Modularity of the Abelian Surface of Conductor 277

David S. Yuen Lake Forest College

joint work with

Armand Brumer Fordham University
Cris Poor Fordham University
John Voight Dartmouth College

Modular Forms and Curves of Low Genus: **Computational** Aspects
ICERM, September 2015

Outline of talk

1. Paramodular Conjecture and evidence.
2. The abelian surface of conductor 277 and the paramodular form f_{277}.
3. Computing eigenvalues by specialization.
4. Making floating point calculations rigorous.
5. Results.

All abelian surfaces $A/\mathbb{Q}$ are paramodular

Paramodular Conjecture (Brumer and Kramer 2009)

Let $N \in \mathbb{N}$. There is a bijection between

1. *isogeny classes of abelian surfaces $A/\mathbb{Q}$ with conductor N and endomorphisms $\mathrm{End}_{\mathbb{Q}}(A) = \mathbb{Z}$,*
2. *lines of Hecke eigenforms $f \in S_2(K(N))^{\mathrm{new}}$ that have rational eigenvalues and are not Gritsenko lifts from $J_{2,N}^{\mathrm{cusp}}$.*

In this correspondence we have

$$L(A, s, \text{Hasse-Weil}) = L(f, s, \text{spin}).$$

Do the arithmetic and automorphic data match up?

Looks like it.

1997: Brumer makes a (short) list of $N < 1000$ that could possibly be the conductor of an abelian surface $A/\mathbb{Q}$.

Theorem (PY 2009)

Let $p < 600$ be prime. If $p \notin \{277, 349, 353, 389, 461, 523, 587\}$ then $S_2(K(p))$ consists entirely of Gritsenko lifts.

This list of primes $\{277, \ldots, 587\}$ exactly matches Brumer's "Yes there is an abelian surface" list for prime levels.

This is a lot of evidence for the Paramodular Conjecture because prime levels $p < 600$ that don't have abelian surfaces over $\mathbb{Q}$ also don't have paramodular cusp forms beyond the Gritsenko lifts.

Proof.

We can inject the weight two space into weight four spaces:

1) For $g_1, g_2 \in \text{Grit}\left(J_{2,p}^{\text{cusp}}\right) \subseteq S_2\left(K(p)\right)$, we have the injection:

$$\begin{aligned} S_2(K(p)) &\hookrightarrow \{(H_1, H_2) \in S_4(K(p)) \times S_4(K(p)) : g_2 H_1 = g_1 H_2\} \\ f &\mapsto (g_1 f, g_2 f) \end{aligned}$$

2) The dimensions of $S_4(K(p))$ are known by Ibukiyama; we still have to span $S_4(K(p))$ by computing products of Gritsenko lifts, traces of theta series and by smearing with Hecke operators.

3) Millions of Fourier coefficients mod 109 later,

$$\dim S_2(K(p)) \leq \dim\{(H_1, H_2) \in S_4(K(p)) \times S_4(K(p)) : g_2 H_1 = g_1 H_2\}$$

4) When the dimension might be bigger, this method also gives a candidate nonlift as a quotient of a known weight 4 form divided by a known weight 2 form.

Equality of L-series modularity examples

- The lift to a paramodular Hecke eigenform of a certain Hilbert modular form over a real quadratic field by Johnson-Leung and Roberts (2012) is modular with respect to a certain abelian surface. Some work of Dembélé and Kumar is related to this. An example has conductor 193^2.

Equality of L-series modularity examples

- The lift to a paramodular Hecke eigenform of a certain Hilbert modular form over a real quadratic field by Johnson-Leung and Roberts (2012) is modular with respect to a certain abelian surface. Some work of Dembélé and Kumar is related to this. An example has conductor 193^2.

- For a similar but different example, constructing a lift from Bianchi modular forms, Berger, Dembélé, Pacetti, Sengun used an imaginary quadratic number field. An example is conductor $N = 223^2$.

Conductor 277

The form and the surface

- (Theorem PY 2009) $\dim S_2(K(277)) = 11$ but $\dim J^{\mathrm{cusp}}_{2,277} = 10$. There is a Hecke eigenform $f_{277} \in S_2(K(277))$ that is not a Gritsenko lift.
- $\mathcal{A}_{277}$ is the Jacobian of the hyperelliptic curve

$$y^2 + y = x^5 + 5x^4 + 8x^3 + 6x^2 + 2x$$

Conductor 277

The form and the surface

- (Theorem PY 2009) $\dim S_2(K(277)) = 11$ but $\dim J^{\mathrm{cusp}}_{2,277} = 10$. There is a Hecke eigenform $f_{277} \in S_2(K(277))$ that is not a Gritsenko lift.
- $\mathcal{A}_{277}$ is the Jacobian of the hyperelliptic curve

$$y^2 + y = x^5 + 5x^4 + 8x^3 + 6x^2 + 2x$$

- Magma will compute lots of Euler factors for $L(\mathcal{A}_{277}, s, \text{H-W})$

Conductor 277

The form and the surface

- (Theorem PY 2009) $\dim S_2(K(277)) = 11$ but $\dim J^{\mathrm{cusp}}_{2,277} = 10$. There is a Hecke eigenform $f_{277} \in S_2(K(277))$ that is not a Gritsenko lift.
- $\mathcal{A}_{277}$ is the Jacobian of the hyperelliptic curve

$$y^2 + y = x^5 + 5x^4 + 8x^3 + 6x^2 + 2x$$

- Magma will compute lots of Euler factors for $L(\mathcal{A}_{277}, s, \mathrm{H\text{-}W})$
- By contrast, it takes much more work to compute Euler factors for $L(f_{277}, s, \mathrm{spin})$, and one of the main goals of this talk is to present one method for computing high eigenvalues of f_{277}.

