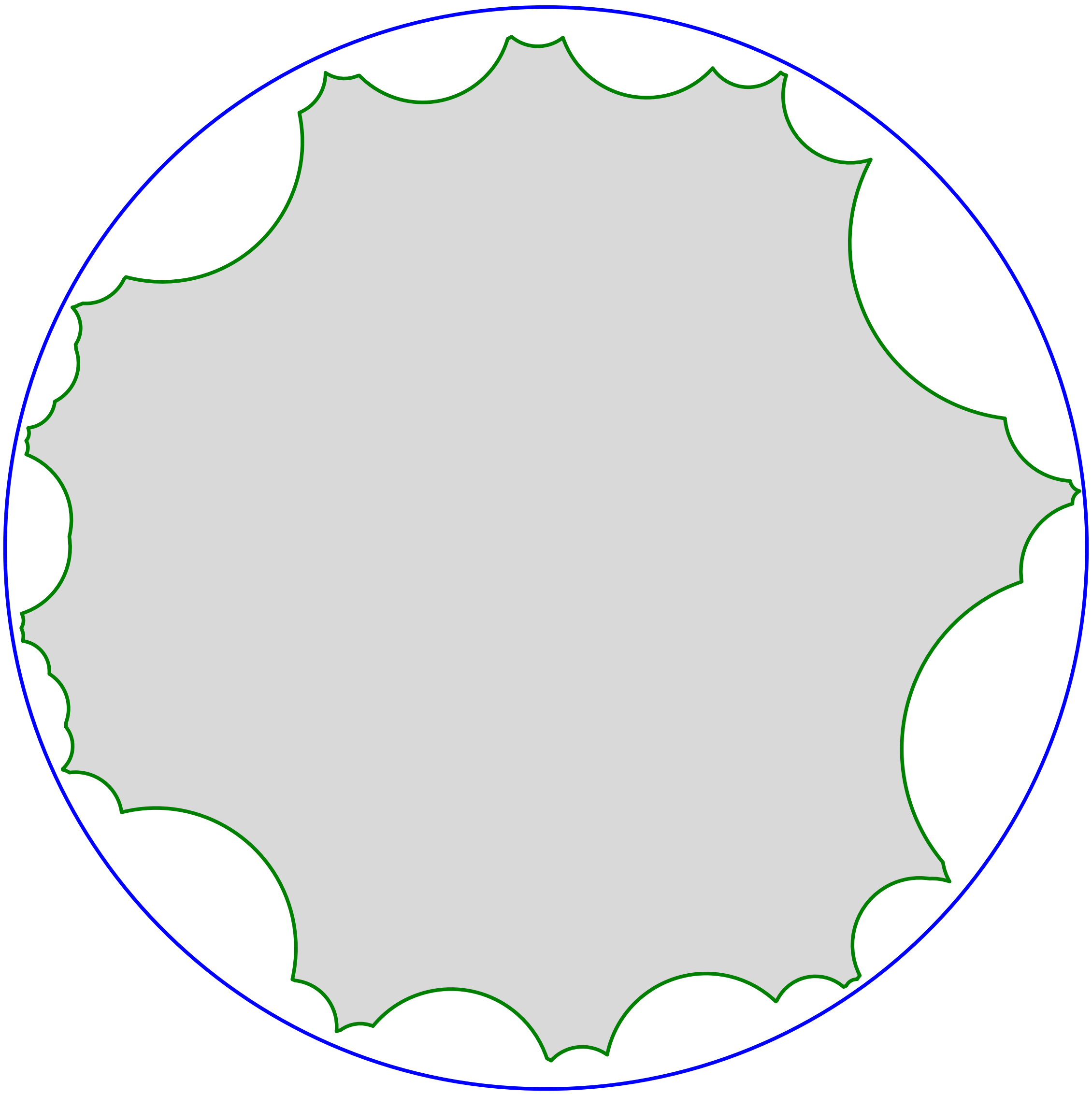
Computing Fundamental Domains for Arithmetic Fuchsian Groups in PARI/GP - User's Manual



James Rickards

University of Colorado Boulder

E-mail: `james.rickards@colorado.edu`

Website GitHub

Contents

1 Introduction

Quaternion algebras over number fields are implemented as part of the algebras package in PARI/GP. This guide is intended as an introduction to how to initialize and use quaternion algebras in PARI/GP, as well as how to compute fundamental domains for Arithmetic Fuchsian groups with the additional package [Ric22].

