Schur Polynomials and Crystal Graphs

Lucas Fagan
Advisor: Dr. Spencer Leslie

A senior thesis submitted to the Department of Mathematics
Duke University, Durham, NC

Spring 2020

Abstract

The Casselman-Shalika formula connects special values of $p$-adic Whittaker functions to characters of Schur polynomials in type A. We study the structure of the expansion of these Whittaker functions in order to derive new formulas for Schur polynomials. In particular, we give an algorithm for computing the necessary data in order to express the integrals that arise in the expansion. Furthermore, we completely compute all terms in a specific case for $GL_4$, which gives a new formula for Schur polynomials in four variables.
Contents

1 Introduction 3

2 Preliminaries and Notation 5
   2.1 Complex Reductive Groups ................................. 5
   2.2 Representations of Complex Groups ......................... 8
   2.3 p-adic Numbers and p-adic Groups ........................ 9
   2.4 Other Notation ........................................... 11

3 Spherical Whittaker Functions 11

4 Computing i-trails 16

5 Computing Integrals From i-trails 18
   5.1 Results .................................................. 19
   5.2 Computational Verification of $\lambda = 0$ Case ............ 24

6 Resonance Families 25

7 Future Directions 28

A Mathematica Code 29
1 Introduction

Schur polynomials are fundamental objects in representation theory and combinatorics, and are connected to lots of different areas of math. Most simply, they are a set of symmetric polynomials that form the basis for the space of all symmetric polynomials. In the context of this paper, Schur polynomials arise in the representation theory of general linear groups by giving characters of representations, which is important because characters determine representations up to isomorphism. This is explained in more detail in the following section. As fundamental as Schur polynomials are, they are still interesting to study.

The goal of this project is to produce new formulas for Schur polynomials. One way to express Schur polynomials is via a connection to the representation theory of \( p \)-adic groups, whereby certain integrals can be evaluated through Schur polynomials. The basis of this project is to reverse this relationship, and use \( p \)-adic integrals to produce new formulas for Schur polynomials.

In particular, the underlying result that allows for this connection is the famous Casselman-Shalika formula, which demonstrates that Schur Polynomials can be expressed in terms of spherical Whittaker functions on a \( p \)-adic group \([CS80]\). Spherical Whittaker functions are discussed in more detail in section 3.

The formula is given by

\[
W_{z_1, \ldots, z_n}(p^\lambda) = p^{-\langle \lambda, \rho \rangle} \prod_{i<j} \left(1 - p^{-1}(z_i z_j^{-1})\right) s_{\lambda}(z_1, \ldots, z_n),
\]

where \( s_{\lambda} \) is a Schur polynomial, \( z_i \) are nonzero complex parameters, \( p \) is a prime, and \( p^\lambda \) is some diagonal matrix. See section 2 for details. Using recent results from \([Les19, McN11]\), we can see a different way of computing the left-hand side, where we write it as a sum over lattice points in a certain convex polyhedral cone. This is given by

\[
p^{-\langle \lambda, \rho \rangle} \prod_{i<j} \left(1 - p^{-1}(z_i z_j^{-1})\right) s_{\lambda}(z_1, \ldots, z_n) = \sum_{\text{lattice points } m} I_\lambda(m),
\]

where the \( I_\lambda(m) \) terms in the sum are integrals over \( p \)-adic manifolds. The formula arises from thinking about the right-hand side of this equation in terms of the geometric theory of crystals. Crystal graphs give combinatorial (crystalized) models of representations of algebraic groups, and in the sum above, the lattice points can be thought of as vertices in a certain infinite crystal graph. This connection—from the special functions to the theory of geometric crystals—is one of the main recent results from the aforementioned papers. Putting this all together, this expansion can be used to give new formulas for Schur polynomials.

Problematically, the \( I_\lambda(m) \) are not explicit—they rely on a choosing some combinatorial data—and even this expression relies on some choice, which is essentially a choice of coordinates on these manifolds \([Les19, Thm 3.2]\). This combinatorial data then allows us to calculate objects called \( i \)-trails, which \([Les19]\) shows us can be used to calculate the terms in the sum above. In general,
\(i\)-trails encode paths of edges in a polytope that is living inside a lattice. They are defined for any connected Lie group, and they are quite complicated in general. However, [BZ01] tells us that they can be reduced to a combinatorial expression in the case of \(GL_n\), the group of \(n \times n\) invertible matrices, which is the context of this project.

However, even though \(i\)-trails can be reduced to a combinatorial expression, it is still nontrivial to compute the \(i\)-trails given a choice of data. The first part of this project is to produce an algorithm to calculate these \(i\)-trails given the choices. This is accomplished through Python.

Once the code takes the data and calculates the \(i\)-trails, the next step is to calculate integrals over \(p\)-adic manifolds that comprise the terms in the aforementioned sum. The values of these integrals is given by the formula

\[
I_\lambda(m) = \int_{C^\ast(m)} d(m) e^{2\pi i s(u)/p} du,
\]

where \(d(m)\) is some explicit complex number that we can compute based on the data \(m\), and \(s(u)\) is expressed as a sum of the \(i\)-trails, as explained below. This integral itself is easy to write down, but computing the \(i\)-trails is necessary for expressing the integral in a way that can be computed.

It is useful to start by computing these integrals for the low-rank cases, using the \(i\)-trails outputted by the Python algorithm. The case of \(GL_2\) is trivial, and \(GL_3\) is relatively simple as well, since there are only two choices, as mentioned above. This means that the resulting integrals are relatively simple and certain pathologies do not arise. In particular, the terms of the integrals work out so that they can be nicely computed in a uniform manner. However, once we reach the case of \(GL_4\), the integrals become interesting. This means that there are additional terms that do not appear in the previous cases, and these terms cause the integrals to exhibit strange behavior in some cases. This mirrors the types of behavior of the exponential sums arising in [BF15], which should be related to this work. Further, the space of choices is much larger in \(GL_4\) than in the previous case, as there are 16 possible choices. This additional behavior means that we can write

\[
\prod_{\substack{(i,j) \\ i < j}} (1 - p^{-1}(z_i z_j^{-1})) s_\lambda(z_1, \ldots, z_n) = \sum_{m \in \text{polytope}} I_\lambda(m) + \sum_{m \notin \text{polytope}} I_\lambda(m),
\]

where the polytope is essentially a finite lattice that is sitting inside the aforementioned convex polyhedral cone, which will be explained in more detail below. Splitting up the sum in this way is important because the first sum are the “expected” terms, while the latter sum is essentially “error terms.” These error terms are precisely the terms that are not appearing in the lower-rank cases, where there is the interesting behavior.

In this paper, we calculate the sum above for a specific choice of long word in \(GL_4\), and demonstrate that we can achieve systematic cancellation in the error terms. In particular, we establish the following theorem, where the details and notation will be made clear later in the paper:
Theorem 1.1. For $G_1(m)$ and $G_{res}(m)$ as defined in Section 5, we have

$$p^{-\langle \lambda, \rho \rangle} \prod_{\alpha \in \Phi^+} (1 - p^{-1}z^{\alpha}) s_\lambda(z)$$

$$= \sum_{m \in B(\lambda+\rho)} G_1(m)x_1^{k_1}x_2^{k_2}x_3^{k_3} + \sum_{RF(m) \lambda \text{-relevant}} \sum_{m \in RF(m)^\circ} G_{res}(m)x_1^{k_1}x_2^{k_2}x_3^{k_3}$$

where $x_i = z^{\alpha_i} = z_i(z_{i+1})^{-1}$.

This is the first case in $GL_n$ where the error terms are nontrivial, and while there are infinitely many error terms with nonzero contributions, our systematic cancellation reduces them to this latter finite sum, which we denote by calling them “relevant.” Furthermore, $G_1(m)$ and $G_{res}(m)$ are combinatorial functions in $p$ which are determined by where the vertex $\lambda$ sits in the polytope.

2 Preliminaries and Notation

2.1 Complex Reductive Groups

In this project, we work with groups over different fields, so the natural context for this is the concept a linear algebraic group, which is a group defined over a field by a system of polynomial equations, where multiplication and inverses are also given by polynomial equations.

More specifically, this project will use one example of such a group, the general linear group over a field $F$, denoted $GL_n(F)$, which is the set of $n \times n$ invertible matrices with elements in $F$, and thus the introduction will be about this particular case. In general, the focus will be cases where the field is the complex or $p$-adic numbers, namely, $GL_n(\mathbb{C})$ and $GL_n(\mathbb{Q}_p)$.

2.1.1 Important Subgroups of $GL_n(F)$

One important subgroup of $GL_n(F)$ is the Borel subgroup $B$, which contains those matrices that are upper-triangular, namely,

$$B = \left\{ \begin{pmatrix} t_1 & * & * \\ & \ddots & * \\ & & t_n \end{pmatrix} : t_1t_2\cdots t_n \neq 0 \right\}.$$

Furthermore, the Borel subgroup has an important subgroup called the Unipotent radical $U \subset B$, defined as elements of $B$ satisfying $t_i = 1$ for all $i$. These matrices are called unipotent because all of their eigenvalues are equal to 1.

Another important subgroup of $GL_n(F)$ is the subgroup of diagonal matrices, known as the diagonal torus

$$T = \left\{ \begin{pmatrix} t_1 & & \\ & \ddots & \\ & & t_n \end{pmatrix} : t_1t_2\cdots t_n \neq 0 \right\}.$$
We observe that $T$ acts on $U$ via conjugation. In particular, a quick calculation demonstrates that if $t = \text{diag}(t_1, \ldots, t_n) \in T$ and $u = \{u_{ij}\}_{ij} \in U$, then the $(i,j)$ entry of $tut^{-1}$ is $t_i t_{j}^{-1} u_{ij}$ where $i < j$.

