Dynamical methods for rapid computations of L-functions

by

Pankaj H. Vishe

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Mathematics

New York University

May 2010

Professor Akshay Venkatesh

To my parents

Acknowledgements

First and foremost, I would like to thank my advisor, Prof. Akshay Venkatesh for everything. Without his support, help and encouragement this project would not be possible.

I would also like to thank Courant institute, New York University for giving me opportunity to conduct research and helping me as much as they could in every problem I faced. I was partially supported by Prof. Yuri Tchinkel's grant. I am very grateful to him for it.

I would like to thank the mathematics department of Stanford University for allowing me stay there and work with the people at Stanford. I learnt a lot during my visit at Stanford. I would also like to thank Prof. Kannan Sounadararajan and Prof. Daniel Bump for some illuminating discussions and encouragement.

It is hard to believe that it has been five years since I started my PhD at the New York University. I would like to thank all my friends at the Courant Institute, Stanford as well as my friends from the Indian Statistical Institute for moral as well as academic support and for making this whole experience so much enjoyable.

Finally, I am really grateful to my family for always supporting me and loving me unconditionally through thick and thin. Without them I would not be here.

Abstract

The primary focus of this thesis is using dynamical ideas to rapidly compute L-functions. The main results can be summarized as:

Rapid algorithm in the T-aspect. Let Γ be a lattice of $\mathrm{SL}(2,\mathbb{R})$ and let f be a holomorphic or Maass cusp form on $\Gamma\backslash\mathbb{H}$. We use the slow divergence of the horocycle flow in $\Gamma\backslash\mathrm{SL}(2,\mathbb{R})$ to get an algorithm to compute $L(f, 1/2+iT)$ up to a maximum error $O(T^{-\gamma})$ using $O(T^{7/8+\eta})$ operations. Here γ and η are any positive numbers and the constants in O are independent of T. We hence improve the current approximate functional equation based algorithms which have complexity $O(T^{1+\eta})$.

Rapid algorithm in the q- aspect. Let $\Gamma = \mathrm{SL}(2,\mathbb{Z})$, f a modular cusp form on $\Gamma\backslash\mathbb{H}$ and χ_q be a Dirichlet character on $\mathbb{Z}/q\mathbb{Z}$. Let $q = MN$. Here $M = M_1, M_2$ such that $M_1|N$ and $(M_2, N) = 1$, where q, M, N, M_1, M_2 are integers. We use the dynamics of the Hecke orbits to get an algorithm to compute $L(f \times \chi_q, 1/2)$ up to any given error $O(q^{-\gamma})$ using $O(M^5 + N)$ operations. In the case when q has a factor less than $q^{1/5}$, we improve current approximate functional equation based algorithms which need $O(q)$ time complexity. Our algorithm is most effective when q has a suitable factor of size $q^{1/6}$.

Contents

Dedication iii

Acknowledgements iv

Abstract v

List of Figures x

1 Introduction **1**

1.1 Contents of the thesis 2

1.2 Model of computation 12

1.3 Outline of the thesis 13

2 Notation **15**

2.1 General notation 15

2.2 Real analytic functions on $\Gamma\backslash \mathrm{SL}(2,\mathbb{R})$ 16

2.3 Hecke orbits 19

3 Rapid computations of ζ (Odlyzko and Schönhage algorithm) **20**

3.1 Introduction 20

3.2 Odlyzko and Schönhage's algorithm 21

4 A 'geometric' approximate functional equation **27**

4.1 Introduction 27
4.2 Preliminaries 28
4.3 A geometric approximate functional equation for $L(f, s)$ 30
4.4 A geometric approximate functional equation for $L(s, f \times \chi_q)$ . . . 36

5 Rapid algorithm in the 'T' aspect **43**
5.1 Proof of theorem 5.1 44
5.2 Proof of theorem 1.1 48

6 Rapid algorithm in the 'q' aspect **54**
6.1 Case $q = MN$ where $(M, N) = 1$ 57
6.2 Case $q = MN$ when $M|N$ 62
6.3 Proof of theorem 1.2 63

7 Computational issues and 'sorting' algorithms **66**
7.1 Numerical integration for analytic functions 67
7.2 Sorting and Reduction 69
7.3 Sorting algorithm 75
7.4 Computational issues 77

8 Appendix **86**
8.1 Special functions 86
8.2 Real analytic functions on $\Gamma\backslash\mathrm{SL}(2, \mathbb{R})$ 90
8.3 Lemmas 5.1.3, 5.2.3 and 6.1.1 and computing $c_{x,y,\beta,l}$ 93
8.4 Computing $c_{x,y,\beta,l}$ and $e_{x,y,\beta,l}$ 100
8.5 Hecke orbits 101
8.6 Computing $\tau(\chi_q)$ 104

Bibliography **107**

List of Figures

1.1 *The approximate horocycle reduced to a fundamental domain.* 5

1.2 The horocycles starting at two "close" points x_1 and x_2 in two close directions ξ_1 and ξ_2 respectively. 6

1.3 *Let* $q = 4 = 2 \times 2$. *The points* $\{A(4,1,k)i\}$ *reduced to* $\mathbb{H}$ *and decomposed into union of* 2 *"arithmetic progressions" of size* 2 *each. Each "arithmetic progression" comes from the* 2^{th} *Hecke orbit of points* v_1 *and* v_2. *The points* v_1 *and* v_2 *themselves come from to the* 2^{th} *Hecke orbit of* i. 9

1.4 N^{th} *Hecke orbits of two "close" points* x *and* y, *when* N *is prime. The Hecke orbits has* 2 *"layers". For a composite* N, *the number of such "layers" will correspond to the divisor function of* N. 11

4.1 *The contour of integration in* (4.3.1). *The branch cut for the logarithm is chosen along the negative real axis. The line* $(1/T + i)t$ *corresponds to the line* $-i\alpha t$. *We take the limit as the radius of the inner circle goes to zero and the radius of the outer circle goes to infinity.* 32

7.1 *Example of an approximate fundamental domain. In this example,* Γ *has three inequivalent cusps* $\mathfrak{a}_0 = \infty$, $\mathfrak{a}_1$ *and* $\mathfrak{a}_2$. *The corresponding approximate fundamental domain is enclosed by bold lines.* D_1 *and* D_2 *correspond to the cuspidal regions and the compact regions respectively.* D_3 *corresponds to the set which consists of points with hyperbolic distance at most* 3 *from* D_2. 71

Chapter 1

Introduction

The primary focus of this thesis is finding rapid algorithms for L-value computations. We consider the problem of computing fast algorithms to compute L-functions in T and q aspect separately in sections 5 and 6 respectively.

In particular, in section 5, we consider the problem of finding a fast algorithm to compute the L-function of a (holomorphic or Maass) cusp form at $1/2 + iT$. We get an efficient algorithm which has time complexity $O(T^{7/8+o(1)})$. We thus improve the current "approximate functional equation" based algorithms which require $O(T^{1+o(1)})$ complexity. In section 6, we consider the problem of computing $L(s, f \times \chi_q)$, where f is a modular (holomorphic or Maass) cusp form, χ_q is a Dirichlet character on $\mathbb{Z}$ with conductor q and s is any fixed point in $\mathbb{H}$. We consider the case where $q = MN$. Let $M = M_1M_2$ where $M_1|N$ and $(M_2, N) = 1$. In this case we get a $O((M^5 + N)^{1+o(1)})$ complexity algorithm. When q has a factor of size $\leq q^{1/5}$, we thus improve the corresponding "approximate functional equation" based algorithms which require $O(q^{1+o(1)})$ time complexity.

Let us start by briefly discussing the model of computation used throughout the

paper. In practice, completely specifying a real number requires an infinite amount of data. For simplicity however, we will use the real number (infinite precision) model of computation that uses real numbers with error free arithmetic having cost as unit cost per arithmetic operation. Our algorithms also work if we work with numbers specified by $O(\log T)$ bits ($O(\log q)$ bits). We will discuss about the floating point error analysis for our algorithms if we use numbers specified by $O(\log T)$ (or $O(\log q)$) bits briefly in 7.4.

Specifying Γ and f requires (*a priori*) an infinite amount of data. Specifying χ_q (*a priori*) requires $O(q)$ amount of data. Throughout the thesis, we will assume that given any point x in $\mathbb{H}$ (or any point y in $\mathrm{SL}(2,\mathbb{R})$) and any $\gamma, T > 0$, we can compute any derivative of f at x (or rather any "derivative" of a "lift" $\tilde{f}$ of f defined in (1.1.4)) up to an error of $O(T^{-\gamma})$ in unit time. We also assume that given any $0 \leq n \leq q-1$, we can compute $\chi_q(n)$ in unit time. The feasibility of these assumptions and what we exactly mean when we say 'given f , co-finite volume subgroup Γ of $\mathrm{SL}(2,\mathbb{R})$ and a Dirichlet character χ_q' will be discussed in section 7.4. We now turn to a more detailed description of contents in this thesis.

1.1 Contents of the thesis

The main focus of this thesis is to use dynamical properties of certain flows on homogeneous spaces to get the desired rapid algorithms.

1.1.1 Background and existing methods

The idea behind "computing" values for the zeta function effectively goes as far back as Riemann. Riemann used an "approximate functional equation" (also called

the *Riemann Siegel formula*) to compute values of the zeta function. The *Riemann Siegel formula*, writes $\zeta(s)$ as the sum of a finite Dirichlet series and some small error term. More explicitly, let $s = 1/2 + iT \in \mathbb{C}$, $a = \sqrt{T/2\pi}$ and $M = \lfloor a \rfloor$, the *Riemann Siegel formula* is given by

$$\zeta(s) = \sum_{1}^{M} \frac{1}{n^s} + \frac{\pi^{s-1/2}\Gamma((1-s)/2)}{\Gamma(s/2)} \sum_{1}^{M} \frac{1}{n^{1-s}} + R(s). \tag{1.1.1}$$

The error was originally bounded by Riemann using the saddle point method. Gabcke in [4] gave better bounds for $R_M(s)$. More explicitly for any positive integer m,

$$R(s) = \frac{(-1)^{M+1}}{a^{\frac{1}{2}}} \sum_{r=0}^{m} \frac{C_r(a)}{a^r} + R_m(s). \tag{1.1.2}$$

Here, C_r are computable constants and $R_m(s) = O(T^{-(2m+3)/4})$. This gives us a $O(\sqrt{T})$ complexity algorithm to compute the zeta function at $1/2 + iT$. Most existing algorithms for computing the zeta function start with the approximate functional equation and try to compute the Dirichlet series faster than $O(\sqrt{T})$.

In general, let $L(f, s)$ be a GL(2) L-function with Dirichlet series $L(f, s) = \sum_1^\infty \frac{b(n)}{n^s}$. The corresponding approximate functional equation is given by

$$L(f, s) = \sum_{1}^{M} \frac{b(n)}{n^s} f_1(s, n) + \sum_{1}^{M} \frac{\overline{b(n)}}{n^{1-s}} f_2(1-s, n) + R_M(s). \tag{1.1.3}$$

We refer the readers to [12, section 3] for more details about (1.1.3). Effective bounds are known for the error term $R_M(s)$ in (1.1.3). Using this we get an algorithm to compute $L(f, s)$ with time complexity $O(T^{1+o(1)})$.

We discuss these and other methods for computing the L-values in the section 2.

1.1.2 Rapid algorithm in the T-aspect

Let Γ be a lattice of $\mathrm{SL}(2,\mathbb{Z})$. Let f be a weight k (holomorphic or Maass) modular cusp form on $\Gamma\backslash\mathbb{H}$ and let $s = 1/2 + iT$. We deal with the problem of computing the corresponding L-function $L(f,s)$ on the critical line up to any given precision.

The main theorem can be given as:

Theorem 1.1. *Given a holomorphic or Maass cusp form for a congruence subgroup of* $\mathrm{SL}_2(\mathbb{Z})$*, and positive reals* γ, η, T*, we can compute* $L(1/2+iT, f)$ *up to an error of* $O(T^{-\gamma})$ *in time* $O(T^{\frac{7}{8}+\eta})$ *using numbers of* $O(\log T)$ *bits. The constants involved in* $O(\ )$ *terms are polynomials in* $\frac{\gamma}{\eta}$ *and are independent of* T*.*

We use an integral representation of $L(f,s)$ to convert the problem of computing L- values to the problem of computing an integral of $\tilde{f}$ on an "approximate horocycle". Here, a lift $\tilde{f} : \Gamma\backslash\mathrm{SL}(2,\mathbb{R}) \to \mathbb{C}$ is defined by

$$\tilde{f}(\begin{pmatrix} a & b \\ c & d \end{pmatrix}) = (ci+d)^{-k} f(\frac{ai+b}{ci+d}). \tag{1.1.4}$$

In particular, in the case of holomorphic cusp forms, we use:

$$\int_0^\infty f((-1+i/T)t)t^{s-1}dt = (2\pi)^{-s}\Gamma(s)(i+1/T)^{-s}L(f, s-(k-1)/2). \tag{1.1.5}$$

The factor $(i+1/T)^{-s}$ takes care of the exponential decay of $\Gamma(s)$ in (1.1.5). We use the "lift" $\tilde{f}$ of f and the exponential decay of f at the cusps to write this integral as an integral of $\tilde{f}$ over an approximate horocycle of hyperbolic length $\approx_\gamma T$ in $\Gamma\backslash\mathrm{SL}(2,\mathbb{R})$ up to any given error $O(T^{-\gamma})$.

We then write the integral over the approximate horocycle as a sum of integrals

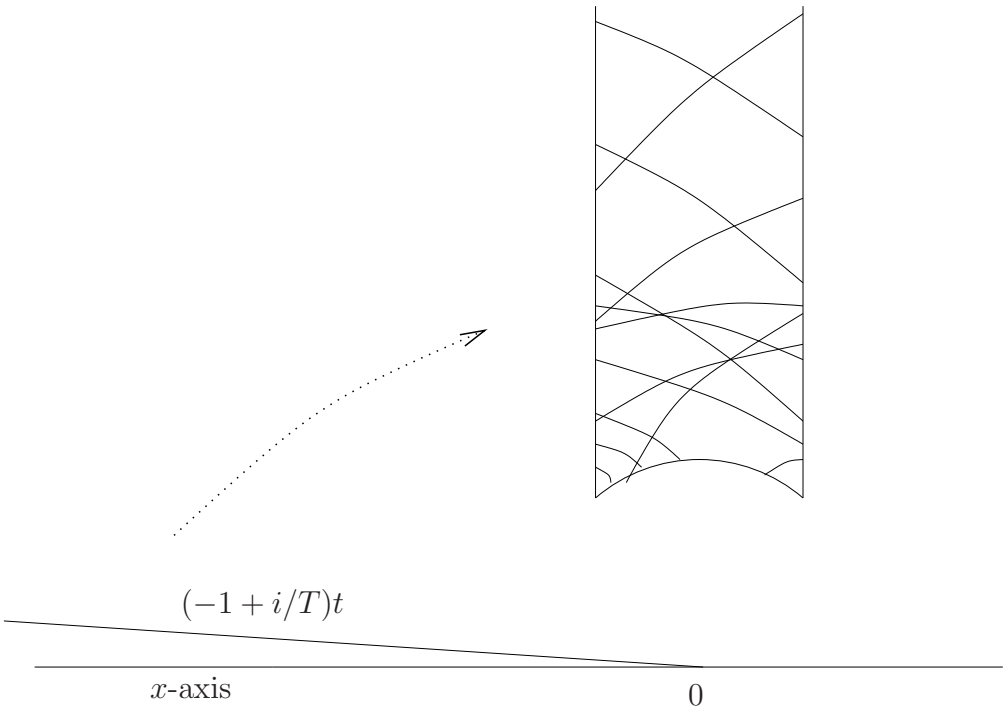

Figure 1.1: *The approximate horocycle reduced to a fundamental domain.*

over smaller segments of equal length M. We sort the starting points of the segments into groups $S_1, S_2, ...$ such that the points in each group are very close to each other. We will give the "sorting" algorithm in detail in subsection 7.2. The key point is that we can compute *all the integrals over all the segments with starting points in* S_i "in parallel".

To accomplish this, we use the critical fact: if two points x_1, x_2 in $\Gamma\backslash \mathrm{SL}(2,\mathbb{R})$ are "sufficiently close" to one other, then the images $x_1(t), x_2(t)$ of these points under the time t horocycle flow stay very close to each other "for a long time." The "approximate" horocycles also have a similar property. We quantify these in propositions 5.1.2 and 5.2.2 later. This is a very special property of the horocycle flow. For example, the geodesic flow does not satisfy such a property.

We return to the description of how to compute the integrals over various

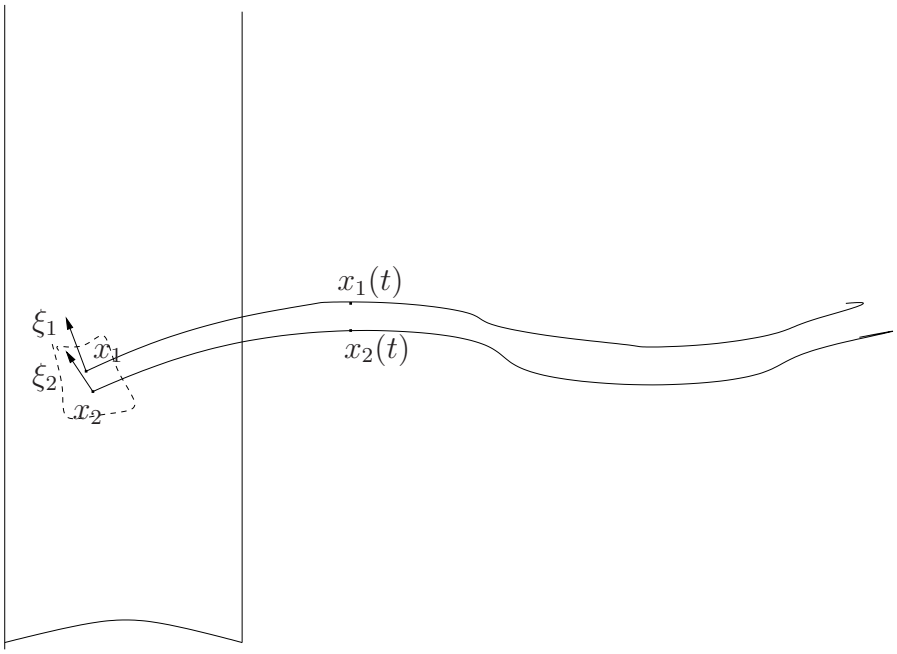

Figure 1.2: The horocycles starting at two "close" points x_1 and x_2 in two close directions ξ_1 and ξ_2 respectively.

segments "in parallel." Consider S_1, for instance. Let I be the segment with starting point v_1, and let J be any other segment with starting point in S_1. Let $F(J)$ denote the value of the integral over segment J. We use proposition 5.2.2 to compute $\tilde{f}$ at each point in J using a power series expansion around a corresponding point in I. In the case of L-value computations, this converts $F(J)$ into sum:

$$F(J) = \sum_{i,j} a_{ij}(J) \int_I t^i f_{ij}(t) g(t) dt. \tag{1.1.6}$$

Here, $f_{ij}(t)$ are smooth functions independent of J. g is a smooth function independent of I and J. a_{ij} are precomputable constants depending on J. We can compute each integral on right hand side of (1.1.6) using $O(M^{1+o(1)})$ operations. Therefore, by precomputing certain constants, we are able to compute all the J-integrals in at most a further $O(|S_1|)$ time.

This grouping leads to a speed-up by a factor that is roughly the size of the groups $|S_i|$. In practice, we cannot make these groups very large, because the amount of time for which the points $x_1(t)$ and $x_2(t)$ stay "very close" to each other depends on how close the points x_1 and x_2 were. Hence, if we try to make the groups "too large", the "admissible" length of the segments decreases, resulting in an increase in the number of segments. This increases the running time.

Upon implementation, this algorithm will allow us to compute the L value for the "high" points on the critical line much faster than the existing methods for a large family of L functions. Thus , it can be extremely useful in numerically checking the GRH, subconvexity bounds, moment conjectures *et al.* for a large class of L- functions.

1.1.3 Rapid algorithm in the q-aspect

Let $\Gamma = \mathrm{SL}(2, \mathbb{Z})$ and f be a (holomorphic or Maass) cusp form of weight k on $\Gamma\backslash\mathbb{H}$. Let χ_q be a Dirichlet character of conductor q. We will assume that q has "large" factors. In particular, we assume that $q = MN$, where $M \leq N$, $M = M_1M_2$ such that $M_1|N$ and $(M_2, N) = 1$. Notice that given any decomposition of $q = M'N'$, we can write $q = MN$ for some $M \leq \sqrt{M'}$ which satisfies our hypothesis. [1]

We deal with the problem of computing $L(1/2, f \times \chi_q)$ up to any given precision with fewer than $O(q)$ arithmetic operations. Our main result can be summarized into

Theorem 1.2. *Let q, M, N be positive integers such that $q = MN$, where $M \leq N$, $M = M_1M_2$ such that $M_1|N$ and $(M_2, N) = 1$. Let f be a modular (holomorphic*

[1] **e.g.** if $q = M'N'$ such that $M' = p^2$ and $N' = pN_1$, for some prime p and $(p, N_1) = 1$. Then we will choose $M = p$ and $N = p^2N'$. Clearly this choice of M and N satisfies our hypothesis.

or Maass) form on $\Gamma\backslash\mathbb{H}$*,* $s \in \mathbb{H}$ *and* χ_q *a character on* $\mathbb{Z}/q\mathbb{Z}$*. Let* γ, η *be any positive reals. Let*

$$E = \min\{\max\{M^5, N\}, q\}.$$

Then we can compute $L(s, f \times \chi_q)$ *up to an error of* $O(q^{-\gamma})$ *in time* $O(E^{1+7\eta})$*. The constants involved in* O *are polynomial in* γ/η*.*

Notice that our method gives us a positive time saving if q has a factor less than $q^{1/5}$. The maximum saving of size $O(q^{1/6})$ can be obtained if q has a suitable factor of size $\approx q^{1/6}$.

Let

$$T(M) = \{(m, k) : m|M, 0 \leq k < M/m\}$$

and

$$A(M, m, k) = \frac{1}{M^{1/2}} \begin{pmatrix} m & k \\ 0 & M/m \end{pmatrix}.$$

The left action of $\{\Gamma A(M, m, k)|(m, k) \in T(M)\}$ on $\mathrm{SL}(2, \mathbb{R})$ corresponds to the M^{th} Hecke orbits in $\Gamma\backslash\mathrm{SL}(2, \mathbb{R})$. We refer the readers to the section 2 and section 8.5 for more details about the Hecke orbits.

We use an integral representation of $L(f \times \chi_q, s)$ to convert the problem of computing $L(f \times \chi_q, s)$ into the problem of computing

$$S = \sum_{k=0}^{q-1} \chi(k) g(A(q, 1, k)x_0)$$

for a small number (for approximately $O(q^{o(1)})$) of values of x_0 in $\mathrm{SL}(2, \mathbb{R})$ and for a real analytic function g on $\Gamma\backslash\mathrm{SL}(2, \mathbb{R})$ with bounded derivatives. We next use

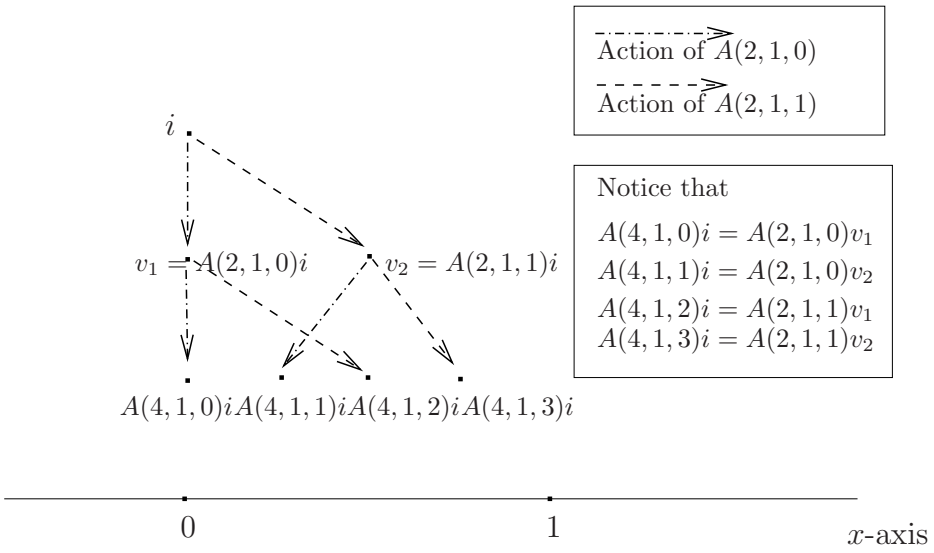

Figure 1.3: *Let $q = 4 = 2 \times 2$. The points $\{A(4,1,k)i\}$ reduced to $\mathbb{H}$ and decomposed into union of 2 "arithmetic progressions" of size 2 each. Each "arithmetic progression" comes from the 2^{th} Hecke orbit of points v_1 and v_2. The points v_1 and v_2 themselves come from to the 2^{th} Hecke orbit of i.*

the fact that $A(q,1,j+kN) = A(M,1,k)A(N,1,j))$ to rewrite S as

$$S = \sum_{j=0}^{N-1} S_j. \tag{1.1.7}$$

Here, S_j is defined as

$$S_j = \sum_{k=0}^{M-1} \chi_q(j+kN) g(A(M,1,k)v_j). \tag{1.1.8}$$

Here

$$v_j = A(N,1,j)x_0.$$

Let us further assume that $(M,N) = 1$. This implies that $\chi_q = \chi_M \chi_N$, χ_M and

χ_N are characters modulo M and N respectively. Using this, we get

$$S_j = \chi_N(j) \sum_{k=0}^{M-1} \chi_M(j+kN) g(A(M,1,k)v_j).$$

Notice that S_j can be rewritten as

$$S_j = \chi_N(j) \sum_{k=0}^{M-1} h_j((1,k)) g(A(M,1,k)v_j);$$

where $h_j((1,k)) = \chi_M(j+kN)$. Notice that h_j is defined on a subset of $T(M)$. Let us extend h_j to the whole $T(M)$ by defining $h_j((m,k)) = 0$ if $m \neq 1$. Hence we rewrite S_j as

$$S_j = \chi_N(j) \sum_{(m,k)\in T(M)} h_j(m,k) g(A(M,1,k)v_j). \tag{1.1.9}$$

Notice that $h_j = h_{j'}$, if $j \equiv j_1 \mod M$. Hence there are only M distinct such functions on the Hecke orbit $T(M)$.

