COMPUTING CLASSICAL MODULAR FORMS

ALEX J. BEST, JONATHAN BOBER, ANDREW BOOKER, EDGAR COSTA, JOHN CREMONA, DAVID ROE, ANDREW V. SUTHERLAND, AND JOHN VOIGHT

Abstract. We discuss practical aspects of computing a database of classical modular forms.

Contents

[[JV: The list of authors is provisional...]]

1. Introduction

1.1. **Motivation.** Databases of classical modular forms have been used for a variety of mathematical purposes and have almost a 50 year history (see section 2). In this article, we report on a recent effort in the L-functions and Modular Forms DataBase (LMFDB, `http://lmfdb.org`) [3].

1.2. **Organization.** The paper is organized as follows. In section 2 we begin with a short history. Next, in section 3 we detail what we mean by computing (spaces of) modular forms and then in section 4 we give a short overview of algorithms for computing modular forms. In section 5 we discuss issues concerning Dirichlet characters. In section 6, we sample the available implementations and make some comparisons. Next, in section 7 we discuss some computational, theoretical, and practical issues that arose in our efforts and in section 8 we explain how we (rigorously) computed the L-functions.

Turning to our main effort, in section 9 we provide an overview of the computation we performed and make some notes on the data. In sections 10–11 we treat twists and issues specific to modular forms of weight 1. We conclude in section 12 with some final remarks.

Date: December 14, 2018.

1.3. **Acknowledgements.** This research was undertaken under the support of Simons Collaboration Grants (550029, ...). The authors would like to thank Karim Belabas, Henri Cohen, Alan Lauder, David Loeffler, David Platt, Mark Watkins, ...

2. History

In this section, we briefly indicate the history of computing tables of modular forms. For more history and discussion, see Kilford [4, Chapter 7].

Perhaps the first systematic tabulation of modular forms was performed by Wada [15, 16]. As early as 1971, he used the Eichler–Selberg trace formula to compute a factorization of the characteristic polynomial of the Hecke operator T_p on $S_2(\Gamma_0(q), \chi)$ for $q \equiv 1 \pmod 4$ prime where χ was either trivial or the quadratic character of conductor q. The total computation time was about 300 hours on a TOSBAC-3000.

The next major step was made in the famous "Antwerp IV" tables (published in 1975), which contains ... and was computed using modular symbols. These were later extended by Cremona [2] (the first edition came out in 1992), with applications to the tabulation of elliptic curves.

Miyake [8] published some numerical tables as appendices, again using the trace formula. These tables included: dimensions of $S_k(\Gamma_0(N))$ for $k \geq 2$ even and N small, eigenvalues and characteristic polynomials of Hecke operators on $S_2(\Gamma_0(N))$ for N a small prime, and Fourier coefficients of a primitive form in $S_2(\Gamma_0(N), \chi_N)$ for $N = 29, 37$.

In the 1990s, Henri Cohen, Nils-Peter Skoruppa, and Don Zagier compiled tables of eigenforms in weights 2 through 12, level up to 1000 in weight 2 and lower in higher weight, some tables with character. They followed a paper by Skoruppa–Zagier on the trace formula [11], but these tables were not published.

In the early 2000s, William Stein create an online modular forms database [12], computed primarily using a modular symbols package [13] he implemented in Magma [5] starting in the late 1990s. The data was computed using a rack of six custom-built machines and a Sun V480; it was stored in a PostgreSQL database (more than 10 GB) and provided a (Python-based) web interface to the data. These tables included dimensions, characteristic polynomials, and q-expansions in a variety of weights and levels. Using the Magma implementation, Christian Meyer [6, 7] computed a table of newforms for $\Gamma_0(N)$ with rational coefficients: in weight $k = 2$ he went to $N \leq 3000$ and for $k = 4$ to $N \leq 2000$.

Prior to our work, the LMFDB had a database of classical modular forms computed by Stephan Ehlen and Fredrik Strömberg. This dataset included partial information on $S_k(\Gamma_0(N))$ for (k, N) in the ranges $[2, 12] \times [1, 100]$ and $[2, 40] \times [1, 25]$, and on $S_k(\Gamma_1(N))$ in the ranges $[2, 10] \times [1, 50]$ and $[2, 20] \times [1, 16]$.

3. Computing modular forms

In this section, we try to make precise what it means to compute modular forms.

3.1. **Setup.** As basic input, we suppose we are given as input a weight $k \in \mathbb{Z}_{\geq 1}$, a level $N \in \mathbb{Z}_{\geq 1}$, and a Dirichlet character $\chi\colon (\mathbb{Z}/N\mathbb{Z})^\times \to \mathbb{C}^\times$ (of modulus N). For the specification of the character in bits, see section 5.

