevé à une puissance négative plus petite strictement que $\frac{n-1}{2}$, ce qui revient à une puissance positive strictement entre $\frac{n-1}{2}$ et n, donc Y ne fait pas partie des $b_{i,j}$, et sinon (les m_i valant 0 ou 1), $(\hat{\tau}(P,Q)^k)^{2^{(i-1)r}(m_{(i-1)r+1}+...+m_{ir}2^{r-1}-2^{r-1})} = (\hat{\tau}(P,Q)^k)^{2^{(i-1)r}(m_{(i-1)r+1}+...+m_{ir-1}2^{r-2}}$, et $k(2^{(i-1)r}(m_{(i-1)r+1} + ... + m_{ir-1}2^{r-2}) < \sum_{i=1}^{ur}(k2^{i-1}) < \frac{n-1}{2}$, et alors Y est bien un $b_{i,j}$.

On rappelle que $f_i(x) = (x-1) \times \prod_{k=1}^{2^{r-1}} b_{ik}$, on peut donc en déduire :

- Soit $f_i(Y) = 0$, auquel cas, Y est un $b_{i,j}$ et donc $m_{ir} = 1$;
- Soit $f_i(Y) \neq 0$, et alors Y n'est pas un $b_{i,j}$ et donc $m_{ir} = 0$.

Dans le premier cas, on prend maintenant $Y = \frac{Y}{\hat{\tau}(P,Q)^{a_{ir}}}$, et dans le second cas, on conserve le même Y et on itère le processus en calculant $f_i(Y/\hat{\tau}(P,Q)^{a_{ir-1}}$

(c'est-à-dire qu'on regarde si $(\hat{\tau}(P,Q)^k)^{2^{(i-1)r}(m_{(i-1)r+1}+...+m_{ir-1}2^{r-2}-2^{r-2})}$ est dans les $b_{i,j}$), et ainsi de suite pour décrypter de $m_{(i-1)r+1}$ à m_{ir}.

Il reste une dernière subtilité pour décrypter de $m_{(u-1)r+1}$ à m_{ur} : comme on n'a pas défini de C_u plus tôt, on va simplement considérer

$$C_u = C - C_1 - ... - C_{u-1},$$

et alors appliquer la même méthode que précèdemment avec

$$Y_u = \hat{\tau}(C,Q)/(\hat{\tau}(C_1,Q)...\hat{\tau}(C_{u-1},Q)) = \hat{\tau}(C_u,Q).$$

Finalement, en appliquant cela de C_1 à C_u, on a décrypté M.

3.3.4 implémentation

Les `Algorithm` 9 à 11 de l'annexe permettent de générer des clés, encrypter et décrypter des messages selon ce cryptosystème.

```
> K<X>:=PolynomialRing(GF(p));
> T,R,S,r,d,F,e,A,p,n,Q:=knapkey(p,curv,a_1,a_2,X^2+1,n,u,r);
> M;
[ 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1,
1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1,
1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1,
0, 0, 0, 0, 0 ]
> C,Ci1,Ci2:=knapencryption(T,R,S,r,M);
> C;
(482698757065238876272437026059638628320669604772 5 :
5709672466186017389825233035724924526285730605581 : 1)
> Ci1;
[ (803699351254174540902230185143791260265265124074 0 :
6867285714877750809412627593152304355649340751259 : 1),
(5689380324941397505455031007976449937451339448695 :
3958046536973800466549306745306926269343210466310 : 1),
(6891719571685671566919661931911341355386214814727 :
3218968101576543814173061259205865899793913291956 : 1),
(7081464274373795762213901665973357340311716903 94 :
5300235758397123663335652656884998542714893238684 : 1),
(6295240974504063501360615690679316203686834030 66 :
5300092908353126327969603161463584575394210817491 : 1),
(6798025850615457068305737793513080514161382259885 :
```

```
821322999582972998041806349429522259212920066881 : 1) ]
> Ci2;
[ (111616868453814953655822456416065712516889781389 9 :
3451075413053052394614609961850643697078354371344 : 1),
(605119124929382586972544174875186986241002383898 9 :
5152902986653157225346612540853081828954865050 87 : 1),
(224492139210524272977021967888522093221740249348 9 :
9286380026897457918118779726635158294641490774396 : 1),
(929182866671410725529342353879191401800876326351 8 :
9397853819506905762559551118079356727241859705462 : 1),
(902188761988516144598878568769410861424355864606 2 :
9326683249488672267678312279147018001313932510527 : 1),
(410360287619176063630297128186578477989438360867 2 :
8153612429311064037726477242786709279064465616843 : 1) ]
> knapdecryption(d,F,e,A,C,Ci1,Ci2,n,r,Q);
[ 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1,
1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1,
1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1,
0, 0, 0, 0, 0 ]
> M2:=knapdecryption(d,F,e,A,C,Ci1,Ci2,n,r,Q);
> M2 eq M;
true
```

Remarque. L'implémentation n'est pas optimale (le calcul des $b_{i,j}$ est fait par force brute, alors qu'un calcul récursif serait bien plus efficace).

Conclusion

Ainsi, on a pu voir la puissance et la simplicité d'usage que peut avoir MAGMA en tant que logiciel de calcul formel : MAGMA peut non seulement réaliser des calculs efficaces sur des objets abstraits comme des groupes de Galois, des quotients d'anneaux par des idéaux ou le traitement d'ensembles finis, mais sa syntaxe permet aussi de les faire facilement. On a pu aussi avoir une première approche de la théorie des courbes elliptiques, et là encore, de l'intérêt que peu avoir MAGMA pour calculer dans celles-ci, et finalement, comment on peut produire un cryptosystème en MAGMA reposant sur des courbes elliptiques.

Au-delà des éventuelles applications, je retiendrai surtout la grande élégance de la théorie de Galois et la beauté du groupe d'une courbe elliptique. Cependant, je ne peux que regretter que mon stage n'ai pas été suffisament long pour aller plus loin dans la théorie de Galois, ou démontrer les théorèmes que j'ai évoqué sur

les courbes elliptiques (comme ceux de Mordell-Weil ou de structure des points de n-torsion), mais peut-être sera-ce l'objet d'un de mes futurs autre stage, un peu plus long ...

Références

[1] Escofier, Jean-Pierre Théorie de Galois (Masson)

[2] Noro, Koichiro et Kobayashi, Kunikatsu Knapsack Cryptosystem on Elliptic Curves

[3] Cohen, Henri et Frey, Gerhard Handbook of Elliptic & Hyperelliptic Curve Cryptography (Chapman & Hall / CRC)

[4] Hoffstein, Jeffrey, Pipher, Jill et Silverman, Joseph H. An Introduction to Mathematical Cryptography (Springer)

[5] Silverman, Joseph H. The Arithmetic of Elliptic Curves (Springer)