Visualizing Random Walks on Supercharacter Theories for Uppertriangular Unipotent Matrices

Justin Willson

Defended on April 1, 2019

Thesis Advisor and Honors Council Representative:
Nathaniel Thiem, Department of Mathematics
Thesis Committee:
Richard Green, Department of Mathematics
Joshua Grochow, Department of Computer Science

1 Introduction

The conjugacy classes of some finite groups are provably wild in the sense that there is no general solution to finding their conjugacy classes other than brute force. This makes using character theory to study the group difficult as such methods are aided by an understanding of the conjugacy classes. In order to surmount this difficulty, Diaconis and Isaacs developed the theory of supercharacters in [1]. Constructing a supercharacter theory for a finite group has the effect of gluing conjugacy classes together into superclasses in order to make the structure of the group easier to understand. This gluing also has the effect of infusing the group with interesting combinatorial properties that can be visualized as familiar object. We can also perform a random walk on the supercharacter theory in order to further study a group.

In this paper, we will apply these methods both to the subgroup of uppertriangular unipotent matrices of the form

$$\left(\begin{array}{c|c} Id_m & A \\ \hline 0 & Id_n \end{array}\right),$$

where A is an arbitrary $m \times n$ matrix, and to the group of uppertriangular matrices as a whole. One supercharacter theory will be represented by matchings and the other will be represented by Dyck paths. We will construct probability functions for the supercharacter theories in order to perform random walks on them. Further, we will animate these walks in order to better understand them.

The author would like to thank the 2017 CU Math Department REU for the funding to begin this work.

2 Preliminaries

In order to begin, we must define the objects we will be studying as well as the tools we will be using.

2.1 Uppertriangular Unipotent Matrices

The group of unipotent uppertriangular matrices have been proven to have a wild conjugacy class structure. This makes them a perfect candidate to analyze using a supercharacter theory.

Definition 2.1.1. The group of uppertriangular unipotent matrices $UT(n, \mathbb{F}_q)$ over a finite field $\mathbb{F}_q$ with q elements is the subgroup of $GL(n, \mathbb{F}_q)$ defined by

$$UT(n, \mathbb{F}_q) = \{g \in GL(n, \mathbb{F}_q) \mid (g - id)_{ij} \neq 0 \text{ implies } i < j\}.$$

Informally, the group of uppertriangular unipotent matrices, $UT(n, \mathbb{F}_q)$ is the set of uppertriangular matrices with ones along the diagonal and arbitrary elements of $\mathbb{F}_q$ above the diagonal.

2.2 Supercharacter Theory

A *supercharacter theory* of a finite group G is a coarsening of the character theory of G, that combines the conjugacy classes of G into superclasses. In the same way a character function is constant on conjugacy classes, a supercharacter is constant on superclasses.

Definition 2.2.1. A supercharacter theory $(\mathrm{Cl}, \mathrm{Ch})$ of a finite group G is a partition, Cl, of G and a set of supercharacters Ch, such that

1. The number of blocks in Cl is the same as the number of characters in Ch.
2. Each block in Cl is a union of conjugacy classes.
3. The characters in Ch are constant on the blocks of Cl. That is, if $\phi \in$ Ch, and g and h are group elements in the same block of Cl, then $\phi(g) = \phi(h)$.
4. Each irreducible character of G is the constituent of exactly one element in Ch.

The blocks of Cl are called *superclasses*.

While it is not generally true that one can associate each supercharacter to a superclass, because of the examples we will be working with in this paper, such an association can be made. In this case, we will associate each superclass, A, to a supercharacter χ^A. Since supercharacters are constant on superclasses, it is sufficient to define $\chi^A(B)$ for every superclass B in order to define a supercharacter. In order to simplify our calculations, we will also frequently consider functions

$$\frac{\chi^A}{\chi^A(1)}$$

where 1 denotes the superclass of the identity.

2.3 Group Walk

In order to study a probability function, it can be useful to associate it to group walk. A famous example of this is to associate methods of shuffling a deck of cards to a group walk on the symmetric group.

Definition 2.3.1. Given a group G and probability function $\text{pr} : G \to \mathbb{Q}$, a *group walk of length* n is a sequence $(g_i)_{i=0}^n$ of group elements such that g_0 is the identity element in the group and $g_i = g_{i-1}h_i$ where h_i is an element of G that was chosen based on the probability function, pr.

Its helpful to visualize the group as a set of point and the group walk as a particle moving from point to point.

The structure of a group walk is heavily influenced by the choice of probability function. If the function does not have support on a set of generators for the group, then it will not be possible to walk to every group element during a walk. On the other hand, if the probability function has support on too many group elements, the walk will be too chaotic or converge too quickly to the uniform distribution. The following example highlights this fact.

Example 2.3.2. Let $G = \mathbb{Z}_5 \times \mathbb{Z}_5$ and define three probability functions, pr_1, pr_2, and pr_3, as follows

$$\text{pr}_1(g) = \begin{cases} 1 & \text{if } g = (1,0) \\ 0 & \text{else} \end{cases} \quad \text{pr}_2(g) = \frac{1}{25} \quad \text{pr}_3(g) = \begin{cases} \frac{1}{4} & \text{if } g \in \{(1,0),(-1,0),(0,1),(0,-1)\} \\ 0 & \text{else} \end{cases}.$$

In order to analyze the group walks generated by each of the probability functions, we can visualize our group as a 5×5 grid, with the first coordinate representing the x-axis and the second coordinate representing the y-axis. If we walk on G using pr_1, our group walk is always the same and looks like stepping along the x-axis of our visualization. In contrast, if we use pr_2, that is, the probability function where each element has an equal likelihood of being chosen, then our walk would bounce around the grid at random. Finally, if we do a walk with the third probability function,

we see a walk where at each point in the walk, we have an equal likelihood of moving up, down, left, or right. Intuitively, we expect this walk to stay "near" the origin, but occasionally venture out and reach points "farther" away. The walk generated by pr_3, unlike the walk generated by pr_2, has enough structure that we can reason about what will happen at each step in the walk while not having so much structure as to be predetermined in the way pr_1 was.

We can extend the definition of a group walk to a walk on superclasses in order to study the structure of a given supercharacter theory.

Definition 2.3.3. Let G be a group, $(\mathrm{Cl}, \mathrm{Ch})$ be a supercharacter theory on G and let $\mathrm{pr} : G \to \mathbb{Q}$ be a probability function such that $\mathrm{pr}(g) = \mathrm{pr}(h)$ for any g and h in the same block of Cl. A *superclass walk of length* n is a sequence of superclasses determined by identifying each element of a group walk generated by pr with its corresponding superclass.

In this case, rather than a particle moving from group element to group element, we have a particle moving from superclass to superclass.

2.4 Eigenvalues for a Group Walk

Given a probability function and a supercharacter theory, we can compute eigenvalues for the associated walk. Each row A of the supercharacter table is an eigenvector for the probability matrix $(\mathrm{pr}(B, C))$ where B and C are superclasses and $\mathrm{pr}(B, C)$ is the probability of going from B to C during the supercharacter walk. The eigenvalues for these eigenvectors can be computed with the formula

$$\frac{|G|\langle \mathrm{pr}, \chi^A \rangle}{\chi^A(1)},$$

as shown in [2], where pr is the probability function, χ^A is the character function corresponding to the row A, and $\langle -, - \rangle$ is the character inner product. In doing this calculation, it is sufficient to have a formula for

$$\frac{\chi^A(B)}{\chi^A(1)}$$

since we can write our eigenvalue formula as

$$|G|\langle \mathrm{pr}, \frac{\chi^A}{\chi^A(1)} \rangle.$$

3 Matching Supercharacter Theory

We can define a supercharacter theory on block matrices of the form

$$\left(\begin{array}{c|c} Id_m & A \\ \hline 0 & Id_n \end{array}\right)$$

by associating each matrix to a matching which can be represented by non-attacking rooks on a chessboard. We can then visualize each matching as a point in m dimensions in order to visualize a walk on this supercharacter theory.

3.1 Matchings and the Point Presentation

Definition 3.1.1. An $m \times n$ matching is a subset M of $\{1, \ldots, m\} \times \{1, \ldots, n\}$ such that no two distinct elements of M share a first or second component. That is if (i, j) and (k, l) are in M with $i = k$ or $j = l$, then $(i, j) = (k, l)$.

Because we are limited by m and n the pigeon-hole principle tells us that $|M| \leq \min(m, n)$. Further, we can represent a matching, M, as a grid of numbers with a 1 in the i^{th} row and j^{th} column for each (i, j) in M.

Example 3.1.2.

If M is the 3×4 matching given by $\{(1, 3), (3, 4)\}$, then we can represent M as

		1	
			1

.