Recall that a modular form of weight k and level N with (Nebentypus) character χ is a holomorphic function

$$f\colon \mathcal{H} \to \mathbb{C}$$

on the upper half-plane $\mathcal{H}$, bounded in vertical strips, satisfying

$$f\left(\frac{az+b}{cz+d}\right) = \chi(d)(cz+d)^k f(z)$$

for all $\begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma_0(N) \leq \mathrm{SL}_2(\mathbb{Z})$. The space of such forms is denoted $M_k(\Gamma_0(N), \chi)$

Remark 3.1.1. We restrict ourselves to integral weight forms in this article. For forms of half-integral weight, the algorithms, applications, and issues that arise are quite different.

3.2. **Dimensions.** The first thing one may ask for are dimensions. Matrix: old and new, cuspidal and Eisenstein subspaces.

Sturm bound.

3.3. **Eisenstein series.** Um [[JV: There was an exercise]]

3.4. **Decomposition into Hecke orbits.** Dimensions, Hecke field, and trace form

Hecke cutters

See below for a discussion of Maeda and verifying irreducibility.

3.5. **Hecke eigenvalues.** Exact. Issues in how to represent them.

Gives a smaller bound to recognize form (often, just one Hecke eigenvalue is enough).

Complex. Rigor in computing. Satake parameters and angles.

3.6. ***L*-functions.** What we mean by computing L-functions, see section 8 for more.

4. Algorithms

In this section, we give a brief overview of the different algorithmic methods to compute modular forms and where they are implemented. In our effort, we only use the first two (modular symbols and the trace formula).

4.1. **Modular symbols.** Specifically, Manin symbols. The book [13] describes in Sage and the overview in weight 2 by Stein [14].

Stein implemented in Magma in the 1990s with improvements by ... and in Sage in the 2000s with additional improvements by ...

Buzzard and Buzzard–Lauder use modular symbols to get weight 1 in Magma by multiplying by an Eisenstein series to get to weight 2. Currently, you can get a basis in Magma for the cuspidal subspace, but it does not decompose the space into the old and new subspace and does not provide the action of the Hecke operators. (They wrote code to do this.)

```
> chi := Generators(FullDirichletGroup(383))[1];
> M := ModularForms(chi,1);
> Dimension(M);
190
> HeckeOperator(M,5);

>> HeckeOperator(M,5);
                 ^

Runtime error: Hecke operator computation currently only supported for spaces
with a single character that takes values +/-1.
> S := CuspidalSubspace(M);
> Basis(S);
[]
```

[[JV: Get a better example here.]]

In Sage, the Hecke stability method of Schaeffer [?] is implemented to get forms in weight 1. [[JV: Unfortunately, it is slow because modular symbols in Sage is slow.]]

4.2. **Trace formula: exact method.** Hijikata

This is a bit optimistic, but typically OK, yes :

1) you assume that the weight is fixed (otherwise the size of the matrix
entries must be taken into account); and that the Nebentypus has fixed order
as well [otherwise you need to work in cyclotomic fields or large degree, which
increases the cost of "base field" computations]

2) splitting the space may need many (linear combinations of) T_p
[I don't know anything better than the Sturm bound to guarantee that the
T_p, p <= B, generate the Hecke algebra]. So O~(d^4) would be a
worst case [given assumption 1)]

>   * To get further eigenvalues, you typically only need one row of
> T_p, but you still need to multiply this row by each eigenvector, so
> it ends up being basically soft-O(d*p) again.

For the trace formula, here's a quick back-of-the-enveloppe computation.
Will check this with Henri in september :-) :

1) We must first build the space S_k^new:

1.a) we pick successive forms T_j Tr^new until they generate the space.
Assuming the first O~(d) values of j are enough [heuristic for now but it may
be possible to prove this; it's true in practice], this requires
expanding those O~(d) forms up to Sturm bound (O~(d)). So will need
O~(d * max(j)) = O~(d^2) coeffs of Tr^new.

1.b) all Hurwitz class numbers of index O~(d * max(j)) are precomputed
[cost O~(d^3)]; the coefficient Tr(n) [= trace of T_n on the space S_k]
costs O(sqrt(n)). I am assuming that the weight and Nebentypus are
fixed, otherwise we need to take into account the "size" of coefficients.

So computing all Tr(n) up to O~(d^2) costs O~(d^3). The Tr^new(n) are
simple convolutions of the Tr(n) with Moebius function and the like and
costs the same up to log factors (sums over divisors etc.).

