Théorie algébrique des nombres avec GP

B. Allombert et A. Page

IMB
CNRS/INRIA / Université de Bordeaux

15/06/2023

Documentation

- refcard-nf.pdf : liste des fonctions avec une courte description.
- users.pdf Section 3.10 : paragraphe introductif et descriptions détaillées des fonctions.
- dans gp, `?10` : liste des fonctions.
- dans gp, `?nomdefonction` : description courte de la fonction.
- dans gp, `??nomdefonction` : description longue de la fonction.
- dans gp, `???mot` : recherche mot dans la doc.

Pour enregistrer vos commandes pendant le tutoriel :

```
? \l TAN.log
```

Plan

1. Corps de nombres
2. Éléments et idéaux
3. Groupes de classes et groupes des unités

Corps de nombres

Irréductibilité

Dans GP, un corps de nombres K est décrit comme

$$K = \mathbb{Q}[x]/f(x)$$

où $f \in \mathbb{Z}[x]$ est un polynôme irréductible unitaire.

```
? f = x^4 - 2*x^3 + x^2 - 5;
? polisirreducible(f)
%2 = 1
```

GP connait les polynômes cyclotomiques :

```
? g = polcyclo(30)
%3 = x^8 + x^7 - x^5 - x^4 - x^3 + x + 1
```

Polmod

Pour effectuer de simples opérations dans
$K = \mathbb{Q}[x]/f(x) = \mathbb{Q}(\alpha)$ où $f(\alpha) = 0$, on peut utiliser `Mod` :

```
? Mod(x,f)^5
%4 = Mod(3*x^3-2*x^2+5*x+10, x^4-2*x^3+x^2-5)
```

Interprétation : $\alpha^5 = 3\alpha^3 - 2\alpha^2 + 5\alpha + 10$.

```
? lift(Mod(x,g)^15)
%5 = -1
```

Les racines de g sont bien des racines 30ème de l'unité.
On a utilisé `lift` pour avoir une sortie plus lisible.

Compositum

Pour construire un corps de nombre comme

$$\mathbb{Q}(\sqrt{2}, \sqrt{3}),$$

Nous pouvons le construire comme compositum.

```
? polcompositum(x^2-2,x^2-3)
% = [x^4 - 10*x^2 + 1]
```

Le résultat est un vecteur de polynômes représentant les différents compositum possibles.

```
? polcompositum(x^4-2, x^4-2*x^2-1)
% = [x^8 - 4*x^6 - 26*x^4 - 4*x^2 + 1,
    x^8 - 4*x^6 + 22*x^4 - 36*x^2 + 49]
```

Compositum

Nous pouvons aussi ajouter deux racines du même polynôme :

```
? L = polcompositum(x^3-2, x^3-2)
% = [x^3 + 2, x^6 + 40*x^3 + 1372]
```

Le premier correspond au cas ou les deux racines sont égales, et le second où elles sont conjuguées.

```
? nfrootsof1(L[2])
% = [6, -1/36*x^3 - 1/18]
```

Nous vérifions que le second corps contient une racine 6-ème de l'unité.

Isomorphismes et inclusions

Nous pouvons tester si deux corps de nombres sont isomorphes :

```
? nfisisom(x^2-5, x^2-2)
% = 0 \\ Non
? nfisisom(x^2-5, x^2-x-1)
% = [-2*x + 1, 2*x - 1]
```

La fonction renvoie soit 0 soit l'ensemble des automorphismes. Les morphismes sont donnés par l'image du générateur du corps de départ dans le corps d'arrivée.
Nous pouvons tester si un corps est inclus dans un autre :

```
? nfisincl(x^2-5,polcyclo(5))
% = [-2*x^3 - 2*x^2 - 1, 2*x^3 + 2*x^2 + 1]
? nfisincl(x^2+5,polcyclo(5))
% = 0
```

Sous-corps

Nous pouvons calculer les sous-corps d'un corps

```
? nfsubfields(x^6 + 40*x^3 + 1372,,1)
% = [x, x^2 + 120*x + 12348, x^3 - 128, x^3 + 250,
%    x^3 - 2, x^6 + 40*x^3 + 1372]
```

L'option "`(...,,1)`" évite d'afficher les inclusions.
Nous pouvons préciser un degré :

```
? nfsubfields(x^8 + 3*x^4 + 9, 2)
% = [[x^2 + 36, 2/3*x^6 + 4*x^2],
%    [x^2 - 12, -2/3*x^6],
%    [x^2 + 12*x + 144, 4*x^4]]
```

Pour chaque corps, le second polynôme représente l'inclusion.