We call this presentation of a matching the rook presentation, because it demonstrates a second understanding of $m \times n$ matchings as a way of placing rooks on an $m \times n$ chessboard such that no two rooks attack each other. If two rooks were to attack each other, then they would have the same first or second component and therefore they would not represent a valid matching. In light of this understanding, we call the 1s in the diagram *rooks*. This presentation is very useful as it gives us a method to count the number of possible matchings.

Lemma 3.1.3. *Assume that $n \leq m$. The number of $m \times n$ matchings is*

$$1 + \sum_{k=1}^{n} \binom{n}{k} (m)(m-1)\ldots(m-(k-1)).$$

Proof. First, there is exactly one way to place no rooks on the chessboard. If we do place rooks on the board, we can place k of them for each $1 \leq k \leq n$. Then, for each k, we choose k rows to place rooks in. Next, we have m choices of where to place the rook in the first row, then $(m-1)$ choices for the second, and so on up until the k^{th} rook for which we have $(m-(k-1))$ choices. Thus our final count is

$$1 + \sum_{k=1}^{n} \binom{n}{k} (m)(m-1)\ldots(m-(k-1)).$$

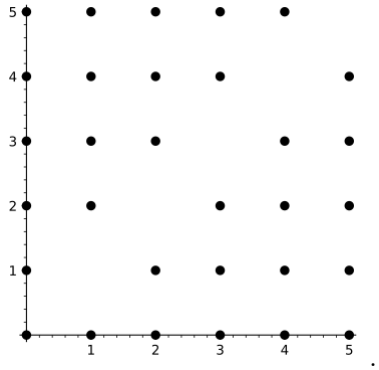
□

Now, one problem with the rook presentation of a matching is that it cannot represent all possible matchings at once. This makes it more difficult to see the structure of the superclasses. A solution to this problem is to convert an $m \times n$ matching into an m-dimensional lattice point. To do this, we will create an m-tuple, $(x_1, \ldots x_m)$ from a matching M. For each element (i, j) of a matching M we let $x_i = j$. If there is no element corresponding to some x_i, we take $x_i = 0$.

Example 3.1.4. Let M be the 3×4 matching given by $\{(1, 3), (3, 4)\}$, then we can represent M as the point $(3, 0, 4)$.

We will call this presentation the point presentation of a matching. With the point presentation, we can display all possible matchings simultaneously.

Example 3.1.5. The set of all 2×5 matchings looks like



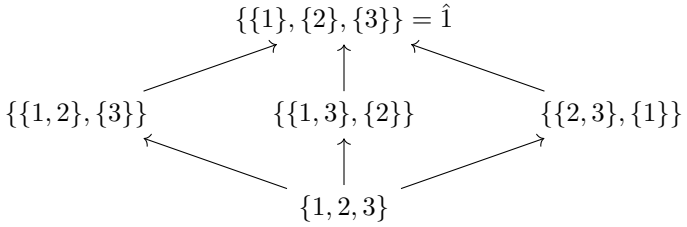
.

Notice how there are no points on the diagonal as such a point would represent a matching with two rooks in the same column.

This new visualization shows us another possible way of counting the number of possible $m \times n$ matchings. Looking at Example 3.1.5, we see we can count the number of $2 \times n$ matchings by looking at the square of lattice points and then subtracting the diagonal not including the origin. This yields the formula $(n+1)^2 - n$. Moving to the 3-dimensional case, we find planes of missing points rather than a line. Here, we see that simply subtracting the planes is not sufficient to count the number of possible matchings, as if we do, we subtract the diagonal too many times. In this instance, our formula is given by $(n+1)^3 - 3n(n+1) + 2n$. We can generalize this method of adding and subtracting various dimensional hyperplanes, but in order to do this, we need to define a poset.

Definition 3.1.6. A *poset*, or partially ordered set, is a set P with a binary relation $\leq$ that is reflexive, antisymmetric, and transitive.

Example 3.1.7. Let $\mathcal{P}_m$ be the poset of partitions of $\{1, \ldots, m\}$ where a partition p is less than a partition r if p is coarser than r. In this case let $\hat{1}$ represent the finest partition. We can visualize $\mathcal{P}_3$ as



where an element higher in the diagram is larger than all the elements below it.

With this definition, we are prepared to give a novel method of counting the number of valid matchings.

Theorem 3.1.8. *Let $m \leq n$ and let $\mathcal{P}_m$ be defined as in example 3.1.7. Then the number of valid $m \times n$ matchings is given by*

$$\sum_{\substack{p \text{ a partition of} \\ \{1,\ldots,m\}}} \mu(p, \hat{1}) n^{\gamma(p)} (n+1)^{\sigma(p)},$$

where μ is the Möbius function on $\mathcal{P}_m$, $\gamma(p)$ is the number of non-singleton blocks of p, and $\sigma(p)$ is the number of singleton blocks of p.

Proof. We will say that a point $(x_1, \ldots, x_m)$ satisfies the partition p if x_i and x_j are equal and non-zero whenever i and j are in the same block of p. A point represents an invalid matching if it satisfies any partition other than $\hat{1}$. This is because if a point, $(x_1, \ldots, x_m)$, satisfies a partition with any non-singular blocks, it has some x_i equal to some x_j with $i \neq j$. Based on how we construct a point from a matching, the point would not represent a valid matching. For a more concrete example, notice that all the points that are missing in example 3.1.5 satisfy the partition $\{\{1, 2\}\}$ and all the points that are present only satisfy the partition $\{\{1\}, \{2\}\}$.

So, our goal will be to count all the points that only satisfy the finest partition, $\hat{1}$. To that end, define $f(p)$ to be the number of points that satisfy p and no coarser partition. Then, $f(\hat{1})$ is the number of partitions that only satisfy $\hat{1}$ as there are no finer partitions. Next, we define

$$g(p) = \sum_{r \leq p} f(r).$$

Then, $g(p)$ represents the number of points that satisfy p as every point that satisfies p only satisfies p or satisfies some other minimal partition less that p. We also see that

$$g(p) = n^{\gamma(p)}(n+1)^{\sigma(p)},$$

as there are n choices for each non-singleton block and $n+1$ choices for each singleton block. Thus, by Möbius inversion, we see that

$$f(\hat{1}) = \sum_{\substack{p \text{ a partition of} \\ \{1,\ldots,m\}}} \mu(p, \hat{1}) g(p) = \sum_{\substack{p \text{ a partition of} \\ \{1,\ldots,m\}}} \mu(p, \hat{1}) n^{\gamma(p)}(n+1)^{\sigma(p)}.$$

□

This counting argument gives us some insight into the geometry of the point presentation. We see a generalization of the image in example 3.1.5, where the points that represent valid fillings are separated by hyperplanes of empty space that are enumerated by set partitions. In this generalized case, for an $m \times n$ matching we have an m dimensional hypercube of lattice points that has side lengths of $n+1$. This hypercube is "missing" hyperplanes of dimension $m-1$ which are enumerated by by the set of partitions of m with $m-1$ blocks. The intersections of these hyperplanes are enumerated by the number of partitions that have $m-2$ blocks and so on until the partition with exactly one block. We also see that the shape of these intersections is dependent on the size of the blocks of the corresponding partition. More precisely, we see the size of the intersection corresponding to the partition p is $n^{\gamma(p)}(n+1)^{\sigma(p)}$.

3.2 Matching Supercharacter Theory

There exists a supercharacter theory on the subgroup of $UT(m+n, \mathbb{F}_q)$ composed of block matrices of the form

$$\left(\begin{array}{c|c} Id_m & A \\ \hline 0 & Id_n \end{array}\right)$$

whose superclasses are indexed by the set of $m \times n$ matchings. We will call this subgroup M_{nm}. This supercharacter theory was first defined in [3]. However, the point presentation allows us to more effectively visualize a group walk in order to get an understanding of it.

In order to construct this supercharacter theory, we must begin by defining some notation.

Definition 3.2.1. If A is an $m \times n$ matrix over a field $\mathbb{F}_q$, let $A_{[i,j]}$ denote the submatrix of A formed by all entries below and to the left of the entry A_{ij}. If A_{ij} is not a valid entry, i.e. one of i or j is greater than m or n respectively, or one of them is less than 0, then we take the empty matrix.

The following example illustrates this notation.

Example 3.2.2. Let

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix},$$

then

$$A_{[1,2]} = \begin{pmatrix} 1 & 2 \\ 4 & 5 \\ 7 & 8 \end{pmatrix} \quad \text{and} \quad A_{[3,3]} = \begin{pmatrix} 7 & 8 & 9 \end{pmatrix}.$$

With this notation, we are able to convert a matrix into a matching. The matrix A is represented by the matching consisting of the (i, j) that satisfy

$$\operatorname{rank}(A_{[i,j]}) - \operatorname{rank}(A_{[i+1,j]}) - \operatorname{rank}(A_{[i,j-1]}) + \operatorname{rank}(A_{[i+1,j-1]}) = 1.$$

In order to understand this conversion, its helpful to look at a few examples.