1.c) we compute the rank of the matrix made up by the coefficients of
the T_j Tr^new, and hope to get maximal rank in O(1) trials with O~(d)
forms: O(d^3) [or whatever exponent: no soft-Oh because we expect to
detect the rank by projecting Z[\chi] to a small finite field]

1.d) we precompute base change matrices from and to Miller's basis: at
least O~(d^\omega+1) [the T_j Tr^new form a somewhat random basis and the
coefficients in the original -> Miller base change matrix are huge]

Total [heuristic] cost for this phase: O~(d^\omega+1)

2) To compute the matrix of T_p on our basis for S_k^new, we now need

```
coefficients of Tr^new up to O~(d * max(j) * p). The Hurwitz class
number precomputation and subsequent coefficients computation jumps to
O~(d^3 p^{3/2}).

3) Then it's the same as in the other methods: characteristic polynomial,
factorization over Q(\chi), splitting, etc.


Thus, in theory, I would expect the trace formula to be slower than modular
symbols because of

- the cost to convert to Miller basis (or to express a random form in
  terms of the T_j Tr^new basis)

- the extra costs (extra coefficients) involved in hitting T_j Tr^new by T_p

In practice, as long as p doesn't get too large (and the linear algebra
involved in converting T_j Tr^new -> Miller basis doesn't get dominant),
I'm not sure at all that this is the case. It also depends on how you
get S_k^new from modular symbols when N is highly composite : kernels of
degeneracy maps can get expensive since they apply on "huge" S_k (of
dimension D), not "tiny" S_k^new (of dimension d).

I'm *very* interested in data points if you compare the above guesstimates
with Sage or Magma running times. :-)
```

4.3. Trace formula: complex analytic method.
Still rigorous, but need to make exact matches for Galois orbits

4.4. Brandt matrices.
Theta series, Pizer. Kohel.

4.5. Method of graphs.
Mestre and Osterlé.

4.6. Ternary quadratic forms.
Birch, and Hein–Tornaria–Voight.

5. Characters

Conductor versus modulus.

5.1. Galois orbits and labels.
Orbit codes.

5.2. Conrey labels.

5.3. Compatibility.
It is not enough just to embed the character field into the coefficient field (which is already hard when the coefficient field is big), you need to embed it in such a way that the character values are compatible with the Hecke action, and it can take a non-trivial amount of time to work this out (matching up roots of unity).

We could avoid this by keeping track of the coefficient field of the form as a relative extension of a cyclotomic field, but that creates some headaches comparing across implementations and does not allow us to represent eigenvalues in terms of a nice LLL-reduced basis (see below).

So we compute the values of χ on generators for $(\mathbb{Z}/N\mathbb{Z})^\times$ in terms of the Hecke field.

6. A sample of the implementations

Three methods: modular symbols in Magma and Sage, exact trace formula in Pari, Brandt matrices, ternary Birch. Not really a user's guide, but something.

Weight 1 in Sage is slow.

7. Issues: computational, theoretical, and practical

7.1. **Analytic conductor.**
Ordering proposed by Sarnak. ..., Roberts, roughly Nk^2.

7.2. **Atkin–Lehner eigenvalues.**
The Atkin–Lehner operators W_M for $M \parallel N$ map $S_k(N,\chi) \to S_k(N,\chi')$ with some explicit dependence of χ' on χ, M, N. In general, χ' is different from χ so they are no use for splitting up spaces. We have $\chi' = \chi$ when the character is quadratic, but Magma computes the operator only for trivial character so we follow suit.

[[JV: Mention the Fricke involution.]]

