Théorie algébrique des nombres

Aurel Page

26 novembre 2021

Table des matières

1 Bases de GP

On affecte une variable avec =.

[1]:
```
a = 2 \\ceci est un commentaire couvrant la fin de la ligne
```

[1]:
```
2
```

Toute variable non affectée peut être utilisée comme une variable polynomiale.

[2]:
```
P = x^3 /* ceci est un commentaire délimité à gauche et à droite
 et qui couvre plusieurs lignes */ + 3*x + 7*a
```

[2]:
```
x^3 + 3*x + 14
```

Les valeurs de vérité vrai/faux sont représentées par les entiers 1/0.

[3]:
```
issquarefree(P)
```

[3]:
```
1
```

On peut obtenir la documentation sur une fonction avec ? (aide courte) ou ?? (aide longue).

[3]:
```
?factor
??bestappr
```

```
factor(x,{D}): factorization of x over domain D. If x and D are both integers,
return partial factorization, using primes < D.

bestappr(x, {B}):

   Using variants of the extended Euclidean algorithm, returns a
rational
approximation a/b to x, whose denominator is limited by B, if present.  If B is
omitted, returns the best approximation affordable given the input accuracy; if
you are looking for true rational numbers, presumably approximated to
sufficient accuracy, you should first try that option.  Otherwise, B must be a
positive real scalar (impose 0 < b <= B).

*  If  x  is  a  t_REAL  or  a t_FRAC,  this function
uses continued fractions.
```

```
   ? bestappr(Pi, 100)
   %1 = 22/7
   ? bestappr(0.1428571428571428571428571429)
   %2 = 1/7
   ? bestappr([Pi, sqrt(2) + 'x], 10^3)
   %3 = [355/113, x + 1393/985]
/*-- (type RETURN to continue) --*/
   By definition, a/b is the best rational approximation to x if |b x - a|
< |v
x  - u| for all integers (u,v) with 0 < v <= B.    (Which implies that
n/d is a
convergent of the continued fraction of x.)

* If x is a t_INTMOD modulo N or a t_PADIC of precision
N = p^k,  this function
performs rational modular reconstruction modulo N. The routine then returns the
unique  rational  number a/b in coprime integers |a| < N/2B and b <= B
which is
congruent  to  x  modulo N.   Omitting B amounts to choosing it of the order of
sqrt{N/2}. If rational reconstruction is not possible (no suitable a/b exists),
returns [].

   ? bestappr(Mod(18526731858, 11^10))
   %1 = 1/7
   ? bestappr(Mod(18526731858, 11^20))
   %2 = []
   ? bestappr(3 + 5 + 3*5^2 + 5^3 + 3*5^4 + 5^5 + 3*5^6 + O(5^7))
   %2 = -1/3
/*-- (type RETURN to continue) --*/
In  most  concrete uses,  B is a prime power and we performed Hensel
lifting to
obtain x.

   The function applies recursively to components of complex objects
(polynomials, vectors,...).  If rational reconstruction fails for even a single
entry, returns [].

   The library syntax is GEN bestappr(GEN x, GEN B = NULL).
```

On peut chercher une chaine de caractères dans la documentation avec `???`.

[3]:
```
???galois
```

```
alggroup           alggroupcenter    bestapprnf        bnfisnorm
bnrgaloisapply     bnrgaloismatrix   bnrisgalois       bnrstark
chargalois         forsubgroup       galoischardet     galoischarpoly
galoischartable    galoisconjclasses galoisexport      galoisfixedfield
galoisgetgroup     galoisgetname     galoisgetpol      galoisidentify
galoisinit         galoisisabelian   galoisisnormal    galoispermtopol
galoissubcyclo     galoissubfields   galoissubgroups   idealfrobenius
idealramgroups     iferr             lfunartin         lfunmf
mfdim              mffields          mfgaloisprojrep   mfgaloistype
mfinit             mfisCM            mfsplit           mspadicL
new_galois_format  nfgaloisapply     nfgaloisconj      nfsubfields
polgalois          polsubcyclo       poltschirnhaus    rnfequation
rnfisnorm          rnfisnorminit     subgrouplist

See also:
  "Artin $L$ functions"
```

Les vecteurs ligne sont délimités par des crochets, leurs composantes par des virgules. Ils peuvent contenir n'importe quels types d'objets.