Example 3.2.3. Working over $\mathbb{F}_5$, if

$$A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix},$$

then, as we would expect based on the visual appearance of A, the corresponding matching is

	1
1	

.

If

$$B = \begin{pmatrix} 1 & 2 \\ 1 & 2 \end{pmatrix},$$

then the corresponding matching is

1	

.

The equivalence relation on block matrices of the form

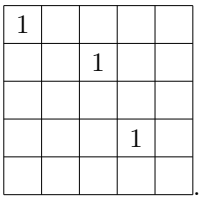
$$\left(\begin{array}{c|c} Id_m & A \\ \hline 0 & Id_n \end{array}\right)$$

generated by identifying A with its representation as a matching defines the superclasses for our supercharacter theory. Because the group M_{mn} is abelian, these superclasses trivially fulfill the requirement of being unions of conjugacy classes. According to [3], if N and M are matchings representing a superclass, then

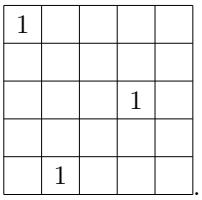
$$\frac{\chi^N(M)}{\chi^N(1)} = \begin{cases} \frac{1}{q^{\operatorname{nst}_M^N}(1-q)^{|N \cap M|}} & \text{if no rook of } N \text{ is directly above or} \\ & \text{directly to the right of a rook of } M \\ 0 & \text{otherwise} \end{cases}$$

where nst_M^N is the number of pairings of a rook in N with a rook in M that is below and to the right of the rook in N and $|N \cap M|$ is the number of rooks that are in the same position of both M and N. Again, it is helpful to give an example to illustrate this formula.

Example 3.2.4. If N is the filling



and M is the filling



then, $\text{nst}_M^N = 2$ because the rook in the fifth row and second column of M is below and to the right of two different rooks in N. We also see that $|N \cap M| = 1$ because there is exactly one rook that is in the same position in both fillings. Hence,

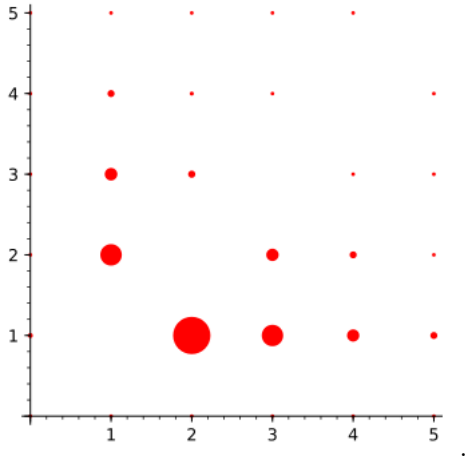
$$\frac{\chi^N(M)}{\chi^N(1)} = \frac{1}{q^2(1-q)}.$$

3.3 Matching Supercharacter Theory Walk

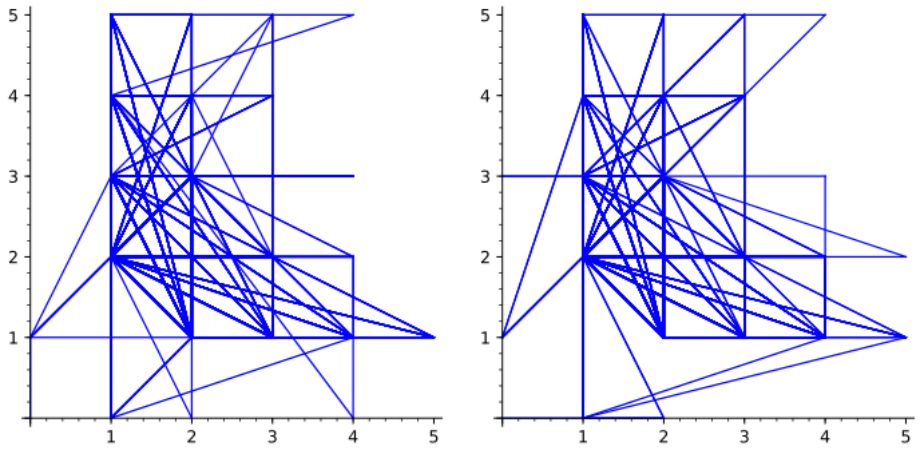
In order to perform a walk on M_{mn}, we need to define a probability function on the group. For this, it is sufficient to define a probability function to choose an $m \times n$ matrix A. In order to keep our walk from being too complex, we will construct a probability function that only has support on matrices of rank less than or equal to one. That is, the matrices represented by a matching with at most one rook. In order to choose such a matrix A, we will randomly choose a column vector v in $\mathbb{F}_q^m$ and a row vector w in $\mathbb{F}_q^n$. Then we will take $A = vw$ to be their product as matrices. As a probability function, this method yields

$$\text{pr}(a) = \begin{cases} \frac{q^m+q^n-1}{q^{m+n}} & \text{if } \text{rank}(a) = 0 \\ \frac{q-1}{q^{m+n}} & \text{if } \text{rank}(a) = 1 \\ 0 & \text{else} \end{cases}.$$

Thus we have all the components necessary to perform a random walk on the matching supercharacter theory. First, we can visualize how frequently our walk lands on each superclass by performing a walk and scaling the size of the points so that the points we land on are larger. The following image is the result for a walk of length 100000 on 2×5 the matching supercharacter theory.



This image gives a nice view of the walk as a whole. We see that we rarely reach superclasses with only one 1. We also see that we spend the most time at the matching which corresponds to the point $(1, 2)$. Further, we can animate the walk using the code in Appendix A. The following images depict the path taken by two different random walks of length 10000 on the 2×5 matching supercharacter theory over $\mathbb{F}_5$.



Notice how the two walks have a very similar structure towards the center of the image, but vary much more towards the edges. Combining this information with our heat map diagram, we see that our walk tends to stay near matchings with two rooks towards the left of the rook diagram and only rarely moves to a diagram with only one rook or two rooks towards the right of the diagram.

Further research into this supercharacter theory could include analyzing the meaning of the character formula with respect to the point presentation of a matching. Because this supercharacter theory has an analog in other subgroups of $UT(n, \mathbb{F}_q)$ represented by placing rooks on non-rectangular chessboards, we could also explore how these supercharacter theories look under the point presentation.

4 Dyck Paths and their Supercharacter Theory

The Dyck path supercharacter theory was defined and first studied in [3]. Here, we will construct a probability function for the supercharacter theory as well as a coarser supercharacter theory, the Dagger Dyck path supercharacter theory, that more closely resembles the probability function we define.

4.1 Dyck Paths

A Dyck path is an object from combinatorics that consists of a series of up and down steps in the $2D$-plane that starts at $(0,0)$, ends at $(2n,0)$ and does not pass below the x-axis. More formally:

Definition 4.1.1. A *Dyck path of length* m is a sequence $(d_n)_{n=1}^{2m}$ of length $2m$ of elements from the set $\{1,-1\}$ such that for any $0 < n \leq 2m$

$$\sum_{i=1}^{n} d_i \geq 0.$$

Another important definition is that of a valley.

Definition 4.1.2. A Dyck path, $(d_n)_{n=1}^{2n}$, of length n has a *valley* at i if $d_i = -1$ and $d_{i+1} = 1$.

Now, this sequence can be turned into a path in the plane by starting at $(0,0)$ and connecting the points $(m, \sum_{i=1}^{m} d_i)$. In this case, a valley is a point where the path transitions from going down to going up, but we can make this definition more formal by looking at the sequence associated with the Dyck path. Alternatively, we can draw this path on an $n \times n$ unipotent uppertriangular matrix by drawing the matrix in a grid and calling the upper left hand corner of the matrix $(0,0)$ and letting positive entries in the Dyck path move right and negative entries move down.

Example 4.1.3. The Dyck path given by the sequence $\{1,1,-1,-1,1,1,-1,-1\}$ can be drawn on a matrix as follows.

$$\left\langle \begin{array}{|cc|cc} 1 & 0 & 2 & 0 \\ 0 & 1 & 3 & 2 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \hline \end{array} \right\rangle.$$

This creates a partition of the entries of the matrix above the diagonal into those above the path and those below it.

Our goal will be to draw such a Dyck path on a unipotent uppertriangular matrix, M, without recourse to the sequential definition and so that the lowest and farthest to the left of the nonzero non-diagonal entries are in the valleys of the Dyck path. To this end, we notice that we can describe a Dyck path of length n by describing its valleys and we construct the abstract notion of a valley as a partition of the horizon.