7.3. **Computing Hecke polynomials.**
Basically, we should be able to do this modulo primes and reconstruct.

```
Regarding your comment about the Hecke cutters, these are actually
minpolys of T_p, and for the big spaces Magma can be really slow at
computing them, even just for T_2 (which for the 4760,4760 example would
already be enough, in fact the trace of a_2 down to Q already
distinguishes these spaces).

I see -- presumably magma had done a similar computation to find the
splitting but does not give back the "certificate" after doing so.
I wonder if we could persuade magma to all for that.

A while back I reported on one case I had where the dimension was
162*80, so the Hecke matrices were 80x80 over Q(chi) of degree 162,
for which computing the char poly of T2 was almost hopeless.  (This
was level 3^6=729 and a character of order 2*3^4.)  I mentioned it
to Bill Hart when I saw him 3 or 4 weeks ago, he asked me to send
him the matrix which I did, and a few days after that I heard from
Claus Fieker who had a probably result (the char poly) which he
had computed mod p for lots of primes (choosing those which split
completely in Q(chi) as I had been doing) and then Chinese Remaindering
these.  He could not certify the result only because I had not told
him that we know a theoretical bound on the eigenvalues and hence
on the coefficients.

It would be nice of Magma (and the others) would use such modular
methods automatically.
```

```
Andrew,

> I can't guarantee this will hit the same code path internally, but
> externally it is doing exactly the same thing:
>
>> chi := FullDirichletGroup(29).1^2;
>> k := 2;
>> time S :=
>> NewformDecomposition(NewSubspace(CuspidalSubspace(ModularSymbols(chi,k,-1))));
> Time: 0.020
>> S;
> [
> Modular symbols space of level 29, weight 2, character $.1^2, and
> dimension 2 over Cyclotomic Field of order 28 and degree 12
> ]

Thanks for that.

The reason for the crash is now understood and a patch is being
developed.

My colleague Allan Steel looked at the slowness of your computation
and found (not surprisingly) that it arises from linear algebra over
Q or a number field. In particular, in this computation the bottlenecks
were the computation of the min polynomial of an element of a Hecke
algebra and then computing a huge resultant in order to factor the
min polynomial (mainly the latter). Allan hacked the code for this
example introducing some parallelism and this reduced the runtime
down to 40 minutes using 16 cores. The answer is

[
     Modular symbols space of level 743, weight 2, character $.1^2, and
dimension
     61 over Cyclotomic Field of order 742 and degree 312
]

This was the internal standard function to factor a polynomial f over a
number field K=Q(alpha).  This uses Trager's algorithm, which computes
the norm N of f over K, factors N which is over Q, and then takes the GCD
of each factor with f back over K to get the true factors of f over K.

To compute the norm, we just use a resultant, of course (to eliminate
alpha); basically we get:

    Resultant(f, g, alpha), where g is min poly of alpha.

Overview here:

https://en.wikipedia.org/wiki/Factorization_of_polynomials#Factoring_over_algebraic_extensions

It would be nice to make improvements for the general case but this
will take some serious development.
```

7.4. **Efficiently recognizing irreducibility.** $N = 2$ is by far the most time-consuming case for magma: for $k > 400$ with $4 \mid k$, each space takes more than 12 hours CPU time.

Conjecture 7.4.1. *For all $k \geq 2$, the space $S_k(\Gamma_0(2))$ decomposes under the Atkin–Lehner operator w_2 into Hecke irreducible subspaces of dimensions $\lfloor d/2 \rfloor$ and $\lceil d/2 \rceil$ where $d = \dim_{\mathbb{C}} S_k(\Gamma_0(2))$.*

This is a Maeda-like conjecture that we verified up to weight $k \leq 400$, but with the additional prediction that the Atkin–Lehner operator splits the space as evenly as possible. For general N, the difference $\dim S_2(\Gamma_0(N))^+ - \dim S_2(\Gamma_0(N))^-$ can be expressed in terms of class numbers of imaginary quadratic fields (fixed points of Atkin–Lehner involutions), and so this difference is $\ll N^{1/2+\epsilon}$. Probably this is explicit enough for $N = 2$ to prove the dimensions above, and something similar can be said for N composite.

Anyway, we should be able to prove that the space is irreducible by working with just one Hecke polynomial (likely to be irreducible).

Question 7.4.2. Given a polynomial $f(x) \in \mathbb{Z}[x]$, is there a fast algorithm that with positive probability correctly identifies if f is irreducible (but gives no information otherwise)?

In other words, if you expect that a polynomial is irreducible, can you verify this quickly without factoring the polynomial? Perhaps factoring it modulo primes until it is clear that the Galois group of the polynomial is transitive (with the expectation that it is typically S_d, and it should be quick to verify this). This is different than the typical factorization methods in computer algebra systems, which compute a factorization p-adically and then reconstruct the factorization over $\mathbb{Z}$. Example spaces that took a long time: 659.2.g (took 86131s in Magma) and 443.3.h (took 845077s). Basically, any space that has dimension in the thousands.

7.5. **Trace form.** For the trace form, in general the coefficients a_n carry information that cannot be seen from just the prime coefficients a_p or even the prime power coefficients a_{p^e}.

In theory, the trace form uniquely determines the Galois orbit of newforms (because these traces determine L-function). Is there an effective bound? The spaces 1500.1.l.a and 1500.1.l.b have the same trace form up to precision 1000.

