MODULAR FUNDAMENTAL DOMAIN

BERNDT E. SCHWERDTFEGER

ABSTRACT. Drawings of modular fundamental domains with METAPOST and Java.

PREFACE

This note contains figures of the modular fundamental domain drawn with METAPOST. In March 2018 I added a short description of Helena VERRILL's Java application `FunDomain` [2].

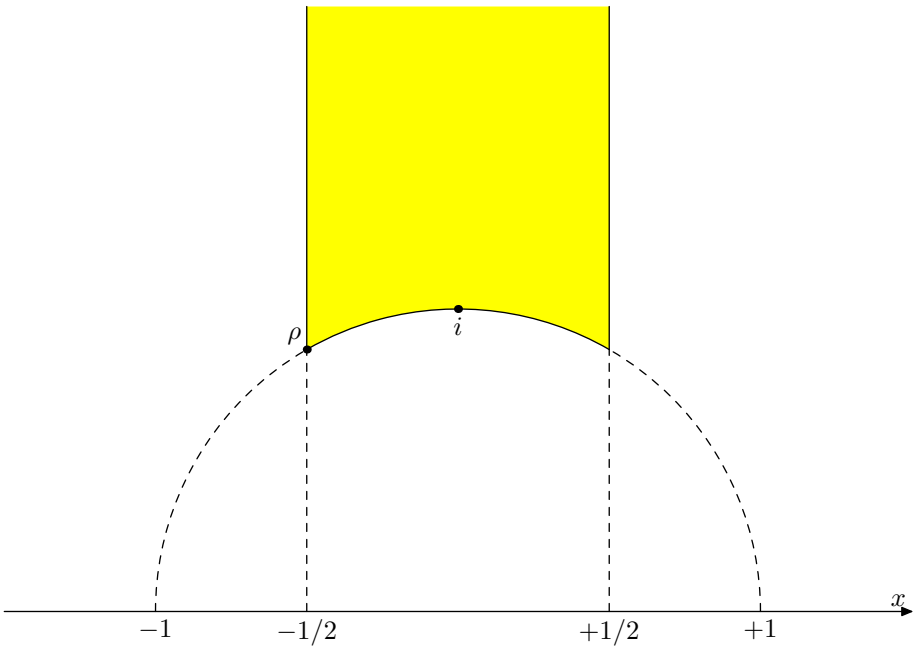
Berlin, 7th October 2018

 v1.3, 8th October 2018

1. THE MODULAR FUNDAMENTAL DOMAIN

This note draws pictures with METAPOST[1] for the *modular fundamental domain* $\mathcal{D}$ of $\mathbf{SL}(2,\mathbf{Z})$ operating on the POINCARÉ upper half plane $\mathcal{H}$, as shown by SERRE in [1, VII, §1.2].

1.1. **Drawing with METAPOST.** Recall that $\mathcal{D} = \{x \in \mathcal{H} \mid |\operatorname{Re} x| \le \frac{1}{2}, |x| \ge 1\}$ is the (*yellow*) region delimited by the unit circle and the two straight lines through $-\frac{1}{2}$ and $+\frac{1}{2}$.



2010 *Mathematics Subject Classification.* Primary 11F06; Secondary 11F11.
Key words and phrases. modular fundamental domain, upper half plane.

[1] see the code in the appendix A

The group $\mathbf{SL}(2,\mathbf{R})$ operates on the upper half plane $\mathcal{H}$