We next use the following fact: If two points x_1 and x_2 in $\Gamma\backslash \mathrm{SL}(2,\mathbb{R})$ are sufficiently close to one another, then their corresponding elements in the Hecke orbit stay "close" to each other . This allows us to look at the points v_j. We then reduce points $\{v_j\}$ to the points $\{x_j\}$ in an approximate fundamental domain and "sort" them in the sets K_i's and choose a representative x_{n_i} from each K_i as in the previous section.

For every j, let $\gamma_j \in \mathrm{SL}(2,\mathbb{Z})$ be such that $\gamma_j x_j = v_j$. We rewrite (1.1.9) as

$$S_j = \chi_N(j) \sum_{(m,k)\in T(M)} h_j((m,k)) g(A(M,1,k)\gamma_j x_j). \tag{1.1.10}$$

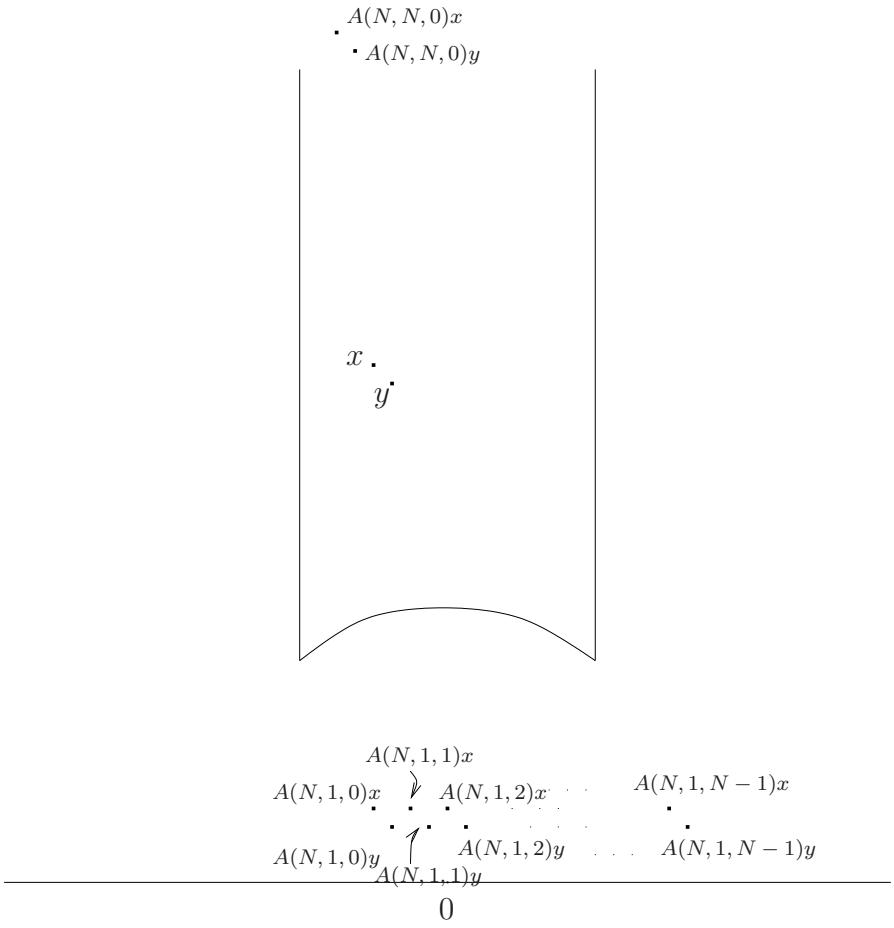

Figure 1.4: *N^{th} Hecke orbits of two "close" points x and y, when N is prime. The Hecke orbits has 2 "layers". For a composite N, the number of such "layers" will correspond to the divisor function of N.*

We will use the fact that the right action of the matrices in $\mathrm{SL}(2,\mathbb{Z})$ permutes the Hecke orbits. In other words, there exists a permutation σ_{γ_j} of $T(M)$ such that for every $(m,k) \in T(M)$,

$$\Gamma A(M,m,k)\gamma_j = \Gamma A(M,\sigma_j((m,k))).$$

Using this we rewrite (1.1.10) as:

$$S_j = \chi_N(j) \sum_{(m,k)\in T(M)} h_j(\sigma_{\gamma_j}((m,k)))g(A(M,1,k)x_j). \tag{1.1.11}$$

We use the fact that the cardinality of $T(M)$ is $\ll M^{1+\eta}$. The number of distinct permutations of $T(M)$ due to the right action of different elements of $\mathrm{SL}(2,\mathbb{Z})$ is at most M^3 (proved in section 8.5). We use this crucial fact to get that the number of distinct functions in the set $\{h_j \circ \sigma_j | j = 0, .., N-1\}$ is at most M^4. We list all these distinct functions as $\{r_1, r_2, ..., r_L\}$. Here $L \leq M^4$ and for every $0 \leq j \leq N-1$, there exists an integer $1 \leq k_j \leq L$ such that $h_j \circ \sigma_j = r_{k_j}$.

We use this result and the fact that the corresponding points in the Hecke orbits of the nearby points stay close, to compute the sums over Hecke orbits of the points in each group "in parallel". This speeds up our computation by a factor depending on M. The maximum saving can be obtained when $q = MN$ where $M \approx q^{1/6}$.

This gives us an algorithm in the case when $q = MN, (M,N) = 1$. We can deal with the general case similarly.

1.2 Model of computation

In practice, specifying a real number completely requires an infinite amount of data. Hence as mentioned before, for simplicity, we will use the real number (infinite precision) model of computation that uses real numbers with error free arithmetic having cost as unit cost per operation. An operation here means addition, subtraction, divison, multiplication, evaluation of logarithm (of a complex

number z such that $|\arg(z)| < \pi$) and exponential of complex numbers.

In reality however, one can not work with the real number model. Computationally, a real number can be specified only by a finite string of digits and a bounded exponent. Hence there are only a very small finite set of rationals that can be exactly represented by such representation. One has to check that the total floating point error in the algorithm does not get too large if we work with finitely represented numbers. The time needed in each arithmetic operations also depends on the size of numbers we are using. For instance, if one is working with numbers specified by 10 digits then adding or subtracting these numbers will approximately need 10 units of time. This adds to the complexity of the algorithm.

Our algorithms will also work if we work with numbers specified by $O(\log T)$ bits ($O(\log q)$ bits, for algorithm in chapter 6). This will at most add a power of $\log T$ (or $\log q$) in the time complexity of the algorithm. We will discuss about the floating point error analysis for our algorithms if we use numbers specified by $O(\log T)$ (or $\log q$) bits briefly in 7.4.

We refer the readers to [16, Chapter 8] and [17] for more details about the real number model of computation.

1.3 Outline of the thesis

A brief account of the notations used in this Thesis is given in chapter 2. We will give a short description of the Schönhage and Odlyzko algorithm for rapid multiple evaluations of $\zeta(s)$ in chapter 3. Our algorithms use a type of "geometric approximate functional equations". They are discussed in detail in chapter 4. In chapters 5 and 6, we will give the proofs of theorems 1.1 and 1.2 respectively. In

sections 8.3 and 8.5, we will prove the exact statements quantifying the "special properties" of the horocycle flow and Hecke orbits. We will give a "sorting" and "reduction" algorithm in chapter 7. In chapter 7, we will also discuss some computational issues regarding the implementation of the algorithms. Issues regarding integral representations for L- functions corresponding to Maass cusp forms will be addressed in section 8.1.

Chapter 2

Notation

2.1 General notation

Throughout, let Γ be a lattice in $\mathrm{SL}(2,\mathbb{Z})$, unless specified otherwise. [1]

Throughout, let T be a positive real [2] and q be a positive integer. We will denote the set of non negative integers by $\mathbb{Z}_+$ and the set of non-negative real numbers by $\mathbb{R}_+$.

We will use the symbol $\ll$ as is standard in analytic number theory: namely, $A \ll B$ means that there exists a constant c such that $A \leq cB$. These constants will always be independent of the choise of T and q.

We will use the following special matrices in $\mathrm{SL}(2,\mathbb{R})$ throughout the thesis:

$$n(t) = \begin{pmatrix} 1 & t \\ 0 & 1 \end{pmatrix}, a(y) = \begin{pmatrix} e^{y/2} & 0 \\ 0 & e^{-y/2} \end{pmatrix}, K(\theta) = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}. \quad (2.1.1)$$

[1] In section 6 however, $\Gamma = \mathrm{SL}(2,\mathbb{Z})$. In the proof theorem 5.1, Γ will denote a lattice in $\mathrm{SL}(2,\mathbb{R})$.

[2] In section 5.1, however T will be a positive integer

$e(x)$ will be used to denote $\exp(2\pi ix)$.

Given a cusp (Maass or holomorphic) form of weight k on $\Gamma\backslash\mathbb{H}$, [3] we will define a lift $\tilde{f}$ of f to $\Gamma\backslash\mathrm{SL}(2,\mathbb{R})$ by $\tilde{f}:\Gamma\backslash\mathrm{SL}(2,\mathbb{R})\to\mathbb{C}$ such that

$$\tilde{f}(\begin{pmatrix} a & b \\ c & d \end{pmatrix}) = (ci+d)^{-k} f(\frac{ai+b}{ci+d}). \tag{2.1.2}$$

$\mathrm{SL}(2,\mathbb{R})$ acts as fractional linear transformation on $\mathbb{H}$. Hence for $X = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ and $z\in\mathbb{H}$, the notation Xz will be used to denote $\frac{az+b}{cz+d}$.

Let x be any element of $\mathrm{SL}(2,\mathbb{R})$. We will frequently abuse the notaition to treat x as an element of $\Gamma\backslash\mathrm{SL}(2,\mathbb{R})$. *i.e.* we will often denote Γx simply by x.

2.2 Real analytic functions on $\Gamma\backslash\mathrm{SL}(2,\mathbb{R})$

Let x be an element of $\mathrm{SL}(2,\mathbb{R})$, let f be a function on $\Gamma\backslash\mathrm{SL}(2,\mathbb{R})$, *a priori* $f(x)$ does not make sense but throughout we abuse the notation to define

$$f(x) = f(\Gamma x).$$

i.e. $f(x)$ simply denotes the value of f at the coset corresponding to x. Using the same notation, we will treat f as a function on $\mathrm{SL}(2,\mathbb{R})$, satisfying $f(\gamma z) = f(z)$ for all $\gamma\in\Gamma$.

[3] The weight corresponding to a Maass form will be 0.

Let ϕ is a bijection (Iwasawa decomposition) given by

$$\phi : (t, y, \theta) \in \mathbb{R} \times \mathbb{R} \times (-\pi, \pi] \to n(t)a(y)K(\theta).$$

Let

Definition 2.2.1. *Given $\epsilon > 0$, set $\mathfrak{U}_\epsilon = ((-\epsilon, \epsilon) \times (-\epsilon, \epsilon) \times (-\epsilon, \epsilon))$ and $U_\epsilon = \phi(\mathfrak{U}_\epsilon) \subset \mathrm{SL}(2, \mathbb{R})$.*

Let us define the following notion of "derivatives" for smooth functions on $\Gamma\backslash\mathrm{SL}(2, \mathbb{R})$:

Definition 2.2.2. *Let f be a function on $\mathrm{SL}(2, \mathbb{R})$ and x any point in $\mathrm{SL}(2, \mathbb{R})$. We define (wherever R.H.S. makes sense)*

$$\begin{aligned}
\frac{\partial}{\partial x_1} f(x) &= \frac{\partial}{\partial t} \mid_{t=0} f(xn(t)); \\
\frac{\partial}{\partial x_2} f(x) &= \frac{\partial}{\partial t} \mid_{t=0} f(xa(t)); \\
\frac{\partial}{\partial x_3} f(x) &= \frac{\partial}{\partial t} \mid_{t=0} f(xK(t)).
\end{aligned}$$

Sometimes, we will also use ∂_i to denote $\frac{\partial}{\partial x_i}$.

Given $\beta = (\beta_1, \beta_2, \beta_3)$, let us define $\partial^\beta g(x)$ by

$$\partial^\beta g(x) = \frac{\partial^{\beta_1}}{\partial x_1^{\beta_1}} \frac{\partial^{\beta_2}}{\partial x_2^{\beta_2}} \frac{\partial^{\beta_3}}{\partial x_3^{\beta_3}} g(x). \tag{2.2.1}$$

For β as above we will define

$$\beta! = \beta_1!\beta_2!\beta_3!$$

and

$$|\beta| = |\beta_1| + |\beta_2| + |\beta_3|.$$

We now define the notion of real analyticity that we will use on $\Gamma\backslash \mathrm{SL}(2,\mathbb{R})$ in the following definition:

Definition 2.2.3. *A function f on $\Gamma\backslash \mathrm{SL}(2,\mathbb{R})$ is called real analytic, if given any point g in $\Gamma\backslash \mathrm{SL}(2,\mathbb{R})$, there exists a positive real number r_g such that g has power series expansion given by*

$$f(gn(t)a(y)K(\theta)) = \sum_{\beta=(\beta_1,\beta_2,\beta_3)\in\mathbb{Z}_+^3} \frac{\partial^\beta f(g)}{\beta!} t^{\beta_1} y^{\beta_2} \theta^{\beta_3} \tag{2.2.2}$$

for every $(t,y,\theta) \in \mathfrak{U}_{r_g}$.

Let us use the following notation for the power series expansion.

Definition 2.2.4. *Let $y, x \in \mathrm{SL}(2,\mathbb{R})$ and t, y, θ be such that $y = xn(t)a(y)K(\theta)$ and $(\beta_1,\beta_2,\beta_3) = \beta \in \mathbb{Z}_+^3$ define*

$$(y-x)^\beta = t^{\beta_1} y^{\beta_2} \theta^{\beta_3}.$$

Hence we can rewrite the Equation (2.2.2) as

$$f(y) = \sum_{\beta=(\beta_1,\beta_2,\beta_3),\beta\in\mathbb{Z}_+^3} \frac{\partial^\beta f(x)}{\beta!} (y-x)^\beta.$$

In the thesis we will assume that given any modular(holomorphic or Maass) cusp form f on $\Gamma\backslash\mathbb{H}$ and $g \in \mathrm{SL}(2,\mathbb{R})$, all the derivatives of $\tilde{f}$ at g are bounded by 1.[4]

[4]In general in claim 8.2.1, we will prove that given a cusp form f, there exists R such that

Given any smooth function f on $\Gamma\backslash \mathrm{SL}(2,\mathbb{R})$ and a compactly supported smooth function h on $\mathrm{SL}(2,\mathbb{R})$, their convolution is defined by:

$$h \star f(x) = \int_{\mathrm{SL}(2,\mathbb{R})} f(z)h(z^{-1}x)dz. \tag{2.2.3}$$

2.3 Hecke orbits

Let $\Gamma = \mathrm{SL}(2,\mathbb{Z})$. Let L be any positive integer and $x \in \mathrm{SL}(2,\mathbb{R})$, let

$$T(L) = \{(m,k) : m|L, 0 \leq k < L/m\}$$

and

$$A(L,m,k) = \frac{1}{L^{1/2}} \begin{pmatrix} m & k \\ 0 & L/m \end{pmatrix}.$$

The L^{th} Hecke orbit is given by $\{\Gamma A(L,m,k), (m,k) \in T(L)\}$. Similarly the L^{th} Hecke orbit of x is given by $\{\Gamma A(L,m,k)x, (m,k) \in T(L)\}$, considered as a subset of $\Gamma\backslash \mathrm{SL}(2,\mathbb{R})$. [5]

$||\partial^\beta \tilde{f}||_\infty \ll R^{|\beta|}$. The case when $R > 1$ can be dealt with analogously. The assumption that all derivates are bounded by 1, allows the proofs to be marginally simpler.

[5] The Hecke orbits generalize the notion of an "arithmetic progression" on $\mathrm{SL}(2,\mathbb{R})$.

Chapter 3

Rapid computations of ζ (Odlyzko and Schönhage algorithm)

3.1 Introduction

The Riemann zeta function is defined by

$$\zeta(s) = \sum_{n=1}^{\infty} \frac{1}{n^s},\ \mathrm{Re}(s) > 1.$$

It can be analytically extended to a meromorphic function on $\mathbb{C}$ with a simple pole at 1. The values of $\zeta(1/2 + iT)$ are important in number theory for various reasons. Understanding the statistical properties of the distribution of values of zeta function on the critical line, verifying moment conjectures and the Riemann hypothesis numerically are just some of many motivations behind the need to compute $\zeta(1/2 + iT)$ up to a "high" precision. In precise words, these motivations led to consideration of the problem of computing $L(1/2 + iT)$ for "large" values of $T > 0$ up to an absolute error bounded by $O(T^{-\gamma})$ for any given $\gamma > 0$.

We refer the readers to [12, Section 2] for some early examples of methods for computing $\zeta(1/2 + iT)$. A real breakthrough was achieved by Odlyzko and Schönhage in [11]. Specifically their method permits evaluation of any single value $\zeta(1/2+iT+it)$ for $t \in [0, \sqrt{T}]$ up to polynomial precision in $O(T^{o(1)})$ time given a precomputation requiring $O(T^{1/2})$ time. This algorithm did not however improve the computational time for a single value computation of the zeta value but it is very useful for multiple evaluations of $\zeta(1/2 + it)$. The problem to reduce the running time for a single evaluation of the zeta function has been tackled succesfully by Schönhage [14], Heath Brown [6], and Hiary [7] *et al.*. The papers with different methods for computing the zeta function include Odlyzko [10], Schönhage [14], Berry and Keating [1], Heath Brown [6], Rubinstein [12] and Hiary [7].

The inspiration for our algorithms comes from [11]. Hence we discuss it in this chapter in detail.

3.2 Odlyzko and Schönhage's algorithm

In [11], the authors proposed new methods to compute the zeta function which are much faster than the Riemann Siegel formula method when many values at closely spaced points are needed. Their main result can be summarized as

Theorem 3.1. *Given any positive constants δ, σ, and c_1 there is an effectively computable constant $c_2 = c_2(\delta, \sigma, c_1)$ and an algorithm that for every $T > 0$ will perform $\leq c_2 T^{1/2+\delta}$ arithmetic operations on the numbers of $\leq c_2 \log T$ bits using $\leq c_2 T^{1/2+\delta}$ bits of storage, and will then be capable of computing any value $\zeta(1/2+it)$, $T \leq t \leq T+T^{1/2}$, to within $\pm T^{-c_1}$ in $\leq c_2 T^{\delta}$ operations using precomputed values.*

Using (1.1.1), the problem of evaluating $\zeta(1/2 + it)$ becomes equivalent to the

problem of evaluating sums of the form

$$g(t) = \sum_{1}^{M} d_n n^{-it}. \tag{3.2.1}$$

Where $|d_n| \leq n^{1/2}$ and $M = O(T^{1/2})$. Notice that

$$g^{(k)}(t) = (-i)^k \sum_{0}^{M} d_n (\log n)^k n^{-it}. \tag{3.2.2}$$

Using explicit bound $|g^{(k)}(t)| \ll T(\log T)^k$, we get that if we know g and its first $O(c_1/\delta)$ derivatives at an "equispaced" grid (with spacing between points= $O(T^{-\delta})$) of $O(T^{1/2+\delta})$ points in $[T, T + T^{1/2}]$, we can compute $g(t)$ for any t in $[T, T+T^{1/2}]$ up to $O(T^{-c_1})$ error. We can do it by using the power series expansion for $g(t)$ at the nearest grid point to evaluate $g(t)$.

We next use fast Fourier transform to compute g at an equispaced grid "in parallel".

3.2.1 Use of FFT

In the last section we proved that the algorithm is equivalent to an algorithm to computing sums of type $\sum_1^M d_n n^{-it}$ at an equispaced "grid" of points in $[T, T+\sqrt{T}]$ (with spacing between points $\theta = O(T^{-\delta})$). Thus after suitable reparametrization, the original problem reduces to the problem of computing $g(k\theta)$ for $0 < \theta < 1$ and $0 \leq k \leq H$. Here

$$g(t) = \sum_{0}^{M} d_n n^{-it}. \tag{3.2.3}$$

for $H = O(T^{1/2+\delta})$ and $M = O(T^{1/2})$. We can further make sure that H is an integer power of 2. Let

$$\omega = exp(2\pi i/H).$$

Let

$$h(j) = \sum_{k=0}^{H-1} g(k\theta)\omega^{kj},\ 0 \leq j \leq H-1, \tag{3.2.4}$$

The inverse discrete Fourier transform yields

$$g(k\theta) = \frac{1}{H}\sum_{j=0}^{H-1} h(j)\omega^{-ij}. \tag{3.2.5}$$

Hence if we can compute all the required $h(j)$'s up to polynomial error, in $O(H^{1+\delta})$ time, then we can use the fast Fourier Transform to compute $g(k\theta)$ for all desired k's.

Hence let us consider the computation of $h(j)$. From 3.2.4 and 3.2.3 we have

$$h(j) = \sum_{k=2}^{M} d_k\omega_k(j); \tag{3.2.6}$$

where

$$\omega_k(j) = \sum_{m=0}^{H-1} k^{im\theta}\omega^{mj}. \tag{3.2.7}$$

If for some $2 \leq k \leq M$, $k^{i\theta}$ is "very close" to some ω^{j_0} for some j_0, then such $k^{i\theta}$ behaves essentially like an H^{th} root of unity. This essentially tells us that $\omega_k(j_0) \approx H$ and that for every other j, $\omega_k(j) \approx 0$.

For every other k, we use geometric sum formula in 3.2.7 to get

$$\omega_k(j) = \frac{1-k^{iH\theta}}{1-k^{i\theta}\omega^j} = \frac{k^{-i\theta}(k^{iH\theta}-1)}{\omega^j - k^{-in\theta}}$$

Let

$$f(z) = \sum_{k=2}^{M} \frac{a_k}{z - k^{-i\theta}}$$

where a_k for $a_k = k^{-i\theta}(k^{iH\theta} - 1)$ and $a_k = 0$ if $k^{i\theta}$ is "very close" to some ω^{j_0} for some j_0.

The problem is now equivalent to computing $f(\omega^j)$ for all $0 \leq j \leq H - 1$ "in parallel". We deal with this problem in a little bit different fashion than Odlyzko and schönhage. We will use the following physical interpretation of this problem to give an intuitive algorithm:

Let H charges having charge a_k each are placed at points $k^{-i\theta}$ respectively. Then the problem of computing $f(\omega^j)$ can be interpreted as finding total "potential" at the points ω^j due to these charges. Such problems are studied in detail by physicists. We will give an outline of the fast multipole algorithm given in [5].

3.2.2 Use of the Fast Multipole Method

Let us consider the fast multipole method in one dimension. Consider M charges with charge $a_1, ..., a_M$ each be placed randomly on the unit circle at angles $b_1, ..., b_M$. Suppose the potential at a point x due to charge a_k is given by

$$f_k(x) = \frac{a_k}{\exp(2\pi x i) - \exp(2\pi b_k i)}. \tag{3.2.8}$$

The main result of the FMM can be given as:

Theorem 3.2. *Let M charges having charge $a_1, ..., a_M$ each be placed randomly on a unit circle at 'angles' $b_1, ..., b_M$. Let the potential function is defined by* (3.2.8)*. Given any positive constants δ, σ, and c_1 there is an effectively computable constant*

$c_2 = c_2(\delta, \sigma, c_1)$ and an algorithm that for every $M > 0$ will perform $\leq c_2 M^{1+\delta}$ arithmetic operations on the numbers of $\leq c_2 \log M$ bits using $\leq c_2 M^{1+\delta}$ bits of storage, and will then be capable of computing total potential at any point x such that $|x - b_k| \geq M^{-\sigma}\ \forall 1 \leq k \leq M$, to within $\pm M^{-c_1}$ in $\leq c_2 M^{\delta}$ operations using precomputed values.

Notice that the condition that x is not "too close" to any charges is equivalent to the condition that $k^{i\theta}$ is not "too close" to any ω^j in 3.2.1. The main idea behind the algorithm comes from the fact that if many charges are concentrated in some part then at a faraway point, the charges "behave" like a single charge. In particular, if our "source panel" is in the interval $[t-h, t+h]$ (a similar argument goes through for a semi closed interval) for some positive constants t, h then for all the points in $[t-3h, t+3h]^c$ the source panel behaves like a single charge. In other words we can use a multipole power series expansion to get $S_{[t-h,t+h]}$ a polynomial in $(x-t)$ of degree $O(\log M)$ such that for any $x \in [t-3h, t+3h]^c$, the potential due to all the charges in $[t-h, t+h]$ is given by $S_{[t-h,t+h]}(x)$. We can compute and store the co-efficients of the polynomial in $O(\log M + C_{t,h})$ steps. Where $C_{t,h}$ denotes the number of charges in $[t-h, t+h]$.

The algorithm is clear now. We start with bisecting the unit interval recursively. At step one we have two intervals $[0, 1/2), [1/2, 1]$. We compute the corresponding $S_{[0,1/2)}$ and $S_{[1/2,1]}$. Notice that the total time needed to precompute and store the coefficients for both the polynomials is bounded by $O(M + \log M)$. In the next step we compute $S_{[0,1/4)}, S_{[1/4,1/2)}, S_{[1/2,3/4)}, S_{[3/4,1]}$. The total time needed in this step is also $O(M + \log M)$. We continue this process for $\frac{\log 3 + \sigma \log M}{\log 2}$ steps. We store all the polynomials. The total time needed for this process is $O(M^{1+\delta})$.

Once we store these values, we can use a 'binary expansion' for x to compute

the total potential at the point x in $O(M^\delta)$ further steps. Notice that in order to use the multipole expansion, $S_{[a,b]}$ at a point x, we need $|x - \frac{a+b}{2}| \geq \frac{3(b-a)}{2}$. We illustrate the algorithm by showing first few steps to compute $S(1/2)$ (total potential at $1/2$).