7.6. **Polredabs.** Unique polynomial, sometimes very hard to get, might try polredbest first and then polredabs to verify. Polredbest uses an LLL-reduced basis for an order, and thereby runs in deterministic polynomial time in the size of the input; whereas polredabs does this for the maximal order (which may require factoring).

Really important: take version of Pari = blank, Sage 8.3.

There were occasions where we could compute a Hecke polynomial, but polredabs did not terminate. For example, with the space 3901.1.j.b, the coefficient field is $\mathbb{Q}(\zeta_{205})$, relative degree 4 over $\mathbb{Q}(\chi) = \mathbb{Q}(\zeta_{41})$. It is not hard to guess this is correct, but to compute an explicit isomorphism required effort. (One can quickly guess the roots of unity in a number field by checking for roots of unity modulo many large primes, like what is done for torsion of abelian varieties.) Without this optimization, the output is an order of magnitude larger.

7.7. **Nicely presenting Hecke eigenvalues.** LLL-basis of Hecke order. Recognizing cyclotomic ring, writing in terms of powers of ζ is nice for weight 1.

Matrix in terms of powers of ν, inverse matrix is smaller in size.

Example 7.7.1. Precision can be an issue in getting 1 as a shortest vector: using default precision might find no basis vector that is a root of unity (including 1). For example, the newforms in the space 14.16.c have an issue when the precision is too low.

In our computations, we always use precision that is at least as large as the discriminant of the Hecke ring.

The inverse matrix is noticeably smaller.

Remark 7.7.2. In the above, we have been concentrating on the case where f is an newform, representing a Galois orbit of newforms, and we write down its q-expansion in terms of a $\mathbb{Z}$-basis for the Hecke ring.

As an alternative, we can consider the $\mathbb{C}$-vector space spanned by f and its conjugates under $\operatorname{Aut}(\mathbb{C})$, making a $\mathbb{C}$-vector space of dimension say d. These conjugates will include conjugates that *do not* preserve the character, so we would either be working implicitly in the direct sum of the spaces over the full Galois orbit of characters, or we need to restrict to quadratic characters, or we only consider conjugates under $\operatorname{Aut}(\mathbb{C} \mid \mathbb{Q}(\chi))$ and get a $\mathbb{Q}(\chi)$-vector space. Anyway, inside this space is a canonical $\mathbb{Q}$-subspace, namely, those forms whose q-expansion belongs to $\mathbb{Q}[[q]]$. So we could instead represent the Galois orbit canonically by an echelonized basis of d individual q-expansions with coefficients in $\mathbb{Q}$. We could then write a representative newform as before as a linear combination of this basis over the Hecke field.

To go from the eigenform to the $\mathbb{Q}$-basis, we apply the operators $\operatorname{Tr}(\beta_i f)$ for β_i any $\mathbb{Q}$-basis for the Hecke field. (To go from the $\mathbb{Q}$-basis to an eigenform one needs to retain sufficiently many eigenvalues to do the linear algebra. In other words, the eigenform contains more information than the $\mathbb{Q}$-basis.) This generalizes the trace form, which is where we take $\beta_i = \beta_0 = 1$.

We could also work integrally and take the $\mathbb{Z}$-module of forms whose q-expansions belong to $\mathbb{Z}$ and then take a LLL-reduced basis which minimizes a (weighted) sum of finitely many coefficients.

It is conceivable that in a world where linear algebra over $\mathbb{Q}$ is much faster than linear algebra over number fields that we could succeed in computing a $\mathbb{Q}$-basis in reasonable time but not succeed in computing an eigenform.

8. Computing L-functions rigorously

Discuss both the $\mathbb{C}$-primitive L-function and $\mathbb{Q}$-primitive (product) L-functions
Labels for complex embeddings

8.1. **Verifying the analytic rank.** Maarten Derickx.

9. An overview of the computation

9.1. **Data extent.** Our data set covers $S_k(\Gamma_1(N))$ for all (N, k) satisfying $Nk^2 \leq 4000$. This data consists of 247 438 spaces $S_k(\Gamma_0(N), \chi)$ of which 30 738 are nonempty, a total of 67 180 Galois orbits of newforms, giving rise to a total of 9 966 498 modular forms as holomorphic functions. For more statistics on distribution of level, weight, dimension, and more, see the stats page.

In Magma, the data was computed using 64 cores [[AS: ...]] and the estimated total CPU time was [[AS: ...]].

For the newforms with coefficient field of absolute degree $d \leq 20$, we computed algebraic a_n for $1 \leq n \leq 1000$ in the LLL-basis described in section 7.7. This step dominated the running time. In all cases, this exceeds the Sturm bound.