Definition 4.1.4. The *horizon of size* m is the set

$$\text{horiz}(m) = \{(i,j) \mid 1 \leq i,j \leq m, (m-i)+j \geq m\}.$$

The horizon represents the set of entries of a matrix above and including the diagonal. We can now define a valley, which corresponds to a partition of the horizon.

Definition 4.1.5. An element (i, j) in horiz(m) is a *potential valley* if $(m - i) + j \geq m + 1$.

Each potential valley, (i, j) corresponds to the partition of the horizon into the sky and the mountainside (mtns) given by

$$\begin{aligned} \text{sky}((i,j)) &= \{(k,l) \mid 1 \leq k, l \leq m, l \leq i, k \geq j\} \\ \text{mtns}((i,j)) &= \text{horiz}(m) \setminus \text{sky}((i,j)). \end{aligned}$$

See example 4.1.9 for a concrete demonstration of the horizon and of valleys.

We are now ready to define a Dyck path on a matrix.

Definition 4.1.6. A *Dyck path of D length m* is a set of potential valleys val(D) such that no two distinct (i, j) and (k, l) in val(D) satisfy $(i, j) \in \text{sky}((k, l))$.

The *sky of a Dyck path D* is the union of the skys of its valleys. That is

$$\text{sky}(D) = \bigcup_{(i,j) \in \text{val}(D)} \text{sky}((i,j)).$$

Next, the *mountainside* of D is everything in the horizon of M that is not in the sky of D. From De Morgan's law, this is equivalent to

$$\text{mtns}(D) = \bigcap_{(i,j) \in \text{val}(D)} \text{mtns}((i,j)).$$

Note that the empty set defines a valid Dyck path and corresponds to the situation where the entire horizon is in the mountainside.

Finally, we must define a peak of a Dyck path D.

Definition 4.1.7. A Dyck path D has a *peak* at (i, j) in mtns(D) if the only entry (k, l) with $l \leq i$ and $k \geq j$ that is in the mountainside is (i, j) itself.

With this definition, we can impose a Dyck path on a matrix. To do this, we have to define which entries of a matrix M in $UT(m, \mathbb{F}_q)$ are valleys.

Definition 4.1.8. A M in $UT(m, \mathbb{F}_q)$ has a *valley* at (i, j) if M_{ij} is the lowest left non-zero non-diagonal entry in its column and the leftmost non-zero non-diagonal entry in its row. We will denote the collection of all valleys of M by val(M).

Let D denote the Dyck path generated by val(M), then,

$$\begin{aligned} \text{sky}(M) &= \text{sky}(D) \\ \text{mtns}(M) &= \text{mtns}(D) \\ \text{peaks}(M) &= \text{peaks}(D). \end{aligned}$$

In order to illustrate these definitions, we provide the following example.

Example 4.1.9. Working over $\mathbb{F}_5$, let

$$M = \begin{pmatrix} 1 & 0 & 0 & 0 & 4 & 0 \\ 0 & 1 & 2 & 3 & 3 & 0 \\ 0 & 0 & 1 & 3 & 0 & 3 \\ 0 & 0 & 0 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

Then, we see we can partition the matrix as

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 4 & 0 \\ 0 & 1 & 2 & 3 & 3 & 0 \\ 0 & 0 & 1 & 3 & 0 & 3 \\ 0 & 0 & 0 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

where the red spaces are the mountainside and the blue spaces are the sky, while the dark blue spaces represent valleys and the dark red spaces represent peaks. Everything that is not colored is not a part of the horizon. We can see the Dyck path more clearly in the following diagram

$$\left\langle \begin{array}{|cc|cccc} 1 & 0 & 0 & 4 & 0 & 0 \\ 0 & 1 & 2 & 2 & 3 & 0 \\ \hline 0 & 0 & 1 & 3 & 0 & 3 \\ \hline 0 & 0 & 0 & 1 & 2 & 1 \\ \hline 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \hline \end{array} \right\rangle .$$

4.2 Dyck Path Supercharacter Theories

We will define two supercharacters on $UT(m, \mathbb{F}_q)$. The first will have its superclasses indexed by Dyck paths and the second will be a coarsening of the first where we glue together the superclasses represented by a Dyck path and its reflection.

We begin by constructing the supercharacter theory associated to Dyck paths as it was defined in [3].

Definition 4.2.1. The *Dyck path supercharacter theory on* $UT(m, \mathbb{F}_q)$ has superclasses given by the sets of matrices that are represented by the same Dyck path. That is M and N are in the same superclass if they have the same associated Dyck path. If λ is a Dyck path, then the supercharacter function associated to λ is given by

$$\frac{\chi^\lambda(M)}{\chi^\lambda(1)} = \begin{cases} \frac{1}{1-q}^{|\text{peaks}(\lambda) \cap \text{val}(M)|} & \text{if no valley of } M \text{ is in the mountainside of } \lambda \\ 0 & \text{otherwise} \end{cases}$$

where M is an element of $UT(m, \mathbb{F}_q)$. We will denote the set of superclasses by Cl and the set of supercharacters by Ch.

Note that, in the definition, we can compare the valleys of M to the peaks of λ even though one is a matrix and one is a Dyck path because they have the same underlying set. This also shows us that the supercharacters are constant on the superclasses as their output only depends on the Dyck path representing their input.

Constructing the second supercharacter theory will be more involved than the first as we will be gluing together the superclasses and supercharacters from the Dyck path supercharacter theory. The idea here is to combine the superclass represented by the Dyck path λ with its reflection.

Definition 4.2.2. If λ is a Dyck path of length m defined in terms of its valleys, then $\dagger\lambda$ is the Dyck path whose valleys are

$$\text{val}(\dagger\lambda) = \{(m - j + 1, m - i + 1) \mid (i, j) \in \text{val}(\lambda)\}.$$

In this case, if λ were a Dyck path that was drawn on a matrix, then $\dagger\lambda$ would be the reflection of λ across the antidiagonal. In order to apply this notion of reflection to matrices, we must define a similar function (represented by the same symbol) that acts on matrices rather than Dyck paths.

Definition 4.2.3. We define the dagger function, $\dagger$, as follows

$$\dagger : UT_n \to UT_n \quad \dagger(A) = w_0 \left(A^{-1}\right)^T w_0,$$

where w_0 is the antidiagonal matrix

$$w_0 = \begin{pmatrix} 0 & \cdots & 1 \\ \vdots & \iddots & \vdots \\ 1 & \cdots & 0 \end{pmatrix}$$

Note that because $w_0 w_0 = id$, we see $\dagger(\dagger(A)) = A$.

The dagger function has the effect of flipping a matrix along the antidiagonal with some modifications to the entries caused by the inversion. The following example demonstrates this effect.

Example 4.2.4. Working over $\mathbb{F}_5$, we see

$$\dagger \begin{pmatrix} 1 & 3 & 3 & 2 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 4 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Looking at the example, we see that the dagger function flips the corresponding Dyck path. In order to prove this fact, we prove the following theorem.

Theorem 4.2.5. *If A is an element of $UT(m, \mathbb{F}_q)$ that has a valley at (i, j), then $(m-j+1, m-i+1)$ is a valley of $\dagger(A)$.*

Proof. We start by proving the claim that if (i, j) is a valley of M and $M_{ij} = s$, then (i, j) is a valley of M^{-1} and $M^{-1}_{ij} = -s$. To do this, we begin by defining some notation. Let $x_{ab}(s)$ be the element of $UT(m, \mathbb{F}_q)$ that has ones along the diagonal, s in the ij entry and zeros everywhere else. Next, we make three observations. First, we notice that $x_{ab}(s)x_{ab}(t) = x_{ab}(s+t)$, and hence that the inverse of $x_{ab}(s)$ is $x_{ab}(-s)$. Next, we notice that

$$x_{ab}(s)x_{cd}(t)x_{ab}(s)^{-1}x_{cd}(t)^{-1} = \begin{cases} id_m & b \neq c \text{ or } a \neq d \\ x_{ad}(st) & b = c \\ x_{cb}(-st) & a = d \end{cases}.$$

Finally, we notice that we can write M as the finite product

$$M = \prod_{a<b\leq m} x_{ab}(M_{ab})$$

so long as we sort the product lexicographical in a and reverse lexicographical in b. That is we start with $x_{1m}(M_{1m})$ then $x_{2m}(M_{2m})$ and so on until we reach the bottom of the column. Then we multiply by $x_{2(m-1}(M_{1(m-1)})$ and repeat the process until we've listed all the entries.