Now let $\mathbb{G}_m$ be the multiplicative group of invertible elements, so, e.g., $\mathbb{G}_m(\mathbb{C}) = \mathbb{C}^\times$. This allows us to define the notion of a root through characters of this matrix $tut^{-1}$.

**Definition 2.1.** The root $\alpha_{ij} : T \to \mathbb{G}_m$ for $1 \leq i < j \leq n$ is given by the character

$$
\alpha_{ij} \begin{pmatrix} t_1 & \cdots & t_n \end{pmatrix} = t_i t_{j}^{-1}.
$$

These $\alpha_{ij}$ satisfying $i < j$ are called the *positive roots* and denoted $\Phi^+$. In a similar manner, one can construct *negative roots*

$$
\Phi^- = \{ \alpha_{ij} = \alpha_{ji}^{-1} : 1 \leq j < i \leq n \},
$$

These combine to form the set of roots $\Phi$ for $GL_n(F)$, and this root system encodes the group up to isomorphism. Because this project entirely focuses on $GL_n(F)$, it should be noted that it is assumed throughout that root systems are of type $A_n$.

**Definition 2.2.** Let $\epsilon_i : T \to \mathbb{G}_m$ be the character

$$
\epsilon_i \begin{pmatrix} t_1 & \cdots & t_n \end{pmatrix} = t_i.
$$

Now define the *fundamental weights* of $T$ by $\Lambda_i = \epsilon_1 + \cdots + \epsilon_i$, namely,

$$
\Lambda_i \begin{pmatrix} t_1 & \cdots & t_n \end{pmatrix} = t_1 t_2 \cdots t_i.
$$

These characters are important in that they help understand representations of $GL_n(F)$. This then leads us to define the weight lattice.

**Definition 2.3.** The *weight lattice*

$$
X^*(T) = \{ \lambda : T \to \mathbb{G}_m : \text{algebraic characters } \lambda \}
$$

where an algebraic character is one that can be expressed as a polynomial function on $T$.

**Example 2.4.** In the case $n = 1$, then $GL_1(F) = \mathbb{G}_m$, which means $X^*(T)$ is just the set of algebraic morphisms from $\mathbb{G}_m$ to itself. The only such morphisms are are $\mathbb{Z} \to \mathbb{Z}^n$ for $n \in \mathbb{Z}$, which shows the sense in which $X^*(T)$ is a lattice.
2.1.2 The Weyl Group and Long Words

An important finite group associated to the roots of \( \text{GL}_n(F) \) is given by permutation matrices

\[
W = \left\{ \begin{pmatrix} e_{\pi(1)} \\ e_{\pi(2)} \\ \vdots \\ e_{\pi(n)} \end{pmatrix} : \pi \in S_n \right\}
\]

where \( e_i \) is the vector of all 0s except for a 1 in position \( i \). We call \( W \) the Weyl group, and it is clearly isomorphic to \( S_n \). Further, it acts naturally on the torus via conjugation.

**Example 2.5.** Consider the \( n = 2 \) case. Then we have

\[
S_2 = \left\{ \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \right\}
\]

is the simple group of order 2 and has one nontrivial element. Let \( w \) be this nontrivial element. Then

\[
w \begin{pmatrix} t_1 & 0 \\ 0 & t_2 \end{pmatrix} w^{-1} = \begin{pmatrix} t_2 & 0 \\ 0 & t_1 \end{pmatrix}.
\]

Because it acts on the torus, the Weyl group will also act on the characters of the torus. That is, for \( w \in W, t \in T \), it acts on the weight lattice by

\[(w \cdot \lambda)(t) = \lambda(w^{-1}tw).
\]

Furthermore, it is useful to consider permutations of these diagonal elements \( t_i \). In particular, for any permutation of the \( t_i \), we can write it as the product of simple reflections \( s_i = (t_i \ t_{i+1}) \). For simplicity, we will henceforth write \( s_i \) as \((i \ i+1)\) and sometimes write a product of simple reflections as \((i_1, \ldots, i_k) = s_{i_1}s_{i_2}\cdots s_{i_k}\).

**Definition 2.6.** Let the length \( l : W \to \mathbb{Z} \) of \( w \in W \) be the number of simple reflections in the minimal expansion of \( w \) into simple reflections.

We observe that the length function is well defined due to the fact mentioned in the above paragraph. Note that this expansion is never unique when \( n > 2 \).

**Example 2.7.** Let \( W = S_3 \). Then the minimal expansion of \((1 \ 3)\) is \((1 \ 2)(2 \ 3)(1 \ 2)\), which means \( l((1 \ 3)) = 3 \).

**Proposition 2.8.** There is a unique element \( w_0 \) of \( W = S_n \) whose minimal expansion into simple reflections is of maximal length. In particular, this element is the antidiagonal element

\[
w_0 = \begin{pmatrix} 1 \\ \ddots \\ 1 \\ 1 \end{pmatrix}.
\]

Finally, we have the following two relations: \( s_is_{i+1}s_i = s_{i+1}s_is_{i+1} \), which we will call the “braid” relation, and \( s_is_j = s_js_i \) for \(|i - j| > 1 \), which will call the “switch” relation.
2.2 Representations of Complex Groups

We will start by letting $F$ in the previous section be $\mathbb{C}$, and thus we are working in the context of $GL_n(\mathbb{C})$. We start by introducing the notion of a partition:

**Definition 2.9.** A partition of $n$ is a sequence $(\lambda_1, \lambda_2, \cdots, \lambda_k)$ that sums to $n$ and satisfies $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_k$.

Furthermore, we define the complete homogeneous symmetric polynomials of degree $k$, given by $h_k(x_1, \cdots, x_n)$, to be sum of all degree-$k$ monomials in the $n$ variables.

**Example 2.10.** We have that $h_2(x_1, x_2, x_3) = x_1^2 + x_2^2 + x_3^2 + x_1 x_2 + x_1 x_3 + x_2 x_3$.

In addition, for a partition $\lambda = (\lambda_1, \cdots, \lambda_k)$, we let $h_\lambda = h_{\lambda_1} h_{\lambda_2} \cdots h_{\lambda_k}$.

This now allows us to formally define Schur polynomials, through the Jacobi-Trudi formula.

**Definition 2.11.** For $x = (x_1, \cdots, x_n)$ and some partition $\lambda = (\lambda_1, \cdots, \lambda_k)$ we define $s_\lambda(x) = \det (h_{\lambda_i + j - 1}(x))_{1 \leq i, j \leq k}$.

**Example 2.12.** One can check that $s_{(2,1,1)}(x_1, x_2, x_3) = x_1^2 x_2 x_3 + x_1 x_2^2 x_3 + x_1 x_2 x_3^2$.

Now we introduce some necessary concepts from representation theory.

**Definition 2.13.** A representation of a group $G$ on a vector space $V$ is a group homomorphism $\rho : G \rightarrow GL(V)$ of $G$ to the group of automorphisms of $V$ that satifies $\rho(g_1 g_2) = \rho(g_1) \rho(g_2)$ for all $g_1, g_2 \in G$, where $\dim V$ is called the dimension of the representation.

**Example 2.14.** Let $G = (\mathbb{R}, +)$. Then for $x \in \mathbb{R}$,

$$\rho(x) = \begin{pmatrix} 1 & x \\ 0 & 1 \end{pmatrix}$$

is a two-dimensional representation of $G$.

**Definition 2.15.** Let $\rho$ be a representation of a group $G$. Then the character of the representation is a function $\chi : G \rightarrow \mathbb{C}$ given by

$$\chi(g) = \text{tr} \rho(g),$$

where $\text{tr}$ refers to the trace.
Definition 2.16. A subrepresentation of some representation $\rho$ is a subspace $U \subset V$ such that $U$ is invariant under the action of $G$.

Definition 2.17. A representation $\rho$ is irreducible if there all subrepresentations are trivial, i.e., that there is no proper, nonempty $U$ which is invariant under the action of $G$. A representation that is not irreducible is called reducible.

Definition 2.18. Every nonnegative integral linear combination of the fundamental weights gives a weight inside the weight lattice. We call these weights dominant.

Furthermore, the highest weight of a representation is the vector on which $B$ acts via a character.

The theory of highest weight says that irreducible representations of $GL_n(\mathbb{C})$ are indexed by dominant weights where the representation associated to dominant weight $\lambda$ is a highest weight representation given by the character $\lambda$. See [HFH91] for additional details.

In addition, it is true that the highest weights that index the irreducible representations are partitions, and moreover, the character is just the Schur polynomial associated to that partition when restricted to $T$.

Definition 2.19. A crystal graph is a decorated graph $(\mathcal{B}(\lambda), e_i, f_i, \text{wt}, \epsilon_i, \varphi_i)$ that satisfies a set of axioms, where $e_i$ and $f_i$ are raising and lowering operators, respectively, and $\text{wt} : \mathcal{B} \to X^*(T)$ is some function.

The additional requirements in the definition are not important to the context of this project and can be found in [HK].

For the purposes of this project, this definition is important because crystal graphs arise in the theory of quantum groups and they give combinatorial models to study highest weight representations in a uniform fashion.

In particular, using results from the work of [BZ01, Les19, McN11], the formula we will derive for $s_\lambda(z)$ is expressed as a sum of over the finite crystal $\mathcal{B}(\lambda + \rho)$, where $\rho$ is the sum of the fundamental weights.

2.3 $p$-adic Numbers and $p$-adic Groups

Recall that $\mathbb{R}$ can be constructed as a completion of $\mathbb{Q}$ with respect to the standard absolute value. In number theory, other completions are important, so we review the construction of these, known as the $p$-adic numbers.

Definition 2.20. An absolute value on a field $K$ is a map $|\cdot| : K \to \mathbb{R}^+$ that satisfies the following properties:

i) $|x| = 0$ if and only if $x = 0$

ii) $|xy| = |x| \cdot |y|$

iii) $|x + y| \leq |x| + |y|$

If the absolute value satisfies
we say it is non-archimedian.