For Pari, the data was [[JC: Just one small comment / reminder: the vast majority of the spaces are fast or very fast, but there is a small minority which take days or worse. If I were to start on the range 4k-5k my gues is that within a week I might get 95% of the spaces, but that the remainder would take another couple of weeks at least. That's only a very rough guess.]]

The exact data from Magma and from Pari was compared and it agrees.

For the complex data (computed using the trace formula), we have data in the following boxes:

- $N \leq 999$, $k \leq 4$, and in this range rigorous a_{p^n} for $p^n < 2000$;
- $N \leq 99$, $k \leq 12$; and
- $N \leq 30$, $k \leq 30$.

[[JV: This is just what I scribbled down, it is probably wrong.]]

9.2. **Comparing implementations.**

```
Stats for Nk^2 <= 1000 in Magma and Pari.

$ wc mfdata_1000.m.txt
    5533     5533 82782397 mfdata_1000.m.txt
$ wc mfdata_1000.g.txt
    5533     5533 82955633 mfdata_1000.g.txt

sage: gdata = read_dtp("t1000")
Read 5533 spaces of which 2653 are nontrivial; 4843 Galois orbits.
3707 orbits have dimension <=20
largest three dimsensions: [1404, 1824, 2016]
Total time = 120960.448
Max time = 11638.884 for space (237, 2, 14)
Average time (all spaces)      = 21.862
Average time (nonzero spaces)  = 45.327

sage: mdata = read_dtp("mfdata_1000.m.txt")
Read 5533 spaces of which 2653 are nontrivial; 4843 Galois orbits.
3707 orbits have dimension <=20
largest three dimsensions: [1404, 1824, 2016]
Total time = 158823.160
Max time = 2685.130 for space (227, 2, 3)
Average time (all spaces)      = 28.705
Average time (nonzero spaces)  = 59.853
```

So the current gp+sage code is faster. However there are some individual spaces for which there is a huge time discrepancy, and we are not 100% sure that there are not tricks left. The wall time for the Pari run for $Nk^2 \leq 1000$ was about 3 hours on 37 cores but the vast majority were done much faster, just a few outliers took ages.

A space where pari was a lot slower than Magma: 237.2.n took 283s in magma and 11638s in pari. The hard case seems to be when there is more than one irreducible piece and (perhaps) at least one piece has quite large dimension.

A space where Magma was a lot slower than pari: 227.2.c took 2685s in magma and 66s in pari. If you know the space is irreducible, then all we computed was the trace form and that is exactly what the trace formula is designed to give you.

Question 9.2.1. Can one get the trace forms when the space is not irreducible without decomposing the space, computing the eigenvalues, and taking a trace?

9.3. **Interesting and extreme behavior.** The largest Hecke irreducible subspace in our data set is 983.2.c.a, having dimension 39690.

The index of the Hecke ring in the Hecke field can be surprisingly large. [[JV: Example]]

9.4. **Data reliability.** The data agrees.

10. Twisting

In this section, we discuss issues of twists. Convenient background references are articles by Ribet [9, 10].

10.1. **Definitions.** Let $f \in S_k(\Gamma_1(N))$ be a newform with $N, k \in \mathbb{Z}_{\geq 1}$ and let $\mathbb{Q}(f) := \mathbb{Q}(\{a_n(f)\}_n) \subseteq \mathbb{C}$ be its Hecke field. Let ψ be a *nontrivial* Dirichlet character. Then there is a unique newform $g := f \otimes \psi$ with the property that

$$a_n(g) = \psi(n) a_n(f) \quad \text{for all } n \text{ coprime to } N \operatorname{cond}(\psi); \tag{10.1.1}$$

we call g the twist of f by ψ. By the Hecke recurrence, (10.1.1) is equivalent to the condition

$$a_p(g) = \psi(p) a_p(f) \quad \text{for all } p \nmid N \operatorname{cond}(\psi). \tag{10.1.2}$$

Lemma 10.1.3. *If $f \in S_k(\Gamma_0(N), \chi)$, then $f \otimes \psi \in S_k(\Gamma_0(M), \chi\psi^2)$ where*

$$M = \operatorname{lcm}(N, \operatorname{cond}(\psi)^2, \operatorname{cond}(\chi)\operatorname{cond}(\psi)). \tag{10.1.4}$$

Proof. Shimura, Proposition 3.64. □

Definition 10.1.5. We say that f has inner twist (or extra twist) by ψ if there exists $\sigma \in \operatorname{Aut}(\mathbb{Q}(f))$ such that $f = \sigma(f) \otimes \psi$. We say that f has self-twist by ψ if $f = f \otimes \psi$.