[4]:
```
v = [-1,y,Mod(2,3),O(x)]
```

[4]:
```
[-1, y, Mod(2, 3), O(x)]
```

Les composantes sont numérotées de 1 au nombre d'éléments `#v`.

[5]:
```
v[#v]
```

[5]:
```
O(x)
```

Les matrices sont délimitées par des crochets, données ligne par ligne ; les composantes d'une ligne sont séparées par des virgules et les lignes par des point-virgules.

[6]:
```
m = [1,2;3,4;7,0;0,-2]
```

[6]:
```
[1  2]

[3  4]

[7  0]

[0 -2]
```

On accède aux composantes avec `[ligne,colonne]`

[7]:
```
m[3,1]
```

[7]:
```
7
```

On transpose les matrices et les vecteurs avec ~.

[8]:
```
m~ * v~
```

[8]:
```
[3*y + Mod(1, 3), (4*y - 2) + O(x)]~
```

La structure de contrôle conditionnelle est codée comme une fonction : `if(condition, code si vrai, code si faux)`

[8]:
```
if(a==0,print("nul"),print(a));
```

```
2
```

De même pour `for`, `while`, et d'autres itérateurs comme `forprime`, `forsquarefree`, `forvec`, etc.

[8]:
```
?for
```

```
for(X=a,b,seq): the sequence is evaluated, X going from a up to b. If b is set
to +oo, the loop will not stop.
```

On peut définir des fonctions ainsi :

[9]:
```
syracuse(n) =
{
    if(n==1,
        1
    ,/*else if*/ n%2==0,
        1+syracuse(n/2)
    ,/*else*/
        1+syracuse(3*n+1)
    )
};
syracuse(97)
```

[9]:
```
119
```

2 Irréductibilité

En GP, on définit un corps de nombres comme $K = \mathbb{Q}[x]/f(x)$ où f est un polynôme irréductible. On teste l'irréductibilité avec `polisirreducible`.

[10]:
```
f = x^4 - 2*x^3 + x^2 - 5;
polisirreducible(f)
```

```
[10]: 1
```

GP connait les polynômes cyclotomiques.

```
[11]: g = polcyclo(30)
```

```
[11]: x^8 + x^7 - x^5 - x^4 - x^3 + x + 1
```

3 Nombres algébriques

On peut effectuer des opérations simples dans $K = \mathbb{Q}[x]/f(x) = \mathbb{Q}(\alpha)$ où $f(\alpha) = 0$ en utilisant `Mod`.

```
[12]: Mod(x,f)^5
```

```
[12]: Mod(3*x^3 - 2*x^2 + 5*x + 10, x^4 - 2*x^3 + x^2 - 5)
```

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

On peut vérifier que les racines de g sont des racines 30èmes de l'unité.

```
[13]: lift(Mod(x,g)^15)
```

```
[13]: -1
```

On a utilisé `lift` pour rendre la sortie plus lisible.

4 Simplification de corps de nombres

Parfois, il peut être utile de trouver un polynôme de définition plus simple pour le même corps de nombres : c'est l'objet de la fonction `polredbest`.

```
[14]: {h = x^5 + 7*x^4 + 22550*x^3 - 281686*x^2 - 85911*x + 3821551};
polredbest(h)
```

```
[14]: x^5 - x^3 - 2*x^2 + 1
```

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

5 Initialisation d'un corps de nombres

La plupart des opérations sur un corps de nombres utilise une structure de données, qui est calculée par la fonction d'initalisation `nfinit`.

[15]:
```
K = nfinit(f);
```

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

[16]:
```
K.pol
```

[16]:
```
x^4 - 2*x^3 + x^2 - 5
```

6 Information calculée

La structure contient plusieurs informations de base sur le corps K.

[17]:
```
K.sign
```

[17]:
```
[2, 1]
```

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

[18]:
```
K.disc
```

[18]:
```
-1975
```

Le corps K est de discriminant -1975.

[19]:
```
K.p
```

[19]:
```
[5, 79]
```

Le corps K est ramifié exactement en 5 et 79.