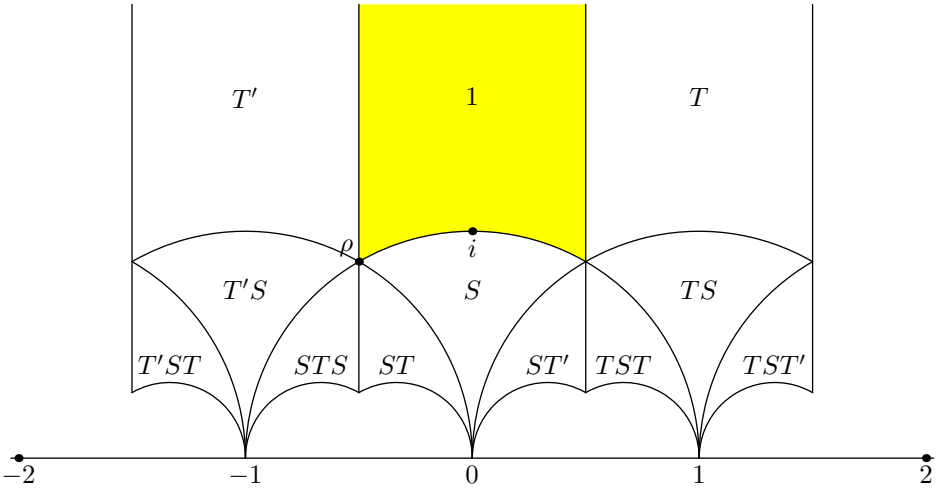
$$\begin{aligned}\mathbf{SL}(2,\mathbf{R})\times\mathcal{H} &\longrightarrow \mathcal{H}\\ (\gamma, x) &\longmapsto \gamma\cdot x\end{aligned}$$

by linear fractional transformations

$$\begin{pmatrix} a & b\\ c & d\end{pmatrix}\cdot x = \frac{ax+b}{cx+d}$$

In fact, as $-1 = \begin{pmatrix}-1 & 0\\ 0 & -1\end{pmatrix}$ operates trivially on $\mathcal{H}$, the group $\mathbf{PSL}(2,\mathbf{R}) = \mathbf{SL}(2,\mathbf{R})/\{\pm 1\}$ operates (faithfully) on $\mathcal{H}$. The *modular group* is the discrete subgroup $\Gamma = \mathbf{PSL}(2,\mathbf{Z}) = \mathbf{SL}(2,\mathbf{Z})/\{\pm 1\}$. As in SERRE [1] we note the elements of Γ given by the matrices $\begin{pmatrix}0 & -1\\ 1 & 0\end{pmatrix}$ and $\begin{pmatrix}1 & 1\\ 0 & 1\end{pmatrix}$ by S and T. i.e. $Sx = -1/x$ and $Tx = x+1$, we have $S^2 = 1$ and $(ST)^3 = 1$.

In the next figure we show the translates of $\mathcal{D}$ by the elements 1, S, T, T^{-1}, ST, TS, ST^{-1}, $T^{-1}S$, STS, TST, TST^{-1}, $T^{-1}ST$. Note: $TST = ST^{-1}S$. Each hyperbolic triangle is labeled by the corresponding transformation (in the figure $T' = T^{-1}$).



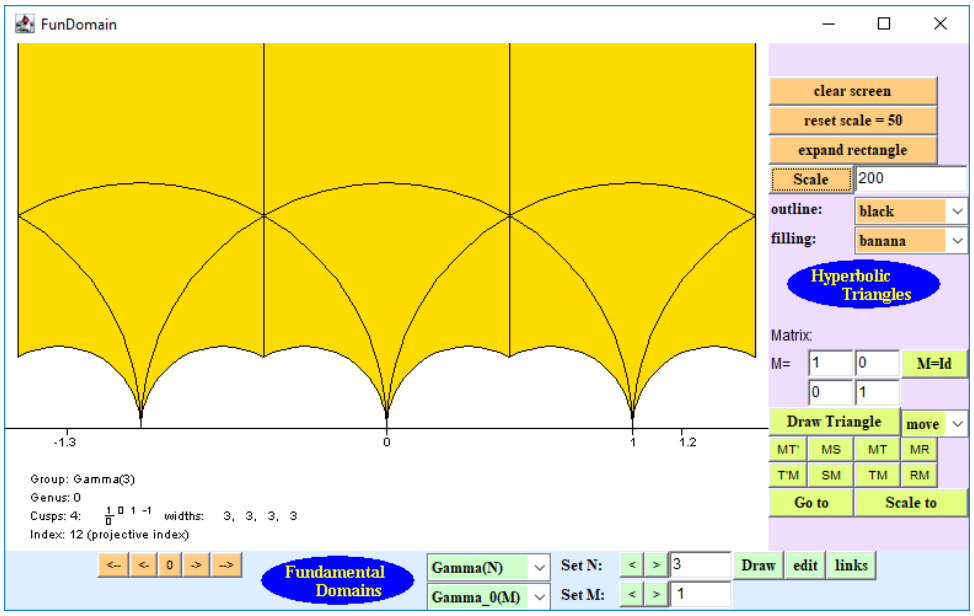
In fact, the 12 triangles together make up a fundamental domain for the congruence subgroup $\Gamma(3)$, the elements above form a set of representatives for $\Gamma/\Gamma(3) = \mathbf{PSL}(2,\mathbf{F}_3)$.