Consider any rational number $x$. Then for any prime $p$, we can write $x$ as $p^m a$, where $a, b, m \in \mathbb{Z}$ and $p$ does not divide $a$ or $b$. We define an absolute value on $x$ with respect to $p$ by

$$|x|_p = p^{-m}.$$

Observe that this absolute value is non-archimedian. Note that the traditional absolute value is an archimedian absolute value.

We can define a metric on $\mathbb{Q}$ by $d(x, y) = |x - y|_p$, and thus induces a metric topology on $\mathbb{Q}$.

**Definition 2.21.** The $p$-adic numbers, denoted $\mathbb{Q}_p$, are the field defined as the set of equivalence classes of all Cauchy sequences in $(\mathbb{Q}, d)$, also known as the completion of $\mathbb{Q}$ with respect to $| \cdot |_p$.

Note that every $x \in \mathbb{Q}_p$ can be expanded in a Laurent series

$$x = a_m p^m + a_{m+1} p^{m+1} + \cdots,$$

where, of course, $m$ can be negative, and $0 \leq a_k \leq p - 1$ for all $k$.

**Definition 2.22.** The $p$-adic integers, denoted $\mathbb{Z}_p$, is the ring defined by the $p$-adic numbers $x$ satisfying $|x|_p \leq 1$.

By simply applying the definition, we see that every $x \in \mathbb{Z}_p$ can be written as

$$x = b_0 + b_1 p + b_2 p^2 + \cdots.$$ 

We can now define the group of units $\mathbb{Z}_p^\times$ to be the invertible elements of $\mathbb{Z}_p$. Observe that any a $p$-adic integer with a nontrivial constant term.

Finally, note that every element $x$ has a unique expansion of the form $a p^k$ where $k = \text{val}(x)$ and $a \in \mathbb{Z}_p^\times$.

Before considering $p$-adic integration, the following theorem is necessary:

**Theorem 2.23.** $\mathbb{Z}_p$ is compact.

*Proof.* See [Gou03, p. 64]. 

For any continuous function $f : \mathbb{Q}_p \to \mathbb{C}$, we would like to ask what it means to integrate the function over $\mathbb{Q}_p$ and $\mathbb{Z}_p$. Viewing $\mathbb{Q}_p$ as a topological group, it is locally compact and Haussdorff, which means that it can be endowed with a Haar measure. While the details of a Haar measure can be found in [Bum04], for the present context it is important to note that a Haar measure is invariant under left translation, which for our purposes means that, for some $\mu$-integrable function $f$ and locally compact group $G$,

$$\int_G f(x) d\mu = \int_G f(gx) d\mu$$
for all $g \in G$.

Moreover, a Haar measure is unique up to scalars. See [Bum04] for more details. The properties of a Haar measure imply that because $\mathbb{Z}_p$ is compact, its volume is finite. Therefore, due to the uniqueness up to scalars, we can normalize the measure such that $\mu(\mathbb{Z}_p) = 1$.

Now, for a measurable function $f : \mathbb{Q}_p \to \mathbb{C}$, we can calculate

$$\int_{\mathbb{Q}_p} f d\mu \text{ and } \int_{\mathbb{Z}_p} f d\mu.$$  

**Example 2.24.** We have that

$$\int_{\mathbb{Z}_p^\times} d\mu = 1 - p^{-1}.$$  

We can see this by first observing that $\mathbb{Z}_p^\times = \mathbb{Z}_p \setminus p\mathbb{Z}_p$, which implies $\mu(\mathbb{Z}_p^\times) = \mu(\mathbb{Z}_p) - \mu(p\mathbb{Z}_p)$ and then writing

$$1 = \int_{\mathbb{Z}_p} d\mu = \int_{p\mathbb{Z}_p} d\mu + \int_{p\mathbb{Z}_p + 1} d\mu + \cdots + \int_{p\mathbb{Z}_p + p-1} d\mu,$$

and because of translation invariance, each of the $p$ integrals on the right-hand side must be equal and therefore equal to $p^{-1}$. Rearranging this gives the result.

More information on $p$-adic integration and additional examples can be found in [Pop11].

We can also endow any vector space over $\mathbb{Q}_p$ with the induced Haar measure. Note that later in this paper, we will consider higher-dimensional vector spaces, and thus we will often let $d\mu = dx$ when we are integrating over $x$. When we integrate over $\mathbb{Q}_p^\times$, the measure is $d\mu = dx/|x|$.

### 2.4 Other Notation

We now fix a finite extension $F$ of $\mathbb{Q}_p$ for the remainder of the paper. Furthermore, fix some uniformizer $\varpi$, and label $\mathbb{Z}_p$ by $\mathcal{O}$ and of course $\mathbb{Z}_p^\times$ by $\mathcal{O}^\times$. Let $q$ be $|\mathcal{O}/\varpi|$.

### 3 Spherical Whittaker Functions

We will now introduce functions on $GL_n(\mathbb{Q}_p)$, which are important in the integrals we will compute. We let $B = B(\mathbb{Q}_p)$ be the Borel subgroup of $GL_n(\mathbb{Q}_p)$.

We first define a character $\chi_{\mathbf{z}} : B \to \mathbb{C}^\times$ of $B$ parametrized by $\mathbf{z}$ given by

$$\begin{pmatrix} t_1 & * & * \\ \vdots & \ddots & * \\ t_n \end{pmatrix} \mapsto |t_1|^{s_1} \cdots |t_n|^{s_n}$$
where $s_i \in \mathbb{C}$ for all $i$.

We now want to use this character to define a function on the entire group. The Iwasawa decomposition tells us that $GL_n(\mathbb{Q}_p) = B \cdot GL_n(\mathbb{Z}_p)$. For a proof, see [Bum97, Prop 4.5.2]. This decomposition always exists, but is not unique; in particular, $B \cap GL_n(\mathbb{Z}_p) \neq \emptyset$. We can now define the function:

**Definition 3.1.** Let $g = b \cdot k$ be the Iwasawa decomposition of $g \in \mathbb{Q}_p$ where $b \in B$ and $k \in GL_n(\mathbb{Z}_p)$. Then we define the spherical Whittaker function $f_{\underline{s}} : GL_n(\mathbb{Q}_p) \to \mathbb{C}$ to be the unique function that is $(B, \chi_{\underline{s}})$ invariant on the left and $GL_n(\mathbb{Z}_p)$ invariant on the right such that $f_{\underline{s}}(I_n) = 1$.

We see that $f_{\underline{s}}(g) = f_{\underline{s}}(b \cdot k) = f_{\underline{s}}(b)$ because of right $GL_n(\mathbb{Z}_p)$ invariance and $f_{\underline{s}}(b) = \chi_{\underline{s}}(b)$ because of left $(B, \chi_{\underline{s}})$ invariance and the fact that $f_{\underline{s}}(I_n) = 1$, giving that $f_{\underline{s}}(g) = \chi_{\underline{s}}(b)$.

**Lemma 3.2.** The formula $f_{\underline{s}}(g) = \chi_{\underline{s}}(b)$ is well-defined. In particular, if $b \cdot k = b' \cdot k'$, then $\chi_{\underline{s}}(b) = \chi_{\underline{s}}(b')$.

**Proof.** Assume $bk = b'k'$. Then $b'b^{-1} = kk'^{-1} \in GL_n(\mathbb{Z}_p)$, which means $b'b^{-1} \in B \cap GL_n(\mathbb{Z}_p)$. We know that $\chi_{\underline{s}}$ is trivial on this, so $\chi_{\underline{s}}$ is trivial on $b'b^{-1}$. Thus $\chi_{\underline{s}}(b') = \chi_{\underline{s}}(b)$. \qed

Next, fix an additive character $\psi : \mathbb{Q}_p \to \mathbb{C}^\times$, where for $x = a_{-k}p^{-k} + \cdots + a_{-1}p^{-1} + a_0 + \cdots$ we have

$$x \mapsto e^{2\pi i(a_{-k}p^{-k} + \cdots + a_{-1}p^{-1})},$$

which means we have $\psi(x+y) = \psi(x)\psi(y)$. From here, we will use $\psi$ to construct a character of the unipotent radical, and in a standard abuse of notation we also call it $\psi : U \to \mathbb{C}^\times$. It is given by summing the main super-diagonal. For example, we have

$$\psi \left( \begin{pmatrix} 1 & u_1 & u_3 \\ 1 & u_2 & 1 \end{pmatrix} \right) = \psi(u_1 + u_2).$$

We can now write down the integral we would like to compute:

$$W_{\underline{s}}(g) = \int_U f_{\underline{s}}(w_0ug)\psi(u)^{-1}du,$$

where, as $p$-adic manifolds, we have $U \cong \mathbb{Q}_p^{(2)}$ by simply looking at the coordinates. This is important because it allows us to identify the Haar measure on
with the standard measure on this vector space \( \mathbb{Q}_p^{(2)} \). It is important to note that this function satisfies the property

\[
W_\mathcal{A}(u'g) = \psi(u')W_\mathcal{A}(g)
\]

for \( u' \in U \), which can be seen directly by applying the change of variables \( u \mapsto u(u')^{-1} \). However, this integral does not converge. This is because \( U \) has infinite volume and \( f_\mathcal{A} \) does not have compact support. To deal with this problem, first observe that there is a filtration of \( \mathbb{Q}_p \)

\[
\mathbb{Z}_p \subset \varpi^{-1}\mathbb{Z}_p \subset \cdots \subset \varpi^{-k}\mathbb{Z}_p \cdots
\]

where \( \varpi^{-k}\mathbb{Z}_p \) is compact for each \( k \). We can show that an integral over \( \mathbb{Q}_p \) is equal to the limit as \( k \to \infty \) of \( \varpi^{-k}\mathbb{Z}_p \).