Conductor 277

The form and the surface

- (Theorem PY 2009) $\dim S_2(K(277)) = 11$ but $\dim J^{\mathrm{cusp}}_{2,277} = 10$. There is a Hecke eigenform $f_{277} \in S_2(K(277))$ that is not a Gritsenko lift.
- $\mathcal{A}_{277}$ is the Jacobian of the hyperelliptic curve

$$y^2 + y = x^5 + 5x^4 + 8x^3 + 6x^2 + 2x$$

- Magma will compute lots of Euler factors for $L(\mathcal{A}_{277}, s, \text{H-W})$
- By contrast, it takes much more work to compute Euler factors for $L(f_{277}, s, \text{spin})$, and one of the main goals of this talk is to present one method for computing high eigenvalues of f_{277}.
- In 2009, we computed the 2, 3 and 5 Euler factors of $L(f_{277}, s, \text{spin})$ and they agree with those of $L(\mathcal{A}_{277}, s, \text{H-W})$.

How can we prove a weight two nonlift cusp form exists?

How can we prove a weight two nonlift cusp form exists?

Proof (2009).

1) We have a candidate $f = H_1/g_1 \in M_2^{\mathrm{mero}}(K(p))$.

2) Find a weight four cusp form $F \in S_4(K(p))$ and prove

$$F\, g_1^2 = H_1^2 \text{ in } S_8(K(p)).$$

Since $F = \left(\frac{H_1}{g_1}\right)^2$ is holomorphic, so is $f = \frac{H_1}{g_1}$.

How can we prove a weight two nonlift cusp form exists?

Proof (2009).

1) We have a candidate $f = H_1/g_1 \in M_2^{\mathrm{mero}}(K(p))$.

2) Find a weight four cusp form $F \in S_4(K(p))$ and prove

$$F\, g_1^2 = H_1^2 \text{ in } S_8(K(p)).$$

Since $F = \left(\frac{H_1}{g_1}\right)^2$ is holomorphic, so is $f = \frac{H_1}{g_1}$. □

Proof (new).

We make a Borcherds product in $S_2(K(p))$ and show the Borcherds product is not a Gritsenko lift. □

Formula for f_{277}

One presentation of f_{277} is

$$\begin{aligned} f_{277} = (&-14G_1^2 - 20G_8G_2 + 11G_9G_2 + 6G_2^2 - 30G_7G_{10} + 15G_9G_{10} \\ &+ 15G_{10}G_1 - 30G_{10}G_2 - 30G_{10}G_3 + 5G_4G_5 + 6G_4G_6 + 17G_4G_7 \\ &- 3G_4G_8 - 5G_4G_9 - 5G_5G_6 + 20G_5G_7 - 5G_5G_8 - 10G_5G_9 - 3G_6^2 \\ &+ 13G_6G_7 + 3G_6G_8 - 10G_6G_9 - 22G_7^2 + G_7G_8 + 15G_7G_9 + 6G_8^2 \\ &- 4G_8G_9 - 2G_9^2 + 20G_1G_2 - 28G_3G_2 + 23G_4G_2 + 7G_6G_2 \\ &- 31G_7G_2 + 15G_5G_2 + 45G_1G_3 - 10G_1G_5 - 2G_1G_4 - 13G_1G_6 \\ &- 7G_1G_8 + 39G_1G_7 - 16G_1G_9 - 34G_3^2 + 8G_3G_4 + 20G_3G_5 \\ &+ 22G_3G_6 + 10G_3G_8 + 21G_3G_9 - 56G_3G_7 - 3G_4^2)/ \\ &(-G_4 + G_6 + 2G_7 + G_8 - G_9 + 2G_3 - 3G_2 - G_1). \end{aligned}$$

for some ten Gritsenko lifts $G_1, \ldots, G_{10}$.