2. FUNDAMENTAL DOMAIN DRAWER

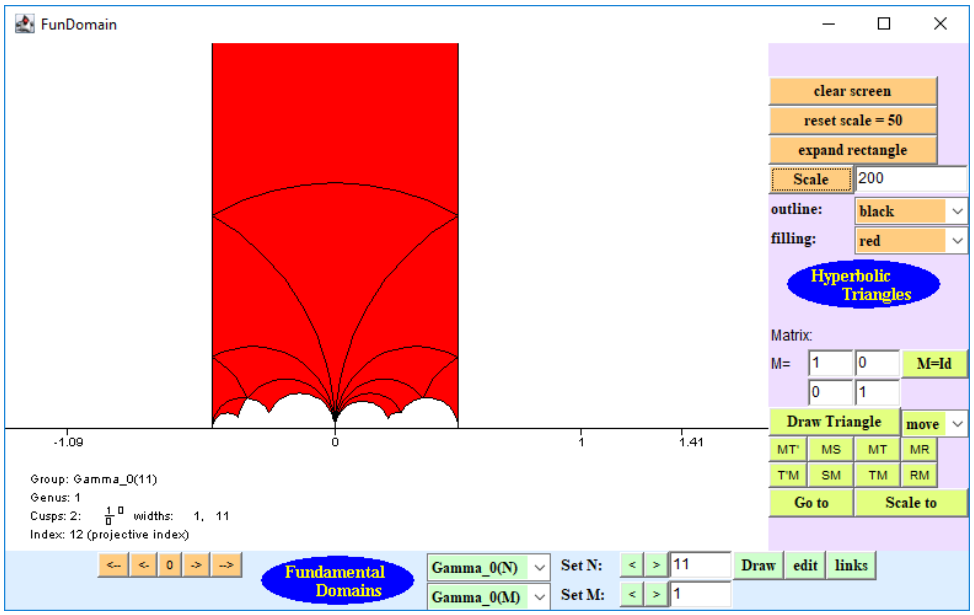
Helena VERRILL in 2000 developed a Java applet for *drawing fundamental domains* [2] (the source code is available at that site). It is still running in today's Java runtime environment[2] as a Java *application* (rather than as *applet*), with some quirks (program does not terminate, you have to kill the process). But that can be repaired: I applied some small changes to class `FunDomain` (see below subsection 2.2).

Here is a screenshot of the fundamental domain for the congruence subgroup $\Gamma(3)$:

[2] my environment is openjdk 11 2018-09-25, OpenJDK Runtime Environment 18.9 (build 11+28)



A fundamental domain for the *modular elliptic curve* $X_0(11)$ corresponding to the HECKE subgroup $\Gamma_0(11)$ is displayed like this:



2.1. **Features and Instructions.** At [2] the features and rough structure of the java code is described, it is included here in subsections 2.1.1–2.1.4.