We use the binary expansion of $1/2$, to get

$$S(1/2) = S_{[0,1/4)}(1/2) + S_{[3/4,1]}(1/2) + S(1/4, 3/4).$$

Here $S(a, b)$ denotes the total potential at $1/2$ due to charges in $[a, b)$. In the second step we write

$$S(1/4, 3/4) = S_{[1/4,3/8)}(1/2) + S_{[5/8,3/4)}(1/2) + S(3/8, 5/8).$$

We do this recursively for $O(\log M)$ steps to compute $S(1/2)$. The algorithm is compute $S(x)$ for any x follows similarly.

Proof of the theorem 3.2, completes the algorithm required for proving theorem (3.1).

Chapter 4

A 'geometric' approximate functional equation

4.1 Introduction

Just like in the proof of theorem 3.1, most algorithms to compute a general L-function start with an approximate functional equation. This is a generalization of the Riemann- Siegel formula for a general L-function. We refer the readers to [12, Section 3] for a detailed discussion of the approximate functional equation based algorithms.

Both our algorithms start with a 'geometric approximate functional equation'. The 'geometric approximate functional equation' for $L(f, s)$ is given (for holomorphic and Maass forms respectively) by (4.3.6) and (4.3.7). The corresponding equation for $L(s, f \times \chi_q)$ is given by (4.4.16) and (4.4.17).

Notice that equations (4.3.6) and (4.3.7) are integrals of 'nice' functions over a curve of hyperbolic length $O(T)$. Hence given any $\gamma, \eta > 0$, using the method

used in proposition 7.1.1, we can write integrals in (4.3.6) and (4.3.7) as a sums of size $O(T^{1+\eta})$ up to an error $O(T^{-\gamma})$. Similarly, the right hand side of (4.4.16) and (4.4.17) consists of sums of q integrals. Each of these integrals is an integral of a 'nice' function on a geodesic of (hyperbolic) length $O(\log q)$. Hence using idea in proposition 7.1.1, we can write each of this integrals (up to an error of $O(q^{-\gamma})$), as a sum of size $O(q^{\eta} \log q)$ terms. Adding all these sums together, the right hand sides of (4.4.16) and (4.4.17) can be written (up to an error of $O(q^{-\gamma})$) a sum of size $O(q^{1+\eta})$. The constants involved in O are polynomial in γ/η and are independent of q and T.

The derivation of the approximate functional equations used here have geometric motivation behind them. Hence we call these equations as "geometric approximate functional equations".

4.2 Preliminaries

Let Γ be a lattice of $\mathrm{SL}(2,\mathbb{Z})$ and let f be a modular (holomorphic or Maass) cusp form of weight k. [1] In the Maass form case, let f be an eigenfunction of the hyperbolic laplacian $\Delta = -y^2(\partial_x^2 + \partial_y^2)$ with eigenvalue $1/4 + r^2$ on $\Gamma\backslash\mathbb{H}$.

In this paper, for simplicity let us assume that f is either holomorphic or an even Maass form. The algorithms will be analogous for the odd Maass cusp form case. For an even Maass cusp form, we will use the following power series expansion :

$$f(z) = \sum_{n>0} \hat{f}(n) W_r'(nz). \tag{4.2.1}$$

[1] In the case of Maass forms let us assume the weight is 0.

Here $W_r'(x+iy) = 2\sqrt{y}K_{ir}(2\pi y)\cos(2\pi x)$. The explicit Fourier expansion for the holomorphic cusp forms is given by

$$f(z) = \sum_{n>0} \hat{f}(n)e(nz). \tag{4.2.2}$$

Given an (even Maass or holomorphic) cusp form f of weight k and the corresponding Fourier coefficients given by (4.2.1) or (4.2.2), let us define the L function by

Definition 4.2.1. $L(s,f) = \sum_{n=1}^{\infty} \frac{\hat{f}(n)}{n^{s+(k-1)/2}}$.

An integral representation for $L(f,s)$ corresponding to a Maass form f and s such that $\mathrm{Re}(s) > 1$ is obtained by:

$$\begin{aligned}
\int_0^\infty f(iy)y^{s-3/2}dy &= \sum_{n\neq 0} \hat{f}(n) \int_0^\infty W_r'(iny)y^{s-3/2}dy \\
&= \sum_{n>0} \hat{f}(n)\sqrt{n} \int_0^\infty 2\sqrt{y}K_{ir}(2\pi ny)y^{s-3/2}dy \\
&= 2\sum_{n>0} \frac{\hat{f}(n)}{n^{s-1/2}} \int_0^\infty K_{ir}(2\pi y)y^{s-1}dy \\
&= 2L(s,f) \int_0^\infty K_{ir}(2\pi y)y^{s-1}dy \\
&= \frac{1}{2}\pi^{-s}L(f,s)\Gamma\left(\frac{s+ir}{2}\right)\Gamma\left(\frac{s-ir}{2}\right).
\end{aligned}$$

Hence using the analytic continuation, we get the integral representation for $L(f,s)$ ($\mathrm{Re}(s) \geq 1/2$) given by

$$\int_0^\infty f(iy)y^{s-1}dy = \frac{1}{2}\pi^{-s}L(f,s)\Gamma\left(\frac{s+ir}{2}\right)\Gamma\left(\frac{s-ir}{2}\right). \tag{4.2.3}$$

We will further assume that f is an eigenfunction of some Fricke involution, as is the case for all the newforms. This will quantify the exponential decay at zero.

This implies that there exists a positive constant C_1 and a real constant C_2 such that:

$$f(-\frac{C_1}{z}) = C_2 z^k f(z). \tag{4.2.4}$$

Using the condition (4.2.4) and the exponential decay of f at ∞, the integral on the LHS of (4.2.3) converges for all s in $\mathbb{H}$. Hence we can use (4.2.3) to compute the values of the L functions on the critical line. The gamma factors on the right hand side are decaying exponentially with T (here $T = \mathrm{Im}(s)$), hence we need to compute the integral to an high precision in order to get moderate accuracy in the computations of the L-values. This problem has been tackled before in several ways. A solution to this has been suggested in [9] and worked out by Rubinstein in [13, Chapter 3].

We will proceed in a different fashion. We change the contour of integration that will add an exponential factor which will take care of the exponential decay of the gamma function. We will derive a 'geometric approximate functional equation' using this idea in the following section.

4.3 A geometric approximate functional equation for $L(f, s)$

Let $s = s_0 + iT$, where $s_0 \geq \frac{1}{2}$ and $\alpha = -1 + \frac{i}{T}$.

As stated before, we change the contour of integration in (4.2.3) so as to add an exponential factor which will take care of the exponential decay of the gamma function (*refer figure* 1.1). The case of holomorphic modular forms is easier to deal with, hence we will consider it first.

Holomorphic case : Let f be a holomorphic cusp form of weight $k > 0$.

$$\begin{aligned}
&\int_0^\infty f(\alpha y) y^{s+(k-1)/2-1} dy \qquad (4.3.1)\\
&= \sum_{n=1}^\infty \hat{f}(n) \int_0^\infty \exp(2\pi i n \alpha y) y^{s+(k-1)/2-1} dy;\\
&= (2\pi)^{-(s+\frac{k-1}{2})} \sum_{n=1}^\infty \frac{\hat{f}(n)}{n^{s+(k-1)/2}} \frac{1}{(-\alpha i)^{s+(k-1)/2}} \int_{z=-i\alpha\mathbb{R}_+} \exp(-z) z^{s+(k-3)/2} dz\\
&= (2\pi)^{-(s+(k-1)/2)} \sum_{n=0}^\infty \frac{\hat{f}(n)}{n^{s+(k-1)/2}} \frac{1}{(-\alpha i)^{s+(k-1)/2)}} \int_{z=\mathbb{R}_+} \exp(-z) z^{s+(k-3)/2} dz
\end{aligned}$$

using holomorphy and suitable contour (ref. fig 4.1).

$$\begin{aligned}
&= (2\pi)^{-(s+(k-1)/2)} \frac{L(f,s)}{(-\alpha i)^{s+(k-1)/2}} \int_0^\infty \exp(-t) t^{s+(k-3)/2} dt\\
&= (2\pi)^{-(s+(k-1)/2)} \frac{L(f,s)}{(-\alpha i)^{s+(k-1)/2}} \Gamma(s + (k-1)/2).
\end{aligned}$$

Here, the branch cut for the logarithm is taken along the negative real axis. Hence the argument of logarithm takes values between $-\pi$ and π. With this choice of the logarithm, the factor $\frac{1}{(-\alpha i)^{s+(k-1)/2}}$ grows like $O(e^{\pi T/2})$ as $T \to \infty$, to compensate for the exponential decay of $\Gamma(s + (k-1)/2)$.

Case of Maass forms : We use a similar idea for the non-holomorphic case. let

$$\alpha_1 = -1 + \frac{i}{T_1}.$$

We will choose a suitable value for T_1 later.

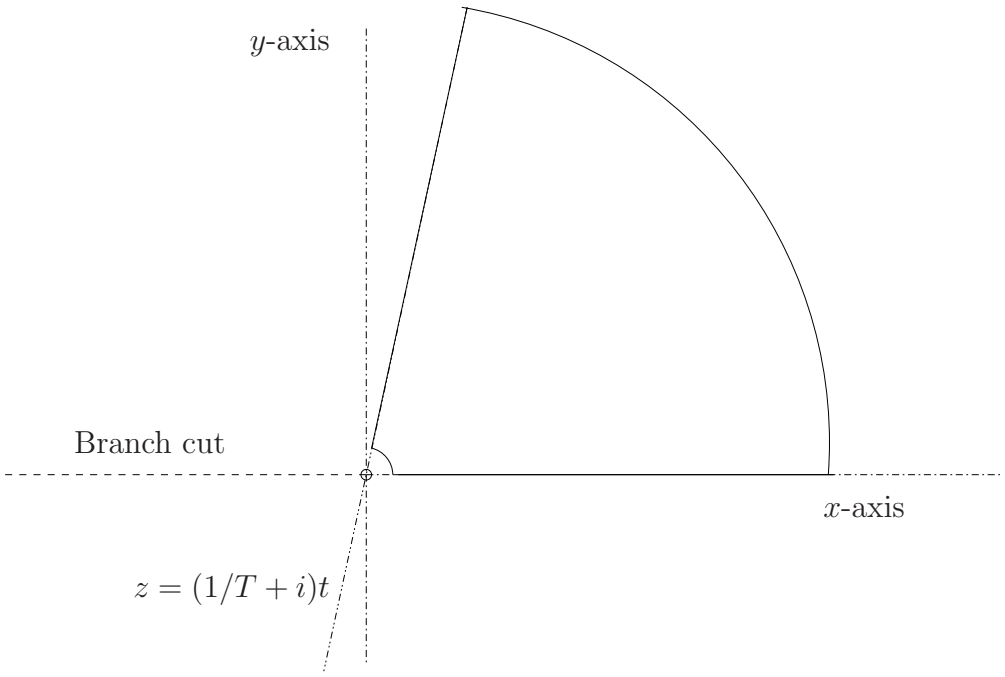

Figure 4.1: *The contour of integration in* (4.3.1)*. The branch cut for the logarithm is chosen along the negative real axis. The line* $(1/T + i)t$ *corresponds to the line* $-i\alpha t$*. We take the limit as the radius of the inner circle goes to zero and the radius of the outer circle goes to infinity.*

$$\begin{aligned}
\int_0^\infty f(\alpha_1 y) y^{s-3/2} dy &= \sum_{n>0} \hat{f}(n) \int_0^\infty W_r'(n\alpha_1 t) t^{s-3/2} dt; \\
&= \sum_{n>0} \frac{\hat{f}(n)}{n^{s-1/2}} \int_0^\infty W_r'(\alpha_1 t) t^{s-3/2} dt; \\
&= \sum_{n>0} \frac{\hat{f}(n)}{n^s} \int_0^\infty W_r'((-1 + \frac{i}{T_1})t) t^{s-3/2} dt; \\
&= 2T_1^{-\frac{1}{2}} \sum_{n>0} \frac{\hat{f}(n)}{n^{s-1/2}} \int_0^\infty cos(2\pi t) K_{ir}(2\pi \frac{t}{T_1}) t^{s-1} dt; \\
&= c' \sum_{n>0} \frac{\hat{f}(n)}{n^{s-1/2}} \int_0^\infty \cos(T_1 t) K_{ir}(t) t^{s-1} dt.
\end{aligned}$$

Here $c' = \frac{2T_1^s}{(2\pi)^s}$. We summarize the result as:

$$\int_0^\infty f(\alpha_1 y) y^{s-3/2} dy = \frac{2T_1^s}{(2\pi)^s} L(f,s) \int_0^\infty \cos(T_1 t) K_{ir}(t) t^{iT-1/2} dt. \tag{4.3.2}$$

In proposition 4.3.1 we will show that we can choose $T_1 \approx T$ such that the integral on the right hand side of (4.3.2) is not too small.

Proposition 4.3.1. *Let r be any fixed complex number such that $|Im(r)| < 1/2$ and $T, T_1 > 0$,*

$$\int_0^\infty \cos(T_1 t) K_{ir}(t) t^{iT-1/2} dt = 2^{iT-3/2} \Gamma(\frac{ir+iT+\frac{1}{2}}{2}) \Gamma(\frac{-ir+iT+\frac{1}{2}}{2}) \tag{4.3.3}$$
$$F(\frac{ir+iT+\frac{1}{2}}{2}, \frac{-ir+iT+\frac{1}{2}}{2}, \frac{1}{2}, -T_1^2).$$

Here F denotes the hypergeometric function. Moreover there exists a computable constant $C' = O(1)$ depending on r such that for $T_1 = C'T$,

$$\Gamma(\frac{ir+iT+\frac{1}{2}}{2}) \Gamma(\frac{-ir+iT+\frac{1}{2}}{2}) F(\frac{ir+iT+\frac{1}{2}}{2}, \frac{-ir+iT+\frac{1}{2}}{2}, \frac{1}{2}, -T_1^2)$$
$$\gg T^{-4}.$$

The implied constant is non zero and depends only on r.

We will prove the proposition 4.3.1 in section 8.1. Proposition 4.3.1 will allow us to take $T_1 = C'T$ for some $C' = O(1)$. Hence for this choice of T_1, once we compute $\int_0^\infty f(\alpha_1 y) y^{s-1} dy$ up to an error of $O(T^{-\gamma})$, we can compute $L(f,s)$ up to an error of $O(T^{4-\gamma})$.

We next use the exponential decay of f at 0 and at ∞ to get:

Lemma 4.3.2. *Given any cusp form f, a real number s_0 and positive reals T, γ, if $\alpha = -1+i/T$ and $s = s_0+iT$ then there exists a computable constant $c = O(1+\gamma)$, independent of T such that :*

$$\int_0^\infty f(\alpha t)t^{s-1}dt = \int_{\frac{1}{cT\log T}}^{Tc\log T} f(\alpha t)t^{s-1}dt + O(T^{-\gamma}). \tag{4.3.4}$$

Proof. Using the exponential decay at ∞, we get that for $t \geq 1$,

$$|f(\alpha t)| = |f(-t + it/T)| \ll \exp(-at/T)$$

for some positive constant a. Hence for all $t > T$, we have

$$|f(\alpha t)t^{s-1}| \leq t^{s_0-1}\exp(-at/T).$$

Hence,

$$t^{s_0-1} < \exp(at/2T) \Leftrightarrow (s_0-1)\log t < at/2T \Leftrightarrow 2T(s_0-1)\log t/a < t.$$

We can see that the condition above holds for $T > e$ and

$$t > t_0 = 20T\log T(1+|s_0-1|)^2(1+a^{-1})^2.$$

This implies that for $t_1 \geq t_0$,

$$|\int_{t_1}^\infty f(\alpha t)t^{s-1}dt| \ll \int_{t_1}^\infty \exp(-at/2T) = (2T/a)\exp(-at_1/2T). \tag{4.3.5}$$

Hence for any $c_1 \leq t_0(\gamma+1)/(aT\log T)$, we get that

$$\int_{c_1 T\log T}^{\infty} f(\alpha t)t^{s-1}dt \ll O(T^{-\gamma}).$$

Using (4.2.4) , we get that for $t < 1 < T$, we have

$$|f(\alpha t)| = |f(-t+\frac{it}{T})| = \frac{|\alpha|^{-k}}{|C_2|}t^{-k}|f(-C_1/(-t+\frac{it}{T}))| \ll t^{-k}\exp(-C_1 a/2Tt).$$

We deal with this case exactly as the previous case to get suitable c. □

Applying Lemma 4.3.2 to (4.3.1) we get that given any $T, \gamma > 0$ there exists a computable constant c such that

$$L(f,s) = \frac{(2\pi)^{s+(k-1)/2}(-\alpha i)^{s+(k-1)/2}}{\Gamma(s+(k-1)/2)}\int_{\frac{1}{cT\log T}}^{Tc\log T} f(\alpha t)t^{s+(k-3)/2}dt + O(T^{-\gamma}). \quad (4.3.6)$$

and similarly after applying Lemma 4.3.2 to (4.3.2) we get that given any $T, \gamma > 0$ there exist computable constants c and $C' = O(1)$ such that if $T_1 = C'T$ then

$$L(f,s) = \frac{(2\pi)^{s+1/2}}{2T_1^s C(T_1)}\int_{1/cT\log T}^{cT\log T} f(\alpha_1 y)y^{s-3/2}dy + O(T^{-\gamma}). \quad (4.3.7)$$

Here, $C(T_1) = \int_0^\infty \cos(T_1 t)K_{ir}(t)t^{s-1}dt$. Notice that the integrals on the right hand side of (4.3.6) and (4.3.7) are over curves of hyperbolic length $\approx T$. We call (4.3.6) and (4.3.7) as geometric approximate functional equations for L- functions corresponding to the holomorphic and the Maass forms respectively. We will see later that the integrals on the right hand side of the equations (4.3.6) and (4.3.7) can be computed directly in $O(T^{1+\delta})$ time, given any $\delta > 0$. Our algorithms for

computing $L(f,s)$ start with these "geometric approximate functional equations".

4.4 A geometric approximate functional equation for $L(s, f \times \chi_q)$

Let f be a modular (holomorphic or Maass) form on $\Gamma\backslash\mathbb{H}$ of weight k. Let us assume that $\Gamma = \mathrm{SL}(2,\mathbb{Z})$. The algorithm will be similar in the case of a congruence subgroup. We will again assume that f is an even Maass form or holomorphic. The algorithm for the odd Maass form case is similar. Let f have a Fourier expansion given by (4.2.1)/(4.2.2). Let q be a positive integer, $s \in \mathbb{H}$ such that $\mathrm{Re}(s) > (k+1)/2$ and χ_q a Dirichlet character on $(\mathbb{Z}/q\mathbb{Z})^*$. For a cusp form of weight k, the corresponding twisted L- function is defined by:

Definition 4.4.1. $L(s, f \times \chi_q) = \sum_{n=1}^{\infty} \frac{\hat{f}(n)\chi_q(n)}{n^{s+(k-1)/2}}$.

Holomorphic case :

$$\begin{aligned}\sum_{k=0}^{q-1} \chi_q(k) f(k/q+iy) &= \sum_{k=0}^{q-1} \chi_q(k) \sum_{n=1}^{\infty} \hat{f}(n) e(nk/q) e(iny) \\ &= \sum_{n=1}^{\infty} \hat{f}(n) e(iny) \sum_{k=0}^{\infty} \chi_q(k) e(nk/q) \\ &= \tau(\chi_q) \sum_{n=1}^{\infty} \hat{f}(n)\chi_q(n) e(iny). \end{aligned} \tag{4.4.1}$$

Here, $\tau(\chi_q)$ is the Gauss sum defined by

$$\tau(\chi_q) = \sum_{k=0}^{q-1} \chi_q(k) e(k/q). \tag{4.4.2}$$

After taking the Mellin transform of (4.4.1), we get

$$\sum_{k=0}^{q-1}\chi_q(k)\int_0^\infty f(k/q+iy)y^{s+(k-3)/2}; \tag{4.4.3}$$

$$= \tau(\chi_q)\sum_{n=1}^{\infty}\hat{f}(n)\chi_q(n)\int_0^\infty e(iny)y^{s+(k-3)/2}dy;$$

$$= \frac{\tau(\chi_q)}{(2\pi)^{s+(k-1)/2}}L(s,f\times\chi_q)\Gamma(s+(k-1)/2). \tag{4.4.4}$$

Even Maass form case :

$$\sum_{k=0}^{q-1}\chi(k)f(k/q+iy)$$

$$= 2\sum_{k=0}^{q-1}\chi_q(k)\sum_{n=1}^{\infty}\hat{f}(n)\sqrt{ny}\cos(2\pi nk/q)K_{ir}(2\pi ny)$$

$$= 2\sum_{n=1}^{\infty}\hat{f}(n)\sqrt{ny}K_{ir}(2\pi ny)\sum_{k=0}^{q-1}\chi_q(k)\frac{e(nk/q)+e(-nk/q)}{2}$$

$$= 2\frac{\tau(\chi_q)+\overline{\tau(\overline{\chi_q})}}{2}\sum_{n=1}^{\infty}\hat{f}(n)\chi_q(n)\sqrt{ny}K_{ir}(2\pi ny)$$

$$= \tau(\chi_q)(1+\chi_q(-1))\sum_{n=1}^{\infty}\hat{f}(n)\chi_q(n)\sqrt{ny}K_{ir}(2\pi ny). \tag{4.4.5}$$

Here, $\tau(\chi_q)$ is the Gauss sum defined by (4.4.2). Taking Mellin transform of (4.4.5), we get

$$\sum_{k=0}^{q-1}\chi(k)\int_0^\infty f(k/q+iy)y^{s-3/2}dy \tag{4.4.6}$$
$$= 2\tau(\chi_q)\frac{1+\chi_q(-1)}{2}\sum_{n=1}^{\infty}\hat{f}(n)\sqrt{n}\chi_q(n)\int_0^\infty K_{ir}(2\pi ny)y^{s-1}dy;$$
$$= \tau(\chi_q)\frac{1+\chi_q(-1)}{4(\pi)^s}L(s,f\times\chi_q)\Gamma\left(\frac{s+ir}{2}\right)\Gamma\left(\frac{s-ir}{2}\right).$$

$\tau(\chi_q)$ is a complex number with absolute value $q^{\frac{1}{2}}$. This implies that we can use (4.4.3) to compute $L(s, f\times\chi_q)$ in the holomorphic case. But if $\chi_q(-1)=-1$, then for an even Maass cusp form f, the right hand side of (4.4.6) is zero. Hence it can not be used to compute the L- values. This is however easily fixable. Recall the Fourier expansion for f given by

$$f(z)=\sum_{n>0}\hat{f}(n)2\sqrt{y}K_{ir}(2\pi ny)\cos(2\pi nx).$$

This implies that

$$\partial_x f(z)=-2\pi\sum_{n>0}n\hat{f}(n)2\sqrt{y}K_{ir}(2\pi ny)\sin(2\pi nx).$$

We use a similar method as before to get:

$$\begin{aligned}
&\sum_{k=0}^{q-1}\chi(k)\partial_x f(k/q+iy)\\
&\qquad=-2\pi\sum_{k=0}^{q-1}\chi_q(k)\sum_{n=1}^{\infty}n\sqrt{ny}\hat{f}(n)\sin(2\pi nk/q)K_{ir}(2\pi ny)\\
&\qquad=-2\pi\sum_{n=1}^{\infty}\hat{f}(n)n^{3/2}\sqrt{y}K_{ir}(2\pi ny)\sum_{k=0}^{q-1}\chi_q(k)\frac{e(nk/q)-e(-nk/q)}{2i}\\
&\qquad=i\pi(\tau(\chi_q)-\overline{\tau(\overline{\chi_q})})\sum_{n=1}^{\infty}\hat{f}(n)n^{3/2}\chi_q(n)\sqrt{y}K_{ir}(2\pi ny)\\
&\qquad=i\pi\tau(\chi_q)(1-\chi_q(-1))\sum_{n=1}^{\infty}\hat{f}(n)n^{3/2}\chi_q(n)\sqrt{y}K_{ir}(2\pi ny). \qquad (4.4.7)
\end{aligned}$$

Taking Mellin transform of (4.4.7), we get

$$\begin{aligned}
&\sum_{k=0}^{q-1}\chi(k)\int_0^{\infty}\partial_x f(k/q+iy)y^{s-1/2}dy \qquad (4.4.8)\\
&=i\pi\tau(\chi_q)(1-\chi_q(-1))\sum_{n=1}^{\infty}\hat{f}(n)\chi_q(n)n^{3/2}\int_0^{\infty}K_{ir}(2\pi ny)y^s dy\\
&=i\pi\tau(\chi_q)\frac{1-\chi_q(-1)}{4(\pi)^{s+1}}L(s,f\times\chi_q)\Gamma\left(\frac{s+1+ir}{2}\right)\Gamma\left(\frac{s+1-ir}{2}\right).
\end{aligned}$$

Notice that (4.4.8) is analogous to (4.4.6). The algorithm to compute $\sum_{k=0}^{q-1}\chi(k)\int_0^{\infty}\partial_x f(k/q+iy)y^{s-1/2}dy$ is completely analogous to the algorithm to compute $\sum_{k=0}^{q-1}\chi(k)\int_0^{\infty}f(k/q+iy)y^{s-3/2}dy$. Hence throughout the rest of the paper, we will assume that $\chi_q(1)=\chi_q(-1)=1$. [2] Using the automorphy of f, we get the following lemma (analogous to the corresponding 4.3.2 of the previous

[2]Note that a similar treatment will give us the corresponding "geometric approximate functional equations" for odd Maass forms.

subsection).

Lemma 4.4.2. *Given any cusp form f (of weight k) on $\Gamma\backslash \mathrm{SL}(2,\mathbb{R})$, positive coprime integers n, q such that $n < q$, χ_q a character on $(\mathbb{Z}/q\mathbb{Z})^*$, $s \in \mathbb{H}$ and positive real γ, there exists a computable constant $c = O(1+\gamma)$, independent of k, q such that:*

$$\int_0^\infty f(n/q+iy)y^s dy = \int_{\frac{1}{cq^2\log q}}^{c\log q} f(n/q+iy)y^s dy + O(q^{-\gamma}). \tag{4.4.9}$$

The constant involved in O is independent of k, q.