It is true that due to the nature of this function, there must be some finite \( k \) where these integrals eventually become constant. For details, see [Bum97] for the case of \( GL_2 \) and references therein.

In order to apply this result to the integral above, we define \( U_k \) to be

\[
U_k := \left\{ \begin{pmatrix} 1 & \cdots & u_{ij} \\ \vdots & \ddots & \vdots \\ 1 \end{pmatrix} : u_{ij} \in \varpi^{-k}\mathbb{Z}_p, 1 \leq i, j \leq n-1 \right\}
\]

and then consider

\[
W_\mathcal{A}(g) = \lim_{k \to \infty} \int_{U_k} f_\mathcal{A}(w_0ug)\psi(u)^{-1}du,
\]

which converges for finite \( k \). Thus this gives a well-defined, continuous function in \( g \) that satisfies the aforementioned invariance property. These integrals play an important role in the global theory of automorphic representations. More information can be found in the introduction of [Les19].

Finally, we must ask which values of \( g \) are important in \( W_\mathcal{A}(g) \). Remembering that we can write \( g \) as \( b \cdot k \) where \( b \in B \) and \( k \in GL_n(\mathbb{Z}_p) \), we know

\[
f_\mathcal{A}(w_0ug) = f_\mathcal{A}(w_0ubk) = f_\mathcal{A}(w_0ub),
\]

which means for all \( g \) we can find a corresponding \( b \in B \) where \( f_\mathcal{A}(g) = f_\mathcal{A}(b) \).

Furthermore, we can write \( b \in B \) as \( b = u't \), where \( u' \in \mathring{U} \) and \( t \in T \). Plugging this in the integral and doing a change of variables gives

\[
W_\mathcal{A}(u't) = \psi(u') \int_{U} f_\mathcal{A}(w_0ut)\psi(u)^{-1}du
\]

which gives the result

\[
W_\mathcal{A}(u'tk) = \psi(u')W_\mathcal{A}(t).
\]
We next observe that we can write \( t \) uniquely as
\[
\begin{pmatrix}
  t_1 \\
  \vdots \\
  t_n
\end{pmatrix}
= \begin{pmatrix}
  \varpi^{\text{val}(t_1)} \\
  \vdots \\
  \varpi^{\text{val}(t_n)}
\end{pmatrix}
\begin{pmatrix}
  u_1 \\
  \vdots \\
  u_n
\end{pmatrix}
\]
where \( u_i \in \mathbb{Z}_p^\times \) for all \( i \). Call the latter matrix on the right-hand side \( M \), and letting \( \lambda = (\text{val}(t_1), \ldots, \text{val}(t_n)) \), call the former matrix \( \varpi^{\lambda} \). Now we can conclude that
\[
W_\varpi(u'tk) = \psi(u')W_\varpi(t) = \psi(u')W_\varpi(\varpi^{\lambda}).
\]
because we have that \( M \in GL_n(\mathbb{Z}_p) \).

Therefore, we can simply consider those \( g \) that are of the form \( \varpi^{\lambda} \), where \((\lambda_1, \ldots, \lambda_n) \in \mathbb{Z}^n\).

However, we can restrict this further with the following theorem.

**Proposition 3.3.** \( W_\varpi(\varpi^{\lambda}) = 0 \) unless \( \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \).

**Proof.** See [CS80, Lemma 5.1] □

Thus we only need to consider the \( \lambda \) that are partitions, as defined in Section 2.2.

Now, set
\[
z := \begin{pmatrix}
  \varpi^{-s_1} \\
  \vdots \\
  \varpi^{-s_n}
\end{pmatrix}
\in T(\mathbb{C}) \subset GL_n(\mathbb{C}).
\]

**Theorem 3.4.** The Casselman-Shalika formula [CS80] says that
\[
W_\varpi(\varpi^{\lambda}) = q^{-\langle \lambda, \rho \rangle} \prod_{\alpha \in \Phi^+} (1 - q^{-1}z^\alpha)s_\lambda(z),
\]
where \( z^\alpha = z_iz_j^{-1} = q^{s_\alpha} - s_{ij} \), noting that \( s_\lambda(z) \) makes sense because \( \lambda \) is a partition.

To reiterate the introduction, the goal of this project is computing the left-hand side in new ways in order to give formulas for the right-hand side.

In order to compute the left-hand side, we can do a change of variables to rewrite \( W_\varpi(\varpi^{\lambda}) \) over the lower unipotent radical \( U^- \) as
\[
W_\varpi(\varpi^{\lambda}) = \int_{U^-} f(u\varpi^{\mu_0\lambda})\psi(u)du.
\]
The details of arriving at this integral can be found in [Les19, Section 3]. From here, [McN11, Theorem 7.2] tells us that \( U^- \) can be written as the union of disjoint cells
\[
U^- = \bigsqcup_{m \in \mathbb{Z}_0^N} C^i(m).
\]
This decomposition is important because it allows us to decompose $W_\mathcal{B}(\varpi^\lambda)$ as a sum $\sum_{\mathbf{m} \in B(-\infty)} I_\lambda(\mathbf{m})$ over an infinite crystal $B(-\infty)$, where the details can be found in [Les19, Section 3]. This infinite crystal $B(-\infty)$ is equipped with a structure such that we can extract the subset that gives the finite crystal $B(\lambda + \rho)$ for a given highest weight. In this project, we exploit this structure to show that most of the terms vanish and be left with a sum over a finite crystal.

Using this decomposition, we get that

$$I_\lambda(\mathbf{m}) = \int_{C^*(\mathbf{m})} f(u) \psi(u) du.$$  

In addition, [McN11, Lemma 6.3] tells us that $f_\mathcal{B}$ is constant on each cell with value

$$\prod_{\alpha \in \Phi^+} (q^{-\langle \rho, \alpha' \rangle} x_\alpha)^{m_\alpha}.$$  

Furthermore, we have from [Les19, Prop 4.2] that on each cell the Haar measure is

$$du = q^{\langle \rho, \text{wt}(\mathbf{m}) \rangle} \prod_{\alpha \in \Phi^+} dt_\alpha$$  

where $t_\alpha$ is the coordinates that we will work with. Combining all of this, [Les19, Section 3] shows that we can re-write $\psi(u)$ in a way that will facilitate calculating the integrals.

From here, we use the result in [Les19, Thm 1.2] that we can compute certain polynomials $s_i$ explicitly as a sum of $i$-trails, which are sequence of weights meeting certain conditions that essentially encodes paths of edges and are defined for any connected Lie group. A certain specialization of these polynomials gives new polynomials $g_i$ that essentially recover as much of the information from the previous polynomials as possible. From here, [Les19, Conj. 4.6] conjectures that these $g_i$ are sufficient to compute $I_\lambda\mathbf{m}$. The details of this computation are covered below, but importantly we have

**Conjecture 3.5.**

$$I_\lambda(\mathbf{m}) = \int_{C^*(\mathbf{m})} f(u) \psi \left( \sum_{i \in I} \varpi^\lambda g_i(t_\alpha) \right) du.$$  

In our special case, this conjecture can be checked in a similar fashion as in [Les19, Appendix A]. Thus, we must compute the $g_i$ by first computing the $s_i$ and enumerating the $i$-trails.

However, the $i$-trails cannot be computed explicitly. In particular, they are dependent on a choice of long word $w_0$. Fortunately, while $i$-trails are quite complicated in general, they can be reduced to a combinatorial expression in the case of $GL_n(F)$.  

15
4 Computing $i$-trails

As described above, the first part of the project was to construct an algorithm that would output the $i$-trails from a chosen long word. This was done using Python code, and the results facilitate calculation of the integrals. The first step in the algorithm is based on a proposition that states that, for each fundamental weight, we can calculate the $i$-trails through essentially looking for subsets of the chosen combinatorial expansion that are equivalent to some $v$ that we calculated through the fundamental weight, in the case of Type A root systems. We can state this special case of the result from [BZ01] as follows:

**Proposition 4.1.** Fix a fundamental weight $\Lambda_i$ and a long word $i$. Let $w_0 s_i \Lambda_i = v_i \Lambda_i$ be a minimal representative. The $i$-trails from $\Lambda_i$ to $w_0 s_i \Lambda_i$ are in bijection with subwords $(i_{k(1)}, \ldots, i_{k(p)})$ of $i$ where $l(v_i) = p$ and $k(1) < \cdots < k(p)$ that are reduced words for $v_i^{-1}$.

In this proposition, $w_0$ is the product of simple reflections that composes the long word. Using this proposition, we wrote a function to calculate the subwords that are reduced words for this $v = v_i^{-1}$. This function has three discrete sub-functions, each of which accomplishes a specific task in building up the subwords.

The first sub-function generates all of the possible long words. Generating the space of long words is important because each long word gives another test case, and looking at all of the long words is important for understanding the structure of the choices in general. The number of long words for $GL_n$ is $a_n$, where $a_n$ is given by

$$a_n = \frac{(n)!}{\prod_{i=0}^{n-2} (2i + 1)^{-i+n-1}}.$$  

This is OEIS sequence A005118. This sequence is exponential in $n$, and increases quite rapidly. The first few terms are 1, 2, 16, 768, which demonstrates this quick growth. By $GL_8$, there are 4860879568960 long words.