2.1.1. *General control.* You can change scale, colour, position of the origin, using the orange coloured buttons. Press "0" to put the origin in the center. (max scale currently fixed to 20,000,000 — but doesn't work so well at this scale in some cases.) Scale is in pixels per unit.

Expand Rectangle: You can click on the screen and drag the mouse to form a rectangle. If you click on 'expand rectangle' the scale changes so the height of the rectangle becomes

the height of the screen. The center of the rectangle moves to the center of the screen (in vertical direction only).

2.1.2. *Fundamental Domain drawing part.*] Will draw domains for $\Gamma^0(N)$, $\Gamma^1(N)$, $\Gamma_0(N)$, $\Gamma_1(N)$, $\Gamma(N)$. Changing N - you can type in N, or you can press the ">" and "<" buttons to increase N in steps.

Edit Mode: In this mode the triangles can be moved to give a different fundamental domain for the same group, by clicking on the yellow circles on the boundary.

Links: These show how the boundary is linked up.

find matrix: You can find what matrix a triangle corresponds to by clicking on it; the colour changes, and the matrix will be printed in the left lower corner. Click again to change colour back.

2.1.3. *Triangle drawing part.* Here you can draw a triangle corresponding to transforming a standard domain by a given matrix. Currently only matrices of determinant 1 are allowed, for convenience at points in computation, but I will probably change this later. You can enter the matrix, or use the buttons TM, RM, etc to transform the matrix M. Matrices are $T = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$, $T' = \begin{pmatrix} 1 & -1 \\ 0 & 1 \end{pmatrix}$, $S = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$, $R = \begin{pmatrix} 0 & -1 \\ 1 & 1 \end{pmatrix}$.

move/copy: If move is selected, when the matrix is applied (eg. T, R, etc) the triangle is moved by this matrix. If copy is selected, a copy is made, which is a translate by the applied matrix.

Move to: this moves the origin so that the triangle just drawn is in the middle of the screen

Scale to: If you click on this, in addition to moving, it also scales, so the triangle just drawn is in the middle of the screen, and at a reasonable size so you can see it.

2.1.4. *Structure of java code.*

FunDomain: Controls for Input, Main routine
ModN: class for working with integers mod (N)
ConFrac: create a continued fraction given a fraction
IntMat: class defines integer 2 by 2 matrices, and various operations on them
ConjClassRep: defines coset representatives of the conjugacy classes; right now, just for $\Gamma_0(N)$, these are given as 2 by 2 matrices.
HypTriangle: drawing triangles
ArcSection: drawing sections for triangles
RepList: computation of the coset representatives

More details on the algorithm used in the computation of the coset representatives (in `RepList.java`) can be found in VERRILL's paper [3].

2.2. **Class FunDomain.** I applied five small changes to VERRILL's code in [2] in class `FunDomain` (in file `FunDomain.java`):

- removed the deprecated classes `//import java.applet.*;`
- changed deprecated `Applet` to `Panel`
- removed deprecated event processing for `Event.WINDOW_DESTROY`
- changed deprecated `show()` to `setVisible(true)`
- added a `WindowListener` to the frame in method `main` of class `FunDomain`