Formula for f_{277}

One presentation of f_{277} is

$$\begin{aligned} f_{277} = (&-14G_1^2 - 20G_8G_2 + 11G_9G_2 + 6G_2^2 - 30G_7G_{10} + 15G_9G_{10} \\ &+ 15G_{10}G_1 - 30G_{10}G_2 - 30G_{10}G_3 + 5G_4G_5 + 6G_4G_6 + 17G_4G_7 \\ &- 3G_4G_8 - 5G_4G_9 - 5G_5G_6 + 20G_5G_7 - 5G_5G_8 - 10G_5G_9 - 3G_6^2 \\ &+ 13G_6G_7 + 3G_6G_8 - 10G_6G_9 - 22G_7^2 + G_7G_8 + 15G_7G_9 + 6G_8^2 \\ &- 4G_8G_9 - 2G_9^2 + 20G_1G_2 - 28G_3G_2 + 23G_4G_2 + 7G_6G_2 \\ &- 31G_7G_2 + 15G_5G_2 + 45G_1G_3 - 10G_1G_5 - 2G_1G_4 - 13G_1G_6 \\ &- 7G_1G_8 + 39G_1G_7 - 16G_1G_9 - 34G_3^2 + 8G_3G_4 + 20G_3G_5 \\ &+ 22G_3G_6 + 10G_3G_8 + 21G_3G_9 - 56G_3G_7 - 3G_4^2)/ \\ &(-G_4 + G_6 + 2G_7 + G_8 - G_9 + 2G_3 - 3G_2 - G_1). \end{aligned}$$

for some ten Gritsenko lifts $G_1, \ldots, G_{10}$.
This can be used to prove f_{277} has integer coefficients.

And if you must know...

And if you must know...
$G_i = \mathrm{GritLift}(\phi_i)$, where ϕ_i is a theta block given by...

$$\phi_1 = \mathrm{TB}_2(2,4,4,4,5,6,8,9,10,14)$$
$$\phi_2 = \mathrm{TB}_2(2,3,4,5,5,7,7,9,10,14)$$
$$\phi_3 = \mathrm{TB}_2(2,3,4,4,5,7,8,9,11,13)$$
$$\phi_4 = \mathrm{TB}_2(2,3,3,5,6,6,8,9,11,13)$$
$$\phi_5 = \mathrm{TB}_2(2,3,3,5,5,8,8,8,11,13)$$
$$\phi_6 = \mathrm{TB}_2(2,3,3,5,5,7,8,10,10,13)$$
$$\phi_7 = \mathrm{TB}_2(2,3,3,4,5,6,7,9,10,15)$$
$$\phi_8 = \mathrm{TB}_2(2,2,4,5,6,7,7,9,11,13)$$
$$\phi_9 = \mathrm{TB}_2(2,2,4,4,6,7,8,10,11,12)$$
$$\phi_{10} = \mathrm{TB}_2(2,2,3,5,6,7,9,9,11,12)$$

where a theta block is a product of theta functions and eta functions:
$\mathrm{TB}_2(d_1,\ldots,d_{10}) = \eta^{-6}\prod_{i=1}^{10}\vartheta(\tau,d_i z)$.

What eigenvalues are needed to prove modularity?

A. Brumer tells us that to prove A_{277} is modular, we need to prove that the eigenvalues λ_p for it and f_{277} are equal for p in the following set:

$$\mathcal{B} = \{2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 53, 59, 97\}$$

Methods for computing eigenvalues.

The smallest determinant nonzero Fourier coefficient of f_{277} is $a(t_0, f_{277})$ where

$$t_0 = \begin{pmatrix} 49 & 233/2 \\ 233/2 & 277 \end{pmatrix}.$$

Note $\det(t_0) = 3/4$. Some methods for computing $T(p)$ are:

Methods for computing eigenvalues.

The smallest determinant nonzero Fourier coefficient of f_{277} is $a(t_0, f_{277})$ where

$$t_0 = \begin{pmatrix} 49 & 233/2 \\ 233/2 & 277 \end{pmatrix}.$$

Note $\det(t_0) = 3/4$. Some methods for computing $T(p)$ are:

- Expand f_{277} using multiplication and division of the Gritsenko lift (3-variable power series) expanded out to coefficient matrices of determinant $3p^2/4$.

Methods for computing eigenvalues.

The smallest determinant nonzero Fourier coefficient of f_{277} is $a(t_0, f_{277})$ where

$$t_0 = \begin{pmatrix} 49 & 233/2 \\ 233/2 & 277 \end{pmatrix}.$$

Note $\det(t_0) = 3/4$. Some methods for computing $T(p)$ are:

- Expand f_{277} using multiplication and division of the Gritsenko lift (3-variable power series) expanded out to coefficient matrices of determinant $3p^2/4$.
- Expand f_{277} to p Fourier-Jacobi coefficients using "Jacobi Restriction".

Methods for computing eigenvalues.

The smallest determinant nonzero Fourier coefficient of f_{277} is $a(t_0, f_{277})$ where

$$t_0 = \begin{pmatrix} 49 & 233/2 \\ 233/2 & 277 \end{pmatrix}.$$

Note $\det(t_0) = 3/4$. Some methods for computing $T(p)$ are:

- Expand f_{277} using multiplication and division of the Gritsenko lift (3-variable power series) expanded out to coefficient matrices of determinant $3p^2/4$.
- Expand f_{277} to p Fourier-Jacobi coefficients using "Jacobi Restriction".
- Specialize by plugging something into the argument for f_{277} and $f_{277}|T(p)$.

Methods for computing eigenvalues.