Now, when we invert M and apply the first and third observations, we see

$$M^{-1} = \prod_{a<b\leq m} x_{ab}(-M_{ab})$$

with the product done in the reverse order. In order to determine the entries of M_{ij}^{-1}, we just have to get our product back in the order we described in the third observation using the commutator rules from our second observation and the rule for combining the x_{ab} from our first observation to deal with the side effects of reordering our product. The concern here is that when we pass one of the x_{ab} by another, we will get some $x_{abj}(t)$ as a biproduct that will modify the entry at M_{ij}^{-1} and thus have the potential to cancel it out and stop it from being a valley. We are also concerned that we will get an x_{ab} that will cause M^{-1} to have a non-zero non-diagonal entry below or to the right of the entry at (i, j). However, both of these are impossible as the result of passing one matrix by another in our product can only result in the identity or an x_{ab} whose non-zero non-diagonal entry is above or to the right of an existing entry. As M had a valley at (i, j), no non-zero non-diagonal entry existed that was below or to the left of M_{ij}. Therefore, $M_{ij}^{-1} = -s$ and M^{-1} has a valley at (i, j). This argument also show us that we cannot get any new valleys as that would require us to multiply by a matrix with a non-zero entry in the mountainside of our matrix. Thus, we have shown that M and M^{-1} have the same valleys.

Next, we claim that if A is in $UT(m, \mathbb{F}_q)$, then $w_0 A^T w_0$ is A flipped along the antidiagonal. That is

$$A_{ij} = \left(w_0 A^T w_0\right)_{m-j+1, m-i+1}.$$

To see this, notice that multiplication on the right by w_0 flips a matrix horizontally, and multiplication on the left results in flipping the matrix vertically. This is because w_0 represents the corresponding series of elementary column and row operations. Thus, flipping horizontally, then along the diagonal, then vertically corresponds to flipping along the antidiagonal.

Putting these two claims together, we have the proof of our theorem.

□

From this, we immediately see the following.

Corollary 4.2.6. *If λ is the Dyck path corresponding to M, then $\dagger\lambda$ is the Dyck path corresponding to $\dagger(M)$.*

This corollary also gives us that there are the same number of matrices that correspond to λ as there are matrices that correspond to $\dagger\lambda$ because $\dagger$, being its own inverse, gives a bijection between the two sets of matrices.

The last task we have to acomplish before we define our new supercharacter theory is to prove the following lemma that will be helpful in showing our new supercharcter theory is in fact a supercharacter theory.

Lemma 4.2.7. *If λ is a Dyck path representing a superclass and M is an element of $UT(m, \mathbb{F}_q)$, then*

$$\frac{\chi^\lambda(\dagger(M))}{\chi^\lambda(1)} = \frac{\chi^{\dagger\lambda}(M)}{\chi^{\dagger\lambda}(1)}$$

Proof. Looking at the character formula, we see

$$\frac{\chi^\lambda(\dagger(M))}{\chi^\lambda(1)} = \begin{cases} \frac{1}{1-q}^{|\text{peaks}(\lambda) \cap \text{val}(\dagger(M))|} & \text{if no valley of } \dagger(M) \text{ is in the mountainside of } \lambda \\ 0 & \text{otherwise} \end{cases}.$$

Because of the flipping effect of the dagger function, we see that if no valley of $\dagger(M)$ is in the interior of λ, then no valley of μ is in the interior of $\dagger\lambda$ and vice versa. Hence, if there is a valley of $\dagger(M)$ in the interior of λ, then

$$\frac{\chi^\lambda(\dagger(M))}{\chi^\lambda(1)} = 0 = \frac{\chi^{\dagger\lambda}(M)}{\chi^{\dagger\lambda}(1)}.$$

Now if no valley of $\dagger(M)$ is in the interior of λ, then

$$\begin{aligned}\frac{\chi^{\lambda}(\dagger(M))}{\chi^{\lambda}(1)} &= \frac{1}{1-q}^{|\text{peaks}(\lambda)\cap\text{val}(\dagger(M))|} \\ &= \frac{1}{1-q}^{|\text{peaks}(\dagger\lambda)\cap\text{val}(M)|} \\ &= \frac{\chi^{\dagger\lambda}(M)}{\chi^{\dagger\lambda}(1)}.\end{aligned}$$

□

We are now ready to define a new supercharacter theory on uppertriangular unipotent matrices.

Definition 4.2.8. The *Dagger Dyck Path supercharacter theory on* $UT(m, \mathbb{F}_q)$ has superclasses given by the unions of the sets $\lambda \cup \dagger\lambda$ where λ and $\dagger\lambda$ are Dyck paths representing sets of matrices. If M is an element of $UT(m, \mathbb{F}_q)$, then the supercharacter formulas for this supercharacter theory are given by

$$\frac{\chi^{\lambda\cup\dagger\lambda}(M)}{\chi^{\lambda\cup\dagger\lambda}(1)} = \frac{\chi^{\lambda}(M)}{2\chi^{\lambda}(1)} + \frac{\chi^{\dagger\lambda}(M)}{2\chi^{\dagger\lambda}(1)},$$

where

$$\frac{\chi^{\lambda}(M)}{\chi^{\lambda}(1)} \quad \text{and} \quad \frac{\chi^{\dagger\lambda}(M)}{\chi^{\dagger\lambda}(1)}$$

are the supercharacter formulas for the Dyck path supercharacter theory. We will denote the set of superclasses for this supercharacter theory by †Cl and the set of supercharacters by †Ch.

Theorem 4.2.9. *The Dagger Dyck Path supercharacter theory on* $UT(m, \mathbb{F}_q)$ *is a supercharacter theory.*

Proof. We must show that this is in fact a supercharacter theory. Criteria 1, 2, and 4 of definition 2.2.1 follow from the fact that Dyck paths define a valid supercharacter theory. Thus, we must show that our character functions are constant on conjugacy classes. Because the character function for Dyck paths is constant on the Dyck path superclasses, we only have to show that if g is represented by the Dyck path μ and h is represented by $\dagger\mu$, then

$$\frac{\chi^{\lambda\cup\dagger\lambda}(g)}{\chi^{\lambda\cup\dagger\lambda}(1)} = \frac{\chi^{\lambda\cup\dagger\lambda}(h)}{\chi^{\lambda\cup\dagger\lambda}(1)}$$

for each supercharacter. To see this, observe

$$\begin{aligned}\frac{\chi^{\lambda\cup\dagger\lambda}(g)}{\chi^{\lambda\cup\dagger\lambda}(1)} &= \frac{\chi^{\lambda}(g)}{2\chi^{\lambda}(1)} + \frac{\chi^{\dagger\lambda}(g)}{2\chi^{\dagger\lambda}(1)} \\ &= \frac{\chi^{\dagger\lambda}(\dagger(g))}{2\chi^{\dagger\lambda}(1)} + \frac{\chi^{\lambda}(\dagger(g))}{2\chi^{\lambda}(1)} \\ &= \frac{\chi^{\dagger\lambda}(h)}{2\chi^{\dagger\lambda}(1)} + \frac{\chi^{\lambda}(h)}{2\chi^{\lambda}(1)} \\ &= \frac{\chi^{\lambda\cup\dagger\lambda}(h)}{\chi^{\lambda\cup\dagger\lambda}(1)}.\end{aligned}$$

Thus, our supercharacters are constant on superclasses and so we have defined a valid supercharacter theory. □

4.3 Dyck Path Supercharacter Theory Walks

In order to construct a walk on the Dyck path and Dagger Dyck path supercharacter theories, we need to construct a probability function. As we discussed earlier, when developing a probability function, it is helpful to limit its support in order to make the walk less chaotic. In our case, we will limit the support of our probability function to the superclasses represented by Dyck Paths that only have one valley.

Our probability function on $UT(m, \mathbb{F}_q)$ will come from picking a random column vector in $\mathbb{F}_q^m$, choosing a point to split it, turning one part into a row vector, and then multiplying the two components together.

Example 4.3.1. Working over $\mathbb{F}_3^4$, say we chose to split

$$\begin{pmatrix} 0 \\ 1 \\ 1 \\ 2 \end{pmatrix}$$

at index 2, then we would get

$$\begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 1 & 2 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

However, if we split it at the 0 index, we would get

$$\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} 0 & 1 & 1 & 2 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

We will weight the probability of splitting at an index, i with $0 \leq i \leq m$, so that the probability is relative to $q^{\min(i,m-i)}$. That is, the probability of choosing index i is given by

$$\frac{q^{\min(i,m-i)}}{\sum_{k=0}^{m} q^{\min(k,m-k)}}.$$

This makes it more likely for us to choose superclasses whose valleys are towards the center and further allows us to distinguish our superclasses from each other.

