Modular Symbols, Modular Forms and Modular Abelian Varieties in MAGMA

William Stein
http://modular.fas.harvard.edu

Two Lecture IHP Minicourse: October 4–8, 2004

Abstract

I found MAGMA frustrating and incomprehensible until Allan Steel visited Berkeley and intensely explained it to me for two days. Since then, MAGMA has made much more sense to me. Now I will share the insights I often tell people when discussing computing with modular forms.

Instead of just telling you how amazing these packages are, I will often **emphasize the subtle problems** with my packages.

Background Assumptions

- You **are** very familiar with the basics of MAGMA.
- I will **not** assume you know about modular symbols.
- I will assume you've heard about modular forms, but will give a definition at some point.

Acknowledgements

- **Kevin Buzzard** had a major influence on how I designed the modular symbols and Dirichlet characters code for MAGMA.
- **David Kohel** wrote an early version of the Dirichlet characters package, and had a constant influence on the design.
- The main reason any of this code is efficient is that **Allan Steel** has massively optimized the exact dense linear algebra core of MAGMA, partly in response to my requests.
- The algorithms owe a major debt to **John Cremona**'s book and **Loic Merel**'s modular symbols article.

1 Dirichlet Characters

A **Dirichlet character** over an integral domain R is a map $\varepsilon : \mathbf{Z} \to R$ such that for some homomorphism $f : (\mathbf{Z}/N\mathbf{Z})^* \to R^*$ we have

$$\varepsilon(a) = \begin{cases} 0 & \text{if } (a, N) \neq 1, \\ f(a \bmod N) & \text{if } (a, N) = 1. \end{cases}$$

Listing 1.1 (Creation of a DirichletGroup).

```
> G<a,b,c> := DirichletGroup(8*13, CyclotomicField(12));
> G;
Group of Dirichlet characters of modulus 104 over Cyclotomic
Field of order 12 and degree 4
> // WARNING: The default ring is Q, not Q(zeta_n).
> DirichletGroup(8*13);
Group of Dirichlet characters of modulus 104 over Rational Field
```

The three generators of G correspond to the decomposition

$$(\mathbf{Z}/104\mathbf{Z})^* = \langle 79 \rangle \times \langle 53 \rangle \times \langle 41 \rangle,$$

where $79 \equiv -1 \pmod 4$ generates $(\mathbf{Z}/4\mathbf{Z})^*$, the element $53 \equiv 5 \pmod 8$ generates the non-± 1 factor of $(\mathbf{Z}/8\mathbf{Z})^*$, and $41 \equiv 2 \pmod{13}$ generates $(\mathbf{Z}/13\mathbf{Z})^*$.

Listing 1.2 (Invariants of characters).

```
> G<a,b,c> := DirichletGroup(8*13, CyclotomicField(12));
> [Order(a), Order(b), Order(c)];
[ 2, 2, 12 ]
> [Conductor(a), Conductor(b), Conductor(c)];
[ 4, 8, 13 ]
> a(3);
-1
> b(3);
-1
> c(3);
zeta_12^2 - 1
```

We can also do arithmetic with Dirichlet characters.

Listing 1.3 (Arithmetic with characters).

```
> G<a> := DirichletGroup(5,CyclotomicField(4));
> H<b> := DirichletGroup(7,RationalField());
> Parent(a*b);
Group of Dirichlet characters of modulus 35 over Cyclotomic Field
of order 4 and degree 2
```

Listing 1.4 (Extension to larger modulus).

```
> c := a*b;
> d := Extend(c,70);  // natural extension to character of modulus 70
> Parent(d);
Group of Dirichlet characters of modulus 70 over Cyclotomic Field
of order 4 and degree 2
> Conductor(d);
35
> Modulus(AssociatedPrimitiveCharacter(d));
35
```

Listing 1.5 (Coercion to bigger base ring).

```
> G := DirichletGroup(35,CyclotomicField(28));  G;
Group of Dirichlet characters of modulus 35 over Cyclotomic Field
of order 28 and degree 12
> e := G!c;
> Parent(e);
Group of Dirichlet characters of modulus 35 over Cyclotomic Field
of order 28 and degree 12
> c(3);
zeta_4
> e(3);
zeta_28^7
> Parent(MinimalBaseRingCharacter(e));
Group of Dirichlet characters of modulus 35 over Cyclotomic Field
of order 4 and degree 2
```

Warnings – Inefficient When N is Large

Dirichlet characters are efficient when N is small, but currently absurdly slow for N large (e.g., 10 digits). I learned yesterday from Allan Steel that this is because "someone" implemented discrete log in a certain context in a very inefficient way. This will presumably be fixed soon.

Source Code: Dirichlet Characters

The implementation of Dirichlet characters is completely contained in the following relatively-short file:

`package/Geometry/ModSym/dirichlet.m`

Remark. Actually Nicole Sutherland did a nice job of moving much of the Dirichlet code to C, which should be an improvement from the point of view of efficieny and memory management. But this code hasn't been released yet.

2 Modular Symbols

2.1 Motivation

Computation of spaces $\boldsymbol{\mathcal{M}}_k(N, \varepsilon)$ of **modular symbols** is the heart of most of the algorithms in MAGMA for computing with modular forms and modular abelian varieties:

- Computing spaces $M_k(N, \varepsilon)$ of **modular forms** involves modular symbols algorithms and enumeration of Eisenstein series.
- We view **modular abelian varieties** as complex vector spaces modulo lattices, where the lattices are naturally viewed as spaces of modular symbols.

References

- My Ph.D. thesis *Explicit Approaches to Modular Abelian Varieties*.
- Loic Merel's *Universal Fourier Expansions of Modular Forms*.

2.2 What Are Modular Symbols?

Fix the following:

- **The Level:** positive integer N
- **The Weight:** integer $k \geq 2$
- **The Character:** Dirichlet character ε of modulus N.

Let $\boldsymbol{\mathcal{M}}_k(N, \varepsilon)$ be the space of modular symbols of level N, weight k, and character ε, which we view as being defined by the algorithm on the next slide. As motivation, there is a non-canonical isomorphism

$$\boldsymbol{\mathcal{M}}_k(N, \varepsilon) \approx S_k(N, \varepsilon)^{\oplus 2} \oplus E_k(N, \varepsilon),$$

where $S_k(N, \varepsilon)$ is the space of cusp forms of type N, k, ε, and E_k is the space of Eisenstein series of that type. (See Merel's article for a proof.)

Algorithm 2.1 (Modular Symbols Presentation). This algorithm computes a presentation for the space $\boldsymbol{\mathcal{M}}_k(N, \varepsilon)$ of modular symbols, as a vector space over $K = \mathbf{Q}(\varepsilon)$.