Proof. If we use the Fourier expansion (4.2.1) for a Maass cusp form f at $z = k/q + ic\log q$, we get

$$f(n/q+ic\log q) = \sum_{m>0} \hat{f}(n) 2\sqrt{c\log q} K_{ir}(2\pi m \log q)\cos(2\pi n/q). \tag{4.4.10}$$

Similarly, using the Fourier expansion (4.2.2) for a holomorphic cusp form f, we get we get

$$f(n/q+ic\log q) = \sum_{m>0} \hat{f}(n) 2\exp(-2\pi cm\log q) e(mn/q). \tag{4.4.11}$$

We use the following quantification of the exponential decay of Bessel functions at infinity :

$$K_\nu(y) = \frac{\pi}{(2y)^{\frac{1}{2}}} e^{-y}(1+O(\frac{1+|\nu|^2}{y})). \tag{4.4.12}$$

We apply the bound $|\hat{f}(m)| \ll m^{k/2}$ to the equation (4.4.11) and use the method in lemma 4.3.2, to find a constant c_1, independent of n, q such that for every

$y \geq c_1 \log q$, we have

$$|f(n/q + c \log qi)| < q^{-|s|-\gamma-2} y^{-2}. \tag{4.4.13}$$

Similarly, for a Maass form, we can use (4.4.12) to (4.4.10) and the bound $|\hat{f}(n)| \ll 1$ to get the result.

Let n', n'' such that $0 \leq n' < q$ and $nn' - qn'' = 1$. The the action of $g = \begin{pmatrix} n' & -n'' \\ -q & n \end{pmatrix}$ on $\mathbb{H}$ maps n/q to infinity. Using the automorphy of f with respect to the action of g, we get

$$\begin{aligned} f(n/q + iy) &= (-q(n/q + iy) + n))^{-k} f(\frac{n'(n/q + iy) - n''}{-q(n/q + iy) + k}) \\ &= (-qiy)^{-k} f(\frac{1/q + in'y}{-qyi}) \\ &= (-qiy)^{-k} f(-n'/q + \frac{i}{q^2 y}). \end{aligned} \tag{4.4.14}$$

We use the exponential decay of f at infinity and the method in lemma 4.3.2, we get a constant c_2, independent of n, q such that for every $y \leq \frac{1}{c_2 q^2 \log q}$, we have

$$|y^{-k} f(n/q + iy)| < q^{-\gamma-|s|-2}. \tag{4.4.15}$$

The equations (4.4.13) and (4.4.15) give us the result.

□

After applying Lemma 4.4.2 to (4.4.3), we get that given any $q, \gamma > 0$ we have a computable constant C', independent of q, such that for any $C \geq C'$,

$$\frac{\tau(\chi_q)}{(2\pi)^{s+(k-1)/2}} L(s, f \times \chi_q)\Gamma(s + (k-1)/2) \tag{4.4.16}$$
$$= \sum_{j=0}^{q-1} \chi(j) \int_{\frac{1}{Cq^2 \log q}}^{C \log q} f(j/q + iy) y^{s+(k-3)/2} dy + O(q^{-\gamma}).$$

and similarly we get that given any $q, \gamma > 0$ we have a computable constant C, independent of q such that such that for any $C \geq C'$,

$$\tau(\chi_q) \frac{1 + \chi_q(-1)}{4(\pi)^s} L(s, f \times \chi_q)\Gamma\left(\frac{s+ir}{2}\right)\Gamma\left(\frac{s-ir}{2}\right) \tag{4.4.17}$$
$$= \sum_{k=0}^{q-1} \chi(k) \int_{\frac{1}{Cq^2 \log q}}^{C \log q} f(k/q + iy) y^{s-1} dy + O(T^{-\gamma}).$$

Notice that the integrals on the RHS of (4.4.16) and (4.4.17) are over hyperbolic curves of length $\approx \log q$, hence they can be computed in $O(q^{o(1)})$ time. The equations (4.4.16) and (4.4.17) denote "geometric approximate functional equations" to compute $L(s, f \times \chi_q)$ in the holomorphic and Maass form case respectively.

Chapter 5

Rapid algorithm in the 'T' aspect

Let Γ be a lattice in $\mathrm{SL}(2,\mathbb{Z})$. Let f be a (holomorphic or Maass) cusp form on $\Gamma\backslash\mathrm{SL}(2,\mathbb{R})$ of weight k. [1] The main goal in this section is to prove theorem 1.1. We use a geometric method to get a rapid algorithm to compute $L(f, 1/2+iT)$ in section 5.2. Before giving this algorithm, we will consider the problem of computing the large Fourier coefficients of f at a cusp. This algorithm is marginally simpler than the desired algorithm and will serve as a "toy model". The main theorem in this case can be given as

Theorem 5.1. *Given a lattice $\Gamma \leqslant \mathrm{SL}(2,\mathbb{R})$, a (holomorphic or Maass) cusp form f on $\Gamma\backslash\mathbb{H}$, positive reals γ, η, and a positive integer T, the T^{th} Fourier co-efficient of f at any cusp can be computed up to an error of $O(T^{-\gamma})$ in time $O(T^{7/8+\eta})$ using numbers of $O(\log T)$ bits. The constants involved in O are polynomials in $\frac{\gamma}{\eta}$ and are independent of T.*

We will give the proof of theorem 5.1 in section 5.1.

[1] For a Maass cusp form, the weight k=0.

5.1 Proof of theorem 5.1

Let Γ be a lattice in $\mathrm{SL}(2,\mathbb{R})$. Given real t, let

$$a(t) = \begin{pmatrix} e^{\frac{t}{2}} & 0 \\ 0 & e^{-\frac{t}{2}} \end{pmatrix},$$

and

$$n(t) = \begin{pmatrix} 1 & t \\ 0 & 1 \end{pmatrix}.$$

Let f be a holomorphic cusp form on $\Gamma\backslash\mathrm{SL}(2,\mathbb{R})$ of weight k. In this section we will only discuss the holomorphic case. The algorithm in the case of a Maass form f will be analogous. Recall the lift $\tilde{f}$ of f defined in (1.1.4): $\tilde{f} : \Gamma\backslash\mathrm{SL}(2,\mathbb{R}) \to \mathbb{C}$ is defined as:

$$\tilde{f} : \Gamma \begin{pmatrix} a & b \\ c & d \end{pmatrix} \to (ci+d)^{-k} f\left(\frac{ai+b}{ci+d}\right).$$

In section 8.2, we will prove that $\tilde{f}$ is "well behaved". In other words we can assume that $\tilde{f}$ has bounded derivatives.[2] Recall the Fourier expansion given by (4.2.2):

$$f(z) = \sum_{n>0} \hat{f}(n) e(nz).$$

[2]We actually prove that there exist a constant $c > 0$, depending on f such that for any $x \in \Gamma\backslash\mathrm{SL}(2,\mathbb{R})$, $|\partial^\beta \tilde{f}(x)| \ll c^{|\beta|}$ on $\Gamma\backslash\mathrm{SL}(2,\mathbb{R})$. The algorithm can be adapted easily if $c > 1$. To keep the algorithm cleaner, we assume that $\tilde{f}$ has bounded derivatives, *i.e* that $c = 1$.

Using this we get for any positive integer T,

$$\begin{aligned} e^{-2\pi}\hat{f}(T) &= \int_0^1 f(x+i/T)e(-Tx)dx \\ &= \int_0^T T^{k/2-1}\tilde{f}(a(-\log T)n(t))e(-t)dt. \end{aligned} \tag{5.1.1}$$

Notice that the integral on the RHS of (5.1.1) is integral of a well behaved smooth function on a horocycle of length T. Hence *a priori* we can 'compute' it in $O(T^{1+o(1)})$ time up to the error at most $O(T^{-\gamma})$, for any given γ. This will denote the corresponding (to (4.3.6) and (4.3.7)) "geometric functional equation" in this case. We will now explain a method to compute the T^{th} Fourier coefficient faster than $O(T)$.

We will write the integral (5.1.1) as a sum of integrals over intervals of length T^ϵ each. For simplicity let's assume T^ϵ is an integer.

$$\begin{aligned} \int_0^T \tilde{f}(x_0 n(t))e(t)dt = &\sum_{j=0}^{\lfloor T^{1-\epsilon}\rfloor-1} \int_0^{T^\epsilon} \tilde{f}(x_o n(jT^\epsilon+t))e(-t)dt \\ &+ \int_{(\lfloor T^{1-\epsilon}\rfloor-1)T^\epsilon}^{T} \tilde{f}(x_o n(jT^\epsilon+t))e(-t)dt. \end{aligned} \tag{5.1.2}$$

Here $x_0 = a(-\log T)$. The second integral on the right hand side of (5.1.2) is an integral of a smooth well behaved function on an interval of size $\leq T^\epsilon$. Hence given $\gamma, \eta > 0$, we can compute it up to an error of $O(T^{-\gamma})$ in time $O(T^{\epsilon+\eta})$, using proposition 7.1.1. (In practice, γ and η will be fixed "small" real numbers and ϵ will eventually be chosen to be 1/8.)

Let

$$M_2 = T^\epsilon.$$

Let us define $I_l(x, \tilde{f})$ the following way

Definition 5.1.1. *Given a cusp form $\tilde{f}$ on $\Gamma\backslash \mathrm{SL}(2,\mathbb{R})$, x in $\Gamma\backslash \mathrm{SL}(2,\mathbb{R})$ and l a non negative integer we define*

$$I_l(x, \tilde{f}) = \int_0^{M_2} t^l \tilde{f}(xn(t))e(-t)dt. \tag{5.1.3}$$

We use a geometric method to compute the inner integrals on the RHS of (5.1.2) "in parallel". We will use the following proposition which quantifies the slow divergence of the Horocycle flow:

Proposition 5.1.2. *Given any $\eta > 0$ and $\epsilon > 0$, for any $\epsilon' \geq 2\epsilon + \eta$ and for any x,y such that $x^{-1}y \in U_{1/T^{\epsilon'}}$ we have a constant c, independent of ϵ and ϵ' such that $yn(t) \in xn(t)U_{cT^{-\eta}}$ for all $0 \leq t \leq T^{\epsilon}$.*

Let $\eta > 0$ be any positive number. We take points $\{x_0 n(jM_2), 0 \leq j < T^{1-\epsilon}\}$ and first we "reduce" the points to an approximate fundamental domain. Then we "sort" the reduced points into sets $S_1, S_2, ..., S_{M_1}$ such that $x, y \in S_j \Rightarrow x^{-1}y \in U_{T^{-(2\epsilon+\eta)}}$. For each j let us choose a representative v_j from each S_j. We give a detailed accounting of the 'reduction' and 'sorting' algorithms in section (7).

If $x \in S_i$, then we use the power series expansion around $v_i n(t)$ to compute values of f at $xn(t)$. Here we use the following lemma:

Lemma 5.1.3. *Given $\gamma > 0, \eta > 0$, any $\epsilon > 0$, $x, y \in S_i$ for some i, then we have constants $c_{x,y,\beta,l}$ and d such that*

$$I_0(y, f) = \sum_{|\beta|<d, \beta \in \mathbb{Z}_+^3} \sum_{l=0}^{d} c_{x,y,\beta,l} I_l(x, \partial^\beta \tilde{f}) + O(M_2(\frac{\gamma}{\eta})^5 T^{-\gamma}). \tag{5.1.4}$$

Here, $d = O(\gamma/\eta)$.

Let us observe that the integrals involved on right hand side of (5.1.4) depend only on x. This lemma will allow us to compute the sum $I_0(y, f)$ in parallel for all the points y in S_i in $O(|S_i| + M_2)$ steps. Now, we will complete the proof of theorem 5.1 using lemma 5.1.3.

Proof of theorem 1. Let us compute the time spent in this algorithm. In the section 7 we will prove that the whole initial sorting process takes $O(T^{1-\epsilon})$ time.

Notice that the inner integrals on the right hand side of (5.1.4) are independent of y. For each $y \in S_i$, there are $O((\frac{\gamma}{\eta})^4)$ terms in the right hand side of equation (5.1.4). We will show in the section 8.4 that for fixed y and v_i we can compute all the required $c_{v_i,y,\beta,l}$'s in $O(d^6)$ time. Hence we can precompute the constants $c_{v_i,x,\beta,l}$, for each x in S_i for all i in $O((\frac{\gamma}{\eta})^6 T^{1-\epsilon})$ steps.

Recall that $M_2 = T^{\epsilon}$. Computing $I_l(v_i, \partial^{\beta} f)$ for $|\beta|, l < d$ takes $O(d^5 T^{\epsilon+\eta})$ operations for each v_i (using that $\tilde{f}$ has bounded derivatives along with proposition 7.1.1). Hence, using Lemma (5.1.3), we need $d^4|S_i|$ more operations to compute $I_0(x, f)$ for all $x \in S_i$. In the section 7.3, we will show that the maximum number of sets $S_i's$ is $O(T^{6\epsilon+3\eta} \log T)$ where the constant only depends on Γ. Notice that the log factor can be absorbed in the exponent η.

The second integral on the right hand side of (5.1.2), can be computed up to an error of at most $O(T^{-\gamma})$, using at most $O(T^{\epsilon+\eta})$ operations. Let us also recall that $d = O(\gamma/\eta)$. Hence the total number of operations needed to compute $\int_0^T \tilde{f}(x_o n(t)) e(t) dt$ up to an error of $O(T^{-\gamma})$ is

$$O((\frac{\gamma}{\eta})^6 (T^{\epsilon+\eta} T^{6\epsilon+3\eta} + T^{1-\epsilon})). \tag{5.1.5}$$

The optimal value for ϵ is $1/8$. This implies that given any $\eta, \gamma > 0$ and real

s_0, $L(f, s_0 + iT)$ can be computed up to an error at most $O(T^{-\gamma})$ using at most $O((\frac{\gamma}{\eta})^6 T^{7/8+4\eta})$ operations. (Notice, we have used that $d = O(\gamma/\eta)$). $\square$

5.2 Proof of theorem 1.1

Proof of this theorem uses an idea very similar to the proof of theorem 5.1. Let Γ be a lattice of $\mathrm{SL}(2, \mathbb{Z})$. Let f be a (holomorphic or Maass) cusp form on $\Gamma\backslash\mathbb{H}$. Let $\alpha = -1 + i/T$ and $c > 0$ be the constant in (4.3.6) and (4.3.7). Let $s = s_0 + iT$ and $M_1 = cT \log T$. We have

$$\int_{M_1}^{\infty} |f(\alpha t) t^{s-1}| dt + \int_{0}^{M_1} |f(\alpha t) t^{s-1}| dt < O(T^{-\gamma}) \tag{5.2.1}$$

Let s_0 be some fixed real. Using the "geometric approximate functional equations" (4.3.6) and (4.3.7), it is enough to give an algorithm to compute

$$\int_{\frac{1}{M_1}}^{M_1} f(\alpha t) t^{s-1} dt.$$

We proceed in the similar manner as in the last section, and write the above integral as a sum of integrals over segments of hyperbolic length $\approx T^{\epsilon}$. In order to do so, let us first define the following quantities:

Definition 5.2.1.

$$x_0 = \frac{1}{cT \log T}(-1 + i/T),$$

$$M = 1 + T^{-1+\epsilon},$$

$$b_j = M^j,$$

$$x_j = b_j x_0.$$

Notice that there exists an integer $M_3 = O(T^{1-\epsilon}\log T)$ such that,

$$M_1 < (1+T^{-1+\epsilon})^{M_3}\frac{1}{cT\log T} = \frac{b_{M_3}}{cT\log T}.$$

Let

$$M_4 = \frac{b_{M_3+1}}{cT\log T}.$$

Recall that constant c is chosen such that it satisfies conditions in (5.2.1). Hence we get that $\int_{M_1}^{M_4}|f(\alpha t)t^{s-1}|dt \ll T^{-\gamma}$. Hence it is enough to give an algorithm to compute

$$\int_{1/M_1}^{M_4} f(\alpha t)t^{s-1}dt$$

up to an error at most $O(T^{-\gamma})$.

$$\begin{aligned}\int_{1/M_1}^{M_4} f(\alpha t)t^{s-1}dt &= \sum_{j=0}^{M_3}\int_0^{b_j(M-1)} f(\alpha(b_j+t))(b_j+t)^{s-1}dt \\ &= \sum_{j=0}^{M_3} b_j^s \int_0^{M-1} f(\alpha(b_j(1+t))(1+t)^{s-1}dt. \end{aligned} \tag{5.2.2}$$

We next define:

$$\kappa(t) = \begin{pmatrix} (\frac{t}{T})^{\frac{1}{2}} & -\sqrt{Tt} \\ 0 & (\frac{t}{T})^{-\frac{1}{2}} \end{pmatrix}. \tag{5.2.3}$$

Notice,

$$\kappa(b_j)^{-1}\kappa(b_j(1+t)) = \begin{pmatrix} (1+t)^{1/2} & T(-(1+t)^{1/2}+(1+t)^{-1/2}) \\ 0 & (1+t)^{-1/2} \end{pmatrix}$$

and that it is independent of j. Let

$$\omega(t) = \begin{pmatrix} (1+t)^{1/2} & T(-(1+t)^{1/2} + (1+t)^{-1/2}) \\ 0 & (1+t)^{-1/2} \end{pmatrix}. \tag{5.2.4}$$

We rewrite equation (5.2.2) as

$$\begin{aligned} \int_{1/M_1}^{M_4} f(\alpha t) t^{s-1} dt &= T^{k/2} \int_{1/M_1}^{M_4} t^{s-k/2-1} \tilde{f}(\kappa(t)) dt \\ &= T^{k/2} \sum_{j=0}^{M_3} (b_j)^{s-k/2} J(f, x_j) \end{aligned} \tag{5.2.5}$$

Here,

$$J(f, y) = \int_0^{M-1} \tilde{f}(y\omega(t))(1+t)^{s-\frac{k}{2}-1} dt$$

We make use of the fact that we are integrating on an "approximate horocycle". We quantify this result in the following proposition. This proposition is analogous to proposition 5.1.2 .

Proposition 5.2.2. *Given any $\epsilon > 0, \eta > 0$, let $x, y \in \mathrm{SL}(2, \mathbb{R})$ such that $x^{-1}y \in U_{T^{-2\epsilon-\eta}}$, then we have a constant c' such that*

$$y\omega(t) \in x\omega(t) U_{c' \frac{1}{T^\eta}} \text{ for all } 0 \leq t \leq T^{-1+\epsilon}.$$

c' can be chosen independent of ϵ and T.

The propositions 5.2.2 will be proved in section 8.3. Let's 'reduce and sort' the

points $\{x_j\}$ into $N = O(T^{6\epsilon+3\eta})$ groups $S_1, ..., S_N$ such that

$$x, y \in S_i \Rightarrow x^{-1}y \in U_{\frac{1}{T^{2\epsilon+\eta}}}.$$

We will discuss these algorithms in sections 7.2 and 7.3. Let us also choose fixed representatives v_i from each S_i. For each group S_i, let us try to compute the inner integral in (5.2.5) "in parallel" .

Let $x, y \in S_i$, then we use power series expansion around points $x\omega(t)$ to compute $\tilde{f}(y\omega(t))$. Hence we get,

Lemma 5.2.3. *Given any $\eta > 0, \gamma > 0, \epsilon > 0$, and $x, y \in S_i$ for some integer i, then there exist constants $e_{x,y,\beta,l}$ and d', independent of T, such that we can write*

$$J(f, y) = \sum_{|\beta|<d', \beta\in\mathbb{Z}^3_+} \sum_{l=0}^{d'} e_{\beta,l,x,y} L_l(\partial^\beta \tilde{f}, x) + O(M_2(d')^4 T^{-\gamma}).$$

Where

$$L_l(g, x) = \int_0^{M-1} (Tt)^l g(x\omega(t))(1+t)^{s-k/2-1} dt$$

and $d' = O(\gamma/\eta)$ The constants involved in O only depends on f, and are independent of x, γ, η and T.

The lemma 5.2.3 converts our problem into the problem of computing $L_l(\partial^\beta \tilde{f}, v_i)$ for each v_i. Change the variable $u = Tt$ to get

$$L_l(g, x) = T^{-1} \int_0^{T^\epsilon} u^l g(x\omega(u/T))(1+u/T)^{s-k/2-1} du.$$

Notice that each integral $L_l(\partial^\beta \tilde{f}, v_i)$ is integral of $\partial^\beta \tilde{f}$ on a segment of hyperbolic length $\approx T^\epsilon$, hence we can compute it up to an error of $O(T^{-\gamma})$ in time $O(T^\epsilon)$.

More rigorously, we prove:

Proposition 5.2.4. *Given, non negative integer l and vector $\beta \in \mathbb{Z}_+^3$, $x \in \Gamma\backslash \mathrm{SL}(2,\mathbb{R})$ and given positive reals γ and η we can compute $L_l(\partial^\beta \tilde{f}, x)$ up to an error at most $O(T^{-\gamma})$ in $O((lT^\epsilon)^{1+\eta})$ time.*

Proof. Let g_l be the function defined by $g_l(u) = u^l(1+u/T)^{s-k/2-1}$. Then there exists a constant C, independent of l such that for any

$$k \in \mathbb{N}, |\partial^k g_l| \leq (lC)^k. \tag{5.2.6}$$

If we prove that for any $k \in \mathbb{N}$, fixed $\beta \in \mathbb{Z}_+^3$ and for any $x \in \Gamma\backslash \mathrm{SL}(2,\mathbb{R})$, there exists a constant D independent of x and l such that,

$$|\partial^k (g_l \partial^\beta \tilde{f}(x\omega(u/T))| \leq (lD)^k$$

then using proposition 7.1.1, we get the result.

Recall that

$$\begin{aligned}\omega(u/T) &= \begin{pmatrix} (1+u/T)^{1/2} & T(-(1+u/T)^{1/2} + (1+u/T)^{-1/2}) \\ 0 & (1+u/T)^{-1/2} \end{pmatrix} \\ &= n(-u)a(\log(1+u/T)). \end{aligned} \tag{5.2.7}$$

Notice that we have assumed that for any β in $\mathbb{Z}_+^3$ and any $x \in \Gamma\backslash \mathrm{SL}(2,\mathbb{R})$, $\partial^\beta \tilde{f}(x) \ll 1$. Hence, for $|u| \leq 1$, We can use the power series expansion for $\tilde{f}$ (ref

2.2.2) to get

$$\partial^{\beta}\tilde{f}(x\omega(u/T)) = \sum_{|\beta'|=0,\beta\in\mathbb{Z}_{+}^{2}\times 0}^{\infty} \frac{\partial^{\beta'}\partial^{\beta}\tilde{f}(x)}{\beta'!}(-u)^{\beta_1}(\log(1+u/T))^{\beta_2}.$$

Differentiate with respect to u and use the claim 8.2.1, to get

$$\left|\partial^{k}|_{u=0}\partial^{\beta}\tilde{f}(x\omega(u/T))\right| \ll 4^{k}.$$

In general, for any $0 \leq u_0 \leq T^{\epsilon}$, we can use the power series expansion for $\partial^{\beta}\tilde{f}$ around $x\omega(u_0)$ to compute $\partial^{\beta}\tilde{f}(x\omega(u_0+u))$ for small u and use similar method as before, along with bounds (5.2.6) to get the desired uniform bound D. □

Proof. proof of theorem 1.1

Using Lemma 5.2.3, proposition 5.2.4 and following exactly similar steps as in the proof of theorem 5.1, we get the result. □

Chapter 6

Rapid algorithm in the 'q' aspect

Let $\Gamma = \mathrm{SL}(2,\mathbb{Z})$ and f be a (holomorphic or Maass) form of weight k on $\Gamma\backslash\mathbb{H}$. Let $q = MN$ where $M \leq N$, $M = M_1M_2$, where $M_1 = \gcd(M,N)$ and $(M_2,N) = 1$. Let χ_q be a Dirichlet character on $\mathbb{Z}/q\mathbb{Z}$. Let γ,η be any given positive numbers. The main goal of this chapter is to prove theorem 1.2.

Theorem 6.1 will prove that the algorithm to compute $L(s, f\times\chi_q)$ faster than $O(q)$ is equivalent to computing the Gauss sum $\tau(\chi_q)$ and

$$\sum_{k=0}^{q-1}\chi_q(k)\partial_2^l\tilde{f}(n(k/q)a(t)) \tag{6.0.1}$$

for approximately $O(q^{o(1)})$ equispaced values of $t \in [-3C\log q, 3C\log q]$ and for $0 \leq l \leq N'$, faster than $O(q)$. Here ∂_2 denotes $\frac{\partial}{\partial x_2}$, $N' = O(\gamma/\eta)$ and C is a suitable constant, independent of q. Hence, in this section we will consider the problem of computing the sum in (6.0.1) for any given $t \in [-3C\log q, 3C\log q]$.

In the section 8.6, we will prove that given any positive γ, we can compute $\tau(\chi_q)$ up to error $O(q^{-\gamma})$, using at most $O(M^2+N)$ operations. The algorithm

in section 8.6 is similar to the algorithm to compute (6.0.1). This algorithm is simpler, and can be helpful in understanding the main algorithm in this section better.

We will deal with computing (6.0.1) for the special cases where $q = MN$ where $(M, N) = 1$ and the case when $M|N$ separately in the sections 6.1 and 6.2 respectively. We use these results to complete the proof of theorem 1.2 in section 6.3.

