README

Ammar Husain

July 16, 2018

1 Modular Forms

1.1 Definition (Modular/Cusp Forms) *$S_k(N)$ transform with a factor of $(c\tau+d)^k$ under $\Gamma_0(N)$ where $c \equiv 0$ modulo N. They must vanish at ∞. $M_k(N)$ don't have to vanish at ∞. In terms of $q = e^{2\pi i\tau}$ S_k gives a q expansion holomorphic on the disc vanishing at the origin. $M_k(N)$ doesn't have to vanish at the origin.*

Acting on $M_k(N = 1)$. That is $\Gamma_0(1) = SL(2,\mathbb{Z})$ without N condition on c entry and k gives the weight.

For $r \geq 1$ and $(n,m) = 1$

$$\begin{aligned}
T_{nm} &= T_n T_m = T_m T_n \\
T_{p^{r+1}} &= T_{p^r} T_p - p^{k-1} T_{p^{r-1}} \\
T_{p^2} &= T_p T_p - p^{k-1} \\
T_{p^3} &= (T_p T_p - p^{k-1}) T_p - p^{k-1} T_p \\
&= T_p^3 - p^{k-1} T_p - p^{k-1} T_p \\
T_{p^4} &= (T_p^3 - 2p^{k-1} T_p) T_p - p^{k-1} (T_p^2 - p^{k-1}) \\
&= T_p^4 - 3p^{k-1} T_p^2 + (p^{k-1})^2
\end{aligned}$$

1.2 Corollary *For a general T_n one can factor n and then use these identities to reduce to a polynomial in the T_p for only primes p.*

Proof

$$\begin{aligned}
n &= p_1^{a_1} p_2^{a_2} \cdots \\
T_n &= T_{p_1^{a_1}} \cdots \\
&= (T_{p_1^{a_1 - 1}} T_p - p^{k-1} T_{p_1^{a_1 - 2}}) \cdots
\end{aligned}$$

$$\begin{aligned}
f &\equiv \sum_{m=0}^{\infty} c_m q^m \\
T_p f &= \sum_{\mu=0}^{\infty} c_{p\mu} q^{\mu} + \sum_{\nu=0}^{\infty} c_{\nu} q^{p\nu}
\end{aligned}$$

If $f \in S_k(1)$, then rescale so starts with $1 * q$.

Now looking for Hecke eigenforms. If we know T_p is acting by a scalar λ_p, then we know that T_{p^r} are acting by scalars as well

$$T_{p^2} f = (\lambda_p^2 - p^{k-1}) f$$

1.3 Theorem (Eichler-Selberg) *Formula for trace of T_n on $S_k(1)$.*

1.4 Corollary *If we know all the traces for T_{p^n} up to $d-1$ where d is dimension of the $S_k(1)$ we can find characteristic polynomial for T_p.*

Proof The first term of T_{p^r} is T_p^r so knowledge of traces of T_{p^r} can be backsubstituted to recover traces of T_p^r then use formula for characteristic polynomial in terms of traces of powers. This is what CharPolyHelper does in the Mathematica file. It takes the list of $\operatorname{tr} T_{p^r}$ and outputs the list of $\operatorname{tr} T_p^r$. CharPolyHelper2 is supposed to transfer that information back into the characteristic polynomial using formula of $\operatorname{tr} A^r$ in terms of symmetric functions of eigenvalues back to the characteristic polynomial which has those eigenvalues as roots. □

2 Point Counts

2.1 Definition (Zeta Function)

$$Z(C,u) = e^{\sum_{m=0}^{\infty} \frac{N_m}{m} u^m}$$

where N_m is the point count over q^m.

2.2 Theorem (Weil)

$$Z(C,u) = \frac{P(u)}{(1-u)(1-qu)}$$

where P is a polynomial that can be written as

$$\begin{aligned} P(u) &= \prod_{i=1}^{2g} (1-\omega_i u) \\ &= 1 + \sum e_k (-u)^k \end{aligned}$$

2.3 Corollary *If you are given N_m for $m = 1 \cdots 2g$, then can recover the e_k symmetric functions of the ω_i. This is SolveForEks in the Mathematica notebook.*

Once you have that, you can plug that back in and recover the count over q^m for even higher m without solving that much harder equation. This is GiveMthPointCount.

If the ω_i are provided, the point counts over all q^m for $1 \cdots m_{max}$ are given through SolveForNms.

2.4 Definition ($Z_{mot,Kapranov}$)

$$Z_{mot,Kapranov}(X,u) = \sum u^n [Sym^n X]$$

where $[Sym^n X]$ is the motive of $Sym^n X$. In particular, can give the points from the motive. Call that map μ.

$$\mu Z_{mot,Kapranov}(X,u) = Z(X,u)$$

2.5 Corollary *This means we can recover the point counts of $Sym^n C$ from the point counts N_m that were given. This is done in symmetricPowerCounts1. In particular, you can get the point count for $Sym^g C$. That is given by the function jacobianSize1.*