1.1 Timings

Make table, talk about Voight/Page. Add citations for me and them.

1.2 Installation instructions

Say how to download, install, and use it.

1.3 Warnings

Will be integrated, that may change things, only works on certain versions.

2 Quaternion Algebras in PARI/GP

2.1 Initializing a quaternion algebra

Let F be a number field, and $a, b \in F^{\times}$. The quaternion algebra $\left(\frac{a,b}{F}\right)$ is the $F-$vector space with basis $1, i, j, k$, with multiplication defined by

$$i^2 = a, \quad j^2 = b, \quad k = ij = -ji.$$

2.1.1 Initialize by a, b, F

In gp, it is easy to initialize the algebra given a, b, for example:

```
F=nfinit(y^2-y+1);
A=alginit(F, [y, y-7]);
```

initializes $A = \left(\frac{y,y-7}{F}\right)$, where $F = \mathbb{Q}(y)$ with $y^2 - y + 1 = 0$.

Warnings:

- The variable you use in defining F must be *lower* in priority than the variable used to define A, which is `'x` by default (can be changed by adding a third input to `alginit`). I suggest using `'y` for F.
- The input a must NOT be a square in F. The reason for this is PARI/GP lets $L = F(\sqrt{a})$, and stores an element as $[e, f]$, representing $e + fj$ for $e, f \in L$ (see Section 2.2 for more on this). There is no way around this limitation, so if a happens to be a square, then you need to swap a, b or modify them suitably.
- Both a and b must not only be integral, but have integral coefficients! If you initialize A by ramification, then this extra limitation is no longer present. This will hopefully be fixed in the future.

2.1.2 Initialize by ramification

Let $\mathrm{Pl}(F)$ be the set of places of F, and for $v \in \mathrm{Pl}(F)$, we say that A is ramified at v if

$$A_v = A \otimes_F F_V$$

is a division ring. Otherwise, A is split/unramified at v.

Every quaternion algebra over F is ramified at a finite even-cardinality set of places. Furthermore, every such set of places corresponds to a unique isomorphism class of quaternion algebras over F. We can initialize a quaternion algebra by specifying the set of ramification places, though it is a little more complicated. All complex places are split, so we only need to specify the splitting behaviour on a set of finite primes, as well as the infinite real places.

Given the number field F, the command `F.roots` gives the roots of the defining polynomial, with the r_1 real roots coming first. When you specify the infinite ramification, you give an r_1-length vector of 0's and 1's, where 1 =ramification, and the order corresponds to the ordering of `F.roots`.

For the finite ramification, you supply two vectors: a vector of prime ideals, and a same-length vector of 0's and 1's, where again, 1 =ramification. The total number of 1's across both vectors must be even. For example:

```
F=nfinit(y^5-y^4-3*y^2+1);\\3 real places, approximately -0.539, 0.564, 1.817
I1=idealprimedec(F, 2)[1];\\A prime ideal lying above 2
I2=idealprimedec(F, 7)[1];\\A prime ideal lying above 7
I3=idealprimedec(F, 11)[1];\\A prime ideal lying above 11
pfin=[I1, I2, I3];
ramfin=[1, 0, 1];\\Ramification at I1 and I3
raminf=[1, 1, 0];\\Ramification at the first two infinite places
A=alginit(F, [2, [pfin, ramfin], raminf]);\\Initialize by ramification.
```

This code initializes the quaternion algebra over F ramified at `I1`, `I3`, and the first two infinite places. The ideal `I2` was of no use, and was included to demonstrate that you may include extraneous places in the initialization. The input of 2 is to specify that quaternion algebras have degree 2 (`alginit` can create more general central simple algebras).

Warnings:

- As before, the variable used in F must have lower priority than the variable in A.
- In version 2.13.3 and earlier, there was a rare bug that could occur. If F had at least 3 real places and A was unramified at all finite places, then the only choice of `raminf` that worked was `[1,1,...,1,0,0,...,0]`, i.e. all the ramified real places came first. All other infinite ramification choices (e.g. `[1, 0, 1]`, `[0, 1, 1]`, etc.) would encounter an infinite loop. This bug has been fixed in version 2.13.4 and beyond.