1. **[Generating Manin Symbols]** Create a list of the Manin symbols $[X^i Y^{k-2-i}, (u, v)]$, where $i = 0, \ldots, k-2$, and $u, v \in \mathbf{Z}/N\mathbf{Z}$ with $\gcd(u, v, N) = 1$. Let V be the K-vector space generated by these Manin symbols modulo the subspace generated by differences $[P, (\lambda u, \lambda v)] - \varepsilon(\lambda)[P, (u, v)]$. (Compute V directly with basis $[0, k-2] \times \mathbf{P}^1(\mathbf{Z}/N\mathbf{Z})$.)
2. **[Subspace of Relations]** The group $\mathrm{GL}_2(\mathbf{Z})$ acts on V on the right by

$$[P(X, Y),\ (u, v)].\left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right) = [P(dX - bY, -cX + aY),\ (au + cv, bu + dv)].$$

   Let $S = \left(\begin{smallmatrix} 0 & -1 \\ 1 & 0 \end{smallmatrix}\right)$ and $T = \left(\begin{smallmatrix} 0 & -1 \\ 1 & -1 \end{smallmatrix}\right)$, and let W be the suspace of V generated by the elements

$$x + xS \qquad \text{and} \qquad x + xT + xT^2 \tag{2.1}$$

   for all generating manin symbols x.
3. **[Quotient]** Using sparse linear algebra, compute and output the quotient $\boldsymbol{\mathcal{M}}_k(N, \varepsilon) \cong V/W$, This is essentially the same as finding the reduced row echelon form of the matrix whose rows are given by the relations (2.1). The output is a list of freely generating Manin symbols $x_1, \ldots, x_n$, and all other Manin symbols written as linear combinations of $x_1, \ldots, x_n$.

Remark. There is a **star involution** $*$ on $\boldsymbol{\mathcal{M}}_k(N,\varepsilon)$, and for many computations it is sufficient to compute in one of the quotients $\boldsymbol{\mathcal{M}}_k(N,\varepsilon)/(*-1)$ or $\boldsymbol{\mathcal{M}}_k(N,\varepsilon)/(*+1)$. We compute this quotient directly by including the relations $x+xI$ or $x-xI$ in W in Step 2, where $I = \left(\begin{smallmatrix} -1 & 0 \\ 0 & 1 \end{smallmatrix}\right)$.

We illustrate modular symbols by using MAGMA to compute a presentation for $\boldsymbol{\mathcal{M}}_5(13,\varepsilon)$, where ε has order 4 and $K = \mathbf{Q}(i)$.

Listing 2.2 (Modular symbols creation).

```
> G<eps> := DirichletGroup(13, CyclotomicField(4));
> eps(-1);
-1
> Order(eps);
4
> M := ModularSymbols(eps, 5);   // eps determines N  !!
> M;
Full modular symbols space of level 13, weight 5, character eps,
and dimension 8 over Cyclotomic Field of order 4 and degree 2
```

Thus $\boldsymbol{\mathcal{M}}_5(13,\varepsilon)$ has dimension 8 as a vector space over $\mathbf{Q}(i)$. Note that the level $N = 13$ is encoded as the modulus of ε, so it is not necessary to specify N when defining $\boldsymbol{\mathcal{M}}_k(N,\varepsilon)$.

Basis of Manin and Modular Symbols

The following command enumerates a basis represented as Manin symbols.

Listing 2.3 (Basis of Manin Symbols).

```
> [ManinSymbol(x)[1] : x in Basis(M)];
[   <X^3, (0 1)>, <X^3, (1 11)>, <X^3, (1 5)>, <X^3, (1 3)>,
    <X^3, (1 4)>, <X^3, (1 6)>, <X^3, (1 12)>, <X^3, (1 0)>]
```

Elements of $\boldsymbol{\mathcal{M}}_k(N, \varepsilon)$ print by default as modular symbols. If $[P(X, Y), (c, d)]$ is a Manin symbol, and $g = \left(\begin{smallmatrix} a & b \\ c' & d' \end{smallmatrix}\right) \in \mathrm{SL}_2(\mathbf{Z})$ satisfies $c \equiv c'$ and $d \equiv d' \pmod{N}$, then the corresponding modular symbol is $P(dX - bY, -cX + aY)\{g(0), g(\infty)\}$.

Listing 2.4 (Corresponding Modular Symbols).

```
> M.1;
X^3*{0, oo}
> M.2;
(1331*X^3 + 363*X^2*Y + 33*X*Y^2 + Y^3)*{-1/11, 0}
```

2.3 Efficiency of Computation of Presentation

The current implementation of computation of the presentation for modular symbols in MAGMA takes way more memory than it should in some cases. However, it is extremely fast.

Listing 2.5 (Computing Presentation is Fast).

```
> M := ModularSymbols(2004);  M; //  1.440 seconds on laptop
Full modular symbols space for Gamma_0(2004) of weight 2 and
dimension 673 over Rational Field
> GetMemoryUsage();
9306624    // about 9.3MB
> time M := ModularSymbols(10000);
Current total memory usage: 419.4MB, failed memory request: 206.0MB
System error: Out of memory.
> time M := ModularSymbols(10007);  Dimension(M);   // prime level, so easier
Time: 6.590
1669
> time t2 := HeckeOperator(M,2);
Time: 3.270
```

Listing 2.6 (Computing Presentations with Nontrivial Character).

```
> G<e> := DirichletGroup(389,CyclotomicField(97));
> Order(e);
194
> time M := ModularSymbols(e^2,2);
Time: 0.320
> M;
Full modular symbols space of level 389, weight 2, character e^2,
and dimension 64 over Cyclotomic Field of order 97 and degree 96
> G<e> := DirichletGroup(37,CyclotomicField(36));
> time M := ModularSymbols(e,7);   // weight 7
Time: 3.260
> M;
Full modular symbols space of level 37, weight 7, character e,
and dimension 38 over Cyclotomic Field of order 36 and degree 12
```

2.4 Efficiency Trick: Work Mod p

For many problems (e.g., related to Galois representations), computing with modular symbols modulo p is enough, and can be **vastly more efficient**. If we create a group `DirichletGroup` of Dirichlet characters over a finite field R, then the corresponding space of modular symbols is a vector space over R. In MAGMA this vector space is **defined** by Algorithm 2.1 with $K = R$.

Listing 2.7 (Modular symbols modulo p).

```
> G<a,b,c> := DirichletGroup(2000,GF(5));
> Conductor(c);
5
> Order(c);
4
> time M := ModularSymbols(c,3);
Time: 5.070
> M;
Full modular symbols space of level 2000, weight 3, character c,
and dimension 1200 over Finite field of size 5
```

Most code views the base field as generic, so is the same for finite fields and $\mathbf{Q}(\zeta_n)$.