Generating all the long words was done through applying the “braid” and “switch” relations in a recursive manner until there is no longer a way to apply either to get a unique long word. This method uses the critical fact, mentioned above, that every long word can be reached from any other through a chain of these relations. Psuedo-code for this function is as follows:

```python
def find_long_words(a_n):
    starting_word = (1,2,1,3,2,1,..., n,n-1,...,2,1)
    long_words_found = [starting_word]
    while len(long_words_found) < a_n:
        long_words_found.append(apply_braid_relations(long_words_found))
    return long_words_found
```

The next sub-function calculates the $v_i^{-1}$, which is done through a recursive process that continued to shorten $w_0 s_i \Lambda_i$ until a minimal representative was
reached. This is done through four techniques: the first two are to remove simple reflections off the right side that are inverses of each other (i.e., if the same reflection appears twice, it can be cancelled), and removing simple reflections that are not $i$, as $s_j \Lambda_i = \Lambda_i$ when $j \neq i$, as can be shown relatively trivially. Once it cannot be shortened through these two techniques, the algorithm applies the braid and switch relation until one of them can apply. If, after applying every possible braid and switch, it cannot be shortened any more, then it must be a minimal representative, and the inverse is returned. The pseudo-code for this is as follows:

```python
def find_v_inv(chosen_long_word):
    current_simplified = chosen_long_word
    while True:
        if can_remove_inverses(current_simplified):
            current_simplified = remove_inverses(current_simplified)
            continue
        if can_remove_simple_refl(current_simplified):
            current_simplified = remove_simple_refl(current_simplified)
            continue
        if can_apply_relations(current_simplified):
            current_simplified = apply_relations(current_simplified)
            continue
    return current_simplified
```

The third step is to use these first two steps to calculate subwords that are reduced words for $v_i^{-1}$, which is done by looking at the powerset of the long word (without the empty set) and checking if each subset is equivalent to $v_i^{-1}$, and then only taking those of minimum length (the reduced ones). The function ends by exporting all this data into a nicely formatted CSV to facilitate analysis. The pseudo-code for this is below:

```python
def find_subwords_reduced_words(v_inv, chosen_long_word):
    possible_subwords = powerset(chosen_long_word)
    final_subwords = []
    for subword in possible_subwords:
        if is_equivalent(subwords, v_inv):
            final_subwords.append(subword)
    return final_subwords
```

We demonstrate what the CSV looks like for a single long word, $(1, 2, 1, 3, 2, 1)$, in the table below:

The second function in the Python code takes the result of this first function (the subwords) and calculates the monomials that are used in the integrals. In the case of $GL_n$, this is done according to the following formula from [Les19,
Prop 4.3]:

\[ s_i(u) = \sum_{i\text{-trails}} \frac{b_{i1}^{c_1(\pi)} \cdots b_{iN}^{c_N(\pi)}}{b_1^{(\Lambda_i, \beta_i)} \cdots b_N^{(\Lambda_i, \beta_N)}}. \]

In this formula, the \( b_i \) are the coordinate system that arise from our choice of coordinates, and the \( c_i(\pi) \) encode the length of edges in the \( i \)-trail. Furthermore, these \( \beta_i \) can be explicitly calculated from the choice of long word, and the exponents in the denominator are calculated by taking this inner product of these \( \beta_i \) and the fundamental weights \( \Lambda_i \). Just like the last function, this one contains three discrete steps, although it is notably less computation than the first function. The first sub-function generates the \( i \)-trails from the subword calculated in the first function, and then uses this to trivially produce the \( c \) vectors. This is important because it determines the exponents in the numerator of the sum that we are using. The second sub-function calculates the \( \beta_i \), and then uses the \( \Lambda_i \) to calculate the denominators of the monomials. The third sub-function combines these and simplifies the resultant fractions to produce the final monomials in terms of the \( b_i \). This allows us to write down the actual sum without any hand calculation, which facilitates calculating the integrals.

It is important to note that this algorithm is relatively inefficient, and this leads to computational limitations in higher-rank cases. As it stands, the algorithm will only run up to \( GL_5 \), because after this, the algorithm reaches the maximum Python recursion depth. The reason for this is directly related to the super-exponential growth of the number of long words—note there are 768 long words for \( GL_5 \), but 292864 for \( GL_6 \). Once we have an understanding of these cases, it is useful to develop a more efficient way to code this algorithm, which can certainly be done. The current algorithm was designed to be able to essentially be the brute-force solution that produces examples, and efficiency was not a primary concern.

5 Computing Integrals From \( i \)-trails

Once the Python program outputs the terms in the integrals, we can apply [Les19, Conj. 4.6] in order to write down and then compute the integrals. The long word for which we calculate the integrals is \((2, 1, 3, 2, 1, 1)\), where \((i, j, \ldots)\) represents the product of simple reflections \((i \ i + 1)(j \ j + 1)\ldots\). Note that this choice of reduced expression is equivalent to giving a convex ordering of the positive roots. In this case, this long word induces the ordering

\[ \alpha_2 > \alpha_2 + \alpha_3 > \alpha_1 + \alpha_2 > \alpha_1 + \alpha_2 + \alpha_3 > \alpha_1 > \alpha_3 \]
on the positive roots. Following the methodology in [Les19], there is one i− trail from \( \Lambda_1 \rightarrow w_0s_1\Lambda_1 \), five from \( \Lambda_2 \rightarrow w_0s_2\Lambda_2 \), and one from \( \Lambda_3 \rightarrow w_0s_2\Lambda_3 \).

Plugging this choice of long word into the Python program gives

- \( s_1 = \frac{1}{b_5} = X_5 \)
- \( s_2 = \frac{1}{b_1} + \frac{b_6}{b_3b_4} + \frac{b_5}{b_2b_3b_4} + \frac{b_4b_6}{b_1b_2b_3} = X_4 + X_3 + \frac{X_2X_4}{X_3} + X_2 + X_1 \)
- \( s_3 = \frac{1}{b_6} = X_6. \)

From here, we can calculate the bounding data \( s_\alpha = \text{val}(\varpi_\lambda X_\alpha) \).

- \( s_1 = \lambda_2 - m_1 - m_2 - m_3 + m_5 + m_6 \)
- \( s_2 = \lambda_2 - m_2 - m_3 - m_4 + m_5 + m_6 \)
- \( s_3 = \lambda_2 - m_3 - m_4 + m_6 \)
- \( s_4 = \lambda_2 - m_4 \)
- \( s_5 = \lambda_1 - m_5 \)
- \( s_6 = \lambda_3 - m_6 \)
- \( s_2 + s_4 - s_3 = \lambda_2 - m_2 - m_4 + m_5. \)

The value of these \( s_\alpha \)—where the subscript on \( s \) is referring to the ordering of the positive roots—tell us whether or not we are in \( B(\lambda + \rho) \) or on the boundary, where we are in \( B(\lambda + \rho) \) if and only if \( s_\alpha \geq -1 \) for each \( \alpha \in \Phi^+ \) and \( s_2 + s_4 - s_3 \geq -1 \). We are on the boundary when there is equality for at least one of the \( s_\alpha \).

### 5.1 Results

Using the computations above, the results from [Les19] allow us to get that

- \( g_1(t_\alpha) = t_5 \)
- \( g_2(t_\alpha) = t_4 + \frac{t_3t_4}{t_6} + \frac{t_2t_4}{t_5} + \frac{t_2t_3t_4}{t_5t_6} + \frac{t_1t_2t_3}{t_5t_6} \)
- \( g_3(t_\alpha) = t_6. \)

We can then directly apply the formula in Section 3, and we get that \( I_\lambda(m) \) equals \( \prod_{\alpha} (q^{-1}x_\alpha)^{m_\alpha} \) times

\[
\int_{C_\lambda(m)} \psi \left( \varpi_\lambda^t_5 + \varpi_\lambda^t_6 + \varpi_\lambda^t_1 + \frac{t_3t_4}{t_6} + \frac{t_2t_4}{t_5} + \frac{t_2t_3t_4}{t_5t_6} + \frac{t_1t_2t_3}{t_5t_6} \right) du,
\]

where the \( x_\alpha \) are the coordinates for the \( z_i \). The domain of integration and the resulting formula is dependent on the values of the \( m_i \), and more details can be
found in [Les19]. We can now simplify this as which we write as

\[ I(s_1, m_1)I(s_5, m_5)I(s_6, m_6)I_\lambda(s_2, s_3; m_2, m_3, m_4) \prod_\alpha x_\alpha^{m_\alpha}, \]

where

\[ I(a, b) = \begin{cases} 
    q^{a+b} \int_{\omega = 0} \psi(t) dt & b = 0 \\
    q^{a+b} \int_{\omega = \varnothing} \psi(t) dt & b > 0 
\end{cases} \]

and from here there are four cases for the value of \( I_\lambda(s_2, s_3; m_2, m_3, m_4) \).

Three of these cases are contained within \( B(\lambda + \rho) \), corresponding to the first sum above. This means that we have \( s_\alpha \geq -1 \) for all \( \alpha \). The fourth case corresponds to the error terms, where \( m \not\in B(\lambda + \rho) \), which have a nonzero contribution by themselves. Note that this means \( s_\alpha < -1 \) for some \( \alpha \).