2.1.3 Retrieving a, b

Given an initialized algebra A, it is easy to retrieve b, but not as easy to retrieve a. We can use `algb(A)` to retrieve b, which outputs b as an element of $\mathbb{Q}[y]/f(y)$, where we initialized F with $f(y)$. For example, if $f(y) = y^2 - y + 1$ and $A = \left(\frac{y, y-7}{F}\right)$, then we get

```
algb(A)=Mod(y-7,y^2-y+1)
```

You may want to lift this (using `lift`) to remove the modulus.

To retrieve a we need a bit more code. The following snippet will work:

```
L=algsplittingfield(A);\\Retrieve L=F(sqrt(a))
a=-subst(L.pol, 'x, 0);\\If you initialized A with a different variable to 'x,
    replace 'x by that variable
```

2.2 Elements of quaternion algebras

If you have never used the algebras package before, then this is likely the most confusing part. PARI/GP uses two main representations of elements, the "algebraic" and the "basis" representations, and *neither* is the traditional $[1, i, j, k]$ basis representation! In the package [Ric22], I have included methods to translate elements to and from the traditional representation.

The nomenclature of the translation methods are:

`algalgtobasis, algbasistoalg, alg1ijktoalg, algbasisto1ijk,`

etc. (the first two methods are in PARI/GP, and the last two are in the extra package).

2.2.1 Algebraic representation

Assume that F was initialized with the variable `'y`, and $A = \left(\frac{a,b}{F}\right)$ with the variable `'x`. The algebra stores the splitting field $L = F(\sqrt{a})$ using $x = \sqrt{a}$, and the algebraic representation of an element α is a length 2 column vector:

$$\alpha = [u, v]\sim \text{ means } \alpha = u + jv, \text{ where } u, v \in L.$$

Note that the j is on the other side of v to the North American convention! In particular, if $u = e + fi$ ($i = \sqrt{a} = x$), and $v = g + hi$, then

$$\alpha = e + fi + j(g + hi) = e + fi + gj - hk.$$

Using one of the methods to translate an element to the algebraic representation will typically produce an element with a lot of `Mod`s, as this is how PARI/GP likes to store elements of number fields. For example,

```
F=nfinit(y^2-3);
A=alginit(F, [y, 2*y-3]);
alpha=algbasistoalg(A, [1, 1, 0, 0, 1, 0, -1, 1]~);
```

produces the element

```
[Mod(Mod(1/6*y + 3/2, y^2 - 3)*x + Mod(y + 1, y^2 - 3), x^2 + Mod(-y, y^2 - 3)), Mod(Mod
    (1/6*y + 1/2, y^2 - 3)*x + Mod(1/3*y + 2, y^2 - 3), x^2 + Mod(-y, y^2 - 3))]~.
```

Use `liftall` to eliminate all the modului, and get the much simpler looking

```
[(1/6*y + 3/2)*x + (y + 1), (1/6*y + 1/2)*x + (1/3*y + 2)]~.
```

2.2.2 Basis representation

A quaternion algebra A comes with a "natural order": let $L = F(\sqrt{a})$, and then $\mathcal{O}_L \oplus j\mathcal{O}_L$ is an order (as b was necessarily integral). By taking a $\mathbb{Z}-$basis of $\mathcal{O}_L$, we obtain a $\mathbb{Z}-$basis of this order. When you initialize an order, PARI/GP also computes a maximal order O_0 which contains the natural order. You can also choose to have `alginit` not compute a maximal order, in which case O_0 stores the natural order. If you don't need the maximal order and are working with an extremely large algebra, then this is a good idea.

If $n = [F : \mathbb{Q}]$, then this basis has length $4n$, and we store an element of A as a length $4n$ column vector of coefficients. The basis representation is the most common form outputted by algebra methods. You can retrieve the basis of O_0 in terms of the natural basis by calling `algbasis(A)`, where the columns are the coefficients.

2.2.3 1ijk representation

This representation is *not* built into PARI, but instead is in the Fundamental Domains package. The element $\alpha \in \left(\frac{a,b}{F}\right)$ is represented as a 4-dimensional column vector `[e, f, g, h]`, where $e, f, g, h \in F$ and

$$\alpha = e + fi + gj + hk.$$

This representation is only there to help input data from a problem or output data into a more palatable format. You should only use this representation at the start or end of a computation, as the PARI/GP library does not handle it.