The smallest determinant nonzero Fourier coefficient of f_{277} is $a(t_0, f_{277})$ where

$$t_0 = \begin{pmatrix} 49 & 233/2 \\ 233/2 & 277 \end{pmatrix}.$$

Note $\det(t_0) = 3/4$. Some methods for computing $T(p)$ are:

- Expand f_{277} using multiplication and division of the Gritsenko lift (3-variable power series) expanded out to coefficient matrices of determinant $3p^2/4$.
- Expand f_{277} to p Fourier-Jacobi coefficients using "Jacobi Restriction".
- Specialize by plugging something into the argument for f_{277} and $f_{277}|T(p)$.
  - Plug in a specific numerical point and evaluate numerically. Get numbers.

Methods for computing eigenvalues.

The smallest determinant nonzero Fourier coefficient of f_{277} is $a(t_0, f_{277})$ where

$$t_0 = \begin{pmatrix} 49 & 233/2 \\ 233/2 & 277 \end{pmatrix}.$$

Note $\det(t_0) = 3/4$. Some methods for computing $T(p)$ are:

- Expand f_{277} using multiplication and division of the Gritsenko lift (3-variable power series) expanded out to coefficient matrices of determinant $3p^2/4$.
- Expand f_{277} to p Fourier-Jacobi coefficients using "Jacobi Restriction".
- Specialize by plugging something into the argument for f_{277} and $f_{277}|T(p)$.
  - Plug in a specific numerical point and evaluate numerically. Get numbers.
  - Specialize to a modular curve. Get elliptic modular forms.

Methods for computing eigenvalues.

The smallest determinant nonzero Fourier coefficient of f_{277} is $a(t_0, f_{277})$ where

$$t_0 = \begin{pmatrix} 49 & 233/2 \\ 233/2 & 277 \end{pmatrix}.$$

Note $\det(t_0) = 3/4$. Some methods for computing $T(p)$ are:

- Expand f_{277} using multiplication and division of the Gritsenko lift (3-variable power series) expanded out to coefficient matrices of determinant $3p^2/4$.
- Expand f_{277} to p Fourier-Jacobi coefficients using "Jacobi Restriction".
- Specialize by plugging something into the argument for f_{277} and $f_{277}|T(p)$.
  - Plug in a specific numerical point and evaluate numerically. Get numbers.
  - Specialize to a modular curve. Get elliptic modular forms.
  - Specialize to a Humbert surface. Get Hilbert modular forms.

Specialization to modular curves

Fix any positive definite 2×2 symmetric matrix s_0.
Let $f(\Omega) = \sum_t a(t; f) \exp(2\pi i \langle t, \Omega \rangle)$. Then

$$f(s_0\tau) = \sum_n \left(\sum_{t:\langle s_0, t\rangle = n} a(t; f) \right) q^n$$

Specialization to modular curves

Fix any positive definite 2×2 symmetric matrix s_0.
Let $f(\Omega) = \sum_t a(t; f) \exp(2\pi i \langle t, \Omega \rangle)$. Then

$$f(s_0\tau) = \sum_n \left(\sum_{t:\langle s_0, t\rangle = n} a(t; f) \right) q^n$$

For example, for $s_0 = \begin{pmatrix} 554 & 233 \\ 233 & 98 \end{pmatrix}$,

$$f_{277}(s_0\tau) = -3q^3 + 6q^6 + 6q^9 + O(q^{10})$$
$$(f_{277}|T(2))(s_0\tau) = 6q^3 - 12q^6 - 12q^9 + O(q^{10})$$

and hence $\lambda_2(f_{277}) = -2$.

Specialization to modular curves

Fix any positive definite 2×2 symmetric matrix s_0.
Let $f(\Omega) = \sum_t a(t; f) \exp(2\pi i \langle t, \Omega \rangle)$. Then

$$f(s_0\tau) = \sum_n \left(\sum_{t:\langle s_0, t\rangle = n} a(t; f) \right) q^n$$

For example, for $s_0 = \begin{pmatrix} 554 & 233 \\ 233 & 98 \end{pmatrix}$,

$$f_{277}(s_0\tau) = -3q^3 + 6q^6 + 6q^9 + O(q^{10})$$
$$(f_{277}|T(2))(s_0\tau) = 6q^3 - 12q^6 - 12q^9 + O(q^{10})$$

and hence $\lambda_2(f_{277}) = -2$.

How is the second of above two series computed?

Coset representatives for $T(p)$

When $p \nmid N$, we can use the following coset representatives in expressing $T(p)$ as a sum of cosets.

$$T(p) = K(N)\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & p & 0 \\ 0 & 0 & 0 & p \end{pmatrix} K(N) = \sum_{\ell=1}^{p^3+p^2+p+1} K(N)\begin{pmatrix} A_\ell & B_\ell \\ 0 & D_\ell \end{pmatrix}$$

$$= K(N)\begin{pmatrix} p & 0 & 0 & 0 \\ 0 & p & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} + \sum_{a \mod p} K(N)\begin{pmatrix} 1 & 0 & a & 0 \\ 0 & p & 0 & 0 \\ 0 & 0 & p & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$+ \sum_{a,b \mod p} K(N)\begin{pmatrix} p & 0 & 0 & 0 \\ a & 1 & 0 & b \\ 0 & 0 & 1 & -a \\ 0 & 0 & 0 & p \end{pmatrix} + \sum_{a,b,c \mod p} K(N)\begin{pmatrix} 1 & 0 & a & b \\ 0 & 1 & b & c \\ 0 & 0 & p & 0 \\ 0 & 0 & 0 & p \end{pmatrix}$$