Spurious Torsion

BIG WARNING: *These spaces need* not *be a "mod p reduction" of the space in characteristic 0. In particular when p is small it is possible that there is "spurious" torsion.*

Listing 2.8 (Spurious torsion).

```
> function f(N,k)
     return Dimension(ModularSymbols(N,k,GF(2))) -
            Dimension(ModularSymbols(N,k));
  end function;
> [N : N in [2..100] | f(N,2) gt 0];
[ 5, 10, 13, 17, 25, 26, 29, 34, 37, 41, 50, 53, 58, 61, 65, 73,
  74, 82, 85, 89, 97 ]
```

In each case the dimension is one bigger, except for 65 and 85, when it is off by 3.

If you want to use modular symbols mod p as computed in MAGMA to "prove" a theorem, you must understand the underlying theory.

For an application of using modular symbols mod p, see Buzzard-Stein, *A Mod 5 Approach to Artin's Conjecture*, which was the paper that motivated me writing all this code in the first place.

2.5 Hecke Operators on Modular Symbols

The spaces $\boldsymbol{\mathcal{M}}_k(N, \varepsilon)$ are equipped with a commuting ring of **Hecke operators** T_n, for all positive integers n. MAGMA computes these Hecke operators using Merel's Heilbronn matrix formulas. For each n, Merel defines a computable set S_n (in fact various sets) of matrices of determinant n such that

$$T_n(x) = \sum_{g \in S_n} x.g.$$

The sets S_n only depend on n, not on k, N, ε, and the cardinality of S_n is about $O(n \log(n))$.

Heilbronn Matrices

The command to list the set S_n for a given n, returns the matrices as a sequence whose entries are the integer sequences corresponding to the elements of S_n.

Listing 2.9 (Heilbronn matrices).

```
> HeilbronnMerel(2);
[
[ 1, 0, 0, 2 ],
[ 1, 0, 1, 2 ],
[ 2, 0, 0, 1 ],
[ 2, 1, 0, 1 ]
]
> #HeilbronnMerel(29);
199
> #HeilbronnMerel(10007);
337977
> #HeilbronnCremona(10007);    // in some cases these can be used...
67698
```

Computing Hecke Operators

We next compute a Hecke operator on the space $\boldsymbol{\mathcal{M}}_5(13, \varepsilon)$:

Listing 2.10 (Hecke operators on modular symbols).

```
> G<eps> := DirichletGroup(13, CyclotomicField(4));
> M := ModularSymbols(eps, 5);
> T2 := HeckeOperator(M,2);
> Nrows(T2);
8
> T2[1];
(zeta_4 + 16 -3/4 1/4 3/4 -3/4 0 2 -3/2)
> F := CharacteristicPolynomial(T2);
> R<X> := Parent(F);
> Factorization(F);
[   <X - 16*zeta_4 - 1, 1>,
    <X - zeta_4 - 16, 1>,
    <X^3 + (zeta_4 + 1)*X^2 - 23*zeta_4*X - 29*zeta_4 + 29, 2>]
```

The characteristic polynomial has 2 factors that appear with multiplicity one, which correspond to Eisenstein series, and a factor with multiplicity 2, which corresponds to $S_5(13, \varepsilon)$.

2.6 Subspaces of Modular Symbols

The spaces $\boldsymbol{\mathcal{M}}_k(N, \varepsilon)$ have many important subspaces. MAGMA computes the cuspidal subspace $\boldsymbol{\mathcal{S}}_k(N, \varepsilon)$ as the kernel of a natural map to a space of *boundary modular symbols*; this subspace is isomorphic to $S_k(N, \varepsilon)^{\oplus 2}$.

Listing 2.11 (Cuspidal subspace).

```
> G<eps> := DirichletGroup(13, CyclotomicField(4));

> M := ModularSymbols(eps, 5);

> S := CuspidalSubspace(M); S;
Modular symbols space of level 13, weight 5, character eps, and
dimension 6 over Cyclotomic Field of order 4 and degree 2

> Factorization(CharacteristicPolynomial(HeckeOperator(S,2)));
[  <X^3 + (zeta_4 + 1)*X^2 - 23*zeta_4*X - 29*zeta_4 + 29, 2>  ]
```

New and Old Subspaces of Modular Symbols

There are other interesting subspaces of modular symbols spaces, such as the new and old subspaces. The new subspace is the kernel of all maps to lower level, and the old subspace is the space generated by all images of maps from lower level. In the following example we compute the new subspace of $\boldsymbol{\mathcal{M}}_2(33, 1)$, where 1 denotes the trivial character.

Listing 2.12 (New subspace).

```
> M := ModularSymbols(33); M;
Full modular symbols space for Gamma_0(33) of weight 2 and
dimension 9 over Rational Field

> NewSubspace(M);
Modular symbols space for Gamma_0(33) of weight 2 and dimension 3
over Rational Field

> Complement(NewSubspace(M));
Modular symbols space for Gamma_0(33) of weight 2 and dimension 6
over Rational Field
```

2.7 Decomposition of Modular Symbols Spaces

The most important operation on spaces of modular symbols is `NewformDecomposition`, which involves writing $\boldsymbol{S}_k(N, \varepsilon)$ as a sum of spaces that cannot be split further using Hecke operators of index coprime. Since the Hecke algebra is commutative, each subspace is preserved by the Hecke operators.

Listing 2.13 (Newform Decomposition).

```
> S := CuspidalSubspace(ModularSymbols(33,2));

> NewformDecomposition(S);
[
    Modular symbols space for Gamma_0(33) of weight 2 and
    dimension 2 over Rational Field,                        // new
    Modular symbols space for Gamma_0(33) of weight 2 and
    dimension 4 over Rational Field                         // old
]
```

Listing 2.14 (Newform Decomposition II).

```
> time M := ModularSymbols(700,2,+1);
Time: 0.270
> time S := CuspidalSubspace(M);
Time: 0.190
> time D := NewformDecomposition(S);
Time: 10.990
> #D;
34
> D;
[
    Modular symbols space for Gamma_0(700) of weight 2 and
    dimension 1 over Rational Field,
    Modular symbols space for Gamma_0(700) of weight 2 and
    dimension 1 over Rational Field,
    ...
    Modular symbols space for Gamma_0(700) of weight 2 and
    dimension 6 over Rational Field
]
```

Remarks about the complexity of NewformDecomposition

- Probably the complexity of decomposition is about $O((Nk)^6)$, the running time being dominated by the factorization of characteristic polynomials on a space of dimension $O(Nk)$. No such complexity analysis has been done, as far as I know, except that Giesbrecht has given a algorithm (with complexity analysis) for computing the rational Jordan form, which is a problem closely related to decomposing modular symbols spaces.