2.2.4 Basic operations on elements

The normal `+,*,/,^` symbols do not work on elements of quaternion algebras. Instead, you should use

```
algadd(A, elt1, elt2);
algsub(A, elt1, elt2);
algneg(A, elt1); \\Only useful for the algebraic representation
algmul(A, elt1, elt2);
algsqr(A, elt1);
algpow(A, elt1, -10);
alginv(A, elt1);
algdivr(A, elt1, elt2); \\Returns elt1*elt2^(-1)
algdivl(A, elt1, elt2); \\Returns elt1^(-1)*elt2
algnorm(A, elt1); \\Reduced norm of elt1, an element of F.
algnorm(A, elt1, 1); \\Absolute norm of elt1, an element of Q.
algtrace(A, elt1); \\Reduced trace of elt1, an element of F.
algtrace(A, elt1, 1); \\Absolute trace of elt1, an element of Q.
```

The elements can be in either the algebraic or basis representations (mixed is fine). The output will be in basis form, unless all input elements were in algebraic form.

3 Fundamental domain background

3.1 Setup

Let F be a totally real number field, and let $A = \left(\frac{a,b}{F}\right)$ be split at a *unique* real place. If $\sigma_1, \sigma_2, \ldots, \sigma_n$ are the embeddings of $F \Rightarrow \mathbb{R}$, then this is equivalent to

$$\sigma_i(a) < 0 \text{ and } \sigma_i(b) < 0 \text{ for exactly } n - 1 \text{ choices of } i.$$

Let O be an order in A, and define

$$N_{A^\times}(\mathrm{O}) := \{x \in A^\times : x\mathrm{O}x^{-1} = \mathrm{O}\}$$

to be the normalizer of O (see Section 3.3 and Section 28.9 of [Voi21] for more on this group). Define

$$\mathrm{O}^1 = \{x \in \mathrm{O} : \mathrm{nrd}(x) = 1\}$$

to be the set of elements of norm 1. Let $\tilde{\Gamma}$ be any group for which

$$\mathrm{O}^1 \leq \tilde{\Gamma} \leq N^+_{A^\times}(\mathrm{O}),$$

where the + indicates that we only keep the elements x with $\mathrm{nrd}(x)$ being totally positive.

Given any element $x \in \tilde{\Gamma}$, the unique split real place maps x to an element of $\mathrm{Mat}(2, \mathbb{R})$ with positive determinant, which can be uniquely scaled to live in $\mathrm{SL}(2, \mathbb{R})$. By quotienting by $\{\pm 1\}$, the image of $\tilde{\Gamma}$ in $\mathrm{PSL}(2, \mathbb{R})$ is denoted by Γ.

This group Γ is a discrete subgroup of $\mathrm{PSL}(2, \mathbb{R})$, and is called an Arithmetic Fuchsian group. The package can compute a fundamental (Dirichlet) domain for Γ when O is an Eichler order (it can work in some instances for non-Eichler orders, but this behaviour is not guaranteed).

3.2 Dirichlet domains

Talk about them, elements generating sides, side pairing.

3.3 Structure of the normalizer group

Talk about it for Eichler orders, cite Voight, 2-group.

4 Computing the fundamental domain

4.1 Finding fields and algebras

To begin, we need to specify a totally real number field F. These can be found in a few places, including:

- On John Voight's website, where tables of fields with small discriminant are listed;
- On LMFDB [LMF23], where you can search for number fields with a variety of properties.

4.2 Specifying Gamma

General input, then explain them.

4.3 Other options

Changing p Displaying partial output

5 Computations using the fundamental domain

Once you have a fundamental domain, you can compute many things, including the signature, a group presentation with a minimal set of generators, closed geodesics, or a visualization of it. These three things are currently implemented.

For the rest of this section, it's assumed that we have a quaternion algebra $A = \left(\frac{a,b}{F}\right)$ with fundamental domain U.

5.1 Signature

Call `algfdomsignature(U)` to compute the signature. The format is $[g, V, s]$, where g is the genus, $V = [m_1, m_2, \ldots, m_t]$ are the orders (≥ 2) of the elliptic cycles, and s is the number of parabolic cycles. In particular, there exists a group presentation of Γ_{O} in the following format:

- The group is generated by $a_1, a_2, \ldots, a_g, b_1, b_2, \ldots, b_g, g_1, g_2, \ldots, g_{t+s}$;
- The g_i satisfy the relations $g_i^{m_i} = 1$ for $1 \leq i \leq t$;
- We have the relation
$$[a_1, b_1][a_2, b_2] \cdots [a_g, b_g] g_1 g_2 \cdots g_{t+s} = 1,$$
where $[x, y] = xyx^{-1}y^{-1}$ is the commutator.