Let us start with the "geometric approximate functional equations" (4.4.16)/ (4.4.17) for holomorphic/Maass case respectively. Notice that the constant C in (4.4.16)/ (4.4.17) can be chosen to be greater than 1. We use the lift $\tilde{f}$ defined in (1.1.4) to get $\tilde{f}(n(x)a(\log y)) = y^{k/2}f(x+iy)$ to rewite (4.4.16) as:

$$\frac{\tau(\chi_q)}{(2\pi)^{s+(k-1)/2}}L(s, f\times\chi_q)\Gamma(s+(k-1)/2)$$
$$= \sum_{j=0}^{q-1}\chi(j)\int_{\frac{1}{Dq^2\log q}}^{D\log q}\tilde{f}(n(j/q)a(\log y))y^{s-3/2}dy + O(q^{-\gamma}).$$

Notice that the above equation will be valid if the constant D is replaced by any $C \geq D$. We will choose a suitable C later. Substitute $\log y = t$ in the above equation to get that for any $C \geq e^D$ we have:

$$\frac{\tau(\chi_q)L(s, f\times\chi_q)\Gamma(s+(k-1)/2)}{(2\pi)^{s+(k-1)/2}} = C'\sum_{j=0}^{q-1}\chi_q(j)\int_{-3C\log q}^{3C\log q} g_j(t)dt + O(q^{-\gamma}). \tag{6.0.2}$$

Here $C' = q^{|3C\mathrm{Re}(s-\frac{1}{2})|}$ and

$$g_j(t) = \frac{1}{C'}\tilde{f}(n(j/q)a(t))\exp(t(s-1/2)).$$

Notice that $\frac{d}{dt}\mid_{t_0}(\tilde{f}(n(j/q)a(t))) = \frac{\partial}{\partial x_2}\tilde{f}(n(j/q)a(t_0))$. Here $\frac{\partial}{\partial x_2}$ denote the derivative of $\tilde{f}$ in the "geodesic direction" defined in definition 2.2.2. We have assumed that $\tilde{f}$ has bounded derivatives (ref. chapter 2). Hence for a fixed $s \in \mathbb{C}$ and each t in $[-3C\log q, 3C\log q]$,

$$\frac{\partial^n}{\partial t^n} g_j(t_0) \ll_f (|s-1/2|+1)^n.$$

The constant involved is independent of q. Hence for each t in $[-3\log q, 3\log q]$, g_j is real analytic with radius of convergence at least $1/(|s-1/2|+1)$. Hence, using the method in the proof of theorem 7.1.1 (choosing a grid of $O(q^\eta)$ equispaced points and using power series expansion at the nearest grid point on the left to compute g_k at any given point), given any $\gamma, \eta > 0$ we get

$$\begin{aligned}&\int_{-3C\log q}^{3C\log q} \tilde{f}(n(j/q)a(t))exp(t(s-1/2))dt \\ &= C' \sum_{x=-3Cq^\eta \log q}^{3Cq^\eta \log q - 1} \sum_{l=0}^{N'} \int_0^{q^{-\eta}} \partial_2^l(g_j(xq^{-\eta}))\frac{t^l}{l!}dt + O(q^{-1-\gamma}) \qquad (6.0.3)\end{aligned}$$

Here $N' = O((1+\gamma)/\eta)$. Notice that the above equation is true for any $C > e^D$. Hence , given η, we can choose C such that $\{3Cq^\eta \log q\} = 0$. Multiplying (6.0.3) by $\chi_q(k)$ and summing over k, we get

$$L(s, f\times\chi_q) = C'' \sum_{x=-3Cq^\eta\log q}^{3Cq^\eta\log q-1} \sum_{l=0}^{N'} d_l \sum_{j=0}^{q-1} \chi_q(j)\partial^l(g_j(xq^{-\eta})) + O(q^{-\gamma}) \qquad (6.0.4)$$

Here $d_l = \int_0^{q^{-\eta}} \frac{t^l}{l!}dt$ and $C'' = \frac{C'(2\pi)^{s+(k-1)/2}}{\Gamma(s+(k-1)/2)\tau(\chi_q)}$.

(6.0.4), along with the definition of $g_j(t)$ implies that the problem of computing $L(s, f\times\chi)$ is equivalent to computing $\tau(\chi_q)$ and $\sum_{j=0}^{q-1}\chi(j)\partial_2^l\tilde{f}(n(j/q)a(t))$ for any

$t \in [-3\log q, 3\log q]$ and for $0 \leq l \leq N'$ faster than $O(q)$. Here ∂_2 denotes $\frac{\partial}{\partial x_2}$. Exactly same method will work for the Maass forms as well. [1]

We have thus proved

Theorem 6.1. *Let f be a modular (holomorphic or Maass) cusp form on $\Gamma\backslash\mathbb{H}$, and χ_q be a Dirichlet character on $\mathbb{Z}/q\mathbb{Z}$, s be any complex number. Let γ, η be any positive reals, then there exists a positive integer $N' = O(\gamma/\eta)$ such that if for any $t \in [-3C\log q, 3C\log q]$ and for any $0 \leq l \leq N'$, we can compute $\sum_{j=0}^{q-1} \chi(j)\partial_2^l \tilde{f}(n(j/q)a(t))$ up to a maximum error $O((\eta/\gamma)^5 q^{-\gamma})$ in time $D(q)$, then we can compute $\tau(\chi_q)L(s, f \times \chi_q)$ using $O(D(q)q^\eta)$ operations.*

In the following sections, we prove that $D(q) = (M^{5+\eta} + N)^{1+o(1)}$. We use this result, along with the algorithm in section 8.6, to finish the proof of theorem 1.2.

6.1 Case $q = MN$ where $(M, N) = 1$

Let g be a real analytic function with bounded derivatives on $\Gamma\backslash\mathrm{SL}(2,\mathbb{R})$. Let $q = MN$ where $(M, N) = 1$. Let t be a real number in $[-3C\log q, 3C\log q]$, $t_1 = t + \log q$. Let

$$x_0 = a(t_1).$$

[1] For an even Maass form if $\chi_q(-1) = -1$, then we need to compute sums of type $\sum_{j=0}^{q-1} \chi(j) \int_{\frac{1}{Dq^2\log q}}^{D\log q} \partial_x f(j/q + iy) y^{s-1/2} dy$. Notice that $\tilde{f}(n(x)a(\log y)) = f(x + iy)$. Hence for any x_0 we have , $\partial_x|_{x_0} \tilde{f}(n(x)a(\log y)) = \partial_x f(x_0 + iy)$. But we have $n(x)a(\log y) = a(\log y)n(x/\sqrt{y})$. This implies that $\partial_x|_{x_0} \tilde{f}(n(x)a(\log y)) = \partial_x|_{x_0} \tilde{f}((\log y)n(x/y))$. This implies that $\partial_x|_{x_0} \tilde{f}(n(x)a(\log y)) = y^{-1}\partial_1 \tilde{f}((\log y)n(x_0/y))$. Recall ∂_1 is defined in section 2.2. This implies that

$$\partial_x f(x_0 + iy) = y^{-1}\partial_1 \tilde{f}(n(x_0)a(\log y)).$$

Using this equation, we can follow exactly the same method in this chapter, to get the required algorithm.

In this section, we will consider the problem of computing the sum

$$\begin{aligned} S &= \sum_{k=0}^{q-1} \chi_q(k) g(n(k/q)a(t)) \\ &= \sum_{k=0}^{q-1} \chi_q(k) g(A(q,1,k)a(t_1)) \\ &= \sum_{j=0}^{N-1} \sum_{k=0}^{M-1} \chi_q(j+Nk) g(A(q,1,j+kN)a(t_1)). \end{aligned} \tag{6.1.1}$$

Computing the sum of this type is equivalent to computing the sum (6.0.1). Recall that $A(q,1,k)$ is defined in section 2.3. Use, $A(q,1,j+kN) = A(M,1,k)A(N,1,j)$ to rewrite (6.1.1) as

$$S = \sum_{j=0}^{N-1} S_j. \tag{6.1.2}$$

Here, S_j is defined as

$$S_j = \sum_{k=0}^{M-1} \chi_q(j+kN) g(A(M,1,k)v_j). \tag{6.1.3}$$

Here

$$v_j = A(N,1,j)x_0.$$

(Ref. figure 1.3). We now use $\chi_q = \chi_M \chi_N$ to get

$$\begin{aligned} S_j &= \sum_{k=0}^{M-1} \chi_q(j+kN) g(A(M,1,k)v_j) \\ &= \chi_N(j) \sum_{k=0}^{M-1} \chi_M(j+kN) g(A(M,1,k)v_j). \end{aligned} \tag{6.1.4}$$

Let us "reduce" the points $v_1, ..., v_n$ to an approximate fundamental domain

using the algorithm in chapter 7. Hence we have matrices $\{\gamma_j, x_j\}$ such that $v_j = \gamma_j x_j$ and x_j lies in the "approximate fundamental domain".

Notice that χ_M is a character on $\mathbb{Z}/M\mathbb{Z}$. Given j let b_j is defined by $j \equiv b_j($ mod $M)$. Hence we rewrite (6.1.4) as

$$S_j = \chi_N(j) \sum_{m|M} \sum_{k=0}^{M/m-1} h_{b_j}(m,k) g(A(M,m,k)\gamma_j x_j). \tag{6.1.5}$$

Here for $\{0 \leq j \leq q-1\}$, $h_j : T(M) \to \mathbb{C}^\times$ is defined as:

$$h_j((m,k)) = \delta_1(m)\chi_M(j+kN). \tag{6.1.6}$$

Where, δ_1 is the characteristic function of $\{1\}$. [2]

The number of points in $T(M)$ are at most $M^{1+\eta}$ (using the fact that the number of divisors of M are at most $O(M^\eta)$). We know that the right action by $a \in \Gamma$ permutes the $T(M)$ in $\Gamma\backslash \mathrm{SL}(2,\mathbb{R})$. In other words, given any element a of Γ, there exists a permutation σ_a on $T(M)$ such that for each $(m,k) \in T(M)$, we have an $a' \in \Gamma$ such that

$$A(M,m,k)a = a' A(M, \sigma_a(m,k)).$$

We prove in the section 8.5 that if $a \equiv b \mod M$, then $\sigma_a = \sigma_b$. This implies that the total number of permutations of $T(M)$ due to the right action of Γ is at

[2] In particular, for a prime number M, the functions h_j are given explicitly by: $h_j(1,k) = \chi_M(j+kN)$ and $h_j(M,0) = 0$

most M^3. Using this, we rewrite (6.1.5) as

$$S_j = \chi_N(j) \sum_{(m,k)\in T(M)} h_{b_j}(\sigma_{\gamma_j}(m,k)) g(A(M,m,k)x_j). \tag{6.1.7}$$

Notice that the number of distinct functions $h_{b_j}\sigma_{\gamma_j}$ is at most M^4 (ref. section 8.5). Let us enumerate them as $r_1, ..., r_L$ (say) , where $L \leq M^4$.

Notice that if the Hecke orbits "preserve" the distance between the points. In other words, for any positive integer M, and any $x, y \in \mathrm{SL}(2,\mathbb{R})$ such that $x \in yV$ for some open neighbourhood V of the identity. Then $A(M,m,k)x$ lies in $A(M,m,k)yV$, for all $(m,k) \in T(M)$. Hence given any positive η, we "sort" the points $\{x_j\}$ into sets $K_1, ..., K_Q$ such that $x, y \in K_k$ implies that $x^{-1}y \in U_{q^{-\eta}}$. Here $Q = O(q^{3\eta})$. We choose a representative w_j from each K_j.

Let us focus on K_1. Let us use a power series expansion around the point $A(M,m,k)w_1$ to compute $g(A(M,m,k)x)$ for all $x \in K_1$. We summarize the result in the following lemma:

Lemma 6.1.1. *Given any positive reals η, γ, positive integer M, x and $y \in K_i$ for some integer i. Let r be any complex valued function defined on the the set $T(M)$. Then there are explicitly computable constants $c_{\beta,x,y}$ and d such that,*

$$J(0,r,y) = \sum_{|\beta|=0}^{d} c_{\beta,x,y} J(\beta,r,x) + O(2M^2 q^{-\gamma}). \tag{6.1.8}$$

Here $d = O(\gamma/\eta)$ and $J(\beta,r,x)$ is defined by

$$J(\beta,r,x) = \sum_{(m,k)\in T(M)} r((m,k))\partial^\beta g(A(M,m,k)x). \tag{6.1.9}$$

The constant in O is independent of M, q and is a polynomial in γ/η.

We will prove the lemma 6.1.1 in section 8.3. Notice that for fixed β, r and a fixed x, we can compute $J(\beta, r, x)$ in time $O(M^{1+\eta})$. This gives us an exact idea of the algorithm. The exact algorithm is given by:

6.1.1 Explicit algorithm

1. Compute and store the functions $r_1, ..., r_L$.
2. Reduction of points $v_1, ..., v_N$ into points $x_1, ..., x_N$ and sorting of the points into sets $K_1, ..., K_Q$ and choose a representative x_{n_i} from each K_i.
3. For each $0 \leq l < N$ find j_l such that $h_{b_l}\sigma_{\gamma_l} = r_{j_l}$.
4. $i \leftarrow 1$.
5. for all x_l in K_i, and for all $|\beta| \leq d$, compute and store c_{β, x_{n_i}, x_l}.
6. for all $|\beta| \leq d$, and for all $0 \leq t \leq L-1$, compute and store $J(\beta, r_t, x_{n_i})$.
7. for each x_l in K_i, compute

$$S_l = \chi_N(l) \sum_{|\beta|=0}^{d} c_{\beta, x_{n_i}, x_l} J(\beta, r_{j_l}, x_{n_i},).$$

8. if $i = Q$ compute $S = S_1 + ... + S_N$ else $i \leftarrow i + 1$ and go to step 5.

6.1.2 Time complexity

Let us compute the time complexity in the algorithm:

Computing and storing r_i for all i takes at most (roughly) $O(LM^{1+\eta})$ time. But recall that $L \leq M^4$. Hence step 1 takes $O(M^{5+\eta})$ time. The reduction and sorting process in step 2 takes $O(N)$ time. Step 3 also takes $O(N)$ time. Notice that for each x_l, step 5 takes $O(d^3)$ steps. Hence for all l, step 5 takes $O(d^3N)$ steps. Notice that for a fixed i, and fixed t, step 6 takes $O(d^3M^{1+\eta})$ time. Hence for fixed i, and for all t, step 6 takes $O(d^3LM^{1+\eta})$ time. Recall that $L \leq M^4$ hence total time spent for fixed i and all t is $O(d^3M^{5+\eta})$. Hence for all i, for all j and for all t, the step 6 takes $O(d^3QM^{5+\eta}) \approx O(d^3M^{5+4\eta})$ time. Recall here that $Q \approx M^{3\eta}$ and $d = O(\gamma/\eta)$. Steps 7 and 8 take $O(d^3N)$ steps.
Hence the total time spent is $O((\gamma/\eta)^3(M^{5+7\eta} + N))$.

6.2 Case $q = MN$ when $M|N$

Let $q = MN$ where $(M, N) = 1$. Unless redefined in this section, we borrow the notations from previous section.

We proceed as in previous section and rewrite (6.1.1) as

$$S = \sum_{j=0}^{N-1} S_j. \tag{6.2.1}$$

Here, S_j is defined as

$$S_j = \sum_{k=0}^{M-1} \chi_q(j + kN) g(A(M, 1, k)v_j). \tag{6.2.2}$$

v_j as in the previous section. We now use the fact that there exists a constant b

such that $\chi_q(1+kN) = e(bk/M)$ (8.6.2). Using this we have

$$S_j = \chi_q(j) \sum_{k=0}^{M-1} e(bj^{-1}k/M) g(A(M,1,k)v_j). \tag{6.2.3}$$

Notice that j^{-1} denotes the multiplicative inverse of $j \mod m$. Let b_j be defined by $b_j \equiv j^{-1} \mod M$, where $0 \leq b_j \leq q-1$. We rewrite (6.2.3) as

$$S_j = \chi_q(j) \sum_{(m,k)\in T(M)}^{M/m-1} h_{b_j}((m,k)) g(A(M,m,k)v_j). \tag{6.2.4}$$

Here $h_j : T(M) \to \mathbb{C}^\times$

$$h_j((m,k)) = \delta_1(m) e(bjk/M). \tag{6.2.5}$$

Notice again that there are M distinct functions h_{b_j}. We can proceed exactly similarly to get the required algorithm.

6.3 Proof of theorem 1.2

Let $q = MN$. Let $M_1 = (M,N)$ and $M_2 = M/M_1$ such that $(M_2,N) = 1$. In chapter 8 Appendix, we will give a $O(M^2)$ algorithm to compute $\tau(\chi_q)$. Hence it only remains to give an algorithm to compute (6.1.1).
We proceed as in previous sections and rewrite (6.1.1) as

$$S = \sum_{j=0}^{N-1} S_j. \tag{6.3.1}$$

Here, S_j is defined as

$$S_j = \sum_{k=0}^{M-1} \chi_q(j+kN) g(A(M,1,k)v_j). \tag{6.3.2}$$

v_j as in the previous sections. Notice that $\chi_q = \chi_{M_2}\chi_{M_1N}$ and that there exists b such that $\chi_{M_1N}(1+kN) = e(kb/M_1)$. We rewrite (6.3.2) as

$$\begin{aligned}
S_j &= \sum_{k=0}^{M_1-1}\sum_{l=0}^{M_2-1} \chi_q(j+(k+lM_1)N) g(A(M,1,k+lM_1)v_j) \\
&= \sum_{k=0}^{M_1-1}\sum_{l=0}^{M_2-1} \chi_q(j+kN+lM_1N) g(A(M,1,k+lM_1)v_j) \\
&= \sum_{k=0}^{M_1-1} \chi_{M_1N}(j+kN) \sum_{l=0}^{M_2-1} \chi_{M_2}(j+kN+lM_1N) g(A(M,1,k+lM_1)v_j) \\
&= \chi_{M_1N}(j) \sum_{k=0}^{M_1-1} e(\frac{bj^{-1}k}{M_1}) \sum_{l=0}^{M_2-1} \chi_{M_2}(j+kN+lM_1N) g(A(M,1,k+lM_1)v_j).
\end{aligned} \tag{6.3.3}$$

Notice that for fixed k and l, the quantity $e(bj^{-1}k/M_1)$ is uniquely determined for $j \mod M_1$ and $\chi_{M_2}(j+kN+lM_1N)$ is uniquely determined for $j \mod M_2$. Hence if $j_1 \equiv j_2 \mod M$ then for all $0 \le k \le M_1-1$ and $0 \le l \le M_2-1$ we have that,

$$e(\frac{j_1^{-1}bk}{M_1})\chi_{M_2}(j_1+kN+lM_1N) = e(\frac{j_2^{-1}bk}{M_1})\chi_{M_2}(j_2+kN+lM_1N).$$

Let b_j be defined by $b_j \equiv j \mod M$, where $0 \le b_j \le M-1$. We can rewrite (6.3.3) as

$$S_j = \chi_{M_1N}(j) \sum_{(m,k)\in T(M)} h_{b_j}((m,k)) g(A(M,m,k)v_j). \tag{6.3.4}$$

Here $h_{b_j} : T(M) \to \mathbb{C}^\times$ is defined by:

$$h_{b_j}((m,k)) = \delta_1(m)e(bb_j^{-1}k/M_1)\chi_{M_2}(b_j + k_0N + l_0M_1N); \tag{6.3.5}$$

where,

$$k = k_0 + l_0M_1,\ 0 \leq k_0 \leq M_1 - 1,\ 0 \leq l_0 \leq M_2 - 1.$$

The inverse in (6.3.5) is $\mod M_1$. Notice again that there are only M distinct functions h_{b_j} on $T(M)$. Hence using the same algorithm as in the previous sections along with theorem 6.1 we get the result.

Chapter 7

Computational issues and 'sorting' algorithms

In this chapter, we will deal with the various issues regarding implementation of the algorithms. In section 7.1 we will state and prove proposition 7.1.1, which is used frequently in the thesis to compute inetgrals of "well behaved" functions on a finite interval. In sections 7.2 and 7.3, we explicitly give the reduction and sorting algorithm used in proving theorems 1.1, 1.2 and 5.1. Throughout the thesis we have assumed the real number model of computation. The algorithms however work if we work with numbers specified by $O(\log T)$ (or $O(\log q)$) digits. In section 7.4, we deal with issues regarding the machine error in our algorithms due to working with such finite precision numbers. We will also deal with issues regarding the input of f, Γ and χ_q in section 7.4.

7.1 Numerical integration for analytic functions

We have used the following theorem about numerical integration of a real analytic function repeatedly in the paper. We will state and prove it in this section.

Proposition 7.1.1. *Let f be a real analytic function on (a,b), a,b extended real numbers. Let R be such that for any $k \in \mathbb{N}$ of $|\partial^k f(x)| \ll R^k$ for all x in (a,b). Let $[c,d]$ in (a,b) and let $T \geq 1$ be any positive constant. If at any given point in $[c,d]$ and $n \in \mathbb{N}$ and any $\gamma > 0$, we can compute n^{th} derivative of f up to an error of $O(T^{-\gamma})$ in polynomial (in n and γ) time, then for any given $\epsilon, D > 0$, we can compute $\int_c^d f(t)dt$ up to an error at most $O(R|d-c|T^{-D})$, using at most $O(R|d-c|T^{\epsilon})$ arithmetic operations. The constant involved is polynomial in D/ϵ.*

Proof. Let us assume that f is defined on an interval of type $[0, d-c]$. The idea is to use a fine grid of $|c-d|T^{\epsilon}R$ equispaced points and use power series expansion around the nearest left grid point to compute f. In particular, let us split the integral into integrals of size $T^{-\epsilon}/R$ each.

Let us define

$$M = T^{-\epsilon}/R, M_2 = \lfloor |d-c|/M \rfloor - 1.$$

Hence

$$\int_0^{|d-c|} f(t)dt = \sum_{x=0}^{M_2} \int_0^M f(xM+t)dt + \int_{M(M_2+1)}^{|d-c|} f(t)dt. \tag{7.1.1}$$

Let us use power series expansion around each xM to compute the value of f at a nearby point. Hence we get

$$f(xM+t) = \sum_{l=0}^{\infty} \partial^l(f)(xM)\frac{t^l}{l!}.$$

For $|t| < M$ and any non negative integer N, we get

$$|f(xM+t) - \sum_{l=0}^{N} \partial^l(f)(x)\frac{t^l}{l!}| \leq \sum_{l=N+1}^{\infty} \frac{T^{-\epsilon l}}{l!} \leq eT^{-\epsilon N}. \tag{7.1.2}$$

Substituting this in equation(7.1.1), we get

$$\begin{aligned}
\int_0^{d-c} f(t)dt &= \sum_{x=0}^{M_2}\sum_{l=0}^{N}\int_0^M \partial^l(f)(x)\frac{t^l}{l!}dt \\
&\quad + \sum_{l=0}^{N}\partial^l f(M(M_2+1))\int_0^{d-c-MM_2-M}\frac{t^l}{l!} + E \\
&= \sum_{x=0}^{M_2}\sum_{l=0}^{N}\partial^l(f)(x)\int_0^M \frac{t^l}{l!}dt \\
&\quad + \sum_{l=0}^{N}\partial^l f(M(M_2+1))\int_0^{d-c-MM_2-M}\frac{t^l}{l!} + E.
\end{aligned} \tag{7.1.3}$$

Here,

$$|E| \ll e(M_2+1)T^{-N\epsilon}.$$

Notice that any $\partial^l f(x)$ can be computed at any x up to $T^{-D-\epsilon}$ error in polynomial in l and $D+\epsilon$ time. Hence the total error in (7.1.3) due to the error in computing $\partial^l(f)(x)$ is at most $O(R|d-c|T^{-D})$. The constant involved in O is polynomial in $D+\epsilon$ and N. Notice that the total number of operations needed to compute the sum on the RHS of (7.1.2) is $O(M_2N)$. We choose $N = \frac{D}{\epsilon}$ to get the result. □

7.2 Sorting and Reduction

In this section, we will present an algorithm for 'reducing' a point z in $\mathbb{H}$ to an approximate fundamental domain. Throughout the section, given $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ and z in $\mathbb{H}$, let us denote

$$Az = \frac{az+b}{cz+d}.$$

Let us recall some properties of the Fuchsian groups of first kind that we will be using.

Throughout this section we will use Gothic characters $\mathfrak{a}, \mathfrak{b}, \mathfrak{c}, ...$ to denote the cusps. Recall that a number $\mathfrak{a}$ in $\mathbb{R} \cup \infty$ is called a cusp for Γ if the stability group of $\mathfrak{a}$ in Γ is generated by an infinite cyclic group generated by a parabolic motion. In particular,

$$\Gamma_{\mathfrak{a}} = \{\gamma \in \Gamma : \gamma\mathfrak{a} = \mathfrak{a}\} =< \gamma_{\mathfrak{a}} > .$$

Moreover, there exists matrix $\sigma_{\mathfrak{a}}$ such that

$$\sigma_{\mathfrak{a}}\infty = \mathfrak{a}, \sigma_{\mathfrak{a}}^{-1}\gamma_{\mathfrak{a}}\sigma_{\mathfrak{a}} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}. \tag{7.2.1}$$

We shall call $\sigma_{\mathfrak{a}}$ scaling matrix for $\mathfrak{a}$. Notice that if $\mathfrak{a} = \infty$ then corresponding scaling matrix is the identity matrix. We will also use the following result [8, Chapter 2]:

Result 7.2.1. *Given any cusp $\mathfrak{a}$ of Γ, let $c_{\mathfrak{a}} = min\{|c| > 0 : \begin{pmatrix} * & * \\ c & * \end{pmatrix} \in \sigma_{\mathfrak{a}}^{-1}\Gamma\sigma_{\mathfrak{a}}\}$. Then $c_{\mathfrak{a}}$ is positive if Γ is a Fuchsian group of first kind.*

We use the previous result to get:

Claim 7.2.1. *Given any cusp* $\mathfrak{a}$ *of* Γ *and any* $z = x + iy$ *in* $\mathbb{H}$ *such that* $y < 1$ *then, we have a constant* $d_{\mathfrak{a}}$ *depending only on* Γ *and* $\mathfrak{a}$ *such that* $Im(\sigma_{\mathfrak{a}}^{-1}\gamma z) \ll \frac{d_{\mathfrak{a}}}{y}$ *for any* $\gamma \in \Gamma$.