- The precise algorithm used for decomposition in MAGMA was created and implemented by Allan Steel, and I don't completely understand it and think it's not published anywhere. Allan and I put a lot of work into optimizing decomposition, since it is **the main bottleneck** in computations.

Source Code: Modular Symbols

The implementation of modular symbols is in the directory magma/package/Geometry/ModSym/.
I encourage you to browse the source code, starting with modsym.m and core.m.

Listing 2.15 (Source Code).

```
$ ls magma/package/Geometry/ModSym/*.m
analytic.m       cusps.m       inner_twists.m          operators.m
arith.m          decomp.m      intersection_pairing.m  period.m
boundary.m       derivative.m  linalg.m                qexpansion.m
calc.m           dims.m        maps.m                  representation.m
charpolyhecke.m  dirichlet.m   mestre.m                subspace.m
compgrp.m        eisenstein.m  modsym.m                tests.m
core.m           elliptic.m    multichar.m             verbose.m
```

These files total about 19000 lines, including comments. I think the only code that is closed off in C are some of the functions from core.m, but the MAGMA implementations are still in core.m, just commented out.

3 Modular Forms

3.1 Definitions

- For integers N, let

$$\Gamma_0(N) = \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \mathrm{SL}_2(\mathbf{Z}) \,:\, \begin{pmatrix} a & b \\ c & d \end{pmatrix} \equiv \begin{pmatrix} * & * \\ 0 & * \end{pmatrix} \pmod{N} \right\}.$$

- The finite-dimensional complex vector space $M_k(N, \varepsilon)$ of **modular forms** of level N, weight k, and character ε is the set of holomorphic functions f on the extended upper half plane

$$\mathfrak{h}^* = \{z \in \mathbf{C} \,:\, \mathrm{Im}(z) > 0\} \cup \mathbf{P}^1(\mathbf{Q})$$

such that for all $g = \left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right) \in \Gamma_0(N)$,

$$f|_{[g]_k}(z) := \det(g)^{k-1}(cz+d)^{-k} f(g(z)) = \varepsilon(g) f(z),$$

where $\varepsilon(g) = \varepsilon(a)$.

- A **cusp form** is a modular form such that $f(\mathbf{P}^1(\mathbf{Q})) = \{0\}$, and we denote the subspace of cusp forms by $S_k(N, \varepsilon)$.

q-Expansions and Hecke Operators

- Any $f \in M_k(N, \varepsilon)$ has a **q-expansion** (Fourier series)

$$f(z) = \sum_{n=0}^{\infty} a_n q^n, \qquad q(z) = e^{2\pi i z}.$$

  The cusp forms all satisfy $a_0 = 0$, but not conversely.

- The **Hecke operators** T_p, for p prime, act on $M_k(N, \varepsilon)$ by

$$T_p(f) = \sum_{n=0}^{\infty} a_{np} q^n + \varepsilon(p) p^{k-1} f(q^n),$$

  and there is a similar formula for T_n for any n.

- If $f \in S_k(N, \varepsilon)$ is an eigenform for T_n with $a_1 = 1$, then $T_n(f) = a_n f$. So to give such an f is the same as giving a **system of Hecke eigenvalues**.

3.2 Computing Modular Forms Using MAGMA

Given N, k, ε, MAGMA can compute a basis of power series expensions as above, modulo a power of q. One way to compute a basis for all cusp forms is to use the `qExpansionBasis` command, applied to the cuspidal subspace of a space of modular symbols.

Listing 3.1 (Basis of q-Expansions).

```
> G<eps> := DirichletGroup(13,CyclotomicField(6));
> M := ModularSymbols(eps,2, +1);
> S := CuspidalSubspace(M);
> S;
Modular symbols space of level 13, weight 2, character eps, and
dimension 1 over Cyclotomic Field of order 6 and degree 2
> qExpansionBasis(S,4);
[   q + (-zeta_6 - 1)*q^2 + (2*zeta_6 - 2)*q^3 + O(q^4) ]
```

The `+1` in the `ModularSymbols` command computes the quotient

$$\boldsymbol{\mathcal{M}}_k(N, \varepsilon)/(* - 1),$$

which is all that is needed to compute $S_k(N, \varepsilon)$, since $\boldsymbol{\mathcal{M}}_2(13, \varepsilon)/(*-1) \cong S_k(N, \varepsilon) \oplus E_k(N, \varepsilon)'$, where $E_k(N, \varepsilon)'$ is a certain subspace of the Eisenstein space.

Listing 3.2 (More q-Expansions).

```
> M := ModularSymbols(33,2);
> S := CuspidalSubspace(M);

> qExpansionBasis(S,10);
[   q - q^5 - 2*q^6 + 2*q^7 - 2*q^8 - q^9 + O(q^10),
    q^2 - q^4 - q^5 - q^6 + 2*q^7 - q^8 + q^9 + O(q^10),
    q^3 - 2*q^6 - q^9 + O(q^10)   ]

> qExpansionBasis(OldSubspace(S),10);
[   q - 2*q^2 + 2*q^4 + q^5 - 2*q^7 - 3*q^9 + O(q^10),
    q^3 - 2*q^6 - q^9 + O(q^10) ]

> qExpansionBasis(NewSubspace(S),10);
[   q + q^2 - q^3 - q^4 - 2*q^5 - q^6 + 4*q^7 - 3*q^8 + q^9 +
        O(q^10) ]
```

In this example we compute a basis of q-expansion corresponding to a simple factor of $S_2(389, 1)$.

Listing 3.3 (Basis of q-Expansions).

```
> M := ModularSymbols(389,2, 1);
> S := CuspidalSubspace(M);
> D := Decomposition(S,2);
> V := D[3]; V;
Modular symbols space for Gamma_0(389) of weight 2 and dimension
3 over Rational Field
> qExpansionBasis(V,10);
[   q - q^5 - 2*q^6 - q^7 + 2*q^8 - q^9 + O(q^10),
    q^2 - q^3 + O(q^10),
    q^4 - q^5 - q^6 + q^9 + O(q^10) ]
> qEigenform(V,6);    // eigenform in span of above q-expansions
q + a*q^2 - a*q^3 + (a^2 - 2)*q^4 + (-a^2 + 1)*q^5 + O(q^6)
> BaseRing(Modulus(Parent($1)));
Univariate Quotient Polynomial Algebra in a over Rational Field
with modulus a^3 - 4*a - 2
```

Modular Forms Package

I wrote a package for computing with modular forms that makes no direct reference to modular symbols, but which is mostly built on the modular symbols machinery. The command `ModularForms(N,k,eps)` creates the free $\mathbf{Z}$ module

$$\left(\bigoplus_{\mathrm{Gal}(\overline{\mathbf{Q}}/\mathbf{Q})\text{-conjugates } \varepsilon'} M_k(N, \varepsilon') \right) \bigcap \mathbf{Z}[[q]],$$

where the sum is over all $\mathrm{Gal}(\overline{\mathbf{Q}}/\mathbf{Q})$-conjugates ε' of ε. This has rank $\dim(M_k(N, \varepsilon)) \cdot d$, where d is the number of conjugates of ε.