In other words, a self-twist is an inner twist where we may take $\sigma = \mathrm{id}$ the identity. The twist is said to be "inner" because twisting stays "within" the Galois orbit of f. Note we do not include the trivial character, so there is no self-twist by the trivial character.

Lemma 10.1.6. *Having inner twist or self-twist is well-defined on the Galois orbit of f.*

Proof. Easy check. □

Proposition 10.1.7 (Ribet)**.** *If $f \in S_k(\Gamma_1(N))$ has self-twist by ψ, then ψ is a quadratic character; if moreover $k \geq 2$, then ψ is associated to an imaginary quadratic field.*

Proof. By Lemma 10.1.3, for a self-twist by ψ we must have $\chi = \chi\psi^2$ so ψ is quadratic. For the second statement, see Ribet [9, Theorem (4.5)]. □

In light of Proposition 10.1.7, we make the following definition.

Definition 10.1.8. We say f has real multiplication (RM) if f has self-twist by a character attached to a real quadratic field and complex multiplication (CM) if f has self-twist by a character attached to an imaginary quadratic field.

Example 10.1.9. Forms of weight 1 have self-twist if and only if they have dihedral projective image. Indeed, if $f \in S_1(\Gamma_1(N))$ has self-twist by the quadratic character ψ, then $a_p(f) = 0$ for all p that are inert in $\mathbb{Q}(\psi)$ (asymptotically half of the primes), and by classification (see below) this happens if and only if the projective image is dihedral.

We can say more: suppose f is dihedral, and let K be the fixed field of the kernel of the projective Galois representation associated to such a form f, so $\operatorname{Gal}(K \,|\, \mathbb{Q}) \simeq D_n$ (a group of order $2n$). Then for each quadratic subfield $F \subseteq K$, the form f has self-twist by the character associated to F. Accordingly, when $n > 2$ the subfield F and associated self-twist character are unique, and when $n = 2$ (so K is biquadratic) there are three distinct subfields and corresponding characters.

These quadratic subfields may be real or imaginary, so there are weight 1 forms with RM or CM (or both). The forms with RM arise precisely from ray class characters of a real quadratic field that are of mixed signature (so even at one real place and odd at another).

Remark 10.1.10. In view of Proposition 10.1.7, it is common in the literature to just replace the term *self-twist* by *complex multiplication*. There is no harm for weight $k \geq 2$ (where the associated forms may be constructed using Hecke Grössencharacters of imaginary quadratic fields), but for weight $k = 1$ we think this is potentially confusing, and we want to avoid saying "f has complex multiplication by the character attached to $\mathbb{Q}(\sqrt{5})$."

Remark 10.1.11. Among the forms of weight $k = 2$, trivial character, and dimension 2, we can search for forms with inner twist and we should see a table that matches Cremona [1, Table 3] up to level $N \leq 300$. The lists match with one exception: we found one form 169.2.a.a that was missed by Cremona.

Example 10.1.12. CM modular forms may also have an inner twist that is not a self-twist: the smallest example by analytic conductor is 52.1.j.a has CM by $\mathbb{Q}(\sqrt{-1})$ and inner twist. This phenomenon is not restricted to weight 1, for example 20.2.e.a.

10.2. Computing inner twists.

Lemma 10.2.1. *If $f \in S_k(\Gamma_0(N), \chi)$ has inner twist by ψ, then ψ is unramified away from N.*

Proof. If $p \mid \operatorname{cond}(\psi)$ and $p \nmid N$, then p will divide the level of $f \otimes \psi$, contradicting that the level of $\sigma(f)$ is N. □

The simplest algorithm is just loop over the finite list of characters and test equality up to the Sturm bound for level M in (10.1.4).

Another algorithm computes $\operatorname{Aut}(\mathbb{Q}(f))$ and then tries to match a Dirichlet character $\psi(p) = \sigma(a_p(f))/a_p(f)$ for $a_p(f) \neq 0$ and for each $\sigma \in \operatorname{Aut}(\mathbb{Q}(f))$.

One trick: computing the minimal polynomial of $a_p(f)$ and writing it as a polynomial in x^s with $s \in \mathbb{Z}_{\geq 1}$ as large as possible, the order of $\psi(p)$ as a root of unity is a divisor of s.

Another possible trick: compute a minimal twist, compute its inner twists (hopefully easier because the level is smaller), then recover the full set of inner twists.

10.3. Sato–Tate group.

The twisting behavior determines the Sato-Tate group in an explicit way.