Subfields

Nous pouvons demander seulement les sous-corps maximaux.

```
? nfsubfieldsmax(x^8-4*x^5+7*x^4-x^2+x+1, 1)
% = [x^2 + 197*x - 199, x^4 - 10*x^2 - 37*x + 121]
```

Ils n’ont pas forcément tous le même degré.

polredbest

Parfois, on peut trouver un polynôme de définition plus simple pour le même corps de nombres, en utilisant `polredbest` :

```
? {h = x^5 + 7*x^4 + 22550*x^3 - 281686*x^2
  - 85911*x + 3821551};
? polredbest(h)
%7 = x^5 - x^3 - 2*x^2 + 1
```

Interprétation : $\mathbb{Q}[x]/h(x) \cong \mathbb{Q}[x]/(x^5 - x^3 - 2x^2 + 1)$.

nfinit

La plupart des opérations sur les corps de nombres nécessitent d'avoir calculé l'anneau des entiers, ce qui est fait par la fonction d'initialisation `nfinit` (nf = number field).

```
? K = nfinit(f);
```

`K` contient la structure représentant le corps de nombres $K = \mathbb{Q}[x]/f(x)$.

```
? K.pol
%9 = x^4 - 2*x^3 + x^2 - 5
? K.sign
%10 = [2, 1]
```

K est de signature $(2, 1)$: il admet deux plongements réels et une paire de plongements complexes conjugués.

Informations calculées

```
? K.disc
%11 = -1975
? K.zk
%12 = [1,1/2*x^2-1/2*x-1/2,x,1/2*x^3-1/2*x^2-1/2*x]
? w = K.zk[2];
```

K est de discriminant -1975, et son anneau d'entiers est

$$\mathbb{Z}_K = \mathbb{Z}+\mathbb{Z}\frac{\alpha^2-\alpha-1}{2}+\mathbb{Z}\alpha+\mathbb{Z}\frac{\alpha^3-\alpha^2-\alpha}{2} = \mathbb{Z}+\mathbb{Z}w+\mathbb{Z}\alpha+\mathbb{Z}w\alpha.$$

Factorisation de polynômes sur un corps de nombres

Nous pouvons factoriser les polynômes sur un corps de nombre. Pour cela nous devons nous assure que la variable du polynôme a une priorité supérieure.

```
? Kz = nfinit(subst(K.pol,x,z));
? nffactor(Kz, K.pol)
% =
[           x + Mod(-z, z^4 - 2*z^3 + z^2 - 5) 1]
[        x + Mod(z - 1, z^4 - 2*z^3 + z^2 - 5) 1]
[x^2 - x + Mod(z^2 - z, z^4 - 2*z^3 + z^2 - 5) 1]
```

Le résultat est une matrice avec deux colonnes ; la première contient les facteurs irréductibles, et a seconde les exposants.

Racines de polynômes sur un corps de nombre

Nous pouvons demander simplement les racines.

```
? lift(nfroots(K, subst(K.pol,x,z)))
% = [-x + 1, x]
```

Nous voyons que K a un automorphisme donné par $\alpha \mapsto 1 - \alpha$.