WARNING. Defining `ModularForms(N,k,eps)` this way was probably a **bad design decision** on my part(!), and I should have defined `ModularForms(N,k,eps)` to be $M_k(N, \varepsilon)$ as a $\mathbf{Q}(\varepsilon)$-vector space. I might change this (while keeping the old definition as an option). The advantage of the choice I made is that, e.g., reduction modulo p make sense.

Listing 3.4 (Spaces of Modular Forms).

```
> M := ModularForms(33,2); M;
Space of modular forms on Gamma_0(33) of weight 2 and
dimension 6 over Integer Ring.
> Basis(M);
[   1 + O(q^8),
    q - q^5 + 2*q^7 + O(q^8),
    q^2 + 2*q^7 + O(q^8),
    q^3 + O(q^8),
    q^4 + q^5 + O(q^8),
    q^6 + O(q^8)   ]
> SetPrecision(M,15);
> Basis(M);
[   1 + 12*q^11 + O(q^15),
    q - q^5 + 2*q^7 - 2*q^8 + q^9 - 2*q^10 - q^11 + 4*q^12 + 4*q^14 + O(q^15),
    q^2 + 2*q^7 + q^8 + q^9 + 2*q^10 + q^11 + q^12 + 2*q^14 + O(q^15),
    q^3 + q^9 - 2*q^11 + 4*q^12 + O(q^15),
    ... ]
```

Newforms

A **newform** is an element $f \in S_k(N, \varepsilon)$ that is in the kernel of the maps to all levels properly dividing N, is an eigenvector for every Hecke operator normalized so the coefficient of q is 1.

One can compute Eisenstein series and a list of all newforms (gathered together in $\mathrm{Gal}(\overline{\mathbf{Q}}/\mathbf{Q})$-conjugacy classes) using the `Newforms` command.

In the following example, we compute the two newforms in $M_{12}(1)$. One newform is the Ramanajun Δ function, and the other is the normalized Eisenstein series of weight 12.

Listing 3.5 (Newforms).

```
> M := ModularForms(1,12);
> Newforms(M);
[* [*
q - 24*q^2 + 252*q^3 - 1472*q^4 + 4830*q^5 - 6048*q^6 - 16744*q^7
+ O(q^8)
*], [*
691/65520 + q + 2049*q^2 + 177148*q^3 + 4196353*q^4 +
48828126*q^5 + 362976252*q^6 + 1977326744*q^7 + O(q^8)
*] *]
```

In the next example, we compute $S_2(\Gamma_0(23))$, which is spanned by two conjugate eigenforms.

Listing 3.6 (Conjugate eigenforms).

```
> M := ModularForms(23,2);
> S := CuspidalSubspace(M);
> S;
Space of modular forms on Gamma_0(23) of weight 2 and dimension 2
over Integer Ring.
> Newforms(S);
[* [*
q + a*q^2 + (-2*a - 1)*q^3 + (-a - 1)*q^4 + 2*a*q^5 + (a - 2)*q^6
  + (2*a + 2)*q^7 + O(q^8),
q + b*q^2 + (-2*b - 1)*q^3 + (-b - 1)*q^4 + 2*b*q^5 + (b - 2)*q^6
  + (2*b + 2)*q^7 + O(q^8)
*] *]
> Parent($1[1][1]);
Space of modular forms on Gamma_0(23) of weight 2 and dimension 2
over Number Field with defining polynomial x^2 + x - 1 over the
Rational Field.
```

Conjugate Eigenforms??

Listing all the conjugate newforms is somewhat silly, because they all look identical; they are just defined over different copies of the field generated by the Fourier coefficients. In particular, adding the two newforms together is not defined in MAGMA (it doesn't give the trace).

Listing 3.7 (Arithmetic Problem).

```
> f := Newforms(S)[1][1];  g := Newforms(S)[1][2];
> f + g;
>> f + g;
     ^
Runtime error in '+': Arguments 1 and 2 have incompatible
coefficient rings.
```

Given how well-developed number fields are in MAGMA, I could probably improve this.

Computation of Embeddings

For many computations that really require arithmetic with all the conjugates of a form, one should just embed the forms in the complex numbers or a p-adic field, where p split:

Listing 3.8 (Complex and p-adic Embeddings).

```
> ComplexEmbeddings(f);
[* [* q - 1.6180339887498948482045868343656381177 20*q^2 +  ...
      q + 0.6180339887498948482045868343656381177 20*q^2 -  ... *] *]
> $1[1][1] + $1[1][2];
2*q - q^2 - q^4 - 2.0...*q^5 -
> pAdicEmbeddings(f,2);
[* [* O(2^20) + (1 + O(2^20))*q + ((1 + O(2^20))*a + O(2^20))*q^2 + ...
      O(2^20) + (1 + O(2^20))*q + ((1 + O(2^20))*b + O(2^20))*q^2 + ... *] *]
> pAdicEmbeddings(f,11);
[* [*  O(11^20) + (1 + O(11^20))*q + (273946294811098331671 + ...
*], [* O(11^20) + (1 + O(11^20))*q - (273946294811098331672 + ... *] *]
> $1[1][1] + $1[2][1];
O(11^20) + (2 + O(11^20))*q - (1 + O(11^20))*q^2 + ...
```

The output of the embedding commands are also modular forms, so we can compute them to higher precision, etc.

Computation of Reductions Modulo p

We can also reduce newforms to characteristic p. In general, this uses computation of a p-maximal order in a number field (via Lenstra's algorithm?).