This is a minimal presentation if $t + s = 0$. If $t + s > 0$, then removing g_{t+s} makes it a minimal presentation.

5.2 Presentation

Call `P=algfdompresentation(U)` to compute a presentation. The output is a length 3 vector, where:

- $P[1]$ is the list of elements generating Γ_{O}, given in basis representation;
- $P[2]$ is the vector of relations;
- $P[3]$ represents $P[1][i]$ as a word in $U[1]$. This is a technical entry used to compute elements as words in the presentation, and will likely not be needed by a user.

Let $P[1] = \{g_1, g_2, \ldots, g_k\}$, and then the relation $[a_1, a_2, \ldots, a_i]$ means
$$1 = g_{a_1}^{\mathrm{sign}(a_1)} g_{a_2}^{\mathrm{sign}(a_2)} \cdots g_{a_i}^{\mathrm{sign}(a_i)}.$$

For example, the relation $[1, 4, -5, -5, 3]$ corresponds to
$$1 = g_1 g_4 g_5^{-2} g_3.$$

Each element of $P[2]$ and $P[3]$ is a Vecsmall.

5.3 Elements as words in the presentation

Once you have computed a fundamental domain U, a presentation P, and have an element $g \in \mathrm{O}^1$, you can compute g as a word in P with

```
w=algfdomword(g, P, U)
```

The format of w is exactly the same as a relation. Note that if you actually multiply out the representation, then you will either get g or $-g$, since they are indistinguishable in Γ_{O}.

5.4 Closed geodesics

Given a primitive hyperbolic element $g \in \mathrm{O}^1$, there is a corresponding closed geodesic on $\Gamma_{\mathrm{O}} \backslash \mathbb{H}$. We can compute this geodesic with `algfdomrootgeodesic(g, U)`.

5.5 Visualization

Currently, we can print a fundamental domain to a LaTeX document or to a Python application.

5.6 LaTeX

Call `fdom_latex(U, filename)` to print the domain to the file "plots/build/filename.tex". In order to compile the file, you need the document class standalone, as well as the pgf package.

The command `fdom_latex` includes 3 further options (in order):

- `boundcircles`, which is 1 by default. Changing it to 0 will not print the bounding circle.
- `compile`, which is 1 by default. If you are working with WSL, this will compile the LaTeX document, and move the pdf up one folder to "plots/filename.pdf".
- `open`, which is 1 by default. If we compiled the picture, this also opens it.

5.7 Python

We require the packages matplotlib and numpy to run "fdviewer.py", which contains the code to visualize a fundamental domain and geodesics. To write a fundamental domain U to a Python-readable file, call `python_printfdom(U, filename)`, which prints the requisite data to "fdoms/filename.dat". You must start the filename with "fd" for this to work correctly.

You can also print geodesics to a file by

```
geod=algfdomrootgeodesic(g, U);
python_printarcs(g[2], filename);
```

where the filename must not start with "fd".

To open the file, call (from the terminal) `py fdviewer.py filenames`. If you are working in WSL, you can also call `python_plotviewer(filenames)`. The filenames include at most one fundamental domain, and any number of geodesics (more than 5 is not suggested), separated by spaces.

In addition to the normal matplotlib commands, you have access to:

- Click on a side to highlight it in red, as well as the paired side in blue. You can use the right/left arrow keys to change sides (as well as clicking on a new side);
- Click on a geodesic to highlight it, highlight the next side in orange, and put an arrow in the middle specifying the orientation. You can use the up/down arrow keys to move between consecutive segments of the geodesic;
- Press "t" to hide the text box with information;
- Press "m" to hide the bounding axes;
- Press "c" to hide the bounding circle.

References

[LMF23] The LMFDB Collaboration. The L-functions and modular forms database. `http://www.lmfdb.org`, 2023. [Online].

[Ric22] James Rickards. Fundamental domains for Shimura curves. `https://github.com/JamesRickards-Canada/Fundamental-Domains-for-Shimura-curves`, 2022.

[Voi21] John Voight. *Quaternion algebras*, volume 288 of *Graduate Texts in Mathematics*. Springer, Cham, [2021] ©2021.