Éléments et idéaux

Éléments d'un corps de nombres

On a vu qu'on pouvait représenter les éléments d'un corps de nombres comme polynômes en α. On peut aussi utiliser des combinaisons linéaires de la base d'entiers. On change de représentation avec `nfalgtobasis` et `nfbasistoalg`.

```
? nfalgtobasis(K,x^2)
%14 = [1, 2, 1, 0]~
```

Interprétation : $\alpha^2 = 1 \cdot 1 + 2 \cdot w + 1 \cdot \alpha + 0 \cdot w\alpha = 1 + 2w + \alpha$.

```
? nfbasistoalg(K,[1,1,1,1]~)
%15 = Mod(1/2*x^3 + 1/2, x^4 - 2*x^3 + x^2 - 5)
```

Interprétation : $1 + w + \alpha + w\alpha = \frac{\alpha^3+1}{2}$.

Éléments d'un corps de nombres : opérations

Les opérations sur les éléments sont les fonctions `nfeltxxxx`, et acceptent les deux représentations.

```
? nfeltmul(K,[1,-1,0,0]~,x^2)
%16 = [-1, 3, 1, -1]~
```

Interprétation : $(1-w)\cdot\alpha^2 = -1+3w+\alpha-w\alpha$.

```
? nfeltnorm(K,x-2)
%17 = -1
? nfelttrace(K,[0,1,2,0]~)
%18 = 2
```

Interprétation : $N_{K/\mathbb{Q}}(\alpha-2) = -1$, $\mathrm{Tr}_{K/\mathbb{Q}}(w+2\alpha) = 2$.

Polynômes caracteristiques et minimaux

Nous pouvons calculer les polynômes caractéristiques et minimaux d'un polynôme :

```
? charpoly(nfbasistoalg(K,[1,2,0,0]~))
% = x^4 - 10*x^2 + 25
? minpoly(nfbasistoalg(K,[1,2,0,0]~))
% = x^2 - 5
```

Les deux polynômes sont identiques sauf si l'élément appartient à un sous-corps.

Plongements

Nous pouvons calculer les plongements réels et complexes d'un élément avec `nfeltembed`.

```
? nfeltembed(K,x^3+x)
% = [-2.3250207137883080622303986499385818825,
%    11.033224646287677151457919656132410589,
%    -2.3541019662496845446137605030969143532
%    - 0.33268570002014959478470322160341519810*I]
```

Décomposition des nombres premiers

On décompose un nombre premier avec `idealprimedec` :

```
? dec = idealprimedec(K,5);
? #dec
%20 = 2
? [pr1,pr2] = dec;
```

Interprétation : $\mathbb{Z}_K$ a deux idéaux premiers au-dessus de 5, qu'on appelle $\mathfrak{p}_1$ et $\mathfrak{p}_2$.

```
? pr1.f
%22 = 1
? pr1.e
%23 = 2
```

$\mathfrak{p}_1$ est de degré résiduel 1 et d'indice de ramification 2.

Décomposition des nombres premiers

```
? pr1.gen
%24 = [5, [-1, 0, 1, 0]~]
```

$\mathfrak{p}_1$ a pour générateurs 5 et $-1 + 0 \cdot w + \alpha + 0 \cdot w\alpha$, c'est-à-dire $\mathfrak{p}_1 = 5\mathbb{Z}_K + (\alpha - 1)\mathbb{Z}_K$.

```
? pr2.f
%25 = 1
? pr2.e
%26 = 2
```

$\mathfrak{p}_2$ est aussi de degré résiduel 1 et d'indice de ramification 2.

Réduction modulo un idéal premier

Nous pouvons calculer l'image d'un élément dans le corps résiduel avec `nfmodpr`.

```
? p11 = idealprimedec(K,11)[1]; p11.f
% = 2
? modpr = nfmodprinit(K,p11,v);
? a = nfmodpr(K,x^2+x+1,modpr)
% = 2*v + 5
? a^11
% = 9*v + 7
? a^121
% = 2*v + 5
```

Réciproquement nous pouvons relever les éléments du corps résiduel avec `nfmodprlift`.

```
? nfmodprlift(K,a,modpr)
% = 2*x + 5
```

Idéaux

Un idéal arbitraire est représenté par sa forme normale de Hermite (HNF) par rapport à la base d'entiers. On peut obtenir cette forme avec `idealhnf`.

```
? idealhnf(K,pr1)
%27 =
% [5 3 4 3]
% [0 1 0 0]
% [0 0 1 0]
% [0 0 0 1]
```