Listing 3.9 (Reductions Modulo p).

```
> Reductions(f,2);
[* [*  q + $.1*q^2 + q^3 + $.1^2*q^4 + $.1*q^6 + O(q^8),
       q + $.1^2*q^2 + q^3 + $.1*q^4 + $.1^2*q^6 + O(q^8) *] *]
> Reductions(f,11);
[* [* q + 7*q^2 + 7*q^3 + 3*q^4 + 3*q^5 + 5*q^6 + 5*q^7 + O(q^8) *],
   [* q + 3*q^2 + 4*q^3 + 7*q^4 + 6*q^5 + q^6 + 8*q^7 + O(q^8) *] *]
> f11 := Reductions(f,11)[1][1];
> Type(f11);
ModFrmElt
> f11;
q + 7*q^2 + 7*q^3 + 3*q^4 + 3*q^5 + 5*q^6 + 5*q^7 + O(q^8)
> PowerSeries(f11,15);
q + 7*q^2 + 7*q^3 + 3*q^4 + 3*q^5 + 5*q^6 + 5*q^7 + 7*q^8 + 2*q^9
    + 10*q^10 + 4*q^11 + 10*q^12 + 3*q^13 + 2*q^14 + O(q^15)
```

Confusing Definition

This example illustrates how `ModularForms(N,k,eps)` is (confusingly!?) defined to be the direct sum of spaces for the conjugates of ε.

Listing 3.10 (Modular forms are over the integers).

```
> G<eps> := DirichletGroup(13, CyclotomicField(6));
> M := ModularForms(eps);
> BaseRing(M);
Integer Ring
> S := CuspidalSubspace(M);
> S;
Space of modular forms on Gamma_1(13) with character all
conjugates of [eps], weight 2, and dimension 2 over Integer Ring.
> Basis(S);
[
    q - 4*q^3 - q^4 + 3*q^5 + 6*q^6 + O(q^8),
    q^2 - 2*q^3 - q^4 + 2*q^5 + 2*q^6 + O(q^8)
]
```

Recall from before though that the dimension of $S_2(13, \varepsilon)$ as a $\mathbf{Q}(\zeta_6)$-vector space is 1, hence the confusion.

Source Code: Modular Forms

The implementation of modular forms is in the directory magma/package/Geometry/ModFrm/.
I encourage you to browse the source code, starting with creation.m.

Listing 3.11 (Source Code).

```
$ ls magma/package/Geometry/ModFrm/*.m
abelian_varieties.m  eisenstein.m      modular_symbols.m  relations.m
arithmetic.m         elliptic_curve.m  newforms.m         subspaces.m
bases.m              hecke_algebras.m  operators.m        tests.m
categories.m         input_output.m    p-adic.m           verbose.m
congruences.m        l_series.m        predicates.m       weight1table.m
creation.m           level1.m          q-expansions.m
degeneracy_maps.m    misc.m            qexp_mappings.m
```

These files total about 11000 lines, including comments. Nothing has been moved to C code.

4 Modular Abelian Varieties

This section is about modular abelian varieties, and some fairly general algorithms for computing with them in MAGMA. Much of this is new, and hasn't been explained anywhere, so I will focus more on the background, the algorithms, and what is implemented, rather than on usage details.

An **abelian variety** is a projective variety that is equipped with an algebraic group structure. (The group structure is necessarily abelian.)

The abelian varieties of dimension 1 are exactly the elliptic curves. Jacobians of curves of genus > 1 are examples of abelian varieties of dimension > 1, and it is a theorem that every abelian variety over an infinite field is a quotient of a Jacobian.

The modular Jacobians $J_1(N)$ are a special class of Jacobians that are very well understood because of their connection with modular forms. The abelian variety $J_1(N)$ is the Jacobian of the modular curve $X_1(N)$, which over $\mathbf{C}$ is the quotient of the extended upper half plane $\mathfrak{h}^*$ by

$$\Gamma_1(N) = \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \mathrm{SL}_2(\mathbf{Z}) \ : \ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \equiv \begin{pmatrix} 1 & * \\ 0 & 1 \end{pmatrix} \pmod{N} \right\}.$$

An abelian variety A over a number field K is a **modular abelian variety** of level N if it is a quotient of $J_1(N)$.

More Background

- **Generalized Modularity Conjecture (Ribet):** There is an analogue of the Shimura-Taniyama-Weil conjecture for abelian varieties. Over $\mathbf{Q}$, the simple modular abelian varieties A are supposed to be the simple abelian varieties of GL_2-type, i.e., those whose endomorphism ring is an order in a number field of degree $\dim(A)$. For more about this *open conjecture*, see Ribet's beautiful paper *Abelian Varieties over* $\mathbf{Q}$ *and Modular Forms*.
- The **new quotient** of $J_1(N)$ breaks up as a product $\prod A_f$ corresponding to the $\mathrm{Gal}(\overline{\mathbf{Q}}/\mathbf{Q})$-conjugacy classes of newforms f. The newform abelian varieties A_f are simple abelian varieties over $\mathbf{Q}$ and the dimension of A_f is the degree of the field generated by the coefficients of f. For example, Wiles et al. proved that the isogeny classes of elliptic curves over $\mathbf{Q}$ are in bijection with the A_f with $f \in \mathbf{Z}[[q]]$.

Motivated by my research on visibility of Shafarevich-Tate groups, last summer I designed and implemented a MAGMA package for doing fairly general computations with modular abelian varieties, which builds on the modular symbols machinery described earlier in this paper. Also, there are many functions that take spaces of modular symbols as input, and compute some quantity associated to the corresponding modular abelian varieties.

What we wish we could compute...

1. (*) The *modular degree*, i.e., the square root of the degree of the natural map $A \to A^\vee$ induced by virtual of A being modular.
2. Defining equations for A.
3. (*) The Birch and Swinnerton-Dyer quotient $L(A,1)/\Omega_A$.
4. (*?) The order of vanishing $r = \mathrm{ord}_{s=1} L(A,s)$ and leading coefficient $L^r(A,1)/r!$ of the expansion of $L(A,s)$ about $s = 1$.
5. The Mordell-Weil group of A and the regulator Reg_A.
6. The p-adic L-functions attached to A.
7. (*?) The Tamagawa numbers c_p of A.
8. (*?) The torsion subgroup $A(\mathbf{Q})_{\mathrm{tor}}$.
9. (*) The intersection $A \cap B$, where $A, B \subset J$.
10. (*) Whether A is isomorphic to $A^\vee$, and the minimal degree of a homomorphism $A \to A^\vee$.
11. (?) Enumeration of the isogeny class of A over $\mathbf{Q}$. (Perhaps via images of A under natural maps into other modular Jacobians, and quotients of A by carefully chosen finite subgroups.)

The Situation for $\dim(A) = 1$: Except for (4 and 6), methods to solve the above problems are known when $\dim(A) = 1$, and they would provably terminate if we knew finiteness of $Ш(A)$. So far the situation for modular abelian varieties isn't nearly as complete. My impression is that even for elliptic curves, unfortunately nobody has any clue about how to *provably* compute $r = \mathrm{ord}_{s=1} L(A, s)$ when $r > 3$; however, if we don't care about provable correctness, then computing r is straightforward (see, e.g., Cremona's book).