[20]:
```
w = K.zk[2];
K.zk
```

[20]:
```
[1, 1/2*x^2 - 1/2*x - 1/2, x, 1/2*x^3 - 1/2*x^2 - 1/2*x]
```

L'anneau des entiers de K est

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

7 Eléments d'un corps de nombres

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

[21]:
```
nfalgtobasis(K,x^2)
```

[21]:
```
[1, 2, 1, 0]~
```

Interprétation :

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

[22]:
```
nfbasistoalg(K,[1,1,1,1]~)
```

[22]:
```
Mod(1/2*x^3 + 1/2, x^4 - 2*x^3 + x^2 - 5)
```

Interprétation :

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

Les opérations sur les éléments sont effectuées par les fonctions `nfeltxxx`, qui acceptent les deux représentations en entrée.

[23]:
```
nfeltmul(K,[1,-1,0,0]~,x^2)
```

[23]:
```
[-1, 3, 1, -1]~
```

Interprétation :

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

[24]:
```
nfeltnorm(K,x-2)
```

[24]:
```
-1
```

Interprétation :

$$N_{K/\mathbb{Q}}(\alpha - 2) = -1.$$

[25]:
```
nfelttrace(K,[0,1,2,0]~)
```

[25]:
```
2
```

Interprétation :

$$\mathrm{Tr}_{K/\mathbb{Q}}(w + 2\alpha) = 2.$$

8 Décomposition des nombres premiers

On calcule la décomposition des nombres premiers avec la fonction `idealprimedec`.

[26]:
```
dec = idealprimedec(K,5);
#dec
```

[26]:
```
2
```

Interprétation : $\mathbb{Z}_K$ possède deux idéaux premiers au-dessus de 5. Appelons-les $\mathfrak{p}_1$ et $\mathfrak{p}_2$.

[27]:
```
[pr1,pr2] = dec;
[pr1.f,pr1.e]
```

[27]:
```
[1, 2]
```

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

[28]:
```
pr1.gen
```

[28]:
```
[5, [-1, 0, 1, 0]~]
```

Interprétation : $\mathfrak{p}_1$ est engendré par 5 et $-1 + 0 \cdot w + \alpha + 0 \cdot w\alpha$, c'est-à-dire qu'on a $\mathfrak{p}_1 = 5\mathbb{Z}_K + (\alpha - 1)\mathbb{Z}_K$.

[29]:
```
[pr2.f,pr2.e]
```

[29]:
```
[1, 2]
```

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

9 Idéaux

On représente les idéaux arbitraires par leur forme normale de Hermite (HNF) par rapport à la base d'entiers. On peut obtenir cette forme avec la fonction `idealhnf`.

[30]:
```
idealhnf(K,pr1)
```

[30]:
```
[5 3 4 3]

[0 1 0 0]

[0 0 1 0]

[0 0 0 1]
```

Interprétation : $\mathfrak{p}_1$ peut être décrit comme

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

[31]:
```
a = idealhnf(K,[23, 10, -5, 1]~)
```

[31]:

```
[260   0 228 123]

[  0 260 123 105]

[  0   0   1   0]

[  0   0   0   1]
```

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

[32]:
```
idealnorm(K,a)
```

[32]: 67600

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

Les opérations sur les idéaux sont effectuées par les fonctions `idealxxx`, qui acceptent comme entrées des formes HNF, des idéaux premiers comme calculés par `idealprimedec`, et des éléments du corps (interprétés comme des idéaux principaux).

[33]:
```
idealpow(K,pr2,3)
```

[33]:

```
[25 15 21 7]

[ 0  5  2 4]

[ 0  0  1 0]

[ 0  0  0 1]
```

On a calculé la HNF de $\mathfrak{p}_2^3$.

[34]:
```
idealnorm(K,idealadd(K,a,pr2))
```

[34]: 1

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

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

[35]:
```
fa = idealfactor(K,a);
matsize(fa) \\ [nombre de lignes, nombre de colonnes]
```

[35]:
```
[3, 2]
```

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

[36]:
```
[fa[1,1].p, fa[1,1].f, fa[1,1].e, fa[1,2]]
```

[36]:
```
[2, 2, 1, 2]
```

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

[37]:
```
[fa[2,1].p, fa[2,1].f, fa[2,1].e, fa[2,2]]
```

[37]:
```
[5, 1, 2, 2]
```

Le second est un idéal premier au-dessus de 5 : on sait donc que c'est $\mathfrak{p}_1$ ou $\mathfrak{p}_2$.

[38]:
```
fa[2,1]==pr1
```

[38]:
```
1
```

C'est $\mathfrak{p}_1$.