We start by making a change of variables to get

\[ I_\lambda(s_2, s_3; m_2, m_3, m_4) = \int \int \psi \left( \omega^{s_2} y_2 + \omega^{s_3} y_3 + \omega^{s_2 + s_4 - s_3} y_2 y_4 y_3 + \omega^{s_4} y_4 \right) dy_2 dy_3 dy_4. \]

### 5.1.1 Interior Case

Observe that when \( s_2 + s_4 - s_3 \geq 0 \), we can simplify the above to

\[ I_\lambda(s_2, s_3; m_2, m_3, m_4) = \int \int \psi \left( \omega^{s_2} y_2 + \omega^{s_3} y_3 + \omega^{s_4} y_4 \right) dy_2 dy_3 dy_4, \]

which means that in this case the behavior is standard in this case. We get the standard contribution, defined in [Les19] as

\[ I_\lambda(m) = x_1^{k_1} x_2^{k_2} x_3^{k_3} \prod_{\alpha \in \Phi^+} G(s_\alpha, m_\alpha) \]

where we define \( x^{k(m)} = x_1^{k_1} x_2^{k_2} x_3^{k_3} := \prod_\alpha x_\alpha^{m_\alpha} \) and

\[ G(s_\alpha, m_\alpha) = \begin{cases} 
    1 - q^{-1} & m_\alpha > 0, s_\alpha \geq 0 \\
    -q^{-1} & m_\alpha > 0, s_\alpha = -1 \\
    1 & m_\alpha = 0, s_\alpha \geq 0 \\
    0 & \text{otherwise}. 
\end{cases} \]

### 5.1.2 Partial Boundary Case

This is where \( s_2 + s_4 - s_3 = -1 \) but we don’t have that \( s_2 = s_3 = s_4 = -1 \), which constitutes the full boundary case. Note that when \( s_2 + s_4 - s_3 \leq -1 \), there is contribution from the additional term in the integrand which means we will not get the standard contribution. There are two ways to get this case which will be broken up below.
Case 1: $s_2 = -1, s_3 = s_4 \geq 0$ or $s_4 = -1, s_3 = s_2 \geq 0$

The symmetry between $s_2$ and $s_4$ in $s_2 + s_4 - s_3$ means that these two cases operate quite similarly, which is why they are grouped together. If $s_4 = -1$, we must have that $m_4 > 0$, but when $s_2 = -1$, we have the weaker restriction that $m_4 > 0$ or $m_2 > 0$. Thus the former case contains four possibilities while the latter contains six. To demonstrate how these integrals compute, we show an example with $s_2 = s_3 \geq 0, s_4 = -1, m_2 > 0, m_3 > 0, m_4 > 0$. Note the $\varpi^{s_2}y_2$ and $\varpi^{s_3}y_3$ do not contribute, giving

$$I_\lambda(s_2, s_3, s_4; m_2, m_3, m_4) = \int_{\varpi^s} \int_{\varpi^s} \int_{\varpi^s} \psi \left( \frac{-1}{y_3} y_4 + \varpi^{-1} y_4 \right) dy_2 dy_3 dy_4,$$

and doing a change of variables $y_2 = y_2 y_4 / y_3$ gives

$$(1 - q^{-1}) \int_{\varpi^s} \int_{\varpi^s} \psi \left( \varpi^{-1} y_2 + \varpi^{-1} y_4 \right) dy_2 dy_4$$

$$= (1 - q^{-1}) \int_{\varpi^s} \psi \left( \varpi^{-1} y_2 \right) dy_2 \int_{\varpi^s} \psi \left( \varpi^{-1} y_4 \right) dy_4,$$

because the volume of $\varpi^s$ is $1 - q^{-1}$. Since each of these integrals is equal to $-q^{-1}$, we get the result of $q^{-2}(1 - q^{-1})$. In total, we get

$$I_\lambda(s_2, s_3, s_4; m_2, m_3, m_4) = \begin{cases} 
q^{-2} (1 - q^{-1}) & m_2 > 0, m_3 > 0, m_4 > 0 \\
q^{-2} & m_2 > 0, m_3 = 0, m_4 > 0 \\
0 & \text{otherwise}
\end{cases}$$

Case 2: $s_2 = s_4 \geq 0, s_3 = 1$

In this case, we do not have any restrictions on the $m$s, and considering the same $m_2 > 0, m_3 > 0, m_4 > 0$ example as above, the integral follows similarly as the previous case. We observe that because $s_2, s_3, s_4 \geq 0$, the only term that contributes is the $y_2 y_4 / y_3$ term, and we get

$$I_\lambda(s_2, s_3, s_4; m_2, m_3, m_4) = \int_{\varpi^s} \int_{\varpi^s} \int_{\varpi^s} \psi \left( \varpi^{-1} y_2 y_4 / y_3 \right) dy_2 dy_3 dy_4.$$

The same change of variables as the previous case gives

$$(1 - q^{-1})^2 \int_{\varpi^s} \psi \left( \varpi^{-1} y_2 \right) dy_2 = -q^{-1}(1 - q^{-1})^2.$$

Working out the other cases gives the final expression:

$$I_\lambda(s_2, s_3, s_4; m_2, m_3, m_4) = \begin{cases} 
-q^{-1} (1 - q^{-1})^2 & m_2 > 0, m_4 > 0 \\
-q^{-1} & m_2 > 0, m_3 = 0, m_4 = 0 \\
0 & \text{otherwise}.
\end{cases}$$
5.1.3 Full Boundary Case

This case is where we have $s_2 = s_3 = s_4 = -1$, and here we see from the above computations that we must have $m_4 > 0$, or else $s_4 \neq -1$. Because each of the relevant $s_i$ is -1, this integral contains all four terms. We demonstrate the calculation in the $m_2 > 0, m_3 > 0$ case. Plugging in the $s_i$ gives

$$\int_{\mathcal{O}} \int_{\mathcal{O}} \int_{\mathcal{O}} \int_{\mathcal{O}} \psi \left( \varpi^{-1} (y_2 + y_3 + \frac{y_2 y_4}{y_3} + y_4) \right) dy_2 dy_3 dy_4.$$

And after a change of variables $y_2 = w y_3$ we get

$$\int_{\mathcal{O}} \int_{\mathcal{O}} \psi(\varpi^{-1} (y_1 + y_3)) \int_{\mathcal{O}} \psi(\varpi^{-1} w(y_1 + y_3)) dw dy_1 dy_3.$$

Now apply a change of variables $t = y_1 + y_3$. We see that $t \in \mathcal{O}$ and when $t \in \mathcal{O} \times \mathcal{O}$ then $t \not\equiv y_1 \mod \varpi$. Subtracting this case off, we get

$$(1 - q^{-1}) \sum_{j=0}^{\infty} \left( \int_{\mathcal{O} \times \mathcal{O}} \psi(\varpi^{-1} t) \int_{\mathcal{O} \times \mathcal{O}} \psi(\varpi^{-1} w t) dw dt \right) - I$$

where

$$I = \int_{\mathcal{O} \times \mathcal{O} \times \mathcal{O} \times \mathcal{O}} \psi(\varpi^{-1} t) \int_{\mathcal{O} \times \mathcal{O}} \psi(\varpi^{-1} w t) dw dt dx.$$

In the case of the first integral, a change of variables $a = wt$ gives

$$(1 - q^{-1}) \sum_{j=0}^{\infty} \varpi^j \int_{\mathcal{O} \times \mathcal{O}} \psi(\varpi^{-1} t) \int_{\mathcal{O} \times \mathcal{O}} \psi(\varpi^{-1} a) da dt.$$

where we observe that the first term in the sum is just $q^{-2}$. Thus we write

$$(1 - q^{-1}) q^{-2} + (1 - q^{-1}) \sum_{j=1}^{\infty} \varpi^j \int_{\mathcal{O} \times \mathcal{O}} \psi(\varpi^{-1} t) \int_{\mathcal{O} \times \mathcal{O}} \psi(\varpi^{-1} a) da dt.$$

and we see that the remaining terms in the sum are $\varpi^{-2j} (1 - \varpi^{-1})^2$. This leaves

$$(1 - q^{-1}) q^{-2} + (1 - q^{-1})^3 \sum_{j=1}^{\infty} \varpi^j = (1 - q^{-1}) q^{-2} + (1 - q^{-1})^2 q^{-1} = q^{-1} (1 - q^{-1}).$$

Now consider $I$. Applying the change of variables $y = wt$ gives

$$\int_{\mathcal{O} \times \mathcal{O} \times \mathcal{O} \times \mathcal{O}} \psi(\varpi^{-1} y) \int_{\mathcal{O} \times \mathcal{O}} \psi(\varpi^{-1} y) dy dt dx$$

$$= -q^{-1} \int_{\mathcal{O} \times \mathcal{O} \times \mathcal{O} \times \mathcal{O}} \psi(\varpi^{-1} t) dt dx.$$
The volume of one residue class is $q^{-1}$, so we can write it

$$-q^{-1} \sum_{x_0 \in \text{residues}} q^{-1} \int_{O \times x_0 \equiv t(\wp)} \psi(\wp^{-1} t) dt.$$  

Now set $t = x_0 + \wp u$ for $u \in O$, which gives

$$-q^{-3} \sum_{x_0 \in \text{residues}} \int_{O} \psi(\wp^{-1}(x_0 + \wp u)) dt = -q^{-3} \sum_{x_0 \in \text{residues}} \psi(\wp^{-1} x_0) \int_{O} \psi(u) du$$

$$= -q^{-3} \sum_{x_0 \in \text{residues}} \psi(\wp^{-1} x_0)$$

$$= q^{-3}.$$  

Combining the two terms gives the final result of

$$q^{-1}(1 - q^{-1}) - q^{-3}.$$  

Across all cases, we get that

$$I_\lambda(s_2, s_3; m_2, m_3, m_4) = \begin{cases} q^{-1}(1 - q^{-1}) - q^{-3} & m_2 > 0, m_3 > 0 \\ q^{-1}(1 - q^{-1}) & \text{otherwise} \end{cases}.$$  

This accounts for all of the cases where $m \in B(\lambda + \rho)$.