Coset representatives for $T(p)$

When $p \nmid N$, we can use the following coset representatives in expressing $T(p)$ as a sum of cosets.

$$T(p) = K(N)\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & p & 0 \\ 0 & 0 & 0 & p \end{pmatrix} K(N) = \sum_{\ell=1}^{p^3+p^2+p+1} K(N)\begin{pmatrix} A_\ell & B_\ell \\ 0 & D_\ell \end{pmatrix}$$

$$= K(N)\begin{pmatrix} p & 0 & 0 & 0 \\ 0 & p & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} + \sum_{a \mod p} K(N)\begin{pmatrix} 1 & 0 & a & 0 \\ 0 & p & 0 & 0 \\ 0 & 0 & p & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$+ \sum_{a,b \mod p} K(N)\begin{pmatrix} p & 0 & 0 & 0 \\ a & 1 & 0 & b \\ 0 & 0 & 1 & -a \\ 0 & 0 & 0 & p \end{pmatrix} + \sum_{a,b,c \mod p} K(N)\begin{pmatrix} 1 & 0 & a & b \\ 0 & 1 & b & c \\ 0 & 0 & p & 0 \\ 0 & 0 & 0 & p \end{pmatrix}$$

So

$$f|T(p) = \sum_{\ell=1}^{p^3+p^2+p+1} f|\begin{pmatrix} A_\ell & B_\ell \\ 0 & D_\ell \end{pmatrix}.$$

Specializing after slashing by upper triangular

$$\begin{aligned}\left(f|\left(\begin{smallmatrix}A & B\\ 0 & D\end{smallmatrix}\right)\right)(s_0\tau) =& (\det D)^{-k}\det(AD)^{2k-3}f(As_0D^{-1}\tau + BD^{-1})\\ =& (\det D)^{-k}\det(AD)^{2k-3}\\ & \cdot\sum_{n\in\mathbb{Q}}\left(\sum_{t:\langle As_0D^{-1},t\rangle=n} a(t;f)\exp(2\pi i\langle BD^{-1},t\rangle)\right)q^n.\end{aligned}$$

Specializing after slashing by upper triangular

$$\begin{aligned}\left(f|\left(\begin{smallmatrix}A & B\\ 0 & D\end{smallmatrix}\right)\right)(s_0\tau) =& (\det D)^{-k}\det(AD)^{2k-3}f(As_0D^{-1}\tau + BD^{-1})\\ =& (\det D)^{-k}\det(AD)^{2k-3}\\ & \cdot\sum_{n\in\mathbb{Q}}\left(\sum_{t:\langle As_0D^{-1},t\rangle=n} a(t;f)\exp(2\pi i\langle BD^{-1},t\rangle)\right)q^n.\end{aligned}$$

Note that $\exp(2\pi i\langle BD^{-1},t\rangle)$ could yield a p^{th} root of unity.

Specialization of $f_{277}|T(p)$

Using the formula

$$f_{277} = \left(\sum_{i,j} \alpha_{ij} G_i G_j\right) / \left(\sum_j \beta_j G_j\right),$$

Specialization of $f_{277}|T(p)$

Using the formula

$$f_{277} = \left(\sum_{i,j} \alpha_{ij} G_i G_j\right) / \left(\sum_j \beta_j G_j\right),$$

we have

$$(f_{277}|T(p))(s_0\tau) = \sum_{\ell=1}^{p^3+p^2+p+1} \left(\left(\sum_{i,j} \alpha_{ij} g_{i\ell} g_{j\ell}\right) / \left(\sum_j \beta_j g_{j\ell}\right)\right),$$

where $g_{i\ell} = \left(G_i|\begin{pmatrix} A_\ell & B_\ell \\ 0 & D_\ell \end{pmatrix}\right)(s_0\tau)$ is a power series in one variable.

Specialization of $f_{277}|T(p)$

Using the formula

$$f_{277} = \left(\sum_{i,j} \alpha_{ij} G_i G_j\right) / \left(\sum_j \beta_j G_j\right),$$

we have

$$(f_{277}|T(p))(s_0\tau) = \sum_{\ell=1}^{p^3+p^2+p+1} \left(\left(\sum_{i,j} \alpha_{ij} g_{i\ell} g_{j\ell}\right) / \left(\sum_j \beta_j g_{j\ell}\right)\right),$$

where $g_{i\ell} = \left(G_i|\begin{pmatrix} A_\ell & B_\ell \\ 0 & D_\ell \end{pmatrix}\right)(s_0\tau)$ is a power series in one variable.

Of course, for computations, we compute with truncated power series.

The issue.

The individual slashes in the sum for $T(p)$ might have p^{th} roots of unity.
Solutions:

The issue.

The individual slashes in the sum for $T(p)$ might have p^{th} roots of unity. Solutions:

- Keep track of the primitive p^{th} root of unity ζ_p symbolically and perform operations in the field $\mathbb{Q}[\zeta_p]$, storing elements as polymomials with rational coefficients.

