Factorization tests arising from counting modular forms and automorphic representations

Miao Gu¹ and Greg Martin

1: University of British Columbia / Duke University

Background

- Let k be a positive even integer.
- \( A(k, N) \) is the number of non-isomorphic automorphic representations associated with the space of weight-k cusp forms on \( \Gamma_0(N) \). Equivalently, it is the dimension of the space of weight-k newforms of level dividing N. (complicated)
- \( G(k, N) \) is the function from Gekeler’s Theorem. (simple)
- \( B(k, N) \) is the dimension of the space of weight-k newforms on \( \Gamma_0(N) \). (complicated)
- \( H(k, N) \) is a modified version of \( G(k, N) \). (simple)

Gekeler’s Theorem (1995)

Using the Dirichlet characters \( \chi_4 \) and \( \chi_3 \), define

\[
G(k, N) = \frac{k-1}{12}N - \frac{1}{2} + c_4(k)\chi_4(N) + c_3(k)\chi_3(N).
\]

(Note: \( c_2(k), \chi_4 \) have period 4, and \( c_3(k), \chi_3 \) have period 3.)

Theorem: If N is squarefree, then \( A(k, N) = G(k, N) \).

Spaces of Modular Forms

Let \( S(k, N) \) be the dimension of weight-k cusp forms on \( \Gamma_0(N) \). By the Atkin–Lehner decomposition of spaces of cusp forms

\[
S(k, N) = \sum_{d|N} A(k, d) = \sum_{d|N} B(k, d).
\]

Dimension Formulas (Martin, 2005)

- \( A(k, N) = \frac{k-1}{12}N s_0^-(N) - \frac{1}{2}v_0(N) + c_2(k)\nu_2(N) + c_3(k)\nu_3(N) \)
- \( B(k, N) = \frac{k-1}{12}N s_0^+(N) - \frac{1}{2}\omega(N) + c_2(k)\nu_2^+(N) + c_3(k)\nu_3^+(N) \)

where \( s_0^-, v_0^-, \nu_2^-, \nu_3^-, \omega^-, \) and \( v_0^+, \nu_2^+, \nu_3^+, \omega^+ \) are multiplicative functions which require the factorization of N to compute.

(Note: extra term needed when \( k = 2 \))

Main Theorems

- The converse of Gekeler’s Theorem is true with one small exception \((k = 2, N = 9)\).
- If we have an oracle that quickly computes \( A(k, N) \), even for a single \( k \) (or a positive linear combination of several \( A(k, N) \), or even a sufficiently tight upper bound for \( A(k, N) \)), we have a polynomial-time test for squarefreeness.
- Similarly, we have a polynomial-time test for primality if we can compute \( B(k, N) \) quickly.
- We can probabilistically obtain the complete factorization of the squarefull part of N if we have fast access to \( A(k, N) \) for two distinct weights \( k_1 \) and \( k_2 \).
- If in addition we have fast access to \( B(k, N) \) for a single weight \( k \), we can probabilistically obtain complete factorization of N.

Main Algorithm \((k_1, k_2 \text{ are distinct weights})\)

Write \( N = EL \) where E is squarefree, L is squarefull and \((E, L) = 1\).

Factorization Algorithm 1 & 2

- Obtain values for \( \nu_2^+(N), \nu_3^+(N) \) (see Note 1)
- Write out the system of two linear equations using the formula for \( A(k_1, N), A(k_2, N) \)
- Solve the system for the two unknown values \( N\nu_0(N), \omega_0(N) \)
- Probabilistically factor L in polynomial time using \( L = N/L \) and \( E_s\nu_0^0 = \nu_0(N)/\nu_0(L) \) (= \( \phi(L) \))
- Combine complete factorization of E and L

Factoring Algorithm 1

- Start
- Input \( N, A(k, N) \)
- Compute \( G(k, N) \)
- Factorization of L
- Factorization Algorithm 2
- Input \( B(k, N) \)
- \( A = G? \)
- \( \text{yes} \)
- \( \text{N is squarefree} \)
- \( \text{Factorization of N} \)
- \( \text{Complete factorization of N} \)
- \( \text{End} \)
- \( \text{Note 1: by definition, } \nu_2^+(N), \nu_3^+(N) \in \{-1, 0, 1\}, \text{ and we can figure out which from } A(k, N) \text{.} \)
- \( \text{Note 2: the only possible values for } \nu_2^+(N), \nu_3^+(N), \text{ and } \omega(N) \text{ are } 0 \text{ or } \pm 2^m \text{ for } m \leq \omega(N). \text{ Trying all these (polynomially many) values, we can verify the right factorization.} \)
- \( \text{Note 3: the denominator of } s_0^0 \text{ is a nontrivial divisor of } L; \text{ the value of } L \text{ can be found by iterating this algorithm.} \)

Calculating Dimensions/Further Research

- Calculating \( S(k, N) \): classical (Riemann–Roch) trace formula (Ross, 1992)
- Calculating \( A(k, N) \) and \( B(k, N) \): recursively, starting with values of \( S(k, N) \) (traditional)
- \( A(k, N) = \mu(N) = S(k, N), B(k, N) = \mu(N) + S(k, N) \) (Martin, 2005)
- Further Research: finding ways to quickly obtain \( A(k, N) \) and \( B(k, N) \) without using the factorization of N

arXiv: 1709.02411 Email: miao.gu@duke.edu