Here is the listing of the class FunDomain with these modifications (marked by *bs*):

```
import java.awt.*;
import java.util.*;
import java.awt.event.*;
//import java.applet.*;                          // bs commented out

public class FunDomain extends Panel {           // bs changed Applet to Panel
    DomainControls controls;
    UpperHalfPlane upperplane;
    ColourControls colcontrol;
    private Button clear_button;

    public void init() {
        setLayout(new BorderLayout());
        upperplane = new  UpperHalfPlane();
        add("Center", upperplane);
        controls = new DomainControls(upperplane);
        controls.setSize(800,200);
        add("South", controls);
        add("East", colcontrol = new ColourControls(upperplane));

        //      clear_button = new Button("clear");

    }

    public void destroy() {
        remove(controls);
        remove(upperplane);
    }

    public void start() {
        controls.setEnabled(true);
        colcontrol.setEnabled(true);
    }

    public void stop() {
        controls.setEnabled(false);
    }

/*  public void processEvent(AWTEvent e) {          bs commented out
        if (e.getID() == Event.WINDOW_DESTROY) {    bs
            System.exit(0);                         bs
        }                                           bs
    } */                                          // bs

    public static void main(String args[]) {
        Frame f = new Frame("FunDomain");
        FunDomain  domains = new FunDomain();

        domains.init();
        domains.start();
```

```
        f.add("Center", domains);
        f.setSize(800, 500);
        f.setVisible(true);                          // bs changed from f.show()
        f.addWindowListener(new WindowClosingAdapter(true)); // bs added line
    }
}
```

2.3. **WindowClosingAdapter.** Here follows the listing of `WindowClosingAdapter` taken from http://www.javabuch.de/ for closing the window and graceful termination of the program.

```
// WindowClosingAdapter.java

/* ------------------------------------------------------------
   WindowClosingAdapter: source from http://www.javabuch.de/
   ------------------------------------------------------------ */

import java.awt.event.*;

public class WindowClosingAdapter
extends WindowAdapter
{
  private boolean exitSystem;

/** Generates a WindowClosingAdapter for closing the window.
    If exitSystem is true, the program is terminated.    */
  public WindowClosingAdapter(boolean exitSystem)
  {
    this.exitSystem = exitSystem;
  }

/** Generates a WindowClosingAdapter for closing the window.
    The program is not terminated.    */
  public WindowClosingAdapter()
  {
    this(false);
  }

  public void windowClosing(WindowEvent event)
  {
    event.getWindow().setVisible(false);
    event.getWindow().dispose();
    if (exitSystem) {
      System.exit(0);
    }
  }
}
```