The issue.

The individual slashes in the sum for $T(p)$ might have p^{th} roots of unity. Solutions:

- Keep track of the primitive p^{th} root of unity ζ_p symbolically and perform operations in the field $\mathbb{Q}[\zeta_p]$, storing elements as polymomials with rational coefficients.
- Use floating point numbers with a specified precision and keep track of their radii of error.

The issue.

The individual slashes in the sum for $T(p)$ might have p^{th} roots of unity. Solutions:

- Keep track of the primitive p^{th} root of unity ζ_p symbolically and perform operations in the field $\mathbb{Q}[\zeta_p]$, storing elements as polymomials with rational coefficients.
- Use floating point numbers with a specified precision and keep track of their radii of error.

We choose the floating point number method.

IEEE standards for floating point calculations

- We use a programming package (such as GMP, MPFR, MPC) that follows IEEE standards for floating point calculations.

IEEE standards for floating point calculations

- We use a programming package (such as GMP, MPFR, MPC) that follows IEEE standards for floating point calculations.
- Choose a precision B. Denote the set of numbers representable by floating numbers of precision B by $\mathcal{C}_B$. So

$$\mathcal{C}_B \subset \pm(\{0, \ldots, 2^B - 1\} + i\{0, \ldots, 2^B - 1\}) * 2^{\mathbb{Z}}.$$

IEEE standards for floating point calculations

- We use a programming package (such as GMP, MPFR, MPC) that follows IEEE standards for floating point calculations.
- Choose a precision B. Denote the set of numbers representable by floating numbers of precision B by $\mathcal{C}_B$. So

$$\mathcal{C}_B \subset \pm(\{0, \ldots, 2^B - 1\} + i\{0, \ldots, 2^B - 1\}) * 2^{\mathbb{Z}}.$$

- Every standard operation (arithmetic, trigonometric, etc.) has user-specified rounding modes for the real and imaginary parts.

IEEE standards for floating point calculations

- We use a programming package (such as GMP, MPFR, MPC) that follows IEEE standards for floating point calculations.
- Choose a precision B. Denote the set of numbers representable by floating numbers of precision B by $\mathcal{C}_B$. So

$$\mathcal{C}_B \subset \pm(\{0, \ldots, 2^B - 1\} + i\{0, \ldots, 2^B - 1\}) * 2^{\mathbb{Z}}.$$

- Every standard operation (arithmetic, trigonometric, etc.) has user-specified rounding modes for the real and imaginary parts.
  - RNDN: Round to nearest

IEEE standards for floating point calculations

- We use a programming package (such as GMP, MPFR, MPC) that follows IEEE standards for floating point calculations.
- Choose a precision B. Denote the set of numbers representable by floating numbers of precision B by $\mathcal{C}_B$. So

$$\mathcal{C}_B \subset \pm(\{0, \ldots, 2^B - 1\} + i\{0, \ldots, 2^B - 1\}) * 2^{\mathbb{Z}}.$$

- Every standard operation (arithmetic, trigonometric, etc.) has user-specified rounding modes for the real and imaginary parts.
  - RNDN: Round to nearest
  - RNDU: Round up (towards $+\infty$).

IEEE standards for floating point calculations

- We use a programming package (such as GMP, MPFR, MPC) that follows IEEE standards for floating point calculations.
- Choose a precision B. Denote the set of numbers representable by floating numbers of precision B by $\mathcal{C}_B$. So

$$\mathcal{C}_B \subset \pm(\{0,\ldots,2^B-1\}+i\{0,\ldots,2^B-1\})*2^{\mathbb{Z}}.$$

- Every standard operation (arithmetic, trigonometric, etc.) has user-specified rounding modes for the real and imaginary parts.
  - RNDN: Round to nearest
  - RNDU: Round up (towards $+\infty$).
  - RNDD: Round down (towards $-\infty$).

IEEE standards for floating point calculations

- We use a programming package (such as GMP, MPFR, MPC) that follows IEEE standards for floating point calculations.
- Choose a precision B. Denote the set of numbers representable by floating numbers of precision B by $\mathcal{C}_B$. So

$$\mathcal{C}_B \subset \pm(\{0,\ldots,2^B-1\}+i\{0,\ldots,2^B-1\})*2^{\mathbb{Z}}.$$

- Every standard operation (arithmetic, trigonometric, etc.) has user-specified rounding modes for the real and imaginary parts.
  - RNDN: Round to nearest
  - RNDU: Round up (towards $+\infty$).
  - RNDD: Round down (towards $-\infty$).
- Example usage: $\mathrm{add}(z, w, \mathit{RNDNN})$.
- The result of an operation should be as if the calculation is done exactly (with infinite precision) and then rounded as requested to an element of $\mathcal{C}_B$.

Write a new class in an object oriented language

Call our class **MyComplex**. For speed, the precision B should be fixed at compile time. Also, set $M = \lfloor B/4 \rfloor$. (Other choices could work.)

Write a new class in an object oriented language

Call our class **MyComplex**. For speed, the precision B should be fixed at compile time. Also, set $M = \lfloor B/4 \rfloor$. (Other choices could work.)

