HOMEWORK 6

DAN YASAKI

From Homework 5, we see that we have a map

$$\pi\colon \{\text{unimodular symbols}\} \to \mathbb{M}_2(\Gamma_0(N)).$$

We can extend this by linearity to get a surjective linear map from

$$\texttt{cycles(N)} \to \mathbb{M}_2(\Gamma_0(N)).$$

The purpose of this exercise will be to code π and a pullback π^*. Specifically, $\pi(\pi^*(\mathbf{u})) = \mathbf{u}$ for all $\mathbf{u} \in \mathbb{M}_2(\Gamma_0(N))$.

Along the way you can use Sage commands to check your work. For example, if you try

```
sage: M = ModularSymbols(11)
sage: P1 = M.manin_generators()
sage: images = M.manin_gens_to_basis()
```

then `M` is 3 dimensional. We have `P1` is $\mathbb{P}^1$, and `images` is a list of the images of π.

1. Define a function `voronoi_symbols(N)` which takes as input a positive integer N and returns $[V, \pi, \pi^*]$, where `V` is the rational vector space `cycles(N)/boundaries(N)` and π is a map that takes $[u, v]$ with $u, v \in \mathbb{Z}^2$ to the corresponding point in V. You can think of the map, theoretically, as follows using the notation of the notes:
   The symbol $[u, v] = g \cdot \mathbf{u}_0$ for some $g \in \mathrm{SL}_2(\mathbb{Z})$. This goes to the coset $\Gamma_0(N)g$, which corresponds to the point $(0 : 1) \cdot g \in \mathbb{P}^1(\mathbb{Z}/N\mathbb{Z})$. Use the ideas developed in Homework 5, problem 6 to map to V. You should return π and π^* as matrices, acting on the right, so that the rows of π and π^* express the basis elements of the domain in terms of the basis elements of the range.
2. Run several checks on your code. First, you map from $\mathbb{P}^1(\mathbb{Z}/N\mathbb{Z})$ to V should match identifications `P1` to `images`, up to overall sign. In other words, the maps should either be the same, or -1 times each other.

   One thing you should notice from these computation is that it may be easier if we start attaching a coefficient on our symbols. For example, we should think of $[u, v]$ as $(1, [u, v])$ and $[u, v] + [u, v]$ as $(2, [u, v])$.
3. Write a function `hecke_operator(N,p)`, which takes as input a level `N` and prime `p` and outputs the matrix representing the Hecke operator T_p on $\mathbb{M}_2(\Gamma_0(N))$. It may be helpful to use your `hecke_action` map from Homework 4.
4. Create a file *voronoi_symbols.py* that defines a class `VoronoiSymbols(N)`, which takes as input a level `N`. Mimic the `ModularSymbols` class in Sage. Our class will be much simpler. Specifically, it will only deal with $\Gamma_0(N)$ and weight 2. The functions you defined in earlier should help flesh out this class. We should be able to compute the dimension of the space, as well as compute Hecke operators on this space. No other features need to be available to the user at this time.
5. Check your class against the existing `ModularSymbols` code. Your Hecke operators should be the same map, up to change of basis. Specifically, your matrix should be

similar to the output of the Sage code. In particular, the characteristic polynomials should be the same. Try level 11 and compute the characteristic polynomials for T_p for a range of primes p and compare with the characteristic polynomials from your code. (Note: The matrices used to compute T_p for $p = 11$ are different than the usual for $p \nmid 11$.)

Dan Yasaki, Department of Mathematics and Statistics, University of North Carolina at Greensboro, Greensboro, NC 27412, USA
E-mail address: `d_yasaki@uncg.edu`