Computing class groups and class fields

A tutorial

B. Allombert

IMB
CNRS/Université de Bordeaux

14/01/2026

Introduction to `bnfinit`

`bnfinit` initializes the class group and the unit group of a number field.

```
? bnf = bnfinit(x^3-11);
? bnf.pol
%2 = x^3-11
? bnf.cyc
%3 = [2]
? bnf.reg
%4 = 5.587206626060907761855413020519975370 1
? bnf.fu
%5 = [Mod(-2*x^2+4*x+1,x^3-11)]
```

Fundamental units

To compute the fundamental units (needed by `bnfunits` and `bnfisprincipal`) use the flag 1.

```
? bnf = bnfinit(x^3+1048583,1);
? sizebyte(bnf.fu)
%7 = 283432
? bnfunits(bnf)
? sizebyte(bnfunits(bnf))
%9 = 8232
? bnfsignunit(bnf)
%10 = Mat(-1)
```

S-units

`bnfunits` can also be used to compute *S*-units:

```
? bnf = bnfinit(x^3-17);
? bnfunits(bnf)
? bnfunits(bnf)[1]
%13 = [[[2,1,0]~,-1;[-2,1,0]~,1;[-1,1,0]~,2;
%       [0,1,1]~,-4;[2,-1,1]~,2],Mat([-1,1])]
? id = idealprimedec(bnf,2);
? bnfunits(bnf,id)[1]
%15 = [Mat([[2,1,1]~,1]),Mat([[-2,1,0]~,1]),
%      [[2,1,0]~,-1;[-2,1,0]~,1;[-1,1,0]~,2;
%       [0,1,1]~,-4;[2,-1,1]~,2],Mat([-1,1])]
```

Finding relations

Let S be a finite set of prime ideals that generate the class group, and I an ideal. The basic strategy is to look for elements $\alpha \in I$ of small T_2-norm and try to factor them over S:

$$(\alpha) = \prod \mathfrak{p}_i^{a_i}$$

when this happens, then $\prod \mathcal{C}\ell(\mathfrak{p}_i)^{a_i} = 1$ and α is a S-unit.
There are two strategies to chose I.
"small norm relations" : I is a product of at most 2 elements of S.
"random relations": I is a product of at least 3 elements of S with random exponents.
Unfortunately the probability of finding a relation is very hard to estimate.

The Bach constant

The factor basis is chosen as the set of all primes ideal of norm at most $c_1 \log(D)^2$. Increasing c_1 increases the probability of finding relations but increase the number of relations to find and the cost of linear algebra.

Imaginary quadratic fields, random rel. (PARI 2.17)

```
? quadclassunit(1-2^127)
  *** Bach constant: 1.3340686047266399061
  Time factor base: 1
  Time subFBquad = Vecsmall([2,11,13,17,23,31,37,41
  Time powsubFBquad: 1
  KC = 642, need 647 relations
  Time random rel [#rel/#test = 647/450475]: 7681
  #### Tentative class number: 573727530352480587 5
  Time be honest: 340
 %16 = [573727530352480587 5,[573727530352480587 5],
 %       [Qfb(10177,9881,417955152452759240767631187
  ***   last result computed in 10,421 ms.
```

Imaginary quadratic fields, sieving (PARI 2.18)

```
? quadclassunit(1-2^127)
  *** Bach constant: 1.3340686047266399061
  Time factor base: 2
  Time subFBquad = Vecsmall([2,11,13,17,23,31,37,41
  Time powsubFBquad: 1
  KC = 642, need 645 relations
  Time MPQS rel [#rel = 645]: 33
  Time hnfadd: 1808
  #### Tentative class number: 5737275303524805875
  Time be honest: 0
 %17 = [5737275303524805875,[5737275303524805875],
 %       [Qfb(10151,8649,41902567101878935999331914
  ***   last result computed in 1,881 ms.
```

The Bach constant

```
? \g1bnf
? bnf = bnfinit(x^3+nextprime(2^52));
 PREC = 192
 Time nfinit & nfrootsof1: 3
 Floating point bnf: R1 = 1, R2 = 1
 D = 608472288109555787253948025418 67
 LIMC2 = 4591
 Time computing Bach constant: 5
 Time computing inverse of hR: 1
 *** Bach constant: 0.857140923514784325 3
```

The tech parameter

`bnfinit` has a an optional third argument: `tech = [c_1, c_2, nrpid, max_fact, idex, usethr]` that allows to tune the algorithm.
`c_1=c_2` is the Bach constant for the chosen factor basis.

nrpid

`nrpid` can be set to -1 to disable search for small norm relations and only use random relations. Otherwise this is the maximum number of relations by ideals (4 by default).

max_fact

`max_fact` is the number of factorizations to attempt for each ideal before giving up. Increasing this value slows down the processing of each ideal but increases the probability of finding relations. The default value is 500. This is the most useful parameter.

idex

When searching for small norm relations we look in ideals of the shape $\mathfrak{p}^e\mathfrak{q}$ with `e=idex`. This is a middle-ground alternative to switching to random relations with `nrpid=-1`.

parallelism

Set `usethr` to 1 to use the parallel implementation. This requires to deactivate early-abort strategies when searching in ideals.