Interprétation : $\mathfrak{p}_1$ s'écrit

$$\mathfrak{p}_1 = \mathbb{Z}\cdot 5 + \mathbb{Z}\cdot(w+3) + \mathbb{Z}\cdot(\alpha+4) + \mathbb{Z}\cdot(w\alpha+3).$$

Idéaux

```
? a = idealhnf(K,[23, 10, -5, 1]~)
%28 =
% [260   0 228 123]
% [  0 260 123 105]
% [  0   0   1   0]
% [  0   0   0   1]
```

On obtient la HNF de l'idéal $\mathfrak{a} = (23 + 10w - 5\alpha + w\alpha)$.

```
? idealnorm(K,a)
%29 = 67600
```

On a $N(\mathfrak{a}) = 67600$.

Idéaux : opérations

Les opérations sur les idéaux sont les fonctions `idealxxxx` et acceptent des HNF, des structures représentant des idéaux premiers (sortie de `idealprimedec`), et des éléments.

```
? idealpow(K,pr2,3)
%30 =
% [25 15 21 7]
% [ 0  5  2 4]
% [ 0  0  1 0]
% [ 0  0  0 1]
? idealnorm(K,idealadd(K,a,pr2))
%31 = 1
```

On a $\mathfrak{a} + \mathfrak{p}_2 = \mathbb{Z}_K$: les idéaux $\mathfrak{a}$ et $\mathfrak{p}_2$ sont premiers entre eux.

Représentation par deux générateurs

Dans un anneau de Dedekind, tout idéel peut être engendré par deux éléments. avec `idealtwoelt`.

```
? [n,b] = idealtwoelt(K,a)
% = [260, [-32, 123, 1, 0]~]
? idealadd(K,n,b) == a
```

Nous vérifions que nous obtenons bien le bon idéal.

Opérations en relation avec les idéaux premiers

Nous pouvons calculer la valuation d'un idéal en un idéal premier avec `idealval`.

```
? idealval(K,a,pr2)
% = 0
```

L'idéal $\mathfrak{a}$ n'est pas divisible par $\mathfrak{p}_2$.
Nous pouvons tester si l'idéal est premier avec `idealismaximal`.

```
? idealismaximal(K,a)
% = 0
? idealismaximal(K,idealhnf(K,pr1)) != 0
% = 1
```

Si l'ideal est premier, la fonction renvoie la même structure qu'`idealprimedec`.

Idéaux : factorisation

On factorise un idéal en produit d'idéaux premiers avec `idealfactor`. Le résultat est une matrice à deux colonnes, la première contenant les idéaux premiers, la seconde contenant les exposants.

```
? fa = idealfactor(K,a);
? #fa[,1]
%33 = 3
```

L'idéal $\mathfrak{a}$ est divisible par trois idéaux premiers.

```
? [fa[1,1].p, fa[1,1].f, fa[1,1].e, fa[1,2]]
%34 = [2, 2, 1, 2]
```

Le premier est un idéal premier au-dessus de 2, de degré résiduel 2 et non ramifié, et apparaît avec exposant 2.

Idéaux : factorisation

```
? [fa[2,1].p, fa[2,1].f, fa[2,1].e, fa[2,2]]
%35 = [5, 1, 2, 2]
? fa[2,1]==pr1
%36 = 1
```

Le deuxième est $\mathfrak{p}_1$, et il apparaît avec exposant 2.

```
? [fa[3,1].p, fa[3,1].f, fa[3,1].e, fa[3,2]]
%37 = [13, 2, 1, 1]
```

Le troisième est un idéal premier au-dessus de 13, de degré résiduel 2 et non ramifié, et apparaît avec exposant 2.