The items in the above list indicated with a (*) are implemented in MAGMA for simple modular abelian varieties over $\mathbf{Q}$, and those with (*?) are partially implemented:

- **Equations:** Substantial work has been done on finding defining equations when $\dim(A)$ is small and $A = \mathrm{Jac}(X)$ for some curve X, e.g., by the researchers in Essen. The paper *Empirical evidence for the Birch and Swinnerton-Dyer conjectures for modular Jacobians of genus 2 curves* then discussed methods for computing such an A when X has genus 2, along with the Cassels-Flynn book and many other papers. In contrast to this trend, my approach has been to compute as much as possible about A without writing down any defining questions; instead I exploit the special structure coming from the modularity of A. Computing the regulator and Mordell-Weil group seems to require having defining equations.
- **p-adic L-functions:** Nothing is implemented for p-adic L-functions, but the foundation is there, and Robert Pollack would likely make an amazingly p-adic L-function package if John Cannon invites him to Sydney (hint, hint).

- **Tamagawa Numbers:** The algorithm for computing component groups and Tamagawa numbers is in my thesis. It computes the order of the component group at primes p such that A_f has multiplicative reduction and A_f is a quotient of $J_0(N)$. It can find the Tamagawa number up to a power of 2, and in many cases it can also find the exact Tamagawa number. The reason it doesn't find the Tamagawa number in all cases is because the algorithm finds the order of the group, not its group structure, and the Tamagawa number is either the order of the group or the order of the 2-torsion subgroup. When the level is not prime, this algorithm relies on David Kohel's MAGMA package for computing with ideal classes in an Eichler order in a quaternion algebra. (Kohel's package initially attracted me to MAGMA.)

- **Isogenies:** When A is simple, I implemented an algorithm in MAGMA for computing the exact endomorphism ring of A. Then computing the minimal degree of any homomorphism $A \to A^\vee$ can be reduced to finding all solutions to a Diophantine norm equation, up to units. Fortunately MAGMA contains code for solving norm equations. This summer Tseno Tselkov did a project with me, in which he implemented an algorithm to find the minimal degree of an isogeny between A_f and $A_f^\vee$ (which is not yet in the standard MAGMA distribution). Very surprisingly, in all the data he computed, when $f \in S_2(\Gamma_0(N))$, the minimal degree is a power of 2 (!!?). This work is a hopeful first step toward an algorithm to enumerate every element of the isogeny class of any A_f.

4.1 Modular Abelian Varieties and Modular Symbols

The following example illustrates computation of most of the starred items in the list. In each case we create a simple space V of modular symbols; associated to V there is a newform f, and associated to f there is an abelian variety A_f, which in the examples is an optimal quotient of $J_0(N)$. The space of modular symbols (with integral structure) "is" the homology of A_f, which is a rich object equipped with structure coming from the modularity of A_f. For example, the homology has a Hecke action.

We compute with simple factors of $J_0(389)$ using commands applied to spaces of modular symbols (note that 389 is prime). In Section we will do the same computations but using the modular abelian varieties package, which provides a nicer wrapper around these functions (and adds other things that are not possible using just modular symbols). However, to most effectively use MAGMA, it is best to know about both ways of doing these computations.

We have

$$J_0(389) = A_1 \times A_2 \times A_3 \times A_6 \times A_{20},$$

where A_d is a simple abelian variety of dimension d.

Listing 4.1 (Modular abelian varieties via modular symbols).

```
> M := ModularSymbols(389);
> S := CuspidalSubspace(M);
> D := NewformDecomposition(S);
> [Dimension(A)/2 : A in D];          // dimensions of abvars A_f
[ 1, 2, 3, 6, 20 ]
> [ModularDegree(D[i]) : i in [1..#D]];
[ 40, 144, 992, 17856, 20480 ]
> [LRatio(D[i],1) : i in [1..#D]];    // BSD Ratios L(A_f,1)/Omega
[ 0, 0, 0, 0, 51200/97 ]
> Factorization(51200);
[ <2, 11>, <5, 2> ]
> LSeriesLeadingCoefficient(D[1],1,100);
0.75931650029224679065762600319 2
> E := EllipticCurve(A);  AnalyticRank(E);   // Watkin's new code
2 0.7593000000
> LSeriesLeadingCoefficient(D[2],1,100);
1.48718462131934683691665432667 1
```

Listing 4.2 (Modular abelian varieties via modular symbols (cont)).

```
> TamagawaNumber(D[1],389);          // c_{389} = 1 for elliptic curve
1
> TamagawaNumber(D[5],389);          // c_{389} = 97 for 20-dim quotient
97
> TorsionBound(D[5],13);             // multiple of order of torsion
97
> #RationalCuspidalSubgroup(D[5]);   // divisor of order of torsion
97
> Invariants(IntersectionGroup(D[1],D[2]));
[ 2, 2 ]
> Invariants(IntersectionGroup(D[1],D[5]));
[ 20, 20 ]
```

Remark 4.3. If E is the elliptic curve factor and A is the 20-dimensional factor, then the above computation, the BSD conjecture, and visibility theory imply that $Ш(A) = 5^2 \cdot 2^?$. Assuming no conjectures I can also prove that

$$(\mathbf{Z}/5\mathbf{Z})^2 \cong E(\mathbf{Q})/5E(\mathbf{Q}) \subset Ш(A).$$

4.2 The Modular Abelian Varieties Package

Given a Dirichlet character ε of modulus N, there is an abelian variety $J(N,\varepsilon)$ whose rational homology corresponds to $\boldsymbol{S}_2(N,\varepsilon)$ viewed as a $\mathbf{Q}$-vector space; thus $J(N,\varepsilon)$ is an abelian variety over $\mathbf{Q}$ (not $\mathbf{Q}(\varepsilon)$!) that is isogenous to a product of abelian varieties A_f attached to the $\mathrm{Gal}(\overline{\mathbf{Q}}/\mathbf{Q})$-conjugacy classes of newforms of level dividing N with character ε.

If $A \subset J(N,\varepsilon)$ is an abelian subvariety, then the image of $\mathrm{H}_1(A,\mathbf{Q})$ in $\boldsymbol{S}_2(N,\varepsilon)$ is a vector subspace V_A, and A is determined by V_A.

An **explicitly given modular abelian variety** is a modular abelian variety B that is specified by giving a vector subspace $V \subset \boldsymbol{S}_2(N,\varepsilon)$ that corresponds to an abelian variety $A \subset J(N,\varepsilon)$ and a finite subgroup $G \subset A$. Then $B \cong A/G$. Every modular abelian variety can be specified in this way, though it can be a highly nontrivial problem to figure out how. For example, Jaocbians of Shimura curves are modular abelian varieties, but determining them explicitly in terms of abelian varieties $J(N,\varepsilon)$ is nontrivial (cf. David Helm's Ph.D. thesis).