Proof. Let us prove the claim for the case when $\mathfrak{a} = \infty$. For any other cusp, the proof will follow very similarly. Let $\gamma = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ be any element of Γ then we have

$$\mathrm{Im}(\gamma z) = \frac{y}{|cz + d|^2}.$$

But

$$|cz + d| > |c|y \geq c_\infty y$$

(using result (7.2.1)). Hence we get the result. □

Let us assume we have a set $\{\gamma_i\}$ of generators of Γ. Let $\mathfrak{a}_0 = \infty, \mathfrak{a}, \mathfrak{a}_1, ..., \mathfrak{a}_k$ be the set of all the inequivalent cusps. Without loss of generality, let us assume that all the cusps $\mathfrak{a}_j$'s lie in the interval $0 \leq x \leq 1$. Let us now define the regions $\mathbb{D}_\infty, \mathbb{D}_1, ..., \mathbb{D}_k$ by

$$\mathbb{D}_\infty^t = \{z : 0 \leq x < 1, t \leq y\}, \mathbb{D}_j = \sigma_{\mathfrak{a}_j}\mathbb{D}_\infty^1 \, , j = 1, ..., k. \tag{7.2.2}$$

Let

$$y_0 = \inf_{0 \leq x \leq 1} \inf_j \{\sigma_{\mathfrak{a}_j}(x + i)\},$$

$$a = \inf\{y_0, 1\}.$$

Notice, using claim 7.2.1, we have that $a \neq 0$. Let

$$\mathbb{D}_0 = \mathbb{D}_\infty^a$$

and

$$F = \bigcup_{j=0,\ldots,k} \mathbb{D}_j.$$

Notice that F is an approximate fundamental domain (a type of Siegel set) and for every point x in $\mathbb{H}$, there exists a $\gamma \in \Gamma$ such that γx is in F. In particular, when $\Gamma = \mathrm{SL}(2,\mathbb{Z})$, $F = \mathbb{D}^1_\infty$.

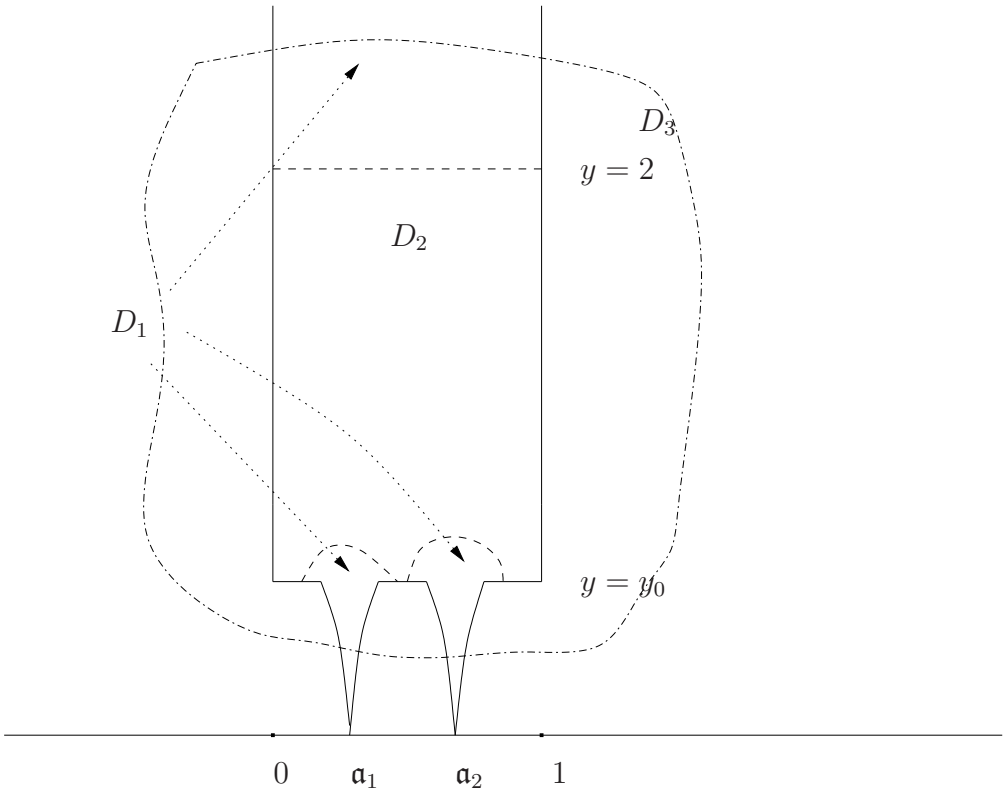

Figure 7.1: *Example of an approximate fundamental domain. In this example,* Γ *has three inequivalent cusps* $\mathfrak{a}_0 = \infty$, $\mathfrak{a}_1$ *and* $\mathfrak{a}_2$. *The corresponding approximate fundamental domain is enclosed by bold lines.* D_1 *and* D_2 *correspond to the cuspidal regions and the compact regions respectively.* D_3 *corresponds to the set which consists of points with hyperbolic distance at most* 3 *from* D_2.

Given any point $z_0 = x + iy$ such that $0 \leq x \leq 1$ we find γ in Γ such that $\gamma z_0 \in F$. We will call this as the 'reduction' of z to F. We will use the following proposition:

Proposition 7.2.2. *Given a finite set of generators* $\{\gamma_i\}$ *of* Γ *and any compact subset* K *of* $\mathbb{H}$*, there exists a constant* C_K *and a subset* $C(K) = \{g_1, ..., g_{C_K}\}$ *of* Γ *such that given any point* x *in* K*,there exists a* $g \in C(K)$ *such that* $gx \in F$*.*

Proof. Let us fix a fundamental domain $\mathfrak{F}$ contained in F. We know that $\mathbb{H}$ is a equal to $\bigcup_{\gamma\in\Gamma} \gamma\mathfrak{F}$. The proof follows easily from the observation that given any compact set K, there are only finitely many Γ-translates of $\mathfrak{F}$ that interset K. □

7.2.1 Reduction Algorithm

Let us first divide F in cuspidal part and compact part, the following way:

Let

$$D_1 = \cup_{j=0,...,k} \sigma_{\mathfrak{a}_j} \mathbb{D}^2_\infty.$$

Let

$$D_2 = F \cap D_1^c.$$

Here D_1^c correspond to the compliment of D_1. Let

$$D_3 = \{z \in \mathbb{H} | d(z, D_2) \leq 3\}$$

Here $d(\ ,\ )$ denotes the hyperbolic distance on $\mathbb{H}$. As D_2 is compact, the distance of a point from D_2 makes sense. Let $C(D_3)$ be the set of 'reductors' of D_3 as in proposition 7.2.2.

Notice that if $y > y_0$ then we are done. So without loss of generality let us assume $y < y_0$. Let $\gamma = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$, $z = \gamma(x + it)$.

Input: γ_i for $i = 1, ..., k$ generators of Γ. $\mathfrak{a}_0, \mathfrak{a}_1, ..., \mathfrak{a}_k$ be the corresponding set

of inequivalent cusps. Let $\mathfrak{a}_0 = \infty$. Any point $z_0 = x + iy$ such that $0 \leq x \leq 1$.

Output: γ in Γ such that $\gamma(z_0) \in F$. Initial condition:

1. [initialize] $t = 1$, $z \leftarrow x + it$, $\begin{pmatrix} a & b \\ c & d \end{pmatrix} \leftarrow \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$.

2. [initialize j] set $j \leftarrow 0$.

3. [check if we checked all the cusps] if $j = k + 1$ then go to step (8).

4. [Check if in jth cusp] If $\mathrm{Im}(\sigma_{\mathfrak{a}_j}^{-1} \begin{pmatrix} a & b \\ c & d \end{pmatrix} z) > 2$ then go to step (5). Otherwise set $j \leftarrow j + 1$ and go to step (3).

5. [When in cusp] $\begin{pmatrix} p & q \\ r & s \end{pmatrix} \leftarrow \sigma_{\mathfrak{a}_j}^{-1} \begin{pmatrix} a & b \\ c & d \end{pmatrix}$, $t_1 \leftarrow \frac{1-\sqrt{1-4r^2(rx+s)^2}}{2r^2}$. (Here $t_1 < t$ is chosen such that , $\mathrm{Im}(\begin{pmatrix} p & q \\ r & s \end{pmatrix} (x + it_1)) = 1$).

6. [check if you have already passed z_0] if $t_1 < y$ then
$n \leftarrow \lfloor \mathrm{Re}(\begin{pmatrix} p & q \\ r & s \end{pmatrix} (x + iy)) \rfloor - 1$,
$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \leftarrow \sigma_{\mathfrak{a}_j} \begin{pmatrix} 1 & -n \\ 0 & 1 \end{pmatrix} \sigma_{\mathfrak{a}_j}^{-1} \begin{pmatrix} a & b \\ c & d \end{pmatrix}$, output $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ and terminate algorithm. Otherwise go to step (7).

7. $n \leftarrow \lfloor \mathrm{Re}(\begin{pmatrix} p & q \\ r & s \end{pmatrix} (x + it_1)) \rfloor - 1$,
$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \leftarrow \sigma_{\mathfrak{a}_j} \begin{pmatrix} 1 & -n \\ 0 & 1 \end{pmatrix} \sigma_{\mathfrak{a}_j}^{-1} \begin{pmatrix} a & b \\ c & d \end{pmatrix}$, $z \leftarrow \begin{pmatrix} a & b \\ c & d \end{pmatrix} z$. Go to step (2).

8. [compact case] $t_1 \leftarrow \frac{t}{e}$.

9. if $t_1 > y$ then $t = t_1$. Then find $\begin{pmatrix} p & q \\ r & s \end{pmatrix}$ in $C(D_3)$ such that $\begin{pmatrix} p & q \\ r & s \end{pmatrix} \begin{pmatrix} a & b \\ c & d \end{pmatrix} z$ lies in F. Set

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \leftarrow \begin{pmatrix} p & q \\ r & s \end{pmatrix} \begin{pmatrix} a & b \\ c & d \end{pmatrix},$$

$z \leftarrow \begin{pmatrix} a & b \\ c & d \end{pmatrix} z$. Go to step (2). Otherwise go to step (10).

10. $t \leftarrow y$. Then find $\begin{pmatrix} p & q \\ r & s \end{pmatrix}$ in $C(D_3)$ such that $\begin{pmatrix} p & q \\ r & s \end{pmatrix} \begin{pmatrix} a & b \\ c & d \end{pmatrix} z$ lies in F. Put

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \leftarrow \begin{pmatrix} p & q \\ r & s \end{pmatrix} \begin{pmatrix} a & b \\ c & d \end{pmatrix}.$$

Output $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$. End algorithm.

We can see that at i^{th} stage of the algorithm, $|t_i| \ll (\frac{2}{e})^i$. Using proposition 7.2.2, we can see that every step takes $O(1)$ operations. Hence we get that the total complexity for the running time of algorithm is $O(|\log y|)$.

7.2.2 Reducing in $\mathrm{SL}(2,\mathbb{R})$

Notice that the above algorithm works for reducing a point of $\mathbb{H}$ to F. For the $\mathrm{SL}(2,\mathbb{R})$ case, let us define an approximate fundamental domain F' for $\Gamma\backslash\mathrm{SL}(2,\mathbb{R})$ by

$$F' = \{n(t)a(y)K(\theta) \in \mathrm{SL}(2,\mathbb{R}) : n(t)a(y)i \in F\}. \tag{7.2.3}$$

Given any g in $\mathrm{SL}(2,\mathbb{R})$, we can first reduce the point $z = gi$ to F, using the previous algorithm. Let $a \in \Gamma$ be such that az is in F. then ag will denote the reduction of g to F'.

7.3 Sorting algorithm

Let F' be an approximate fundamental domain in (7.2.3). Let $x_1, ..., x_N$ be points of F'. Let $x_j = n(t_j)a(y_j)K(\theta_j)$ be the Iwasawa decomposition for each x_j. This implies that $0 \leq t_j \leq 1$ and $-\pi < \theta_j \leq \pi$ for every j. Further let's assume that $|y_j| \ll \log T$. Let $\delta < 1/60$ be any fixed constant.

In this section, we will give an algorithm to sort the points x_j's in the sets $S_1, ..., S_L$, such that $x, y \in S_j$ implies that $x^{-1}y = n(t)a(y)K(\theta)$ (where $-\delta \leq t, y, \theta \leq \delta$), using $O(N + \log T\delta^{-3})$ operations. Moreover, $L \ll \log T\delta^{-3}$. The constants in O and $\ll$ are independent of choice of δ, N and T.

Let us define the sets F'_j as

$$F'_j = \{n(t)a(y)K(\theta) \in \mathrm{SL}(2,\mathbb{R}) : n(t)a(y)i \in \mathbb{D}_j\}. \tag{7.3.1}$$

Clearly

$$F' = \bigcup F'_j$$

Let us write the set $\{x_1, ..., x_N\}$ as $\cup P_l$ where P_l's are disjoint sets such that each P_l is contained in F_l'.

We will use the following lemma here.

Lemma 7.3.1. *Let x be any element of F_0. Let $x = n(t_0)a(y_0)K(\theta_0)$ be the Iwasawa decomposition for x where $0 \leq t_0 \leq 1, 0 \leq y_0 \leq \infty, -\pi < \theta_0 \leq \pi$. Given any $\delta < \frac{1}{60}$, let*

$$F_\delta^x = \{n(t)a(y)K(\theta) : (t, y, \theta) \in [t_0 - \delta, t_0 + \delta] \times [y_0 - \delta, y_0 + \delta] \times [\theta_0 - \delta, \theta_0 + \delta]\}.$$

Then for any A, B in F_x^δ, we have $A^{-1}B \in U_{30\delta}$. Recall U_δ is defined in (2.2.1).

Proof. Given any A, B in F_δ^x, there exist $(t_1, y_1, \theta_1), (t_2, y_2, \theta_2)$ in $[t_0 - \delta, t_0 + \delta] \times [y_0 - \delta, y_0 + \delta] \times [\theta_0 - \delta, \theta_0 + \delta]$ such that $A = n(t_1)a(y_1)K(\theta_1)$, $B = n(t_2)a(y_2)K(\theta_2)$. This implies that

$$A^{-1}B = k(-\theta_1)a(-y_1)n(-t_1)n(t_2)a(y_2)K(\theta_2).$$

It is easy to see that $||A^{-1}B - I||_\infty \leq 5\delta$. Hence using result (8.3.1), we get the result if $5\delta < \frac{1}{12}$. □

Given integers n_1, n_2, n_3 let us define the sets V_{n_1,n_2,n_3} by

$$V_{n_1,n_2,n_3} = \{n(t)a(y)K(\theta) : \lfloor 30t/\delta \rfloor = n_1, \lfloor 30y/\delta \rfloor = n_2, \lfloor 30\theta/\delta \rfloor = n_3\}$$

Let us define the sets $S^j_{n_1,n_2,n_3}$ by

$$S^j_{n_1,n_2,n_3} = \{A \in P_j : \sigma_{\mathfrak{a}_j}^{-1} A \in V_{n_1,n_2,n_3}\}.$$

Lemma 7.2.1 along with the condition that $|y_j| \ll \log T$ for all j, gives us that the total number of sets needed to sort the points in P_j are $O(\log T \delta^{-3})$. Here the constant independent of δ.

This gives us the required algorithm. The 'reduction and sorting' algorithms used in the proof of 1.1, 5.1 and 1.2 follow from the algorithms given in this section.

7.4 Computational issues

7.4.1 Machine error

Throughout the algorithm we have assumed that we work with a real number model of computation. We have therefore assumed that the real numbers are stored completely up to zero error and that the arithmetic operations are error free. In reality as one can not fully specify a real number with finite amount of data, in practice, the real numbers are stored as floating point number.

Let us assume that each number is stored by a floating point number of $O(\delta \log T)$ bits[1]. We will assume that there exists a constant a auch that any arithmetic operation (addition/ multiplication/ subtraction/ division /log /exponential) takes $O((\log T)^a)$ time. As our algorithm has exponential ($O(T^{7/8+\eta})$) complexity, the time involved in each operation is comparatively negligible and can be absorbed in the T^η term. Hence we will assume that each operation takes a unit time instead. Let us now compute the possible rounding off errors.

Let us give examples of the different ways errors can arise due to fixed floating point arithmetic. Suppose we store numbers as a string of 10 digits and an exponent

[1]This means a number is stored as a finite number string and an exponent. In our case exponent will be stored with $O(\log T)$ bits of data and the number string will have $O(\log T)$ bits of data.

of 2 digits, a positive or negative sign for exponent and a positive or negative sign for the number. Notice that the numbers 10^{10} and 1 can be represented exactly by this system but the number $10^{10}+1$ which is equal to $1.0000000001\times 10^{10}$ will also be saved as $1.000000000\times 10^{10}$. Hence the total error in this operation is 1, which is quite large. Similarly the number 1.0000000011 can only be stored as 1.00000001×10^{0}. Notice that this number is specified up to an error of $O(10^{-10})$, which is very small. But the product $10^{10}\times 1.0000000011$ differs from $10^{10}\times 1.00000001\times 10^{0}$ by .1, which is also a fairly large error. Hence one has to be careful about errors arising due to working with very small or very large numbers. in the following we will show that we can choose a suitable δ such that if we work with numbers specified by $O(\delta \log T)$ digits, the total floating point error in our algorithm stays small).

For every real number a, let a be stored as the floating point number $c(a)$, where $c(a)$ is specified by $O(\log T)$ amount of data. This implies that the total error $|a-c(a)| \ll |a|T^{-\delta}$. Hence notice that if we are dealing with numbers which are as large as T^{δ}, the total machine error will be significant. Similarly if we are adding two numbers a and b, the total error $|c(c(a)+c(b))-(a+b)| \leq (2+|a|+|b|)T^{-\delta}$ and the error for multiplication becomes $|c(c(a)c(b))-ab| \leq 2(1+|ab|)T^{-\delta}$. Similarly for division, we have $|c(\frac{c(a)}{c(b)})-\frac{a}{b}| \ll (1+|\frac{a}{b}|)T^{-\delta}$

Let us analyze the proof of theorem 5.1 for the possible rounding off errors. First let's notice that at any stage of the algorithm, we do not multiply/divide more than $r_1 = O(\frac{\gamma}{\eta})$ numbers together. Recall $\epsilon = \frac{1}{8}$. As $\tilde{f}$ is assumed to have bounded derivatives, it is clear that, except for the proof of proposition 4.3.1, there exists a constant $r = O(\max\{1+\eta, r_1\})$ such that all the numbers involved in our calculation are $O(T^r)$ and also notice that at any step other than the proof of

proposition 4.3.1, we do not divide by numbers which are smaller than T^{-r}. As the total number of operations needed for our algorithm is $O(T^{\frac{7}{8}})$ the total error contributed because of multiplication/division is $O(T^{\frac{7}{8}+2r^2}T^{-\delta})$. This implies that total error due to adding/subtracting these numbers is $O(T^{\frac{7}{8}+o(1)}T^{\frac{7}{8}+r^2}T^{-\delta})$. This implies that if we choose $\delta = \frac{7}{4} + r^2 + \gamma$ then we make sure that total machine error involved is $O(T^{-\gamma})$.

This takes care of all the calculation except using the equation (4.3.6) (or (4.3.7)), where we divide numbers $\approx \exp(-\pi T/2)$. By using the error bound for division, we get that we can work with numbers of $O(\delta \log T)$ bits to achieve the desired precision in the computation.

Similar analysis can be done for the proofs of theorems 1.1 and 1.2 as well.

7.4.2 Input for the algorithm

As mentioned before, specifying Γ and f, *a priori* need an infinite amount of data. Similarly, specifying a Dirichlet character χ_q need $O(q)$ amount of data. Here, we will specify what we mean by 'given' f, Γ and χ_q.

7.4.3 Input for χ_q

A priori, specifying χ_q might require $O(q)$ amount of data. We will use an efficient way of specifying χ_q, when q has large factors. As before, let us deal with the following 3 cases separately:

7.4.3.1 Case: $q = MN, (M, N) = 1$.

In this case, we know that $\chi_q = \chi_M \chi_N$. Our input in this case would be χ_M and χ_N. Notice that it takes $O(M + N)$ data to specify χ_M and χ_N. Once χ_M and

χ_N are stored, $\chi_q(n)$ can be computed for any $0 \leq n \leq q-1$, in $O(\log q)$ more operations.

7.4.3.2 Case: $q = MN, M|N$.

In this case, we know $\chi_q(j + kN) = \chi_q(j)\chi_q(1 + j^{-1}kN)$. As noticed before, there exists a such that $\chi_q(1 + kN) = e(ak/M)$. This implies that $\chi_q(1 + kN)$ is specified completely by a. Hence χ_q can be specified completely if $\chi_q(j)$ for $j = 0, ..., N-1$ and a are specified. Notice that $(\mathbb{Z}/N\mathbb{Z})^\times$ is an abelian group. Hence using Jordan decomposition, we can decompose it into a direct sum of cyclic subgroups $C_1, ..., C_L$, (say). From each C_j, choose a generator c_j. If d_j denotes the cardinality of C_j for each j, then we know that for each j, there exists a k_j such that $c_j^{d_j} = 1 + k_jN$. This implies that $(\chi_q(c_j))^{d_j} = \chi_q(1 + k_jN) = e(ak_j/M)$.

This implies that each $\chi_q(c_j)$ is a d_j^{th} root of $e(ak_j/M)$. Also notice that χ_q is completely specified if the integer a and $\chi_q(c_j)$ are specified for all j. Hence the input for χ_q in this case will be integers a, c_j, d_j, k_j and a d_j^{th} root of $e(ak_j/M)$ for $j = 0, ..., L$.

Notice that given N, the integers c_j, d_j, k_j can be computed in $O(N)$ operations. Given $\chi_q(c_j)$ and a, we can compute $\chi_q(n)$ for any integer $0 \leq n \leq q-1$, in $O(\log q)$ time. Hence total input size is $O(N)$.

7.4.3.3 Case of a general q

Let $q = M_1M_2N$ where $M_1|N$ and $(M_2, N) = 1$. Notice that $\chi_q = \chi_{M_2}\chi_{M_1N}$. χ_q is specified completely if χ_{M_2} and χ_{M_1N} are specified. But notice that as per previous case, χ_{M_1N} can be specified by $O(N)$ data. Hence χ_q can be specified by $O(M + N)$ size input.

7.4.4 Input for f

Throughout, we have assumed that given any point g of $F' \bigcap SL(2,\mathbb{R})$, and any β in $\mathbb{Z}^3_+$, we can compute $\partial^\beta \tilde{f}(g)$ up to an error of $O(T^{-\gamma})$ in "unit" time. In reality however, the time needed turns out to be polynomial in $|\beta|, \log T$ and the precision level γ. We quantify the actual result as:

Theorem 7.1. *Let F' be an "approximate fundamental domain" as defined by definition 7.2.3. Let g' be in F' and let $g' = x' + iy'$ be such that $|\log y'| \ll \log T$. Let γ be any given positive real. If we can compute (up to an error $O(T^{-\gamma})$) the n^{th} Fourier coefficient of a cusp form f of weight k at every cusp $\mathfrak{a}$ of F' in polynomial (in n) time, then given any $\beta \in \mathbb{Z}^3_+$, we can compute $\partial^\beta \tilde{f}(g')$ up to an error of $O(T^{-\gamma})$ using $O((\log T)^2 + |\beta|^5)$ operations. The constant involved in O is independent of $|\beta|$ and T.*

Proof. Let $g' = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$. Hence $g'i = \frac{ai+b}{ci+d} = x' + iy'$. We also know that $x' + iy'$ lies in the approximate fundamental domain F (because g' lies in F').

Let us first give an algorithm to compute the derivatives of f at $x' + iy'$. Let us assume that $x' + iy'$ lies in the cuspidal region corresponding to ∞. [2] This implies that $y' \gg 1$. First, let us assume that f is a Maass cusp form. Hence we have

$$\begin{aligned} &|f(x' + iy') - \sum_{n=1}^{N} \hat{f}(n) W'(n(x' + iy'))| \\ &= |f(x' + iy') - \sum_{n=1}^{N} \hat{f}(n) 2\sqrt{ny'} \cos(2\pi n x') K_{ir}(2\pi n y')| \qquad (7.4.1) \\ &\ll \sum_{n=N+1}^{\infty} n^{k/2+1/2} \exp(-\pi y' n/2) \ll e^{-\frac{\pi N y'}{4}}. \end{aligned}$$

[2] A similar method will work in case of a general cusp.

when $N \gg k \log k$. Here we use the exponential decay of Bessel functions quantified by:

$$K_\nu(y) = (\pi/2y)^{\frac{1}{2}} e^{-y}(1 + O(\frac{1+|\nu|^2}{y})). \tag{7.4.2}$$

This implies that in order to compute $f(x' + iy')$ up to an error of $O(T^{-\gamma})$ we need to compute the partial sum with $N \approx \log T$ terms. We deal with the higher derivatives of f, by using the recurrence relation

$$-2K'_\nu(z) = K_{\nu+1}(z) + K_{\nu-1}(z)$$

along with a similar technique to equation (7.4.1) to compute the derivatives of $\partial_x^{\beta_1}\partial_y^{\beta_2} f(x' + iy')$ using at most $O(\log T + |\beta_1 + \beta_2|^2)$ operations.

Notice that a similar method can be employed for a holomorphic cusp form of weight k to get the algorithm to compute $f(x' + iy')$ in the holomorphic case.

We now give an algorithm to compute the $\partial^\beta \tilde{f}(g')$. Let $g' = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$. Using definition 1.1.4, we have

$$\tilde{f}(g') = (ci + d)^{-k} f\left(\frac{ai + b}{ci + d}\right).$$

From the argument we just gave before, we can see that we can compute $\tilde{f}(g')$ in time $O(\log T)$. The constant in O is absolute. It is easy to see that $\tilde{f}(g'K(\theta)) = \exp(k\theta)\tilde{f}(g')$. This implies that for any non negative integer n, we have

$$\partial_3^n \tilde{f}(g') = k^n \tilde{f}(g').$$

Hence for any $\beta = (\beta_1, \beta_2, \beta_3) \in \mathbb{Z}_+^3$, we have

$$\partial^\beta \tilde{f}(g') = k^{\beta_3} \partial^{(\beta_1,\beta_2,0)} \tilde{f}(g'). \tag{7.4.3}$$

Let us define

$$C(j, l, t) = (ci + ct + d)^{-j} |ci + ct + d|^{-l}. \tag{7.4.4}$$

It is easy to see that for any $\beta \in \mathbb{Z}_+^3$,we can write each $\partial^\beta \tilde{f}(g')$ as a linear combination

$$\partial^\beta \tilde{f}(g') = k^{\beta_3} \sum_{\beta'=(\beta'_1,\beta'_2)\in\mathbb{Z}_+^2, |\beta'_1+\beta'_2|\le\beta_1+\beta_2} \sum_{l,n=0}^{3|\beta'|+k} a_{\beta,\beta',n,l} C(n, l, 0) \partial_x^{\beta'_2} \partial_y^{\beta'_3} f(x' + iy').$$

Moreover, all the corresponding $a_{\beta,\beta',n,l}$'s can be computed and stored using at most $O(|\beta|^5)$ arithmetic operations.