Restes chinois

On peut appliquer le théorème des restes chinois avec `idealchinese` :

```
? b = idealchinese(K,[pr1,2;pr2,1],[1,-1]);
```

On cherche un élément $b \in \mathbb{Z}_K$ tel que $b = 1 \bmod \mathfrak{p}_1^2$ et $b = -1 \bmod \mathfrak{p}_2$.

```
? nfeltval(K,b-1,pr1)
%39 = 2
? nfeltval(K,b+1,pr2)
%40 = 1
```

On vérifie le résultat en calculant les valuations : $v_{\mathfrak{p}_1}(b-1) = 2$ et $v_{\mathfrak{p}_2}(b+1) = 1$.

Restes chinois avec signes

On peut calculer le signe des plongements réels de b :

```
? nfeltsign(K,b)
%41 = [-1, 1]
```

On a $\sigma_1(b) < 0$ et $\sigma_2(b) > 0$, où σ_1, σ_2 sont les deux plongements réels de K.
On peut demander à `idealchinese` un élément qui, en plus des congruences, soit totalement positif :

```
? c = idealchinese(K,[[pr1,2;pr2,1],[1,1]],[1,-1]);
? nfeltsign(K,c)
%43 = [1, 1]
```

On a bien $\sigma_1(c) > 0$ et $\sigma_2(c) > 0$.

Fonction zêta de Dedekind

On peut évaluer la fonction zêta de Dedekind avec `lfun`.

```
? L = nfinit(x^3-3*x-1);
? L.sign
%45 = [3, 0]
```

L est totalement réel.

```
? lfun(L,2)
%46 = 1.1722471496117109428809260096356285918
? q = bestappr(lfun(L,2)/Pi^6)
%47 = 8/6561
? lfun(L,2)/(Pi^6*q)
%48 = 1.0000000000000000000000000000000000000
```

$\zeta_L(2)$ est un multiple rationnel de π^6 (théorème de Siegel).

Groupes des classes et groupes des unités

bnfinit

Pour faire des calculs de groupes de classes et d'unités dans un corps de nombres, il faut un calcul plus coûteux que celui de `nfinit`. Ce calcul est effectué par `bnfinit` (b = Buchmann).

```
? K2 = bnfinit(K);
? K2.nf == K
%50 = 1
? K2.no
%51 = 1
```

K est principal (no = nombre de classes).

```
? K2.reg
%52 = 1.7763300299706546701307646106399605586
```

On obtient une valeur approchée du régulateur de *K*.

bnfcertify

La sortie de `bnfinit` n'est a priori correcte que sous GRH (Hypothèse de Riemann Généralisée). On peut la certifier inconditionnellement au prix d'un calcul supplémentaire avec `bnfcertify`.

```
? bnfcertify(K2)
%52 = 1
```

Le calcul est maintenant certifié ! Si `bnfcertify` renvoie `0`, on a trouvé un contre-exemple à GRH (ou plus probablement un bug dans PARI/GP) !

bnfinit : unités

```
? lift(K2.tu)
%54 = [2, -1]
? K2.tu[1]==nfrootsof1(K)[1]
%55 = 1
```

K a deux racines de l'unité (tu = torsion units), ± 1. On peut également les calculer avec `nfrootsof1`.

```
? lift(K2.fu)
%56 = [1/2*x^2-1/2*x-1/2, 1/2*x^3-3/2*x^2+3/2*x-1]
```

La partie libre de $\mathbb{Z}_K^\times$ est engendrée par $\frac{\alpha^2-\alpha-1}{2}$ et $\frac{\alpha^3-3\alpha^2+3\alpha-2}{2}$ (fu = fundamental units).

bnfinit : formule analytique du nombre de classes

```
? lfun(K,1+x+O(x^2))
%57 = 0.50228472605280111386617636567964565169*x^-1
  + O(x^0)
? res = polcoeff(lfun(K,1+x+O(x^2)),-1)
%58 = 0.50228472605280111386617636567964565169
```

On calcule une valeur approchée du résidu de $\zeta_K(s)$ en $s = 1$.

```
? {2^K2.r1*(2*Pi)^K2.r2*K2.no*K2.reg/
    (K2.tu[1]*sqrt(abs(K2.disc))*res)}
%59 = 0.99999999999999999999999999999999999999
```

On vérifie numériquement la formule analytique du nombre de classes.