The following code illustrates the computations that we did above, but instead uses the modular abelian varieties package:

Listing 4.4 (Modular abelian varieties package).

```
> J := JZero(389); J;
Modular abelian variety JZero(389) of dimension 32 and level 389
over Q
> D := Decomposition(J);
> [Dimension(A) : A in D];
[ 1, 2, 3, 6, 20 ]
> [ModularDegree(A) : A in D];
[ 40, 144, 992, 17856, 20480 ]
> [LRatio(A,1) : A in D];
[ 0, 0, 0, 0, 51200/97 ]
> L := LSeries(D[1]); L;
L(389A,s): L-series of Modular abelian variety 389A of dimension
1, level 389 and conductor 389 over Q
> LeadingCoefficient(L,1,200);    // slow, since doesn't use Watkins (but *general
0.75931650029224679065762600319 2
> TamagawaNumber(D[1],389);
1 1 true
```

Listing 4.5 (Modular abelian varieties package (cont)).

```
> TamagawaNumber(D[5],389);
97 97 true
> TorsionLowerBound(D[5]);
97
> TorsionMultiple(D[5]);
97
> G := RationalCuspidalSubgroup(D[5]); G;
Finitely generated subgroup ... with invariants [ 97 ]
> B := D[5]/G; B;     // quotients by anything are defined.
Modular abelian variety of dimension 20 and level 389 over Q
> H := D[1] meet D[5];  H; // takes a while
Finitely generated subgroup ... with invariants [ 20, 20 ]
```

Remark 4.6. I found (trivial-to-fix) bugs in my implementation of `ModularDegree` and `LeadingCoefficient` functions while preparing this talk; if you try the above example in Magma V2.11-6 you will get the wrong answers. I'll upload a fix right after this conference.

The modular abelian varieties package allows for creation of much more general abelian varieties than the modular symbols package; for example, it supports arbitrary finite direct sums and quotients by finite subgroups. Also, it includes explicit computation of endomorphism rings and hom rings over $\mathbf{Q}$.

Listing 4.7 (Computation of endomorphism ring).

```
> J := JZero(22);
> [Matrix(phi) : phi in Basis(End(J))];
...
```

This gives the following four matrices as generators:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad \begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & -1 \\ 0 & 1 & 0 & 1 \end{pmatrix},$$

$$\begin{pmatrix} 0 & 1 & 0 & -1 \\ 0 & 1 & 0 & 0 \\ -1 & 2 & -1 & 1 \\ -1 & 1 & 0 & 0 \end{pmatrix}, \quad \begin{pmatrix} 0 & 1 & -2 & 1 \\ -1 & 2 & -1 & 0 \\ -1 & 0 & 1 & -1 \\ 0 & -1 & 1 & -1 \end{pmatrix}.$$

Also one can do new computations with endomorphism rings. For example, the following is a table of the index of the Hecke algebra in its saturation in $\mathrm{End}(J_0(N))$, which is a quantity that controls the relation between the modular degree and congruences (the "congruence modulus"):

Listing 4.8 (Index of Hecke Algebra in Saturation).

```
> function f(N)
      J := JZero(N);
      T := HeckeAlgebra(J);
      return Index(Saturation(T),T);
  end function;
> for N in [1..120] do print N, f(N); end for;
...
```

This results in the following table, which suggests that if $p \mid f(N)$ is a prime, then $p^2 \mid 4 \cdot N$, a fact closely related to what Ken Ribet proved at the Raynaud birthday conference in Orsay a few years ago. Also, Mazur proved that $f(p) = 1$ when p is prime.

N	$f(N)$	N	$f(N)$	N	$f(N)$	N	$f(N)$	N	$f(N)$
1	1	21	1	41	1	61	1	81	1
2	1	22	1	42	1	62	2	82	1
3	1	23	1	43	1	63	1	83	1
4	1	24	1	44	2	64	2	84	2
5	1	25	1	45	1	65	1	85	1
6	1	26	1	46	2	66	1	86	1
7	1	27	1	47	1	67	1	87	1
8	1	28	1	48	1	68	2	88	8
9	1	29	1	49	1	69	1	89	1
10	1	30	1	50	1	70	1	90	1
11	1	31	1	51	1	71	1	91	1
12	1	32	1	52	1	72	2	92	16
13	1	33	1	53	1	73	1	93	1
14	1	34	1	54	3	74	1	94	4
15	1	35	1	55	1	75	1	95	1
16	1	36	1	56	2	76	2	96	8
17	1	37	1	57	1	77	1	97	1
18	1	38	1	58	1	78	2	98	1
19	1	39	1	59	1	79	1	99	9
20	1	40	1	60	2	80	4	100	1

Toward a Theory over Number Fields

These are three open problems whose solution is needed in order to have a good theory for computing with modular abelian varieties over number fields.

1. **(Endomorphism Ring)** Find an **efficient** way to compute the endomorphism ring $\mathrm{End}(A_f/K)$. By explicit, we mean give generators as a subgroup of $\mathrm{End}(\mathrm{H}_1(A, \mathbf{Z}))$. This ring can be computed using the Ribet-Shimura theory of inner twists, but the formulas they give translated to modular symbols are very slow; one needs a direct Manin symbols formula.

2. **(Simple Decomposition)** Suppose A is an abelian variety over a number field K, that we have explicit generators for $\mathrm{End}(A)$, and that A is isogenous to a power B^n of a simple abelian variety B. Determine B explicitly and find an explicit isogeny between A and B^n. This boils down to a characteristic 0 "meataxe", which is something Allan Steel has been working on.

3. **(General Isomorphism Testing)** Suppose A and B are explicitly given modular abelian varieties. Decide whether two explicitly given modular abelian varieties are isomorphic. (I think I know how to do this unless some simple factor occurs with multiplicity bigger than 1 in the isogeny decomposition of both A and B.)

Source Code: Modular Abelian Varieties

The implementation of modular abelian varieties is in the directory magma/package/Geometry/ModAbVar/. I encourage you to browse the source code, starting with modabvar.m.

Listing 4.9 (Source Code).

```
$ ls magma/package/Geometry/ModAbVar/*.m
arithabvar.m   elt.m       homspace.m      misc.m       periods.m
compgrp.m      endo_alg.m  inner_twists.m  modabvar.m   rings.m
complements.m  fields.m    linalg.m        morphisms.m  subgrp.m
decomp.m       heegner.m   lser.m          new_old.m    test.m
ellcrv.m       homology.m  map.m           operators.m  torsion.m
```

These files total about 14000 lines, including comments. Nothing has been moved to C code.