11. Weight 1

11.1. Computational observations.

A large amount of time is spent to prove that a (new) space is zero-dimensional. For example, the space 3900.1.x has no newforms, which required 215 CPU hours and over 250 GB in Pari.

Some weight 1 forms with projective image D_2 can have very large kernel fields.

11.2. Classifying the projective Galois images.

Described by Buzzard–Lauder. In Magma, there is separate functionality to compute the dihedral forms, so this is a bit of reverse engineering. Apparently Pari (|mfgaloistype—) can classify the projective image (using `mfgaloistype`).

We made some further observations...

11.3. Computing the associated Artin representation.

Along the way, we found small number fields to add to the LMFDB: for example the S_4-quartic field defined by $x^4 - 401x - 8421$ arises as the projective image for a weight 1 form of level 1203.

Pari can only provide projective kernel polynomials for the A_4 and S_4 cases.

Conversely, we matched all 4375 odd 2-dimensional Artin reps of conductor ≤ 4000 stored in the LMFDB to weight one forms in our data.

12. Final remarks

Future work:

- Extend data set: for trivial character, compute $Nk^2 \leq 40000$ (necessarily even weight, will surely be dominated by $k = 2$). This is in progress.
- Make a database of modular and Shimura curves.
- Make a database of modular abelian varieties, including (geometric) endomorphism algebra.
- [[JV: Mention Dohyeong Kim is computing $k = 2$ and trivial character and quadratic Hecke field using linear combinations of T_1, T_p, T_q?]]

References

[1] J. E. Cremona, *Abelian varieties with extra twist, cusp forms, and elliptic curves over imaginary quadratic fields*, J. London Math. Soc. (2) **45** (1992), no. 3, 404–416.

[2] J. E. Cremona, *Algorithms for modular elliptic curves*, 2nd ed., Cambridge University Press, Cambridge, 1997.

[3] John Cremona, *The L-functions and modular forms database project*, Found. Comput. Math. **16** (2016), no. 6, 1541–1553.

[4] Lloyd J.P. Kilford, *Modular forms: A classical and computational introduction*, 2nd ed., Imperial College Press, London, 2015.

[5] W. Bosma, J. Cannon, and C. Playoust, *The Magma algebra system. I. The user language*, J. Symbolic Comput. **24** (3–4), 1997, 235–265.

[6] Christian Meyer, *Newforms of weight two for $\Gamma_0(N)$ with rational coefficients*, http://www2.mathematik.uni-mainz.de/alggeom/cm/weight2.pdf, 2005.

[7] Christian Meyer, *Newforms of weight four for $\Gamma_0(N)$ with rational coefficients*, http://www2.mathematik.uni-mainz.de/alggeom/cm/weight4.pdf, 2005.

[8] Toshitsune Miyake, *Modular forms*, Translated from the 1976 Japanese original by Yoshitaka Maeda, Springer Monographs in Math., Springer-Verlag, Berlin, 2006.

[9] Kenneth A. Ribet, *Galois representations attached to eigenforms with Nebentypus*, Modular functions of one variable, V (Proc. Second Internat. Conf., Univ. Bonn, Bonn, 1976), Lecture Notes in Math., vol. 601, Springer, Berlin, 1977, 17–51.

[10] Kenneth A. Ribet, *Twists of modular forms and endomorphisms of abelian varieties*, Math. Ann. **253** (1980), no. 1, 43–62.

[11] Nils-Peter Skoruppa and Don Zagier, *Jacobi forms and a certain space of modular forms*, Invent. Math. **94** (1988), no. 1, 113–146.

[12] William Stein, *The Modular Forms Database*, http://wstein.org/Tables/.

[13] William Stein, *Modular forms, a computational approach*, with an appendix by Paul E. Gunnells, Graduate Studies in Math., vol. 79, Amer. Math. Soc., Providence, 2007.

[14] William A. Stein, *An introduction to computing modular forms using modular symbols*, Algorithmic number theory: lattices, number fields, curves and cryptography, Math. Sci. Res. Inst. Publ., vol. 44, Cambridge Univ. Press, Cambridge, 2008, 641–652.

[15] Hideo Wada, *Tables of Hecke operators. I*, Seminar on Modern Methods in Number Theory (Inst. Statist. Math., Tokyo, 1971), Paper No. 39, Inst. Statist. Math., Tokyo, 1971.

[16] Hideo Wada, *A table of Hecke operators. II*, Proc. Japan Acad. **49** (1973), 380–384.

Department of Mathematics, Dartmouth College, 6188 Kemeny Hall, Hanover, NH 03755, USA
E-mail address: jvoight@gmail.com
URL: http://www.math.dartmouth.edu/~jvoight/