Hilbert class field

To compute a Hilbert class field, we first need to compute the class group.

```
? \g0bnf
? bnf = bnfinit(a^2-a+50,1);
? bnf.cyc
%22 = [9]
```

The class group is isomorphic to $\mathbb{Z}/9\mathbb{Z}$. We compute a relative defining polynomial for the Hilbert class field with the function `bnrclassfield`.

```
? R = bnrclassfield(bnf)[1]
%23 = x^9 - 24*x^7 + (2*a - 1)*x^6 + 495*x^5
% + (-12*a + 6)*x^4 - 30*x^3 + (18*a - 9)*x^2
% + 18*x + (-2*a + 1)
```

Hilbert class field

Conversely, from an abelian extension, we can recover its corresponding class group `rnfconductor`.

```
? [cond,bnr,subg] = rnfconductor(bnf,R);
? cond
%25 = [[1, 0; 0, 1], []]
? subg
%26 = [9]
```

Here the conductor is trivial, and its norm group is trivial in the class group.

Hilbert class field

We can also ask for an absolute defining polynomial for the Hilbert class field with the optional `flag=2`.

```
? R2 = bnrclassfield(bnf,,2)
%27 = x^18 - 48*x^16 + 1566*x^14 - 23621*x^12
% + 244113*x^10 - 19818*x^8 - 3170*x^6
% + 17427*x^4 - 3258*x^2 + 199
```

Ray class fields

We can also consider class fields with nontrivial conductor.

```
? bnr = bnrinit(bnf,12); bnr.cyc
%28 = [72,2]
```

We can compute in advance the absolute degree, signature and discriminant of the corresponding class field with `bnrdisc`.

```
? [deg,r1,D] = bnrdisc(bnr);
? [deg,r1]
%30 = [288,0]
? D
%31 = 92477896[...538 digits...]84942237696
```

This field is huge!

Ray class fields

For efficiency, the class field is computed as a compositum of several smaller fields.

```
? bnrclassfield(bnr)
%32 = [x^2 - 3, x^8 + (-27*a+24)*x^6
% + (-294*a-3273)*x^4 + (-3*a-3852)*x^2 - 3,
% x^9 - 24*x^7 + (2*a-1)*x^6 + 495*x^5
% + (-12*a+6)*x^4 - 30*x^3 + (18*a-9)*x^2
% + 18*x + (-2*a+1)]
```

We can force the computation of a single polynomial with `flag=1`.

```
? bnrclassfield(bnr,,1)
%33 = [x^144+... huge polynomial ...]
```

Ray class fields

We can also compute a subfield of the ray class field by specifying a subgroup.

```
? bnr = bnrinit(bnf,7);
? bnr.cyc
%35 = [54,3]
? bnrclassfield(bnr,3) \\elementary 3-subextension
%36 = [x^3 + 3*x + (14*a - 7),
% x^3 + (-1008*a - 651)*x + (-1103067*a - 8072813)
```

Without the explicit field

Computing a defining polynomial with `bnrclassfield` can be time-consuming, so it is better to compute the relevant information without constructing the field, if possible.

We already saw the use of `bnrdisc`; we can also compute splitting information without the explicit field.

```
? pr41 = idealprimedec(bnf,41)[1];
? bnrisprincipal(bnr,pr41,0)
%38 = [0,0]~
```

The Frobenius at $\mathfrak{p}_{41}$ is trivial: this prime splits completely in the degree 162 extension (which we did not compute).

Ray class fields

Let's do a full example with an HNF ideal and a subgroup given by a matrix.

```
? bnr = bnrinit(bnf,[102709,43512;0,1]);
? bnr.cyc
%40 = [17010, 27]
? bnrclassfield(bnr,[9,3;0,1]) \\subgroup of index
%41 = [x^9 + (-297*a - 4470)*x^7 + ... ]
```

Modulus with infinite places

If the base field has real places, we can specify the modulus at infinity by providing a list of 0 or 1 of length the number of real embeddings.

```
? bnf=bnfinit(a^2-217,1);
? bnf.cyc
%43 = []
? bnrinit(bnf,1).cyc
%44 = []
? bnrinit(bnf,[1,[1,1]]).cyc
%45 = [2]
```

The field $\mathbb{Q}(\sqrt{217})$ has narrow class number 2.

p-part of the ray class group

`bnfinit` requires the computation of discrete logarithms, which is slow. Often, one only needs the group $\mathcal{C}\ell_f(K)/\mathcal{C}\ell_f(K)^n$ for some small n which is much easier to compute.

```
? bnr = bnrinit(bnf,271*379,,3);
? bnr.cyc
%47 = [3,3,3]
? bnrclassfield(bnr)
%48 = [x^3-1137*x-10991,x^3+...,x^3+...]
```

Multiple subgroups at once

Since `bnrclassfield` can involve costly computations of `bnf`, it is possible to specify several subgroups at once.

```
? S = [s|s<-subgrouplist(bnr,,1),matdet(s)==3];
? L = [bnrclassfield(bnr,s,1)|s<-S];
? L == bnrclassfield(bnr,S,1)
%51 = 1
```