Theorem 4.3.2. *Working with $m \times m$ matrices over $\mathbb{F}_q$ and using this process, the probability of choosing an element of a superclass represented by the Dyck path λ, is given by*

$$pr(\lambda) = \begin{cases} \frac{\sum_{k=0}^{m} q^{\min(k,m-k)}(q^k + q^{m-k} - 1)}{q^m \sum_{k=0}^{m} q^{\min(k,m-k)}} & \text{if } \lambda \text{ has no valleys} \\ \frac{(q-1)^2 q^i q^{m-j-1} \sum_{k=i}^{j-1} q^{\min(k,m-k)}}{q^m \sum_{k=0}^{m} q^{\min(k,m-k)}} & \text{if } \lambda \text{ has exactly one valley at } (i,j) \\ 0 & \text{otherwise} \end{cases}.$$

Proof. This process always yields a rank 1 or rank 0 matrix as it is formed by multiplying two matrices that are at most rank 1. So, it is impossible to choose a matrix with more than one valley.

Next, there are q^m vectors we can choose and $\sum_{k=0}^{m} q^{\min(k,m-k)}$ ways to choose a point to split at, so the total number of choices is given by

$$q^m \sum_{k=0}^{m} q^{\min(k,m-k)}.$$

Next, we count the number of ways our multiplication can result in the zero matrix. From our process, it is possible for any split to result in a multiplication that yields the zero matrix. The only thing that has to happen is for all the entries above or below the split to be zero. So, if we split at index k, in order to get the zero matrix, either all k entries above the split are zero or all $m-k$ entries below the split are zero. There are q^{m-k} vectors that satisfy the first condition and q^k vectors that satisfy the second. However, the zero vector satisfies both, so we have over counted it by 1. Thus the total number of ways of getting the zero matrix is

$$\sum_{k=0}^{m} q^{\min(k,m-k)}(q^k + q^{m-k} - 1).$$

So, the probability of choosing the zero matrix, that is the only matrix with no valleys, is

$$\frac{\sum_{k=0}^{m} q^{\min(k,m-k)}(q^k + q^{m-k} - 1)}{q^m \sum_{k=0}^{m} q^{\min(k,m-k)}}.$$

Next, we determine the probability of choosing a matrix with a valley at (i, j). For this we notice that we can get such a matrix by splitting at any point between i and $j-1$ so long as all the entries between these two indexes are zero. It is also necessary that the i^{th} entry and the j^{th} entry be nonzero. For any choice of a split between i and $j-1$, there are $(q-1)^2$ ways to choose the i^{th} and j^{th} entry. Then there are q^i ways of choosing the entries above the i^{th}, and q^{m-j-1} ways of choosing the entries below the j^{th}. All the entries in between are fixed. Thus the total number of ways of getting a matrix with a valley at (i, j) is

$$(q-1)^2 q^i q^{m-j-1} \sum_{k=i}^{j-1} q^{\min(k,m-k)}.$$

Thus, the probability of choosing such a matrix is

$$\frac{(q-1)^2 q^i q^{m-j-1} \sum_{k=i}^{j-1} q^{\min(k,m-k)}}{q^m \sum_{k=0}^{m} q^{\min(k,m-k)}}.$$

□

This probability function is cumbersome, but we can simplify it using the q version of a number.

Definition 4.3.3. The *q-version of an integer n* is given by

$$[n] = 1 + q + q^2 + \ldots + q^{n-1} = \frac{q^n - 1}{q - 1}.$$

In order to familiarize ourselves with this notation, we will prove a lemma that will be helpful later when we show this probability function is constant on superclasses.

Lemma 4.3.4. *If m is even and $i < \frac{m}{2} < j-1$, then*

$$\left[\frac{m}{2}-i+1\right]q^i + \left[j-\frac{m}{2}-1\right]q^{m-j+1} = \left[\frac{m}{2}-i\right]q^i + \left[j-\frac{m}{2}\right]q^{m-j+1}.$$

Proof. We apply the definition of the q-version of an integer to get the following equality.

$$\begin{aligned}
\left[\frac{m}{2}-i+1\right]q^i + \left[j-\frac{m}{2}-1\right]q^{m-j+1} &= \left(1+\ldots+q^{\frac{m}{2}-i}\right)q^i + \left(q^{j-\frac{m}{2}-2}+\ldots+1\right)q^{m-j+1} \\
&= \left(q^i+\ldots+q^{\frac{m}{2}-1}+q^{\frac{m}{2}}\right) + \left(q^{\frac{m}{2}-1}+\ldots+q^{m-j+1}\right) \\
&= \left(q^i+\ldots+q^{\frac{m}{2}-1}\right) + \left(q^{\frac{m}{2}}+q^{\frac{m}{2}-1}+\ldots+q^{m-j+1}\right) \\
&= \left(1+\ldots+q^{\frac{m}{2}-i-1}\right)q^i + \left(q^{j-\frac{m}{2}-1}+\ldots+1\right)q^{m-j+1} \\
&= \left[\frac{m}{2}-i\right]q^i + \left[j-\frac{m}{2}\right]q^{m-j+1}
\end{aligned}$$

□

Another useful expression for our simplification is the $q-binomial$

Definition 4.3.5. The q-binomial is given by

$$\begin{bmatrix} n \\ m \end{bmatrix} = \prod_{k=0}^{m-1} \frac{q^{n-k}-1}{q^{k+1}-1}.$$

Notice that if $m = 1$, we have

$$\begin{bmatrix} n \\ 1 \end{bmatrix} = [n].$$

Using this new notation, we are able to reduce our probability function to something more manageable.

Theorem 4.3.6. *Working with $m \times m$ matrices over $\mathbb{F}_q$ and using the process described, the probability of choosing an element of a superclass represented by the Dyck path λ is given by*

$$pr(\lambda) = \begin{cases}
\frac{q^{\frac{m}{2}-2\left[\frac{m}{2}\right]+2\left(\left[\frac{m}{2}+1\right]-q\left[\frac{m}{2}\right]\right)}}{q^m\left(2\left[\frac{m}{2}\right]-q^{\frac{m}{2}}\right)} & \text{if } \lambda \text{ has no valleys} \\
\frac{(q-1)q^i q^{m-j-1}(q^j-q^i)}{q^m\left(2\left[\frac{m}{2}\right]-q^{\frac{m}{2}}\right)} & \text{if } \lambda \text{ has exactly one valley at } (i,j) \text{ and } i<j\leq\frac{m}{2} \\
\frac{(q-1)q^i q^{m-j-1}(q^{\frac{m}{2}}-q^i+q^{\frac{m}{2}+1}-q^{m-j+1})}{q^m\left(2\left[\frac{m}{2}\right]-q^{\frac{m}{2}}\right)} & \text{if } \lambda \text{ has exactly one valley at } (i,j) \text{ and } i<\frac{m}{2}<j \\
\frac{(q-1)q^i q^{m-j-1}(q^{m-i+1}-q^{m-j+1})}{q^m\left(2\left[\frac{m}{2}\right]-q^{\frac{m}{2}}\right)} & \text{if } \lambda \text{ has exactly one valley at } (i,j) \text{ and } \frac{m}{2}\leq i<j \\
0 & \text{otherwise}
\end{cases}$$

if m is even. If m is odd, the probability is given by

$$pr(\lambda) = \begin{cases} \frac{q^m - \left[\frac{m+1}{2}\right] + \left[\frac{m}{2}\right] - q^{\left[\frac{m+1}{2}\right]}}{q^{m\left[\frac{m+1}{2}\right]}} & \text{if } \lambda \text{ has no valleys} \\ \frac{(q-1)q^i q^{m-j-1}(q^j - q^i)}{q^{m\left(2\left[\frac{m+1}{2}\right]\right)}} & \text{if } \lambda \text{ has exactly one valley at } (i,j) \text{ and } i < j \leq \frac{m}{2} \\ \frac{(q-1)q^i q^{m-j-1}(q^{\frac{m+1}{2}} - q^i + q^{\frac{m+1}{2}} - q^{m-j+1})}{q^{m\left(2\left[\frac{m+1}{2}\right]\right)}} & \text{if } \lambda \text{ has exactly one valley at } (i,j) \text{ and } i < \frac{m+1}{2} < j \\ \frac{(q-1)q^i q^{m-j-1}(q^{m-i+1} - q^{m-j+1})}{q^{m\left(2\left[\frac{m+1}{2}\right]\right)}} & \text{if } \lambda \text{ has exactly one valley at } (i,j) \text{ and } \frac{m}{2} \leq i < j \\ 0 & \text{otherwise} \end{cases}.$$

A proof of these equalites can be found in Appendix C. Even though this presentation of the probability function looks more complicated, it makes it easier to see the distinction between a Dyck path of odd length and a Dyck path of even length. This new presentation also makes it easy to confirm that our probability function is symmetric. That is if a matrix represented by λ has the same probability of being chosen as a matrix represented by $\dagger\lambda$. This symmetry is why we defined the Dagger Dyck path supercharacter theory in the first place as we would like our probability functions to distinguish between superclasses. Thus, we define a probability function for the Dagger Dyck path supercharacter theory as

$$\dagger\text{pr}(\lambda \cup \dagger\lambda) = \begin{cases} \text{pr}(\lambda) & \text{if } \lambda = \dagger\lambda \\ \text{pr}(\lambda) + \text{pr}(\dagger\lambda) & \text{otherwise} \end{cases}.$$

Note that

$$\text{pr}(\lambda) + \text{pr}(\dagger\lambda) = 2\text{pr}(\lambda)$$

as the two probabilities are the same.