APPENDIX A. METAPOST CODE OF FUNDAMENTAL DOMAIN

```
% ------------------------------------------------------------------------
% modular fundamental domain
```

```
% Date: 2018-10-07
% ---------------------------------------------------------------------------

u=1mm;                                          % basic unit

color yellow;
yellow = (1,1,0);
path sixthcircle;                               % 1/6 circle
sixthcircle=
subpath(0,xpart(quartercircle intersectiontimes (origin--right rotated 60)))
of quartercircle;

% ---------------------------------
%     Figure 1
% ---------------------------------

beginfig(1);
path p[];
s=20u;                                          % scale 2s=1, 4cm
z0=(0,2s);                                      % z0=i=\sqrt{-1}=e(1/4)
x1=x4=-x2=-x3=s;                                % z1=\rho=e(1/3), 3rd root of 1
y1=y2=s*sqrt(3);                                % z2=\rho+1=e(1/6), 6th root of 1
y3=y4=4s;
z5=-z7=(x1,0);
z6=-z8=(2s,0);
p0=sixthcircle scaled 4s;                       % arc from z6 to z1
p1=p0 rotated 60;                               % arc from z1 to z2
p2=p1..(z2--z3) & (z3--z4) & (z4--z1) & cycle;  % closed path around D
fill p2 withcolor yellow;                       % fill it yellow
dotlabel.bot(btex $i$ etex, z0);                % label i
dotlabel.ulft(btex $\rho$ etex, z2);            % label \rho
draw p0 dashed evenly;
draw p0 reflectedabout (origin,z0) dashed evenly;
draw p1;
draw z2--z3;
draw z4--z1;
draw z5--z1 dashed evenly;
draw z7--z2 dashed evenly;
label.bot(btex $-1$ etex,z8);
label.bot(btex $-1/2$ etex,z7);
label.bot(btex $+1/2$ etex,z5);
label.bot(btex $+1$ etex,z6);
drawarrow (-3s,0)--(3s,0);                      % x-axis
label.ulft(btex $x$ etex, (3s,0));

endfig;

% ---------------------------------------------------------------------------

% ---------------------------------
%     Figure 2 from SERRE, p.128
% ---------------------------------
```

```
beginfig(2);
path p[];
s:=10u;                                          % rescale 3s=1, 3 cm
z0=(0,3s);                                       % z0=i=\sqrt{-1}=e(1/4)
x1=x4=-x2=-x3=1.5s;                              % z1=\rho=e(1/3), 3rd root of 1
y1=y2=1.5s*sqrt(3);                              % z2=\rho+1=e(1/6), 6th root of 1
y3=y4=6s;
y5=y6=y7=y8=1/3y1;
x5=-x8=9/2s;
x6=-x7=3/2s;
p0=sixthcircle scaled 6s;                        % arc from 1 to z1
p3=sixthcircle scaled 2s;                        % arc 1/3 radius
p4=p3 .. (p3 rotated 60);                        % .. 120 degree wide
p1=p0 rotated 60;                                % arc from z1 to z2
p2=p1..(z2--z3) & (z3--z4) & (z4--z1) & cycle;   % closed path around D
fill p2 withcolor yellow;                        % fill it yellow
dotlabel.bot(btex $i$ etex, z0);                 % label i
dotlabel.ulft(btex $\rho$ etex, z2);             % label \rho
% big arcs top
draw p1;
draw p1 shifted (-3s,0);
draw p1 shifted ( 3s,0);
% big arcs right
draw p0;
draw p0 shifted (-3s,0);
draw p0 shifted (-6s,0);
%big arcs left
draw p0 rotated 120;
draw p0 rotated 120 shifted (3s,0);
draw p0 rotated 120 shifted (6s,0);
% small arcs
draw p4 shifted (2s,0);
draw p4 shifted (-s,0);
draw p4 shifted (-4s,0);
p5=p4 reflectedabout (origin,z0);
draw p5 shifted (-2s,0);
draw p5 shifted ( s,0);
draw p5 shifted (4s,0);
% vertical lines
draw z7--z3;
draw z6--z4;
draw z5--(z4 + (3s,0));
draw z8--(z3 - (3s,0));

label.bot(btex $T'$ etex,(-3s,5s));
label.bot(btex $1 $ etex,(0  ,5s));
label.bot(btex $T $ etex,( 3s,5s));
label.top(btex $T'S$etex,(-3s,2s));
label.top(btex $  S$ etex,(0 ,2s));
label.top(btex $ TS$ etex,(3s,2s));
label.top(btex $T'ST$ etex,(-4s,s));
```

```
label.top(btex $STS$ etex,(-2s,s));
label.top(btex $ST$ etex,(-s,s));
label.top(btex $ST'$ etex,(s,s));
label.top(btex $TST$ etex,(2s,s));
label.top(btex $TST'$ etex,(4s,s));

dotlabel.bot(btex $-2$ etex,(-6s,0));          % some integer points on x
label.bot(btex $-1$ etex,(-3s,0));
label.bot(btex $ 0$ etex,origin);
label.bot(btex $ 1$ etex,(3s,0));
dotlabel.bot(btex $ 2$ etex,(6s,0));
draw (-6.1s,0)--(6.1s,0);                      % x-axis

endfig;

% ----------------------------------------------------------------------

end
```

REFERENCES

[1] Jean-Pierre Serre, *Cours d'arithmétique*, Presses Universitaires de France, Paris, 1970.

[2] Helena Verrill, *Fundamental Domain drawer* (2001), https://wstein.org/Tables/fundomain/. Accessed March 2, 2018.

[3] ———, *Algorithm for Drawing Fundamental Domains* (2001), available at https://berndt-schwerdtfeger.de/wp-content/uploads/pdf/fundomain-alg.pdf.

INDEX

F

fundamental domain drawer 2

M

modular elliptic curve 3
modular fundamental domain 1
modular group 2