Groupe des classes

```
? L = bnfinit(x^3 - x^2 - 54*x + 169);
? L.cyc
%61 = [2, 2]
```

$\mathcal{C}\ell(L) \cong \mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}$

```
? L.gen
%62 = [[5,0,0;0,5,3;0,0,1],[5,0,3;0,5,2;0,0,1]]
```

Générateurs du groupe des classes, donnés comme idéaux sous forme HNF.

Tester si un idéal est principal

On peut tester si un idéal est principal avec `bnfisprincipal` :

```
? pr = idealprimedec(L,13)[1]
? [dl,g] = bnfisprincipal(L,pr);
? dl
%65 = [1, 0]~
```

`bnfisprincipal` exprime la classe de l'idéal en fonction des générateurs du groupe des classes (logarithme discret). Ici, l'idéal `pr` est dans la même classe que le premier générateur. En particulier, il n'est pas principal, mais son carré l'est.

Tester si un idéal est principal

```
? g
%66 = [-2/5, 1/5, 0]~
? {idealhnf(L,pr) == idealmul(L,g,
   idealfactorback(L,L.gen,dl))}
%67 = 1
```

La seconde composante de la sortie de `bnfisprincipal` est un élément $g \in L$ qui engendre l'idéal principal restant. (idealfactorback = inverse de idealfactor = $\prod_i \texttt{L.gen[i]}^{\texttt{dl[i]}}$)

Calculer un générateur d'un idéal principal

On sait que `pr` est de 2-torsion ; calculons un générateur de son carré :

```
? [dl2,g2] = bnfisprincipal(L,idealpow(L,pr,2));
? dl2
%69 = [0, 0]~
```

L'idéal est bien principal (trivial dans le groupe des classes).

```
? g2
%70 = [1, -1, -1]~
? idealhnf(L,g2) == idealpow(L,pr,2)
%71 = 1
```

`g2` est un générateur de $\mathtt{pr}^2$.

Application : bnfisintnorm

On peut utiliser cela pour trouver des solutions dans $\mathbb{Z}_L$ d'équations aux normes avec `bnfisintnorm` :

```
? bnfisintnorm(L,5)
%72 = []
```

Il n'y a pas d'élément de norme 5 dans $\mathbb{Z}_L$.

```
? bnfisintnorm(L,65)
%73 = [x^2 + 4*x - 36, -x^2 - 3*x + 39, -x + 2]
```

Il y a trois éléments de $\mathbb{Z}_L$ de norme 65, à multiplication près par les éléments de $\mathbb{Z}_L^\times$ de norme positive.

Exprimer une unité en fonction des générateurs

```
? u = [0,2,1]~;
? nfeltnorm(L,u)
%75 = 1
```

On a trouvé une unité $u \in \mathbb{Z}_L^\times$.

```
? bnfisunit(L,u)
%76 = [1, 2, 1]~
? lift(L.fu)
%77 = [-x^2 - 4*x + 34, x - 4]
? lift(L.tu)
%78 = [2, -1]
```

On l'exprime en fonction des générateurs avec `bnfisunit` :
$u = (-\alpha^2 - 4\alpha + 34) \cdot (\alpha - 4)^2 \cdot (-1)^1$.

Grandes unités fondamentales

Par défaut, `bnfinit` ne calcule des unités fondamentales que si elles sont petites.

```
? M = bnfinit(x^2-3019);
? M.fu
%80 = 0 \\valeur sentinelle: non calculées
```

On peut forcer le calcul des unités avec `bnfinit(,1)`.

```
? M = bnfinit(x^2-3019,1);
? lift(M.fu)
%82 = [213895188053752098546071055592725565706690
?  871236169789*x - 117525625416599410184425264152
?  375394603920948258603143330]
```

Note : on peut manipuler les très grandes unités avec `bnfunits` (utilisation avancée, non décrite ici).

Questions ?

À vos claviers !