### 5.1.4 Exterior Case

Every case where $m \not\in B(\lambda + \rho)$ provides 0 contribution, except for $s_2 = s_3 = s_4 = -k$, where in this case we also the restriction that $m_4 > 0$. We again show the computation for the $m_2 > 0, m_3 = 0$ case, noting that the $m_2 > 0, m_3 > 0$ case proceeds identically as above, except that $-k$ is the valuation of the terms instead of $-1$, and this means that we get $I = 0$, giving $q^{-k}(1 - q^{-1})$. This is important because it is used in the following computation. We start similarly as above, except with slightly different sets over which we integrate:

$$\int_{O} \int_{O} \int_{O} \psi \left( \wp^{-1}(y_2 + y_3 + \frac{y_2 y_4}{y_3} + y_4) \right) dy_2 dy_3 dy_4.$$  

which we write as

$$\sum_{j \geq 0} \int_{O} \int_{O} \int_{O} \psi(\wp^{-j}(y_2 + y_3 + y_4 + \frac{y_2 y_4}{y_3})) dy_2 dy_3 dy_4.$$  

And after a change of variables $y_2 = w y_3$ we get

$$\sum_{j \geq 0} \wp^{-j} \int_{O} \int_{O} \psi(\wp^{-j}(y_1 + y_3)) \int_{O} \psi(\wp^{-j} w(y_1 + y_3)) dw dy_1 dy_3.$$  

23
Now, we can separate the first term of the sum as it is simply the $m_2 > 0, m_3 > 0$ case, and in the remaining sum do a change of variables $t = y_2 + y_4$, where now we know $|y_2| \neq |y_4|$. This gives
\[
q^{-k}(1 - q^{-1}) + (1 - q^{-1}) \sum_{j \geq 1} \mathbb{C}^{-j} \int_{\mathbb{O}^x} \psi(\mathbb{C}^{-k}t) \int_{\mathbb{C}^{-j}\mathbb{O}^x} \psi(\mathbb{C}^{-k}w) \, dw dt.
\]
But $-j - k < -1$ so this inner integral is 0, and thus only the first term in the sum contributes. This cancellation happens in every case, and overall we get that
\[
I_\lambda(s_2, s_3, s_4; m_2, m_3, m_4) = q^{-k}(1 - q^{-1}).
\]

These computations prove the following theorem

**Theorem 5.1.** If $m \in \mathcal{B}(\lambda + \rho)$, we get
\[
I_\lambda(m) = G'(s_2, s_3, s_4, m_2, m_3, m_4) \prod_{\alpha \neq 2,3,4} G(s_\alpha, m_\alpha)x_1^{k_1}x_2^{k_2}x_3^{k_3} =: G_1(m)x_1^{k_1}x_2^{k_2}x_3^{k_3}
\]
where we define $G'(s_2, s_3, s_4, m_2, m_3, m_4)$ to be $G(s_2, m_2)G(s_3, m_3)G(s_4, m_4)$ in the on-polytope case $s_2 + s_4 - s_3 \geq 0$, $G(s_3, m_3)G(\min\{s_2, s_4\}, m_2)G(\min\{s_2, s_4\}, m_4)$ in the partial boundary cases $s_2 = s_3 \geq 0, s_4 = -1$ and $s_4 = s_3 \geq 0, s_2 = -1$, and
\[
\begin{cases}
q^{-1}(1 - q^{-1}) & s_2 = s_3 = s_4 = -1, \min\{m_2, m_4\} = 0 \\
q^{-1}(1 - q^{-1}) - q^{-3} & s_2 = s_3 = s_4 = -1, m_2 > 0, m_4 > 0 \\
-q^{-1}(1 - q^{-1})^2 & s_2 = s_4 \geq 0, s_3 = 1, m_2 > 0, m_4 > 0 \\
0 & \text{otherwise}
\end{cases}
\]
in all other cases.

If $m \notin \mathcal{B}(\lambda + \rho)$, then $I_\lambda(m) = 0$ unless $s_2 = s_3 = s_4 = -k < -1$, where we get
\[
I_\lambda(m) = q^{-k}(1 - q^{-1}) \prod_{\alpha \neq 2,3,4} G(s_\alpha, m_\alpha)x_1^{k_1}x_2^{k_2}x_3^{k_3} =: G_{res}(m)x_1^{k_1}x_2^{k_2}x_3^{k_3}.
\]

### 5.2 Computational Verification of $\lambda = 0$ Case

In order to check that this formula satisfies the relationship described in the first section, namely that
\[
\prod_{\substack{(i,j) \\ i < j}} (1 - q^{-1}(z_i z_j^{-1})) s_\lambda(z_1, \ldots, z_n) = \sum_{m \in \mathcal{B}(-\infty)} I_\lambda(m),
\]
it is useful to do implement these calculations in Wolfram Mathematica. In order to simplify the computation, consider the case where $\lambda = 0$, which means that $s_\lambda(z) = 1$, giving
\[
\prod_{\alpha \in \Phi^+} (1 - q^{-1}z^\alpha) = \sum_{\text{lattice points } m} I_\lambda(m).
\]
In a general sense, this program encodes each of the terms in the expanded product on the left-hand side, and then checks if the cases in the right-hand side that correspond to that term generate the same coefficient.

This is accomplished in four concrete steps. The first is to generate all the possible $m$ that correspond to a particular term in the product expansion. Using that $x_{\alpha_i} = x_i$, we can enumerate the terms by looking at the $(k_1, k_2, k_3)$ exponents from $x_1^{k_1} x_2^{k_2} x_3^{k_3}$. From here, we use the ordering of the positive roots to relate $m$s and $k$s. This gives

\[
\begin{align*}
    k_1 &= m_3 + m_4 + m_5 \\
    k_2 &= m_1 + m_2 + m_3 + m_4 \\
    k_3 &= m_2 + m_4 + m_6,
\end{align*}
\]

for which Mathematica can generate solutions over the positive integers. These are precisely the sets of $m$ that correspond to the $x_1^{k_1} x_2^{k_2} x_3^{k_3}$ term.

The next step is to calculate the $s_i$s and determine the relevant values that break up the cases. This is done by simply applying a formula to the relevant $m$s.

Next, there are two functions that encode all of the different aforementioned cases—interior, boundary, and exterior cases. The first function encodes the standard contribution, and thus the interior cases. The latter takes care of the more complex cases.

The final step is to sum up and simplify all the terms for a given weight, and then check if the sum matches the coefficient in the product expansion. This is done iteratively for each of the different weights, including weights that do not actually appear in the expansion, since it is an additional confirmation for these to evaluate to 0. This code is provided in the appendix.

This shows how we can write the sum over $m \in B(\infty)$ as a sum of two terms in a natural way

\[
\sum_{m \in B(-\infty)} I_{\lambda}(m) = \sum_{m \in B(\lambda + \rho)} I_{\lambda}(m) + \sum_{m \notin B(\lambda + \rho)} I_{\lambda}(m).
\]

Importantly, there are $m \notin B(\lambda + \rho)$ that satisfy $I_{\lambda}(m) \neq 0$. In this case, we say $m \in B(\lambda + \rho)_{res}$.

In the case of $i = (2, 1, 3, 2, 1, 3)$, we get a weight of

\[(m_3 + m_4 + m_5)\alpha_1 + (m_1 + m_2 + m_3 + m_4)\alpha_2 + (m_2 + m_4 + m_6)\alpha_3,
\]

which is calculated from the induced ordering of the positive roots.

## 6 Resonance Families

We know that if $m \notin B(\lambda + \rho)$ for all $m$ with nonzero contribution satisfying $k(m) = (k_1, k_2, k_3)$, then the coefficient on the term corresponding to $(k_1, k_2, k_3)$ is 0. Thus, we aim to demonstrate that any nonzero terms must sum to 0.
In order to have \( m \not\in \mathcal{B}(\lambda + \rho) \) and \( I_\lambda(m) \neq 0 \), we need \( s_2 = s_3 = s_4 = -k \) for \( k > 1 \), as shown in the calculations in section 4.

This restriction means that \( \lambda_2 - m_2 - m_3 - m_4 + m_5 + m_6 = \lambda_2 - m_3 - m_4 + m_6 \) and \( \lambda_2 - m_3 - m_4 + m_6 = \lambda_2 - m_4 \), which implies that \( m_2 = m_5 \) and \( m_3 = m_6 \).

This also imposes a restriction on the weight, namely that

\[
    k(m) = (m_2 + m_3 + m_4, m_1 + m_2 + m_3 + m_4, m_2 + m_3 + m_4)
\]

and gives simplified formulas for the bounding data

\[
    s_1 = \lambda_2 - m_1 \\
    s_2 = s_3 = s_4 = \lambda_2 - m_4 \\
    s_5 = \lambda_1 - m_5 \\
    s_6 = \lambda_3 - m_6.
\]

This means we have nonzero contribution for \( 0 \leq m_1 \leq \lambda_2 + 1, 0 \leq m_2 \leq \lambda_1 + 1, 0 \leq m_3 \leq \lambda_3 + 1 \), and \( m_4 \geq \lambda_2 + 2 \).

Furthermore, observe that \( k_1 = k_3 \) and \( m_1 \) is fixed at \( k_2 - k_1 \), which means \( k_2 \geq k_1 \).

**Definition 6.1.** For a fixed weight \( (k_1, k_2, k_3) \) that meets the conditions above, define a resonance family \( RF(m_3) \) to be

\[
    \{(m_1, m_2, m_3, k_1 - m_2 - m_3) : 0 \leq m_2 \leq \min\{\lambda_1 + 1, k_1 - m_3 - \lambda_2 - 2\}\}
\]

for a fixed \( m_1 \). Call \( (m_1, 0, m_3, k_1 - m_3) \) the head of the resonance family.

Observe that these two restrictions on \( m_2 \) in the resonance family differ in that if \( m_2 > \lambda_1 + 1 \), then the \( m_2 \) term will cause the integral to vanish, whereas if \( m_2 > k_1 - m_3 - \lambda_2 - 2 \), then the terms will be on the polytope, and importantly will not vanish.