[39]:
```
[fa[3,1].p, fa[3,1].f, fa[3,1].e, fa[3,2]]
```

[39]:
```
[13, 2, 1, 1]
```

Le troisième est un idéal premier au-dessus de 13, non-ramifié de degré résiduel 2, et apparaît avec un exposant 1.

10 Restes chinois

On peut utiliser le théorème des restes chinois avec `idealchinese`.

[40]:
```
b = idealchinese(K,[pr1,2;pr2,1],[1,-1]);
```

On recherche un élément $b \in \mathbb{Z}_K$ tel que $b = 1 \bmod \mathfrak{p}_1^2$ et $b = -1 \bmod \mathfrak{p}_2$. Vérifions la sortie en calculant des valuations.

[41]:
```
nfeltval(K,b-1,pr1)
```

[41]:
```
2
```

On a $v_{\mathfrak{p}_1}(b-1) = 2$.

[42]:
```
idealval(K,b+1,pr2)
```

[42]:
```
1
```

On a $v_{\mathfrak{p}_2}(b+1) = 1$.

On peut calculer le signe des plongements réels de b avec `nfeltsign`.

[43]:
```
nfeltsign(K,b)
```

[43]:
```
[-1, 1]
```

On a $\sigma_1(b) < 0$ et $\sigma_2(b)$, où σ_1, σ_2 sont les deux plongements réels de K.

On peut demander à `idealchinese` de calculer un élément qui, en plus de satisfaire les congruences, est totalement positif.

[44]:
```
c = idealchinese(K,[[pr1,2;pr2,1],[1,1]],[1,-1]);
nfeltsign(K,c)
```

[44]:
```
[1, 1]
```

On a bien $\sigma_1(c) > 0$ et $\sigma_2(c) > 0$. On peut également vérifier les signes en calculant tous les plongements réels et complexes.

[45]:
```
nfeltembed(K,c)
```

[45]:
```
[8.33454612818026696224192142354332194388,
10.283487860569627885962665398932418679, 8.190983005625052575897706582817180941
1 - 2.280261710651260054736978882921621328 5*I]
```

11 Fonction ζ de Dedekind

On peut évaluer la fonction ζ de Dedekind avec `lfun` (cf aussi l'atelier sur les fonctions L).

[46]:
```
L = nfinit(x^3-3*x-1);
L.sign
```

[46]:
```
[3, 0]
```

Le corps L est totalement réel.

[47]:
```
lfun(L,2)
```

[47]:
```
1.1722471496117109428809260096356285918
```

On a calculé une approximation de $\zeta_L(2)$.

[48]:
```
q = bestappr(lfun(L,2)/Pi^6)
```

[48]:
```
8/6561
```

`bestappr` calcule un nombre rationnel de petit dénominateur proche du nombre flottant donné en entrée.

[49]:
```
lfun(L,2)/(Pi^6*q)
```

[49]:
```
1.0000000000000000000000000000000000000
```

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

12 Groupe des classes et unités

Pour obtenir le groupe des classes et le groupe des unités d'un corps de nombres, il faut faire un calcul plus coûteux que celui effectué par `nfinit`. L'information est contenue dans une structure calculée par la fonction `bnfinit` (`b` = Buchmann, inventeur de l'algorithme utilisé).

[50]:
```
K2 = bnfinit(K);
K2.nf == K
```

[50]:
```
1
```

[51]:
```
K2.no
```

[51]:
```
1
```

Le groupe des classes de K est trivial (`no` = nombre de classes).

[52]:
```
K2.reg
```

[52]:
```
1.7763300299706546701307646106399605586
```

On obtient une approximation du régulateur de K.

La sortie de `bnfinit` n'est a priori correcte qu'en supposant l'hypothèse de Riemann généralisée (GRH). On peut la certifiée inconditionnellement avec `bnfcertify`.

[53]:
```
bnfcertify(K2)
```

[53]:
```
1
```

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

[54]:
```
lift(K2.tu)
```

[54]:
```
[2, -1]
```

Le corps K contient deux racines de l'unité (`tu` = torsion units), ± 1.

[55]:
```
K2.tu[1]==nfrootsof1(K)[1]
```

[55]:
```
1
```

On pouvait également les calculer directement avec `nfrootsof1`, qui est beaucoup plus rapide que `bnfinit`.