Using this probability function, we can animate the Dyck Path Walk using the code in Appendix B. From this animation, we learn a lot about the walk. We see that it tends towards the Dyck path that follows the diagonal, starting by removing a big chunk from the middle and then slowly diminishing the rest of the mountainside.

4.4 Eigenvalues for the Dyck Path Supercharacter Walk

We are now prepared to compute the eigenvalues for the Dyck path supercharacter theory. First, we recall our character formula

$$\frac{\chi^\lambda(\mu)}{\chi^\lambda(1)} = \begin{cases} \frac{1}{1-q}^{|\text{peaks}(\lambda) \cap \text{val}(\mu)|} & \text{if no valley of } \mu \text{ is in the interior of } \lambda \\ 0 & \text{otherwise} \end{cases}.$$

With this and the probability function computed in theorem 4.3.6, we are ready to compute the eigenvalue for a given supercharacter

$$|G|\langle \text{pr}, \frac{\chi^Y}{\chi^Y(1)} \rangle = \sum_{\mu \in \text{Ch}} \text{pr}(\mu) \frac{\chi^\lambda(\mu)}{\chi^\lambda(1)}$$

Because our probability function is 0 unless μ has zero or one valleys, we can rewrite our sum as

$$\mathrm{pr}(1)\frac{\chi^\lambda(1)}{\chi^\lambda(1)} + \sum_{\mu\in \mathrm{Cl}_1} \mathrm{pr}(\mu)\frac{\chi^\lambda(\mu)}{\chi^\lambda(1)} = \mathrm{pr}(1) + \sum_{\mu\in \mathrm{Cl}_1} \mathrm{pr}(\mu)\frac{\chi^\lambda(\mu)}{\chi^\lambda(1)}$$

where Cl_1 represents the set of superclasses in Cl with exactly one valley. Similarly, because μ can only have one valley, the eigenvalue can be written

$$\frac{\chi^\lambda(\mu)}{\chi^\lambda(1)} = \begin{cases} 1 & \text{if the valley of } \mu \text{ is in the sky of } \lambda \\ \frac{1}{1-q} & \text{if the valley of } \mu \text{ intersects a peak of } \lambda \\ 0 & \text{if the valley of } \mu \text{ is in the interior of } \lambda \end{cases}.$$

Now, if we say that $\mu \in \mathrm{sky}(\lambda)$ if μ has a valley in the sky of λ, that $\mu \in \mathrm{peaks}(\lambda)$ if μ has a valley at a peak of λ, and that $\mu \in \mathrm{mtns}(\lambda)$ if it has a valley in the interior of λ. Here mtns stands for mountainside and the $\mathrm{sky}(\lambda)$ includes the valleys of λ. These partitions allow us to write our eigenvalue formula as

$$\mathrm{pr}(1) + \sum_{\mu\in \mathrm{Cl}_1\cap \mathrm{sky}(\lambda)} \mathrm{pr}(\mu) + \sum_{\mu\in \mathrm{Cl}_1\cap \mathrm{peaks}(\lambda)} \frac{\mathrm{pr}(\mu)}{1-q}.$$

Now that we have a formula that gives us eigenvalues, we will compute a few examples.

Example 4.4.1. Let λ be the superclass represented by a Dyck path with exactly one valley in the upper right hand corner. That is a Dyck path with a valley at $(1, m)$. Then in order to compute the corresponding eigenvalue, we need to find all the superclasses in Cl_1 that are represented by a Dyck path that has a valley in the sky of λ and those that have a valley at one of the peaks of λ. The only Dyck path whose valley is in the sky of λ is λ itself. There are two Dyck paths with a valley that intersects one of the peaks of λ, the one which has a valley at $(1, m-1)$ and the one with a valley at $(2, m)$. If m is odd and greater than 1, we see our eigenvalue is given by

$$\begin{aligned} e_\lambda &= \frac{q^m - \left[\frac{m+1}{2}\right] + \sum_{k=0}^{\frac{m+1}{2}} q^{2k}}{q^m \left[\frac{m+1}{2}\right]} + \frac{(q-1)q^1q^{-1}\left(2q^{\frac{m+1}{2}} - 2q^1\right)}{2q^m \left[\frac{m+1}{2}\right]} \\ &\quad - \frac{q^1q^0\left(2q^{\frac{m+1}{2}} - q^2 - q^1\right)}{2q^m \left[\frac{m+1}{2}\right]} - \frac{q^2q^{-1}\left(2q^{\frac{m+1}{2}} - q^1 - q^2\right)}{2q^m \left[\frac{m+1}{2}\right]} \\ &= \frac{q^m - \left[\frac{m+1}{2}\right] + \sum_{k=0}^{\frac{m+1}{2}} q^{2k}}{q^m \left[\frac{m+1}{2}\right]} + \frac{(q-1)\left(q^{\frac{m+1}{2}} - q\right)}{q^m \left[\frac{m+1}{2}\right]} \\ &\quad - \frac{q\left(2q^{\frac{m+1}{2}} - q^1 - q^2\right)}{q^m \left[\frac{m+1}{2}\right]} \\ &= \frac{q^m - \left[\frac{m+1}{2}\right] + \sum_{k=0}^{\frac{m+1}{2}} q^{2k} - q^{\left[\frac{m+1}{2}\right]} - q^{\left[\frac{m+1}{2}\right]+1} + q + q^3}{q^m \left[\frac{m+1}{2}\right]} \\ &= \frac{q^m - \left[\frac{m+1}{2} + 2\right] + \sum_{k=0}^{\frac{m+1}{2}} q^{2k} + q + q^3}{q^m \left[\frac{m+1}{2}\right]} \end{aligned}$$

.

Example 4.4.2. Let λ be the Dyck path that runs along the diagonal with exactly one peak at $(1, 2)$. Then the corresponding eigenvalue is given by

$$\mathrm{pr}(1) + \sum_{\mu \in \mathrm{Cl}_1 \cap \mathrm{sky}(\lambda)} \mathrm{pr}(\mu) + \sum_{\mu \in \mathrm{Cl}_1 \cap \mathrm{peaks}(\lambda)} \frac{\mathrm{pr}(\mu)}{1-q}.$$

Notice how every element in Cl_1 has its valley in the sky of λ except the one whose valley intersects the peak of λ. Call this element α. Adding and subtracting the probability for α from our sum, we see we have all the elements of our probability function, which sum to 1. If m is odd we get

$$\begin{aligned} e_\lambda &= \mathrm{pr}(1) + \mathrm{pr}(\alpha) - \mathrm{pr}(\alpha) \\ &\quad + \sum_{\mu \in \mathrm{Cl}_1 \cap \mathrm{sky}(\lambda)} \mathrm{pr}(\mu) + \sum_{\mu \in \mathrm{Cl}_1 \cap \mathrm{peaks}(\lambda)} \frac{\mathrm{pr}(\mu)}{1-q} \\ &= 1 - \frac{(q-1)q^m \left(q^2 - q^1\right)}{2q^m \left[\frac{m+1}{2}\right]} - \frac{q^m \left(q^2 - q^1\right)}{2q^m \left[\frac{m+1}{2}\right]} \\ &= 1 - \frac{q^3 - q^2}{2\left[\frac{m+1}{2}\right]}. \end{aligned}$$

If m is even, then the calculation proceeds in the same manner, just with a slightly different denominator. In this case, the eigenvalue is

$$e_\lambda = 1 - \frac{q^3 - q^2}{2\left[\frac{m}{2}\right] - q^{\frac{m}{2}}}.$$

Conjecture 4.4.3. *The eigenvalue computed in example 4.4.2 is the largest eigenvalue not equal to* 1 *for the Dyck path supercharacter theory walk.*

References

[1] Persi Diaconis and I. M. Isaacs. Supercharacters and superclasses for algebra groups. *Trans. Amer. Math. Soc*, 360(5):2359–2392, 2008.

[2] Nathaniel Thiem. Notes on rook random walks. *Unpublished Notes*, 2018.

[3] Nathaniel Thiem. Supercharacter theories of type a unipotent radicals and unipotent polytopes. *Algebraic Combinatorics*, (1):23–45, 2018.