**Theorem 6.2.** For a fixed weight \( (k_1, k_2, k_3) \) that meets the conditions above and \( m \) satisfying \( k(m) = (k_1, k_2, k_3) \), we know \( I_\lambda(m) \) can only be nonzero when \( 0 \leq m_3 \leq \lambda_3 + 1 \). For each \( m_3 \leq k_1 - \lambda_1 - \lambda_2 - 3 \), we have that

\[
    \sum_{n \in RF(m_3)} I_\lambda(n) = 0.
\]

**Proof.** Note that these are precisely the resonance families for which \( \lambda_1 + 1 < k_1 - m_3 - \lambda_2 - 2 \), or the families that do not intersect the polytope. Directly applying the computations in section 4, we have

\[
    \sum_{n \in RF(m)} I_\lambda(n) = \sum_{m_2=0}^{\lambda_1+1} (1 - q^{-1}) q^{\lambda_2-k_1+m_3+m_2} G(s_1, m_1) G(s_5, m_5) G(s_6, m_6),
\]

where we again emphasize that the condition on \( m_3 \) means that it makes sense to sum to \( \lambda_1 + 1 \). Because \( m_1 \) and \( m_3 = m_6 \) are constant in this sum, we can
write this as

\[(1 - q^{-1})q^{\lambda_2 - k_1 + m_3}G(s_1, m_1)G(s_6, m_6) \sum_{m_2=0}^{\lambda_1+1} q^{m_2}G(s_5, m_5).\]

Now, assume \(\lambda_1 \geq 1\) and note that \(G(s_5, m_5) = 1\) when \(m_2 = 0\) (and \(s_5 \geq 0\)), \(G(s_5, m_5) = 1 - q^{-1}\) when \(1 \leq m_2 \leq \lambda_1\) because \(s_5 \geq 0\), and finally \(G(s_5, m_5) = -q^{-1}\) when \(m_2 = \lambda_1 + 1\). Thus we expand the sum and write

\[\left(1 \pm (1 - q^{-1}) \sum_{m_2=1}^{\lambda_1} q^{m_2} - q^{\lambda_1}\right) = \left(1 + (1 - q^{-1}) \frac{q^{\lambda_1} - 1}{1 - q^{-1}} - q^{\lambda_1}\right) = 0.\]

On the other hand, if \(\lambda_1 = 0\), we get

\[\left(1 \pm (1 - q^{-1}) \sum_{m_2=0}^{1} q^{m_2} G(s_5, m_5)\right)\]

where in the first case \(G(s_5, m_5) = 1\) and in the second \(G(s_5, m_5) = -q^{-1}\). This gives

\[\left(1 \sum_{m_2=0}^{1} q^{m_2} G(s_5, m_5) = (1 + q(-q^{-1})) = 0.\]

\[\square\]

For a fixed weight \((k_1, k_2, k_3)\) that meets the conditions for nonzero contribution off the polytope, we see that \(RF(m_3) = 0\) for all \(0 \leq m_3 \leq \lambda_3 + 1\) if and only if \(k_1 \geq \lambda_1 + \lambda_2 + \lambda_3 + 4\). This follows directly from letting \(m_3 = \lambda_3 + 1\) in the inequality in the theorem above. Given that this is all the values of \(m_3\) for which the term has a nonzero contribution, this is precisely what we set out to prove at the beginning of the section.

It also means that when \(k_1 < \lambda_1 + \lambda_2 + \lambda_3 + 4\), then there is an \(m_3\) such that \(RF(m_3) \cap B(\lambda + \rho) \neq \emptyset\). We call such resonance families \(\lambda\)-relevant. Now set \(RF(m_3)^\circ = RF(m_3) \setminus (RF(m_3) \cap B(\lambda + \rho))\). This allows us to sum up the results of this paper in the following theorem:

**Theorem 6.3.** For \(G_1(m)\) and \(G_{res}(m)\) as defined in the previous section, we have

\[p^{-\langle \lambda, \rho \rangle} \prod_{\alpha \in \Phi^+} (1 - p^{-1}z^\alpha) s_\lambda(z) = \sum_{m \in B(\lambda + \rho)} G_1(m)x_1^{k_1}x_2^{k_2}x_3^{k_3} + \sum_{RF(m_3)} \sum_{\lambda\text{-relevant}} G_{res}(m)x_1^{k_1}x_2^{k_2}x_3^{k_3}.\]

where \(x_i = z^{\alpha_i} = z_i(z_{i+1})^{-1}\).
7 Future Directions

In this paper, we examined formulas for a specific choice of long word $i = (2, 1, 3, 2, 1, 1)$. Given that it is not desirable to have formulas depend on choices, we also want to see if we can understand the formulas in a canonical way. To do this, we can consider that the combinatorial choices fall into categories in terms of the structure of the $i$-trails that are computed, and thus one idea is to exploit this structure to try to understand these integrals in a canonical way. Furthermore, we can look at $GL_n(F)$ for $n > 4$ and conjecture about what happens for a general $n$. 
A Mathematica Code

GetMs[k1_, k2_, k3_] :=
Solve[m5 + m4 + m3 == k1 && m1 + m2 + m3 + m4 == k2 &&
m2 + m4 + m6 == k3 && m1 >= 0 && m2 >= 0 && m3 >= 0 && m4 >= 0 &&
m5 >= 0 && m6 >= 0, Integers]

GetSsFromMs[mss_] :=
Table[{{m1, -m1 - m2 - m3 + m5 + m6}, {m2, -m2 - m3 - m4 + m5 +
m6}, {m3, -m3 - m4 + m6}, {m4, -m4}, {m5, -m5}, {m6, -m6}, {-m2 - m3 - 2*m4 + m5 +
m6 - (-m3 - m4 + m6)}} /. mss[[j]], {j, 1, Length[mss]}]

StandardContribution[mss, sss_] :=
Piecewise[{{1 - q^(-1), m > 0 && s == 0}, {-q^(-1),
   m > 0 && s == -1}, {1, m == 0 && s == 0}}]

BadTerm[smss_] :=
Piecewise[{{Product[stdcont[smss[[j, 1]], smss[[j, 2]]], {j, {2, 3, 4}}],
   smss[[2, 2]] + smss[[4, 2]] - smss[[3, 2]] >=
   0}, {-q^(-1)*(1 - q^(-1))^2,
   smss[[2, 2]] = smss[[4, 2]] && smss[[2, 2]] == 0 && smss[[3, 2]] == 1 &&
   smss[[2, 1]] > 0 && smss[[4, 1]] > 0 }, {-q^(-1),
   smss[[2, 2]] == smss[[4, 2]] && smss[[2, 2]] == 0 && smss[[3, 2]] == 1 &&
   smss[[2, 1]] > 0 && smss[[3, 1]] == 0 &&
   smss[[4, 1]] ==
   0 }, {q^(-2)*(1 - q^(-1)),
   (smss[[2, 2]] == smss[[3, 2]] && smss[[2, 2]] == 0 &&
   smss[[4, 2]] == -1) || (smss[[4, 2]] == smss[[3, 2]] &&
   smss[[4, 2]] == 0 && smss[[2, 2]] == -1) && (smss[[2, 1]] > 0 &&
   smss[[4, 1]] > 0 && smss[[3, 1]] >= 0)},
   {q^(-2),
   (smss[[2, 2]] == smss[[3, 2]] && smss[[2, 2]] == 0 &&
   smss[[4, 2]] == -1) || (smss[[4, 2]] == smss[[3, 2]] &&
   smss[[4, 2]] == 0 && smss[[2, 2]] == -1) && (smss[[2, 1]] > 0 &&
   smss[[4, 1]] > 0 && smss[[3, 1]] >= 0)},
   {q^(-1)*(1 - q^(-1)),
   smss[[2, 2]] == -1 && smss[[4, 2]] == -1 &&
   smss[[3, 2]] == -1 && smss[[3, 1]] == 0 ||
   smss[[2, 1]] == 0},
   {q^(-1)*(1 - q^(-1)) - q^(-3),
   smss[[2, 2]] == -1 && smss[[4, 2]] == -1 &&
   smss[[3, 2]] == -1 && smss[[3, 1]] == 0 ||
   smss[[2, 1]] == 0},
   {q^(-1)*(1 - q^(-1)) - q^(-3),
   smss[[2, 2]] == -1 && smss[[4, 2]] == -1 &&
   smss[[3, 2]] == -1 && smss[[3, 1]] == 0 ||
   smss[[2, 1]] == 0},
   {q^(-1)*(1 - q^(-1)) - q^(-3),
   smss[[2, 2]] == -1 && smss[[4, 2]] == -1 &&
   smss[[3, 2]] == -1 && smss[[3, 1]] == 0 ||
   smss[[2, 1]] == 0},
   {q^(-1)*(1 - q^(-1)) - q^(-3),
   smss[[2, 2]] == -1 && smss[[4, 2]] == -1 &&
   smss[[3, 2]] == -1 && smss[[3, 1]] == 0 ||
   smss[[2, 1]] == 0},
   {q^(-1)*(1 - q^(-1)) - q^(-3),
   smss[[2, 2]] == -1 && smss[[4, 2]] == -1 &&
   smss[[3, 2]] == -1 && smss[[3, 1]] == 0 ||
   smss[[2, 1]] == 0}}]

total[k1_, k2_, k3_] :=
Expand[Simplify[
  Sum[Simplify[

29
Product[GetSs[GetMs[k1, k2, k3]][[t, j, 1]],
    GetSs[GetMs[k1, k2, k3]][[t, j, 2]], {j, {1, 5, 6}}]*
BadTerm[GetSs[GetMs[k1, k2, k3]][[t]]], {t, 1,
Length[GetSs[GetMs[k1, k2, k3]]]}]]

Table[total[moreweights[[i, 1]], moreweights[[i, 2]],
    moreweights[[i, 3]]] ==
    Coefficient[%23,
        x1^moreweights[[i, 1]]*x2^moreweights[[i, 2]]*
        x3^moreweights[[i, 3]] /. {x3 -> 0, x1 -> 0, x2 -> 0}, {i, 1,
            Length[moreweights]}]
Acknowledgements

I would like to thank my thesis advisor, Dr. Spencer Leslie, for introducing me to this problem, teaching me all of the necessary background, and mentoring me each step of the way. Furthermore, throughout this project, Dr. Leslie’s guidance has significantly strengthened my ability to communicate mathematics clearly and completely.

References