[56]:
```
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).

[57]:
```
lfun(K,1+x+O(x^2))
```

[57]:
```
0.50228472605280111386617636567964565169*x^-1 + O(x^0)
```

On calcule le développement asymptotique de ζ_K au voisinage de 1 : on a bien un pôle simple.

[58]:
```
res = polcoeff(lfun(K,1+x+O(x^2)),-1)
```

[58]:
```
0.50228472605280111386617636567964565169
```

On extrait une approximation du résidu.

[59]:
```
{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.

[60]:
```
M = bnfinit(x^3 - x^2 - 54*x + 169);
M.cyc
```

[60]:
```
[2, 2]
```

On a

$$\mathrm{Cl}(M) \cong \mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}.$$

En général, les diviseurs élémentaires du groupe de classes sont donnés par ordre décroissant pour la divisibilité.

[61]:
```
M.gen
```

[61]:
```
[[5, 0, 0; 0, 5, 3; 0, 0, 1], [5, 0, 3; 0, 5, 2; 0, 0, 1]]
```

Des générateurs du groupe des classes sont donnés en forme HNF.

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

[62]:
```
pr = idealprimedec(M,13)[1];
[dl,g] = bnfisprincipal(M,pr);
dl
```

[62]:
```
[1, 0]~
```

`bnfisprincipal` exprime la classe de l'idéal en fonction des générateurs du groupe des classes (logarithme discret, DL). Ici, l'idéal $\mathfrak{p}$ est dans la même classe que le premier générateur ; en particulier il n'est pas principal, mais son carré l'est.

[63]:
```
g
```

[63]:
```
[-2/5, 1/5, 0]~
```

La seconde composante de la sortie de `bnfisprincipal` est un élément $g \in M$ qui engendre l'idéal principal restant :

[64]:
```
idealhnf(M,pr) == idealmul(M,g,idealfactorback(M,M.gen,dl))
```

[64]:
```
1
```

`idealfactorback` = inverse of `idealfactor` = $\prod_i$ `L.gen[i]^dl[i]`.

Nous savons que la classe de $\mathfrak{p}$ est un élément de 2-torsion ; calculons un générateur de son carré.

[65]:
```
[dl2,g2] = bnfisprincipal(M,idealpow(M,pr,2));
dl2
```

[65]:
```
[0, 0]~
```

L'idéal est bien principal (classe triviale).

[66]:
```
g2
```

[66]:
```
[1, -1, -1]~
```

g_2 est un générateur de $\mathfrak{p}^2$; vérifions-le :

[67]:
```
idealhnf(M,g2) == idealpow(M,pr,2)
```

[67]:
```
1
```

Intéressons-nous maintenant aux unités de M.

[68]:
```
u = [0,2,1]~;
nfeltnorm(M,u)
```

[68]:
```
1
```

Nous avons trouvé une unité $u \in \mathbb{Z}_M^\times$.

[69]:
```
bnfisunit(M,u)
```

[69]:
```
[1, 2, 1]~
```

`bnfisunit` exprime u en fonction des générateurs de $\mathbb{Z}_M^\times$ calculés par `bnfinit`.

[70]:
```
lift(M.fu)
```

[70]:
```
[-x^2 - 4*x + 34, x - 4]
```

Les unités fondamentales de M calculées sont $-\alpha^2 - 4\alpha + 34$ et $\alpha - 4$.

[71]:
```
lift(M.tu)
```

[71]:
```
[2, -1]
```

Le groupe de torsion est d'ordre 2 engendré par -1. Interprétation de la sortie de `bnfisunit` :

$$u = (-\alpha^2 - 4\alpha + 34)^1 \cdot (\alpha - 4)^2 \cdot (-1)^1.$$

Par défaut, `bnfinit` ne calcule les unités fondamentales que si elles ne sont pas trop grandes.

[72]:
```
N = bnfinit(x^2-3019);
N.fu
```

[72]:
```
0
```

Le 0 est une "valeur sentinelle", qui signale que les unités fondamentales n'ont pas été calculées. On peut forcer leur calcul avec `bnfinit(,1)`.

[73]:
```
N = bnfinit(x^2-3019,1);
lift(N.fu)
```

[73]:
```
[213895188053752098546071055592725565706690871236169789*x -
1175256254165994101844252641523753946039209482586031433 0]
```

13 Pour aller plus loin

— Tutoriels PARI/GP couvrant des fonctionnalités plus avancées.
— SageMath Number Fields pour trouver les fonctionnalités équivalentes.