- An object of this class has at least two fields:
  - z, a complex floating point number of precision B.
  - r, a real floating point number of precision B.

Write a new class in an object oriented language

Call our class **MyComplex**. For speed, the precision B should be fixed at compile time. Also, set $M = \lfloor B/4 \rfloor$. (Other choices could work.)

- An object of this class has at least two fields:
  - z, a complex floating point number of precision B.
  - r, a real floating point number of precision B.
- The idea is that a **MyComplex** object represents every value that is within r of z, namely

$$D_{(z,r)} = \{w \in \mathbb{C} : |w - z| \leq r\}.$$

Addition in this class

We want the result of **MyComplex_add**$((z_1, r_1), (z_2, r_2))$ to be (z_3, r_3) such that

$$D_{(z_1,r_1)} + D_{(z_2,r_2)} \subseteq D_{(z_3,r_3)}.$$

Addition in this class

We want the result of **MyComplex_add**$((z_1, r_1), (z_2, r_2))$ to be (z_3, r_3) such that

$$D_{(z_1,r_1)} + D_{(z_2,r_2)} \subseteq D_{(z_3,r_3)}.$$

We achieve this by:

Addition in this class

We want the result of **MyComplex_add**$((z_1, r_1), (z_2, r_2))$ to be (z_3, r_3) such that

$$D_{(z_1,r_1)} + D_{(z_2,r_2)} \subseteq D_{(z_3,r_3)}.$$

We achieve this by:

- $z_3 = \mathrm{add}(z_1, z_2, RNDNN)$

Addition in this class

We want the result of **MyComplex_add**$((z_1, r_1), (z_2, r_2))$ to be (z_3, r_3) such that

$$D_{(z_1,r_1)} + D_{(z_2,r_2)} \subseteq D_{(z_3,r_3)}.$$

We achieve this by:

- $z_3 = \mathrm{add}(z_1, z_2, RNDNN)$
- $r_3 = \mathrm{add}(\mathrm{add}(r_1, r_2, RNDU), \mathrm{const\_adderr})$
- where $\mathrm{const\_adderr} = 2^{1+M-B}$ is a constant.
- Abort with an error if any input $|z_i| > 2^M$. Be sure to check this condition via $\mathrm{norm}(z_i, RNDU) \leq 2^M$.

Addition in this class

We want the result of **MyComplex_add**$((z_1, r_1), (z_2, r_2))$ to be (z_3, r_3) such that

$$D_{(z_1,r_1)} + D_{(z_2,r_2)} \subseteq D_{(z_3,r_3)}.$$

We achieve this by:

- $z_3 = \mathrm{add}(z_1, z_2, RNDNN)$
- $r_3 = \mathrm{add}(\mathrm{add}(r_1, r_2, RNDU), \mathrm{const\_adderr})$
- where $\mathrm{const\_adderr} = 2^{1+M-B}$ is a constant.
- Abort with an error if any input $|z_i| > 2^M$. Be sure to check this condition via $\mathrm{norm}(z_i, RNDU) \leq 2^M$.

It can be proven that the above guarantees

$$D_{(z_1,r_1)} + D_{(z_2,r_2)} \subseteq D_{(z_3,r_3)}.$$

Multiplication in this class

To perform the method **MyComplex_multiply**$((z_1, r_1), (z_2, r_2))$...

Multiplication in this class

To perform the method **MyComplex_multiply**$((z_1, r_1), (z_2, r_2))$...

- Calculate $a_i = \mathrm{norm}(z_i, RNDU)$. Abort if either $a_i > 2^M$.
- Calculate $z_3 = \mathrm{mul}(z_1, z_2, RNDNN)$
- Calculate

$$\begin{aligned} r_3 =& \mathrm{add}(\mathrm{add}(\mathrm{mul}(r_1, a_2, RNDU), \\ & \mathrm{mul}(r_2,\ myadd(a_1, r_1, RNDU), RNDU), \\ & RNDU), \mathrm{const\_mulerr}, RNDU) \end{aligned}$$

where $\mathrm{const\_mulerr} = 2^{2M-B}$ is a constant.

It can be proven that the above guarantees

$$D_{(z_1,r_1)} \cdot D_{(z_2,r_2)} \subseteq D_{(z_3,r_3)}.$$

Division in this class

To perform the method **MyComplex_reciprocal**(z, r))...

Division in this class

To perform the method **MyComplex_reciprocal**(z, r))...

- Calculate $b = \mathrm{norm}(z, RNDD)$.
- Abort if $b < 2^{-M}$. (Thus guaranteeing $|z| > 2^{-M}$.)
- Abort if $r \geq b$. (Thus guaranteeing $r < |z|$ and $0 \notin D(z, r)$.)
- Calculate $z_3 = \mathrm{div}(1.0, z, RNDNN)$
- Calculate

$$r_3 = \mathrm{add}(\mathrm{div}(\mathrm{div}(r, b, RNDU), \mathrm{subtract}(b, r, RNDD), RNDU), \\ \mathrm{const\_diverr}, RNDU)$$

where $\mathrm{const\_diverr} = 2^{B-M}$ is a constant.