Let us prove the above statement using induction. Result is clearly true for $\beta = (0, 0, 0)$ where $a_{\beta,(0,0),0,0} = 1$. Suppose we can compute and store $a_{\beta,\beta',n,l}$ for all $\beta \in \mathbb{Z}_+^3$ such that $|\beta| \le N_0$ using $O(1 + N_0^5)$ arithmetic operations. We have to prove that for any β^1 such that $|\beta^1| = N_0$ and for all $0 \le l, n \le k + 3|\beta| + 3$, we can compute [3] and store $a_{\beta^1,\beta',n,l}$ for all $\beta' \in \mathbb{Z}_+^2$ such that $|\beta'| \le |\beta_1^1 + \beta_2^1|$, using $O(N_0^3)$ operations. Let us define $\beta = \beta^1 - (1, 0, 0)$ if $\beta_1^1 > 0$ and $\beta = \beta_1 - (0, 1, 0)$ if $\beta_1^1 = 0$ and $\beta_2^1 > 0$.

Notice

$$g'n(t) = \begin{pmatrix} a & at + b \\ c & ct + d \end{pmatrix}, \tag{7.4.5}$$

[3] We define $a_{\beta,\beta',n,l} = 0$ when n or $l > 3|\beta| + k$ and when n or $l < k$.

$$g'a(y) = \begin{pmatrix} e^{y/2}a & e^{-y/2}b \\ e^{y/2}c & e^{-y/2}d \end{pmatrix}. \tag{7.4.6}$$

Then

$$\begin{aligned}
&\partial^{\beta}\tilde{f}(g'n(t)) \\
&= k^{\beta_3} \sum_{\beta'=(\beta'_1,\beta'_2)\in\mathbb{Z}^2_+,|\beta'|\le\beta_1+\beta_2} \sum_{l,n=0}^{3|\beta|+k} a_{\beta,\beta',n,l}C(n,l,t)\partial_x^{\beta'_2}\partial_y^{\beta'_3} f(\frac{ai+at+b}{ci+ct+d}) \\
&= k^{\beta_3} \sum_{\beta'\in\mathbb{Z}^2_+,|\beta'|\le\beta_1+\beta_2} \sum_{l,n=0}^{3|\beta|+k} a_{\beta,\beta',n,l}C(n,l,t)\partial_x^{\beta'_2}\partial_y^{\beta'_3} f(\frac{ac+(at+b)(ct+d)+i}{|ci+ct+d|^2}).
\end{aligned} \tag{7.4.7}$$

Differentiate this function with respect to t and evaluate at 0. We can see that for $\beta^1 = \beta + (1,0,0)$ and for all n such that $k \le n \le |\beta_1+\beta_2|+k$, we can compute $a_{\beta^1,\beta',n}$ for all $|\beta'| \le |\beta_1+\beta_2|$, using at most $O(N_0^3)$ operations.

Similarly using (7.4.6), we can prove analogous result for $\partial^{\beta^1}\tilde{f}(g')$ when $\beta = (0,\beta_2,\beta_3)$ and $\beta^1 = \beta + (0,1,0)$.

Hence, if we compute and store $a_{\beta,\beta',n}$ for all $|\beta| \le N_0$, then computing and storing $a_{\beta^1,\beta',n,l}$ for each $|\beta^1| = N_0+1$ and for relevant β', n and l takes $O(N_0^3)$ more operations. Hence, the total operations needed to compute and store $a_{\beta,\beta',n,l}$ for all $|\beta^1| = N_0+1$ is at most $O((N_0+1)^4)$.

We already proved that computing $\partial_x^{\beta_1}\partial_y^{\beta_2} f$ takes $O(\log T + |\beta|^2)$ more steps. This implies that we can compute $\partial^{\beta^1}\tilde{f}(g')$ in $O(\log T + |\beta|^4)$ steps. The constant involved is independent of $|\beta|$.

□

In the theorem 7.1, we have assumed that Fourier coefficient of f can be com-

puted in polynomial time. The class of such cusp forms is large. For example let us recall that the Ramanujan delta function is defined by

$$\Delta(z) = (2\pi)^{12} e(z) \prod_{1}^{\infty} (1 - e(nz))^{24}.$$

Clearly, we can compute the n^{th} Fourier coefficient exactly (up tp a factor of π^{12}) in polynomial time using numbers of $O(\log n)$ bits.

In the proof of theorem 7.1, we have also assumed that we can compute the values of the Bessel functions up to an error of $O(T^{-\gamma})$ in unit time. We refer the readers to [2] for a detailed discussion about high precision computations of Fourier coefficients of Maass forms as well as dealing with the computations involving K-Bessel functions.

7.4.5 Input for Γ

We assume that Γ is given in terms of a finite set of generators and a set of all inequivalent cusps $\{\mathfrak{a}_j\}$ and corresponding scaling matrices $\{\sigma_{\mathfrak{a}_j}\}$. We can assume that generators of Γ are given to us by matrices with numbers of $O(\delta \log T)$ bits.

Chapter 8

Appendix

In this chapter, we prove various facts that we have repeatedly used in this thesis. We deal with some special functions and give a proof of the proposition 4.3.1 in section 8.1. We prove the bounds on the derivatives of the lift $\tilde{f}$ of a cusp form f in section 8.2. In sections 8.3 and 8.5, we will prove the exact statements quantifying the "special properties" of the horocycle flow and Hecke orbits. An algorithm to compute the Gauss sum $\tau(\chi_q)$ will be given in section 8.6.

8.1 Special functions

Let us recall the definitions and some properties of some special functions and prove proposition 4.3.1.

Definition 8.1.1. *Given, a, b, c any complex numbers and $|z| < 1$, we define the hypergeometric function $F(a, b, c, z)$ by*

$$F(a,b,c,z) = \frac{\Gamma(c)}{\Gamma(a)\Gamma(b)} \sum_{n=0}^{\infty} \frac{\Gamma(n+a)\Gamma(n+b)}{\Gamma(n+c)n!} z^n. \tag{8.1.1}$$

There is an analytic continuation of this function to the whole complex plane except a branch cut from 1 to infinity.

We will use the following well known property of the Hypergeometric functions (ref. [15, 9.132]).

Lemma 8.1.2.

$$F(a,b,c,z) = \frac{\Gamma(c)\Gamma(b-a)}{\Gamma(b)\Gamma(c-a)}(1-z)^{-a}F(a,c-b,1+a-b,\frac{1}{1-z}) + \frac{\Gamma(c)\Gamma(a-b)}{\Gamma(a)\Gamma(c-b)}(1-z)^{-b}F(b,c-a,1+b-a,\frac{1}{1-z}). \tag{8.1.2}$$

provided c is non zero and $a-b$ is not an integer.

We also need the following asymptotic expansion of the Gamma function ([15, 8.327]) given by

$$\Gamma(z) = z^{z-\frac{1}{2}}e^{-z}\sqrt{2\pi}(1+O(1/|z|)). \tag{8.1.3}$$

Hence for $z = \sigma + it$, σ fixed, we get,

$$\Gamma(\sigma+it) = e^{i\frac{\pi(\sigma-\frac{1}{2})}{2}}t^{\sigma-\frac{1}{2}}e^{-\frac{\pi t}{2}}(\frac{t}{e})^{it}\sqrt{2\pi}(1+O(t^{-1})). \tag{8.1.4}$$

Here, t is positive and the constant in O depends on σ. We use the following result about the Bessel functions ([15, 6.699]):

Result 8.1.1. *Let r be any fixed complex number such that $|Im(r)| < \frac{1}{2}$ and $T, T' >$*

0, *we have*

$$\int_0^\infty \cos(T_1 t) K_{ir}(t) t^{iT-1/2} dt = 2^{iT-3/2}\Gamma(\frac{ir+iT+\frac{1}{2}}{2})\Gamma(\frac{-ir+iT+\frac{1}{2}}{2}) \quad (8.1.5)$$
$$F(\frac{ir+iT+\frac{1}{2}}{2}, \frac{-ir+iT+\frac{1}{2}}{2}, \frac{1}{2}, -T_1^2).$$

Let us use (8.1.2) to get

$$F(\frac{ir+iT+\frac{1}{2}}{2}, \frac{-ir+iT+\frac{1}{2}}{2}, \frac{1}{2}, -T_1^2) \quad (8.1.6)$$
$$= \frac{\Gamma(\frac{1}{2})\Gamma(-ir)(1+T_1^2)^{\frac{-ir-iT-\frac{1}{2}}{2}}}{\Gamma(\frac{-ir+iT+\frac{1}{2}}{2})\Gamma(\frac{\frac{1}{2}-ir-iT}{2})} F(\frac{ir+iT+\frac{1}{2}}{2}, \frac{ir-iT+\frac{1}{2}}{2}, ir+1, \frac{1}{1+T_1^2})$$
$$+ \frac{\Gamma(\frac{1}{2})\Gamma(ir)(1+T_1^2)^{\frac{ir-iT-\frac{1}{2}}{2}}}{\Gamma(\frac{ir+iT+\frac{1}{2}}{2})\Gamma(\frac{ir-iT+\frac{1}{2}}{2})} F(\frac{iT-ir+\frac{1}{2}}{2}, \frac{-ir-iT+\frac{1}{2}}{2}, -ir+1, \frac{1}{1+T_1^2}).$$

Let us use the definition of hypergeometric function given in equation (8.1.1) to get

$$F(\frac{ir+iT+\frac{1}{2}}{2}, \frac{ir-iT+\frac{1}{2}}{2}, ir+1, \frac{1}{1+T_1^2}) = 1 \quad (8.1.7)$$
$$+ \sum_{n=2}^{\infty} \frac{(n-1+\frac{ir+iT+\frac{1}{2}}{2})...(\frac{ir+iT+\frac{1}{2}}{2})(n-1+\frac{ir-iT+\frac{1}{2}}{2})...(\frac{ir-iT+\frac{1}{2}}{2})}{(n-1+ir+1)...(ir+1)(n-1)...1} \left(\frac{1}{1+T_1^2}\right)^n.$$

Observe that for any complex number ν such that $|\mathrm{Re}(\nu)| < 1$, we have $n-1 < |n+\nu| \leq (2+2|\nu|)|n|$. Using these bounds to (8.1.7), we get that for $T > 1$, then for $T_1 > 16(1+|r|+|r|^2)T$, we have,

$$F(\frac{ir+iT+\frac{1}{2}}{2}, \frac{ir-iT+\frac{1}{2}}{2}, ir+1, \frac{1}{1+T_1^2}) = 1 + r_1. \quad (8.1.8)$$

Where $|r_1| \leq \frac{1}{7}$.

Similarly we get that when $T > 1$, then for $T_1 > 16(1 + |r| + |r|^2)T$, we have,

$$F(\frac{ir + iT + \frac{1}{2}}{2}, \frac{ir - iT + \frac{1}{2}}{2}, ir + 1, \frac{1}{1 + T_1^2}) = 1 + r_2. \tag{8.1.9}$$

Where $|r_2| \leq \frac{1}{7}$.

Using (8.1.6),(8.1.8) and (8.1.9) to (8.1.5) we get

$$\int_0^\infty \cos(T_1 t) K_{ir}(t) t^{iT - \frac{1}{2}} dt = D(T)(1 + r_1 + E(T_1, T)(1 + r_2)). \tag{8.1.10}$$

Here

$$D(T) = \frac{2^{iT-1}\Gamma(\frac{ir+iT+\frac{1}{2}}{2})\Gamma(\frac{-ir+iT+\frac{1}{2}}{2})\Gamma(\frac{1}{2})\Gamma(-ir)(1 + T_1^2)^{\frac{ir-iT-\frac{1}{2}}{2}}}{\Gamma(\frac{-ir+iT+\frac{1}{2}}{2})\Gamma(\frac{-ir-iT+\frac{1}{2}}{2})}. \tag{8.1.11}$$

and

$$E(T_1, T) = \frac{\Gamma(ir)\Gamma(\frac{-ir+iT+\frac{1}{2}}{2})\Gamma(\frac{-ir-iT+\frac{1}{2}}{2})(1 + T_1^2)^{ir}}{\Gamma(\frac{ir+iT+\frac{1}{2}}{2})\Gamma(\frac{ir-iT+\frac{1}{2}}{2})}. \tag{8.1.12}$$

Notice that for a fixed T and real r, the argument of $\frac{\Gamma(ir)\Gamma(\frac{-ir+iT+\frac{1}{2}}{2})\Gamma(\frac{-ir-iT+\frac{1}{2}}{2})}{\Gamma(-ir)\Gamma(\frac{ir+iT+\frac{1}{2}}{2})\Gamma(\frac{ir-iT+\frac{1}{2}}{2})}$ is fixed. On the other hand $(1+T_1^2)^{ir}$ is a rapidly oscillating function of T_1. Hence we can choose a suitable $C' > 16(1+|r|+|r|^2)$ such that $|1+r_1+E(C_1T, T)(1+r_2)| > \frac{1}{2}$. Similarly if ir were real, then using the asympotics of Γ function to (8.1.12), we can see that the behaviour of absolute value of $E(T_1, T)$ is dominated by $(1+T_1^2)^{ir}$, for $T_1 > T$. In other words, we can choose a suitable $C' > 16(1 + |r| + |r|^2)$ such that $|1 + r_1 + E(T_1, T)(1 + r_2)| > \frac{1}{2}$.

Notice that for a real r, C' needs to be chosen such that

$$C' > 16(1 + |r| + |r|^2)$$

and

$$|\arg((1 + C'T)^{ir}) - \arg(\frac{\Gamma(ir)\Gamma(\frac{-ir+iT+\frac{1}{2}}{2})\Gamma(\frac{-ir-iT+\frac{1}{2}}{2})}{\Gamma(\frac{ir+iT+\frac{1}{2}}{2})\Gamma(\frac{ir-iT+\frac{1}{2}}{2})})| \leq \pi/20$$

(roughly). Notice that the second term in the above equation is independent of C' and $(1 + C'T)^{ir}$ is highly oscillating function of C'. Hence after computing $\arg(\frac{\Gamma(ir)\Gamma(\frac{-ir+iT+\frac{1}{2}}{2})\Gamma(\frac{-ir-iT+\frac{1}{2}}{2})}{\Gamma(\frac{ir+iT+\frac{1}{2}}{2})\Gamma(\frac{ir-iT+\frac{1}{2}}{2})})$, in unit time we can compute C_1 satisfying the required property. Similarly we can deal with the case when r is purely imaginary.

Using the asymptotics of Γ function, we can see that $D \gg T^{-4}$. The constant only depends on r. We have thus proved proposition 4.3.1.

8.2 Real analytic functions on $\Gamma\backslash \mathrm{SL}(2, \mathbb{R})$

Throughout, let Γ be a lattice of $\mathrm{SL}(2, \mathbb{R})$. In this section we continue using the notations defined in section 2.2.

Let f be a (holomorphic or Maass) cusp form on $\Gamma\backslash\mathbb{H}$. We recall the definition of $\tilde{f}$ given in (1.1.4) again. $\tilde{f} : \Gamma\backslash\mathrm{SL}(2, \mathbb{R}) \to \mathbb{C}$ is defined as:

$$\tilde{f} : \Gamma \begin{pmatrix} a & b \\ c & d \end{pmatrix} \to (ci + d)^{-k} f\left(\frac{ai + b}{ci + d}\right).$$

Result 8.2.1. *There exists a compactly supported function h on $\Gamma\backslash\mathrm{SL}(2, \mathbb{R})$ such that $\tilde{f}(g) = c\tilde{f} \star h(g)$ for some non zero constant c depending only on h and λ.*

Proof. Let $k = 0$. Recall that f is an eigenfunction of the hyperbolic Lapla-

cian with eigenvalue $\lambda = \frac{1}{4} + r^2$. We know that $\tilde{f}$ belongs to a subspace V_λ of $L^2(\Gamma\backslash \mathrm{SL}(2,\mathbb{R}))$, which is isomorphic to a principal or complementary series representation under the right action of $\mathrm{SL}(2,\mathbb{R})$. Moreover, the Casimir operator Ω acts as a scalar λ on V_λ.

Let h be a compactly supported function on $\mathrm{SL}(2,\mathbb{R})$. We use the fact that the convolution by h is a compact operator on $\Gamma\backslash \mathrm{SL}(2,\mathbb{R})$ ([3, Chapter 2]) along with the fact that the convolution operator commutes with the Casimir operator on $L^2(\Gamma\backslash \mathrm{SL}(2,\mathbb{R}))$, to get that for any f' in V_λ,

$$\Omega(h \star f') = h \star \Omega(f') = \lambda h \star f'.$$

This implies that the subspace V_λ stays invariant under the convolution by h. Notice that $\tilde{f}(gK(\theta)) = e(k\theta)\tilde{f}(g)$ for all $g \in \mathrm{SL}(2,\mathbb{R})$. We use the fact that $\tilde{f}$ is unique (up to scalar) such function in V_λ. In other words, given any f' in V_λ such that for all $g \in \mathrm{SL}(2,\mathbb{R})$, $f'(gK(\theta)) = e(k\theta)f'(g)$ then $f' = c\tilde{f}$ for some scalar c.

Recall that

$$h \star \tilde{f}(g) = \int_{\mathrm{SL}(2,\mathbb{R})} \tilde{f}(z)h(z^{-1}g)dz = \int_{\mathrm{SL}(2,\mathbb{R})} h(z)\tilde{f}(z^{-1}g)dz.$$

It is clear from the definition of $h \star \tilde{f}$ that $h \star \tilde{f}(gK(\theta)) = e(k\theta)h \star \tilde{f}(g)$ for all $g \in \mathrm{SL}(2,\mathbb{R})$. This implies that given any compactly supported function h on $\mathrm{SL}(2,\mathbb{R})$, there exists a constant c_h such that $h \star \tilde{f} = c_h f$. We need to find a function h such that $c_h \neq 0$.

Let $\tilde{f}$ be non zero cusp form. Let us assume that $\tilde{f}(Id) \neq 0$ (the proof will be similar in other cases). Let us choose h such that for all $g \in \mathrm{SL}(2,\mathbb{R})$, $h(K(\theta)g) =$

$e(k\theta)h(g)$. Notice that

$$\begin{aligned} h \star \tilde{f}(Id) &= \int_{\mathrm{SL}(2,\mathbb{R})} h(z^{-1})\tilde{f}(z)dz \\ &= \int_{\mathrm{SL}(2,\mathbb{R})} h(K(-\theta)a(-y)n(-t))\tilde{f}(n(t)a(y)K(\theta))dz \\ &= \int_{\mathrm{SL}(2,\mathbb{R})} e(-k\theta)h(a(-y)n(-t))e(k\theta)\tilde{f}(n(t)a(y))dz \\ &= \int_{\mathrm{SL}(2,\mathbb{R})} h(a(-y)n(-t))\tilde{f}(n(t)a(y))dz. \end{aligned} \tag{8.2.1}$$

Notice that $\tilde{f}(Id) \neq 0$ hence we can choose a suitable smooth positive real valued function h such that the right hand side integral of (8.2.1) is non zero.

The case when $k > 0$ can be solved analogously. Notice that using a $\mathrm{SL}(2,\mathbb{R})$ analogue of proposition 7.1.1, (using a grid of $O(T^{3\eta})$ equispaced points and using real analyticity of $\tilde{f}$) we can compute $h \star \tilde{f}(Id)$ up to an error of at most $O(T^{-\gamma})$ using $O(T^{3\eta})$ operations. □

We refer the readers to [8, Chapter 1] for explicit computation for the case of Maass forms. We now use the result 8.2.1 to bound the derivatives of $\tilde{f}$.

Claim 8.2.1. *For a cusp form f, there exists constant $R > 0$ such that for any $g \in \mathrm{SL}(2,\mathbb{R})$ and $\beta \in \mathbb{Z}^3_+$,*

$$|\partial^\beta \tilde{f}(g)| \ll R^{|\beta|}$$

The implied constant is independent of g or $|\beta|$. Here $|\beta| = \beta_1 + \beta_2 + \beta_3$.

Proof. Using the real analyticity of f, it can easily be shown that $\tilde{f}$ is a real analytic function on $\Gamma\backslash\mathrm{SL}(2,\mathbb{R})$.. We can see from the definition that $\tilde{f}$ is function on $\Gamma\backslash\mathrm{SL}(2,\mathbb{R})$. Using the exponential decay of f at cusps, we get that $\tilde{f}$ is a

bounded function. Using the result 8.2.1 and the definition 2.2.2, we have

$$\partial^{\beta}\tilde{f}(g) = c\tilde{f} \star \partial^{\beta}h(g).$$

As h is a test function, we have a constant R such that $|\partial^{\beta}h(g)| \ll R^{|\beta|}$. This together with the fact that $\tilde{f}$ is bounded, we get the result. □

8.3 Lemmas 5.1.3, 5.2.3 and 6.1.1 and computing

$$c_{x,y,\beta,l}$$

Propositions 5.1.2 and 5.2.2 are analogous. Similarly lemmas 5.1.3,5.2.3 and 6.1.1 are analogous. Our main goal is to prove them in this section.

Let x, y be any two arbitrary points in $\Gamma\backslash \mathrm{SL}(2,\mathbb{R})$ and let $y = xA$ where $A = \begin{pmatrix} p & q \\ r & s \end{pmatrix}$. Let U_{ϵ} be the ϵ neighborhood ball around identity defined by definition (2.2.1). Let $A \in U_{\epsilon}$. Let $\kappa(t) = \begin{pmatrix} \sqrt{\frac{t}{T}} & -\sqrt{Tt} \\ 0 & \sqrt{\frac{t}{T}}^{-1} \end{pmatrix}, n(t) = \begin{pmatrix} 1 & t \\ 0 & 1 \end{pmatrix}$.

We will use the following result to prove the propositions 5.1.2 and 5.2.2:

Result 8.3.1. *Given any* $0 < \epsilon < \frac{1}{12}$*, and any* A *in* $\mathrm{SL}(2,\mathbb{R})$ *such that* $||A-I||_{\infty} < \epsilon$*, then* $A \in U_{6\epsilon}$

Proof. We use the Iwasawa decomposition to write $A = n(t)a(y)K(\theta)$. We have to prove that $|t|, |y|, |\theta| \leq 6\epsilon$. The Iwasawa decomposition implies that

$$A = \begin{pmatrix} e^{y/2}\cos\theta - te^{-y/2}\sin\theta & e^{y/2}\sin\theta + te^{-y/2}\cos\theta \\ -e^{-y/2}\sin\theta & e^{-y/2}\cos\theta \end{pmatrix}.$$

From the given condition, we have $|\tan\theta| \leq 2\epsilon$. This implies that $|\theta| \leq 2\epsilon$. We also have $1+\epsilon \geq e^{-y/2}\cos\theta \geq 1-\epsilon$. The bound on θ implies that for $0 \leq \epsilon \leq 1/12$, $|\cos\theta| \geq \sqrt{1-4\epsilon^2}$. This implies that

$$1-\epsilon \leq e^{-y/2} \leq \frac{1+\epsilon}{\sqrt{1-4\epsilon^2}} \leq \frac{1+\epsilon}{1-\epsilon}.$$

This implies that $-y/2 \leq \log(1+\epsilon) - \log(1-\epsilon) \leq 3\epsilon$ and using the other side of the inequality, we have $-y \geq -4\epsilon$. Hence we have $|y| \leq 6\epsilon$.

Similarly, using the previous bounds and the bound $|e^{y/2}\sin\theta + te^{-y/2}| \leq \epsilon$ we get

$$|t| \leq (\epsilon + |e^{-y/2}\sin\theta|)e^{y/2} \leq 3\epsilon e^{6\epsilon}.$$

We next use the naive bound $e^{6\epsilon} \leq e^{1/2}$ for $|\epsilon| \leq 1/12$, to get

$$|t| \leq 3e^{1/2}\epsilon.$$

Hence for $0 \leq \epsilon \leq 1/12$, we have

$$|t| \leq 5\epsilon.$$

□

Proof. Proof of proposition 5.1.2

The proposition is equivalent to proving that $(n(-t)An(t)) \in U_{cT^{-\eta}}$ for some c. Using result (8.3.1), it is enough to prove that $||n(-t)An(t) - I||_\infty \ll T^{-\eta}$ for all $0 < t < T^\epsilon$ where $||X||_\infty$ denotes the usual infinity norm. But if we carry out

the calculation, we see that

$$n(-t)An(t) = \begin{pmatrix} p - tr & (p-s)t - t^2 r + q \\ r & tr + s \end{pmatrix}. \tag{8.3.1}$$

Using (8.3.1) it is easy to see that if $||A-I||_\infty < T^{-2\epsilon-\eta}$ then $||n(-t)An(t)-I||_\infty \ll T^{-\eta}$ for all $0 < t < T^\epsilon$.

□

If we look at the the equation (8.3.1) and compute the N, A, K coordinates for $n(-t)An(t)$, we get the following immediate corollary

Corollary 8.3.1. *Using the same notation as in the proof of proposition 5.1.2, given any β in $Z_+^2 \times 0$ and any $k \in \mathbb{Z}_+$, we have constants $c_{x,y,\beta,k}$ such that*

$$|c_{x,y,\beta,k}| \ll 2^k k^3 \beta! T^{-k(\epsilon+\frac{\eta}{2})}$$

and

$$(yn(t) - xn(t))^\beta = \sum_{k \in \mathbb{Z}_+} c_{x,y,\beta,k} t^k.$$

This also implies that given any $\gamma, \eta > 0$, we have a constant $M = O(\gamma/\eta)$ such that for all $0 \le t \le T^\epsilon$,

$$\frac{(yn(t) - xn(t))^\beta}{\beta!} = \frac{1}{\beta!} \sum_{k=0}^{M} c_{x,y,\beta,k} t^k + O(T^{-\gamma}). \tag{8.3.2}$$

The O constant is absolute.