It can be proven that the above guarantees

$$1/D_{(z,r)} \subseteq D_{(z_3,r_3)}.$$

Back to calculating $(f_{277}|T(p))(s_0\tau)$

Check that each coefficient, a **MyComplex** object, has only one integer in its disk.

Back to calculating $(f_{277}|T(p))(s_0\tau)$

Check that each coefficient, a **MyComplex** object, has only one integer in its disk.

Example using precision $B = 512$ (binary) digits, we calculate:

$$(f_{277}|T(17))(s_0\tau) = (12 + 9.22337 \times 10^{-82}, 1.9 \times 10^{-69})q^3 + O(q^{3+1/17}).$$

Back to calculating $(f_{277}|T(p))(s_0\tau)$

Check that each coefficient, a **MyComplex** object, has only one integer in its disk.

Example using precision $B = 512$ (binary) digits, we calculate:

$$(f_{277}|T(17))(s_0\tau) = (12 + 9.22337 \times 10^{-82}, 1.9 \times 10^{-69})q^3 + O(q^{3+1/17}).$$

We note that there is only one integer 12 that is inside the coefficient disk of q^3. Because we know the coefficients of f_{277} are integers, we conclude

$$(f_{277}|T(17))(s_0\tau) = 12q^3 + O(q^4).$$

Back to calculating $(f_{277}|T(p))(s_0\tau)$

Check that each coefficient, a **MyComplex** object, has only one integer in its disk.

Example using precision $B = 512$ (binary) digits, we calculate:

$$(f_{277}|T(17))(s_0\tau) = (12 + 9.22337 \times 10^{-82}, 1.9 \times 10^{-69})q^3 + O(q^{3+1/17}).$$

We note that there is only one integer 12 that is inside the coefficient disk of q^3. Because we know the coefficients of f_{277} are integers, we conclude

$$(f_{277}|T(17))(s_0\tau) = 12q^3 + O(q^4).$$

Hence $\lambda_{17}(f_{277}) = -4$.

Back to calculating $(f_{277}|T(p))(s_0\tau)$

Check that each coefficient, a **MyComplex** object, has only one integer in its disk.

Example using precision $B = 512$ (binary) digits, we calculate:

$$(f_{277}|T(17))(s_0\tau) = (12 + 9.22337 \times 10^{-82}, 1.9 \times 10^{-69})q^3 + O(q^{3+1/17}).$$

We note that there is only one integer 12 that is inside the coefficient disk of q^3. Because we know the coefficients of f_{277} are integers, we conclude

$$(f_{277}|T(17))(s_0\tau) = 12q^3 + O(q^4).$$

Hence $\lambda_{17}(f_{277}) = -4$.

The above calculation took 140 seconds on a laptop.

Results: some eigenvalues of f_{277}

p	2	3	5	7	11	13	17	19	23
λ_p	-2	-1	-1	1	-2	3	-4	-1	3

p	29	31	37	41	43	47	53	59	97
λ_p	-1	-10	4	7	4	-8	14	1	-6

Results: some eigenvalues of f_{277}

p	2	3	5	7	11	13	17	19	23
λ_p	-2	-1	-1	1	-2	3	-4	-1	3

p	29	31	37	41	43	47	53	59	97
λ_p	-1	-10	4	7	4	-8	14	1	-6

Note: the calculation of λ_{97} took 28 days on one CPU (on a Fordham HPC machine, done in July 2015).

Results: some eigenvalues of f_{277}

p	2	3	5	7	11	13	17	19	23
λ_p	-2	-1	-1	1	-2	3	-4	-1	3

p	29	31	37	41	43	47	53	59	97
λ_p	-1	-10	4	7	4	-8	14	1	-6

Note: the calculation of λ_{97} took 28 days on one CPU (on a Fordham HPC machine, done in July 2015).
This algorithm is parallelizable as the slash by each coset representative can be computed separately.

Results: some eigenvalues of f_{277}

p	2	3	5	7	11	13	17	19	23
λ_p	-2	-1	-1	1	-2	3	-4	-1	3

p	29	31	37	41	43	47	53	59	97
λ_p	-1	-10	4	7	4	-8	14	1	-6

Note: the calculation of λ_{97} took 28 days on one CPU (on a Fordham HPC machine, done in July 2015).
This algorithm is parallelizable as the slash by each coset representative can be computed separately.

These eigenvalues match those of the abelian surface A_{277}.

Results: some eigenvalues of f_{277}

p	2	3	5	7	11	13	17	19	23
λ_p	-2	-1	-1	1	-2	3	-4	-1	3

p	29	31	37	41	43	47	53	59	97
λ_p	-1	-10	4	7	4	-8	14	1	-6

Note: the calculation of λ_{97} took 28 days on one CPU (on a Fordham HPC machine, done in July 2015).
This algorithm is parallelizable as the slash by each coset representative can be computed separately.

These eigenvalues match those of the abelian surface A_{277}.

Together with the results in Brumer's talk and and assuming that all the details can be written down for associating a Galois representation to a weight 2 paramodular form, this proves the modularity of the abelian surface of conductor 277.

Further work

Work in progress on conductors $349, 353, 389, \ldots$.

Thank you.