Proof. We will use the following result here.

Result 8.3.2. *Given any* $\begin{pmatrix} p & q \\ r & s \end{pmatrix}$ *in* $\mathrm{SL}(2,\mathbb{R})$*, we have*

$$\begin{pmatrix} p & q \\ r & s \end{pmatrix} = n(\frac{pr+qs}{r^2+s^2})a(-\log(r^2+s^2))K(\tan^{-1}(-\frac{r}{s})). \tag{8.3.3}$$

Using result (8.3.2) and equation (8.3.1), we get

$$\begin{aligned} n(-t)An(t) =& n(\frac{r(p-tr)+(tr+s)((p-s)t-t^2r+q)}{r^2+(tr+s)^2}) \\ & a(-\log(r^2+(tr+s)^2))K(\tan^{-1}(-\frac{r}{tr+s})). \end{aligned} \tag{8.3.4}$$

Let $h_{1,\beta_1,x,y}(t) = (\frac{r(p-tr)+(tr+s)((p-s)t-t^2r+q)}{r^2+(tr+s)^2})^{\beta_1}$. As $||A-I||_\infty \ll T^{-(2\epsilon+\eta)}$, we have that $|r|, |p-s| \ll T^{-(2\epsilon+\eta)}$. Using these bounds, it is clear that

$$h^{(n)}_{1,\beta_1,x,y}(0) \ll \beta_1! n! 2^n T^{-(\epsilon+\eta/2)n}. \tag{8.3.5}$$

Similar result will hold for $h_{2,\beta_2,x,y} = (-\log(r^2+(tr+s)^2))^{\beta_2}$ and $h_{3,\beta_3,x,y} = (\tan^{-1}(-\frac{r}{tr+s}))^{\beta_3}$. Using the Taylor series for $h_{1,\beta_2,x,y}$ and $h_{2,\beta_2,x,y}$, we get that

$$c_{x,y,\beta,l} = \sum_{\beta' \in \mathbb{Z}^3_+, |\beta'|=l} \frac{h^{(\beta'_1)}_{1,\beta_1,x,y}(0) h^{(\beta'_2)}_{2,\beta_2,x,y}(0) h^{(\beta'_3)}_{3,\beta_3,x,y}(0)}{\beta'!}. \tag{8.3.6}$$

This gives us an algorithm to compute $c_{x,y,\beta,k}$. Using bound (8.3.5), we can see that

$$c_{x,y,\beta,l} \ll 2^l l^3 \beta! T^{-(\epsilon+\eta/2)n}.$$

Using this bound for $c_{x,y,\beta,l}$ we can prove that the radius of convergence for the

power series $\sum_{k\in\mathbb{Z}_+} c_{x,y,\beta,k}t^k$ is at least $T^{\eta+\epsilon}/2$. Hence using the real analyticity of $h_{j,\beta_1,x,y}$ for $j = 1, 2, 3$, we get that for all $0 \leq t \leq T^{\epsilon+\eta}/2$, we have

$$(yn(t) - xn(t))^\beta = \sum_{l\in\mathbb{Z}_+} c_{x,y,\beta,l}t^l.$$

Using this equation and the bounds on $c_{x,y,\beta,l}$, we get (8.3.2). □

Proof. Proof of proposition 5.2.2

Recall

$$\omega(t) = \begin{pmatrix} (1+t)^{1/2} & T(-(1+t)^{1/2} + (1+t)^{-1/2}) \\ 0 & (1+t)^{-1/2} \end{pmatrix}$$

and $M - 1 = T^{-1+\epsilon}$. Notice that

$$\omega(t) = n(-Tt) + Err.$$

Where $||Err||_\infty \ll T^{-1+2\epsilon}$ for all $0 < t < T^{-1+\epsilon}$ and

$$\omega^{-1}(t) = n(Tt) + Err_2.$$

where $||Err_2||_\infty \ll T^{-1+2\epsilon}$ for all $0 < t < T^{-1+\epsilon}$. Using result 8.3.1, the proposition is equivalent to proving that

$$||\omega^{-1}(t)A\omega(t)||_\infty \ll T^{-\eta}$$

for all $0 < t < T^{-1+\epsilon}$.

Hence by using above estimates and using same technique as in the proof of proposition 5.1.2, we get the result. □

Again, if we compute the NAK co ordinates of $\omega^{-1}(t)A\omega(t)$, we get following corollary for any $\beta \in \mathbb{Z}_+^3$

Corollary 8.3.2. *Using the same notation as in the proof of proposition 5.2.2, given any β in Z_+^3, we have constants $e_{x,y,\beta,k}$ such that*

$$|e_{x,y,\beta,k}| \ll 2^k k^3 \beta! T^{-k(\epsilon+\frac{\eta}{2})}$$

and

$$(y\omega(t) - x\omega(t))^\beta = \sum_{k\in\mathbb{Z}_+} e_{x,y,\beta,k}(Tt)^k.$$

This also implies that given any $\gamma, \eta > 0$, we have a constant $M = O(\gamma/\eta)$ such that for all $0 < t < T^{-1+\epsilon}$,

$$\frac{(y\omega(t) - x\omega(t))^\beta}{\beta!} = \frac{1}{\beta!}\sum_{k=0}^{M} e_{x,y,\beta,k}(Tt)^k + O(T^{-\gamma}).$$

The constant involved is absolute.

Proof. proof of lemma 5.1.3

Let $y = xA$ as defined at the beginning of this section, we use power series expansion for f around points $xn(t)$ to compute the value of $\tilde{f}$ at points $yn(t)$. Using claim 8.2.1 and proposition 5.1.2, For all $0 < t < T^\epsilon$, we have a constant $d = O(\gamma/\eta)$ such that

$$\tilde{f}(yn(t)) = \sum_{|\beta|=0}^{d} \frac{\partial^\beta \tilde{f}(xn(t))}{\beta!}(yn(t) - x(n(t))^\beta + O(T^{-\gamma}),$$

the constant involved in O only depends on f. We use corollary (8.3.1) to get that

$$\tilde{f}(yn(t)) = \sum_{|\beta|=0}^{d} \partial^{\beta}\tilde{f}(xn(t))(\frac{1}{\beta!}\sum_{l=0}^{d} c_{x,y,\beta,l}t^{l} + O(T^{-\gamma})) + O(T^{-\gamma}).$$

Hence we have constants $c_{x,y,\beta,l}$ such that

$$\tilde{f}(yn(t)) = \sum_{|\beta|<d,\beta\in\mathbb{Z}_{+}^{3}} \sum_{l=0}^{d} c_{x,y,\beta,l}t^{l}\frac{\partial^{\beta}\tilde{f}(xn(t))}{\beta!} + O(d^{4}T^{-\gamma}).$$

Here the O constant only depends on f. Now integrating, we get the result. □

Proof. Proof of lemma 5.2.3

By using exactly the same proof as in the proof of lemma 5.1.3 we get that we have constants $e_{x,y,\beta,l}$ and $d' = O(\gamma/\eta)$ such that

$$\tilde{f}(y\omega(t)) = \sum_{|\beta|<d',\beta\in\mathbb{Z}_{+}^{3}} \sum_{l=0}^{d'} e_{x,y,\beta,l}(Tt)^{l}\partial^{\beta}\tilde{f}(x\omega(t)) + O((d')^{4}T^{-\gamma}).$$

Integrating on both sides, we get the result. □

Proof. Proof of lemma 6.1.1:

Let $x, y \in K_i$ for some integer i and let r be a function on the set $T(M)$. This implies that $x^{-1}y \in U_{q^{-\eta}}$. This implies that

$$(A(M,m,k)x)^{-1}A(M,m,k)y = x^{-1}y \in U_{q^{-\eta}}$$

for all $(m,k) \in T(M)$. Let

$$x^{-1}y = n(t_0)a(y_0)K(\theta_0).$$

Here $|\theta_0| \leq \pi$. As g is a real analytic function on $\Gamma\backslash \mathrm{SL}(2,\mathbb{R})$, we can use the power series expansion 2.2.2 for g at points $A(M,m,k)x$ to compute $g(A(M,m,k)y)$ to get

$$g(A(M,m,k)y) = \sum_{\beta=(\beta_1,\beta_2,\beta_3)\in\mathbb{Z}_+^3} \frac{\partial^\beta g((A(M,m,k)x)}{\beta!} t_0^{\beta_1} y_0^{\beta_2} \theta_0^{\beta_3}. \tag{8.3.7}$$

As $n(t_0)a(y_0)K(\theta_0) \in U_{q^{-\eta}}$, we get that there exists a constant $d = O(\gamma/\eta)$ such that

$$g(A(M,m,k)y) = \sum_{|\beta|\leq d} \frac{\partial^\beta g((A(M,m,k)x)}{\beta!} t_0^{\beta_1} y_0^{\beta_2} \theta_0^{\beta_3} + O(q^{-\gamma}). \tag{8.3.8}$$

Substituting in (6.1.7), we get

$$J(0,r,y) = \sum_{|\beta|=0}^{d} c_{\beta,x,y} J(\beta,r,x) + O(2M^2 q^{-\gamma}). \tag{8.3.9}$$

Here $c_{\beta,x,y} = t_0^{\beta_1} y_0^{\beta_2} \theta_0^{\beta_3}/\beta!$. Notice that each $c_{\beta,l,n}$ can be computed in unit time. $\square$

8.4 Computing $c_{x,y,\beta,l}$ and $e_{x,y,\beta,l}$

Recall that $d = O(\gamma/\eta)$ and we need to compute $c_{x,y,\beta,l}$ for all $|\beta|, l \leq d$. Recall that using (8.3.6), we have

$$c_{x,y,\beta,l} = \sum_{\beta'\in\mathbb{Z}_+^3, |\beta'|=l} \frac{h_{1,\beta_1,x,y}^{(\beta_1')}(0) h_{2,\beta_2,x,y}^{(\beta_2')}(0) h_{3,\beta_3,x,y}^{(\beta_3')}(0)}{\beta'!}.$$

Here

$$h_{1,\beta_1,x,y}(t) = (\frac{r(p-tr)+(tr+s)((p-s)t-t^2r+q)}{r^2+(tr+s)^2})^{\beta_1},$$

$$h_{2,\beta_2,x,y} = (-\frac{1}{2}\log(r^2+(tr+s)^2))^{\beta_2},$$

and

$$h_{3,\beta_3,x,y} = (\tan^{-1}(-\frac{r}{tr+s}))^{\beta_3}.$$

Notice that

$$h_{1,\beta_1,x,y}(t) = (r(p-tr)+(tr+s)((p-s)t-t^2r+q))^{\beta_1}(r^2+(tr+s)^2)^{-\beta_1}.$$

We can use the formal power series expansion for $(1+x)^\alpha$ given by $\sum_{n=0}^{\infty}\binom{\alpha}{n}x^n$ to compute $h^{(j)}_{1,\beta_1,x,y}(0)$ for all $j \ll d$, using $O(d^4)$ operations. Similarly it is easy to prove that we can compute $h^{(j)}_{l,\beta_1,x,y}(0)$ for all $l = 1,2,3$ and for all $j \leq d$ can be computed and stored in $O(d^4)$ time. After computing $\{h^{(j)}_{n,\beta_1,x,y}(0) : n = 1,2,3 \text{ and } j \leq d\}$, we can use (8.3.6), to compute $c_{\beta,x,y,l}$ for any $|\beta|, l \leq d$ using $O(d^2)$ more operations. We thus proved that each required all the $c_{x,y,\beta,l}$ can be computed using at most $O(d^6)$ operations.

A similar proof will go through for computing $e_{x,y,\beta,l}$.

8.5 Hecke orbits

Let $\Gamma = \mathrm{SL}(2,\mathbb{Z})$. Recall from subsection 2.3, for a positive integer L and a point x in $\mathrm{SL}(2,\mathbb{R})$, the L^{th} Hecke orbit of the point x is indexed by the set

$$T(L) = \{(m,k) : m|L, 0 \leq k < L/m\}.$$

The L^{th} Hecke orbit of x is explicitly given by the set of cosets

$$\{\Gamma A(L,m,k)x : (m,k) \in T(L)\}.$$

Here,

$$A(L,m,k) = \frac{1}{L^{1/2}} \begin{pmatrix} m & k \\ 0 & L/m \end{pmatrix}.$$

It is well known that the right action of any element a in $\mathrm{SL}(2,\mathbb{Z})$ permutes the cosets $\{\Gamma A(L,m,k), (m,k) \in T(L)\}$. This means that

$$\{\Gamma A(L,m,k), (m,k) \in T(L)\} = \{\Gamma A(L,m,k)a, (m,k) \in T(L)\}$$

for any $a \in \mathrm{SL}(2,\mathbb{Z})$. However, the right action by a permutes the Hecke orbit. This means that there exists a permutation (one to one and onto map) $\sigma_{\mathfrak{a}} : T(L) \to T(L)$ such that $\Gamma A(L,m,k)a = \Gamma A(L, \sigma_{\mathfrak{a}}(m,k))$ as the elemets of $\Gamma\backslash \mathrm{SL}(2,\mathbb{R})$.

Given any $\eta > 0$, it is known that the cardinality of $T(M)$ is at most $O(M^{1+\eta})$. [1] Hence *a priori* there are $M^{1+\eta}!$ possible permutations on $T(L)$. However, Let us prove the following lemma:

Lemma 8.5.1. *Let a_1 and a_2 are matrices in $\mathrm{SL}(2,\mathbb{Z})$ such that $a_1 = a_2 + Lb$, for some b in $M_2(\mathbb{Z})$ (2×2 matrices with integer entries). Then the corresponding permutations σ_{a_1} and σ_{a_2} are equal. In other words, for every $(m,k) \in T(L)$, $\sigma_{a_1}(m,k) = \sigma_{a_2}(m,k)$.*

Proof. Let (m,k) be a any element of $T(L)$. Let $\sigma_{a_1}(m,k) = (m_1,n_1)$. This implies that $\Gamma A(L,m,k)a_1 = \Gamma A(L,m_1,k_1)$. This implies that there exist a matrix

[1] We just use the fact that the number of divisors of M are at most $O(M^{\eta})$.

$c \in \mathrm{SL}(2,\mathbb{Z})$ such that

$$cA(L,m,k)a_1 = A(L,m_1,k_1). \tag{8.5.1}$$

The lemma is equivalent is proving that there exists a matrix d in $\mathrm{SL}(2,\mathbb{Z})$, such that

$$dA(L,m,k)a_2 = A(L,m_1,k_1) \tag{8.5.2}$$

(this would imply that $\sigma_{a_2}(m,k) = \sigma_{a_1}(m,k)$).

We use (8.5.1) to get

$$\begin{aligned}
&cA(L,m,k)a_2 \\
&= cA(L,m,k)(a_1 + Lb) \\
&= cA(L,m,k)a_1 + cA(L,m,k)Lb \\
&= A(L,m_1,k_1) + cA(L,m,k)Lb \\
&= A(L,m_1,k_1) + cA(L,m,k)LbA(L,m_1,k_1)^{-1}A(L,m_1,k_1) \\
&= (I + cA(L,m,k)LbA(L,m_1,k_1)^{-1})A(L,m_1,k_1) \\
&= (I + cL^{-\frac{1}{2}} \begin{pmatrix} m & k \\ 0 & L/m \end{pmatrix} LbL^{-\frac{1}{2}} \begin{pmatrix} L/m_1 & -k_1 \\ 0 & L/m_1 \end{pmatrix})A(L,m_1,k_1) \\
&= (I + c \begin{pmatrix} m & k \\ 0 & L/m \end{pmatrix} b \begin{pmatrix} L/m_1 & -k_1 \\ 0 & L/m_1 \end{pmatrix})A(L,m_1,k_1).
\end{aligned} \tag{8.5.3}$$

(8.5.3) implies that $d = c(I + c \begin{pmatrix} m & k \\ 0 & L/m \end{pmatrix} b \begin{pmatrix} L/m_1 & -k_1 \\ 0 & L/m_1 \end{pmatrix})^{-1}$ will be in $\mathrm{SL}(2,\mathbb{Z})$ and will satisfy condition in equation (8.5.2). (m,k) was any arbitrary

element of $T(L)$. This implies that $\sigma_{a_1} = \sigma_{a_2}$. $\square$

Lemma 8.5.1, implies that the number of possible permutations of the Hecke orbit $\{\Gamma A(L, m, k) | (m, k) \in T(L)\}$ due to right action of $\mathrm{SL}(2, \mathbb{Z})$ are at most $|SL(2, \mathbb{Z}/L\mathbb{Z})| \leq L^3$.

8.6 Computing $\tau(\chi_q)$

Recall the Gauss sum $\tau(\chi_q)$ defined in (4.4.2)

$$\tau(\chi_q) = \sum_{k=0}^{q-1} \chi_q(k) e(k/q).$$

Let $q = MN$. In this section we will give an algorithm to compute $\tau(\chi_q)$ up to any given error $O(q^{-\gamma})$ using a most $O(M^2 + N)$ operations. We deal with the case when $q = MN$, $(M.N) = 1$ and the case when $q = MN$, $M|N$ separately. Recall that the input for χ_q is defined in section 7.4.3.

8.6.1 Computing Gauss sum when $q = MN$

Let $q = MN$, where $(M, N) = 1$. We then have $\chi_q = \chi_M \chi_N$ where χ_M is a character modulo M and χ_N is a character modulo N. Notice

$$\tau(\chi_q) = \sum_{k=0}^{q-1} \chi_q(k) e(k/q).$$

We cut this into a sum of N arithmetic progressions of size M each as:

$$\begin{aligned}\tau(\chi_q) &= \sum_{j=0}^{N-1}\sum_{k=0}^{M-1}\chi_q(j+kN)e((j+kN)/q) \\ &= \sum_{j=0}^{N-1}\chi_q(j)e(j/q)S_{b_j}. \end{aligned} \tag{8.6.1}$$

Here given j, b_j is defined by $j \equiv b_j \mod M$, where $0 \leq b_j \leq M-1$ and for any integer $0 \leq l \leq M-1$, S_l is defined by:

$$S_l = \sum_{k=0}^{M-1}\chi_M(1+kl^{-1}N)e(k/M).$$

l^{-1} denotes the inverse of $l \mod M$. Notice that $\{S_l : 0 \leq l \leq M-1\}$ can be computed in $O(M^2)$ steps, since each S_l takes $O(M)$ time. After this calculation the computation of $\tau(\chi_q)$ takes further $O(N)$ steps. Hence we got an algorithm to compute $\tau(\chi_q)$ in $O(M^2+N)$ steps.

8.6.2 Computing Gauss sum when $q = MN$, $M|N$

Let $q = MN$ where, $M|N$. We start as in previous case and write

$$\begin{aligned}\tau(\chi_q) &= \sum_{j=0}^{N-1}\sum_{k=0}^{M-1}\chi(j+kN)e((j+kN)/q) \\ &= \sum_{j=0}^{N-1}\chi_q(j)\sum_{k=0}^{M-1}\chi_q(1+j^{-1}kN)e((j+kN)/q). \end{aligned} \tag{8.6.2}$$

The j^{-1} here denotes the multiplicative inverse [2] of j modulo M. Even though χ_q does not split here as in the previous case, notice that for any k, k',

$$\chi_q(1+kN)\chi_q(1+k'N) = \chi_q((1+kN)(1+kN)) = \chi_q(1+(k+k')N). \quad (8.6.3)$$

This implies that the function

$$\chi'_q(k) = \chi_q(1+kN) \quad (8.6.4)$$

defines a additive character on $\mathbb{Z}/M\mathbb{Z}$. This implies that there exists a positive integer a such that $\chi'_q(k) = e(ak/M)$. This applied to (8.6.2), we get that

$$\begin{aligned}\tau(\chi_q) &= \sum_{j=0}^{N-1}\sum_{k=0}^{M-1} \chi(j+kN)e((j+kN)/q) \\ &= \sum_{j=0}^{N-1} \chi_q(j) \sum_{k=0}^{M-1} e(aj^{-1}k/M)e((j+kN)/q) \\ &= \sum_{j=0}^{N-1} \chi_q(j)e(j/q)S_{b_j}. \end{aligned} \quad (8.6.5)$$

Here $j \equiv b_j \mod M$, j^{-1} denotes inverse of $j \mod M$ and

$$S_l \sum_{k=0}^{M-1} e(al^{-1}k/M + kN/q).$$

Notice that the inner sum in (8.6.5) depends only on b_j. We can compute $\{S_l : 0 \leq l \leq M-1\}$ in $O(M^2)$ steps. Once we compute these sums, we need $O(N)$ more steps to compute $\tau(\chi_q)$. hence we have an algorithm to compute $\tau(\chi_q)$ in

[2]Notice that $\chi_q(j) = 0$ if $(M, j) \neq 1$. Hence the sum is actually over $(j, N) = 1$, hence taking 'inverse' modulo M is justified.

$O(M^2 + N)$ steps.

8.6.3 Computing Gauss sum in general case

Let $q = M_1M_2N$ where $M_1|N$ and $(M_2, N) = 1$. Let $M = M_1M_2$. We have $\chi_q = \chi_{M_2}\chi_{M_1N}$. Notice $M_1|N$, hence let a be such that $\chi_{M_1N}(1+kN) = e(ak/M_1)$. We use (8.6.5) to get,

$$\begin{aligned}
\tau(\chi_q) &= \sum_{j=0}^{N-1}\sum_{k=0}^{M-1} \chi_q(j+kM)e((j+kM)/q) \\
&= \sum_{j=0}^{N-1}\sum_{l=0}^{M_2-1}\sum_{k=0}^{M_1-1} \chi_q(j+kM_2N+lN)e(j/q+k/M_1+l/M) \\
&= \sum_{j=0}^{N-1} e(j/q)\sum_{l=0}^{M_2-1}\sum_{k=0}^{M_2-1} \chi_{M_2}(j+lN)\chi_{M_1N}(j+(l+kM_2)N)e(k/M_1+l/M) \\
&= \sum_{j=0}^{N-1} \chi_{M_1N}(j)e(j/q)b(j). \qquad (8.6.6)
\end{aligned}$$

Where j^{-1} denotes the inverse of $j \mod M_1$. and

$$b(j) = \sum_{l=0}^{M_2-1}\sum_{k=0}^{M_1-1} \chi_{M_2}(j+lN)e(aj^{-1}(l+kM_2)/M_1)e(k/M_1+l/M).$$

Notice that $b(j) = b(j')$ if $j \equiv j' \mod M$. Hence there are only M distinct sums $b(j)$. We can compute $\{b(j) : 0 \leq b(j) \leq M-1\}$ in $O(M^2)$ time. Given χ as in section 7.4.3, we can compute the sum of (8.6.6) in $O(M^2 + N)$ time.

Hence, we thus have an algorithm to compute $\tau(\chi_q)$ in $O(M^2+N)$ time. Notice that the time required to compute $\tau(\chi_q)$ is much less than the time complexity in theorem 1.2.

Bibliography

[1] M.V. Berry, J.P. Keating, *A new asymptotic representation for ζ(1/2 + it) and quantum spectral determinants.* Proc. Roy. Soc. London Ser. A 437 (1992), no. 1899, 151173.

[2] Andrew Booker, Andreas Strmbergsson and Akshay Venkatesh, *Effective computation of Maass cusp forms*, IMRN vol. 2006, article ID 71281, 34 pages, 2006.

[3] A. Borel, *Automorphic forms on SL2(R)*,Cambridge University Press, 1997.

[4] Gabcke, Wolfgang (1979), *Neue Herleitung und Explizite Restabschtzung der Riemann-Siegel-Formel*, Georg-August-Universitt Gttingen, http://www.worldcat.org/oclc/60563505

[5] L. Greengard, J. Carrier and V. Rokhlin, *A Fast Adaptive Multipole Algorithm for Particle Simulations*,SIAM J. Sci. Stat. Comput. 9, 669 (1988).

[6] D. R. Heath-Brown, *private communication to A. Odlyzko.*

[7] Ghaith Ayesh Hiary, *Fast methods to compute the Riemann zeta function*, http://arxiv.org/pdf/0711.5005.

[8] Henryk Iwaniec, *Spectral Methods Of Automorphic Forms*, (graduate Studies In Mathematics, V. 53), 0821831607.

[9] J.C. Lagarias, A.M. Odlyzko , *On computing artin L-functions in the critical strip*, math.comp. 33(1979), no. 147,1081-1095.

[10] A.M. Odlyzko, *The 1020-th zero of the Riemann zeta function and 175 million of its neighbors.* Manuscript. www.dtc.umn.edu/odlyzko.

[11] A.M. Odlyzko and A. Schönhage, *Fast algorithms for multiple evaluations of the Riemann zeta function.* Trans. Amer. Math. Soc. 309 (1988), no. 2, 797809

[12] M. Rubinstein, *Computational methods and experiments in analytic number theory.* Recent perspectives in random matrix theory and number theory, 425506, London Math. Soc. Lecture Note Ser., 322, Cambridge Univ. Press, Cambridge, 2005.

[13] M. Rubinstein, *Evidence for a spectral interpretation of the zeros of L-functions.* Princeton Ph.D. thesis, June 1998.

[14] A. Schönhage, *Numerik analytischer Funktionen und Komplexitat. Jahresber. Deutsch.Math.- Verein.* 92 (1990), no. 1, 120.

[15] I. Solomonovich Gradshten, I.M. Ryzhik, A. Jeffrey, D. Zwillinger,*Table of integrals, series and products*,

[16] Joseph Frederick Traub, Arthur G. Werschulz,*Complexity and information*,Cambridge University Press, 1998.

[17] Joseph Frederick Traub, *A continuous model of computation*, Physics today, 1998, volume 5, 39-43.

[18] A. Venkatesh, *Sparse equidistribution problems, period bounds, and subconvexity*, 2005.

[19] P. Vishe, *Rapid computations for L-functions for modular forms*, in preparation.

[20] P. Vishe, *Rapid computations for twisted L- functions for modular forms*, in preparation.