A Matching Walk Code

This code was written using the SageMath computer algebra system

```
# The functions in this worksheed are only concerned with rook fillings
#IE square fillings with limits of all 1
from sage.plot.line import Line

n = 5 #Number of columns
field = GF(3)

############################################################
### Constants
m = 2 #number of rows must be 2 for walk to work
MS = MatrixSpace(field ,m,n,sparse=False)
############################################################
### Helper Functions
def Cvr(row):
    #Flips the rows of our matrix so we can go up the rows instead of down
    return m-1-row

def GetLowerLeftSubmatrix(mat, row, column):
    return mat[Cvr(row):m,0:column+1]

def GetFillingValue(mat, row, column):
    #we don't do the conversion of the row here because we want the 0 row to be the
    bottom row and we did the conversion when filling out the matrix
    if column == 0 and row == 0:
        return GetLowerLeftSubmatrix(mat, row, column).rank()
    if column == 0:
        return GetLowerLeftSubmatrix(mat, row, column).rank() - GetLowerLeftSubmatrix
    (mat, row-1, column).rank()
    if row == 0:
        return GetLowerLeftSubmatrix(mat, row, column).rank() - GetLowerLeftSubmatrix
    (mat, row, column-1).rank()

    return GetLowerLeftSubmatrix(mat, row, column).rank() - GetLowerLeftSubmatrix(mat
    , row-1, column).rank() - GetLowerLeftSubmatrix(mat, row, column-1).rank() +
    GetLowerLeftSubmatrix(mat, row-1, column-1).rank()

def GetFilling(mat):
    filling = matrix(m,n)
    for i in range(m):
        for j in range(n):
            #have convert to go up the matrix to match the language of the fillings
            filling[Cvr(i),j] = GetFillingValue(mat, i, j)
    return filling

def MakeUTBlockMatrix(mat):
    return block_matrix([ [1, mat], [0, 1] ])

def GetULBlock(blockMat):
    return blockMat[0:m,m:]

def fillingToPoint(mat):
    toReturn = ();
    for row in range(m):
        x = 0
        for col in range(n):
            if mat[Cvr(row)][col] == 1:
```

```
                x = col + 1
            toReturn += (x,)
        return toReturn

def GetRandomRankOne():
        col = MatrixSpace(field,m,1,sparse=False).random_element()
        row = MatrixSpace(field,1,n,sparse=False).random_element()
        return col * row

def MakeWalk(length):
        pointList = []
        current = MS.matrix(0)
        pointList.append(fillingToPoint(GetFilling(current)))
        for i in range(length):
            toAdd = GetRandomRankOne()
            current = current + toAdd
            nextPoint = fillingToPoint(GetFilling(current))
            pointList.append(nextPoint)

        return pointList

def DrawFramesForLine(li, ns, g, **options):
        resu = []
        x1, y1 = li[0]
        x2, y2 = li[1]
        dx = float(x2 - x1)/float(ns)
        dy = float(y2 - y1)/float(ns)
        for i in range(ns):
            xx = x1 + (i+1)*dx
            yy = y1 + (i+1)*dy
            resu.append(g + sage.plot.line.line([(x1,y1), (xx,yy)], **options))

        return resu

def DrawFramesForCircle(point, ns, g, **options):
        resu = []
        for i in range(ns):
            r = float(0.5)/float(i+1)
            resu.append(g + circle(point, r, **options))
        return resu

#Main function for animating walk
#The function is very slow because of the ammount of frames it has to render
def GetFramesForWalk(length):
        pointList = MakeWalk(length)
        outLines = [[ pointList[i], pointList[i+1] ] for i in range(len(pointList)-1)]

        g = Graphics()
        parts = []
        frames = []
        for x in outLines:
            if x[0] == x[1]:
                frames = frames + DrawFramesForCircle(x[0], 5, g, rgbcolor=(1,0,0))
            else:
                frames = frames + DrawFramesForLine(x, 5, g, color='red')
            g+=sage.plot.line.line(x)
            frames.append(g)
        return frames

animate(GetFramesForWalk(40), xmin = 0, xmax = n, ymin=0, ymax=n)
```

FillingWalk.py

B Dyck Path Walk Code

This code was written using the SageMath computer algebra system

```
import random
import time
##########################
#Settings
#n is the size of the matrix
m = 8
#q is the size of the field, must be a prime power
q = 2

#Constants
field = GF(q)
MS = MatrixSpace(field,m,m,sparse=False)

##########################
#Helper Functions
def getFirstNonZero(row):
    for i in range(m):
        if row[0,i] != 0:
            return i
    return 0

def getMat():
    row = MatrixSpace(field,1,m,sparse=False).random_element()
    row[0,0] = 0
    col = MatrixSpace(field,m,1,sparse=False).random_element()
    for i in range(getFirstNonZero(row), m):
        col[i,0] = 0
    return col*row

def getSplitPoint():
    count = 0
    for i in range(m+1):
        count+= q**min(i,m-i)

    randPoint = random.randint(0, count-1)
    count = 0
    for i in range(m+1):
        count += q**min(i,m-i)
        if randPoint < count:
            return i
    return m

def getRandomOneValley():
    vect = MatrixSpace(field,m,1,sparse=False).random_element()
    split = getSplitPoint()
    row = matrix(field,1,m)
    col = matrix(field,m,1)

    for i in range(split):
        col[i,0] = vect[i,0]
    for i in range(split, m):
```

```
        row[0,i] = vect[i,0]

    return col*row

def getSpot(mat):
    for i in range(m):
        for j in range(m):
            if mat[m-i-1,j] != 0:
                toReturn = matrix(QQ,m,m,{(m-i-1,j): 1})
                return toReturn
    return matrix(QQ,m,m)

def CalculateDenom():
    count = 0
    for i in range(m+1):
        count+= q**min(i,m-i)
    count *= q**m
    return count

def calculateProbability(i,j):
    count = 0
    for k in range(i, j):
        count += q**(min(k,m-k))
    count *= (q-1)**2 * q**(i) * q**(m-j-1)
    return count

def getLowNonZ(mat,i):
    for j in range(m):
        if mat[m-j-1,i] != 0:
            return m-j-1
    return -1

def MatToPath(mat):
    lowNonZ = [0]*m
    for i in range(m):
        lowNonZ[i] = getLowNonZ(mat,i)
    dp = [0]*(2*m)
    location = 0
    currentHeight = -1
    for k in range(0,m):
        if lowNonZ[k] > currentHeight:
            location += lowNonZ[k] - currentHeight
            currentHeight = lowNonZ[k]
        dp[location] = 1
        location+=1
    return dp

#Necessary to multiply two Uppertriangular unipotent matrices
#as our functions do not return matrices with diagonals
def MultUtUp(m1, m2):
    idn = MS.identity_matrix()
    return (m1+idn)*(m2+idn) - idn

#Prints the Dyck path for each matrix in a random walk
#timePerFrame is in seconds
def DoWalk(length, timePerFrame):
    mat = MS.identity_matrix() - MS.identity_matrix()
    for i in range(length):
        print(i)
        p = MatToPath(mat)
```

```
            DyckWord(p).pretty_print()
            mat2 = getRandomOneValley()
            mat = MultUtUp(mat, mat2)
            time.sleep(timePerFrame)
            clear()
        print("done")

#Prints the probability of choosing a matrix with a valley at that location in the matrix
def PrintProbability():
    mat = MatrixSpace(RR,m,m, sparse=False).identity_matrix() - MatrixSpace(RR,m,m, sparse=False).identity_matrix()
    denom = CalculateDenom()
    total = 0
    for i in range(m):
        for j in range(m):
            prob = calculateProbability(i+1,j+1)
            mat[i,j] = 100*(prob / denom)
            total+=prob/denom
    print(mat.str(rep_mapping=lambda x : str(x.n(digits=3))) )
    print(round(100*total,3))

DoWalk(120, 0.25)
PrintProbability()
```

DyckPathWalk.py

C Probability Reduction

In order to reduce the probability function we break it up into the case where m is even and the case where m is odd. This is because when m is odd, the probability of splitting our vector achieves its maximum value twice, while when m is even, it only achieves the maximal value once. That is, when m is odd, there are two indexes where the probability of splitting there is $\frac{m+1}{2}$, while when m is even, there is only one index where the probability of splitting is $\frac{m}{2}$. In each case, we have to compute the denominator, the case where μ has no valleys, and the case where μ has a valley at i, j. If μ has a valley at (i, j), then it further reduces into three cases: $i < j \leq \bar{m}$, $i < \bar{m} < j$, $\bar{m} \leq i < j$, where $\bar{m}$ is $\frac{m}{2}$ if m is even and $\frac{m+1}{2}$ if m is odd.

We start with the case where m is odd.

- Denominator

$$q^m \sum_{k=0}^{m} q^{\min(k,m-k)} = q^m \left(\sum_{k=0}^{\frac{m+1}{2}} q^k + \sum_{k=\frac{m+1}{2}+1}^{m} q^{m-k} \right)$$
$$= 2q^m \left[\frac{m+1}{2} \right]$$

- μ has no valleys

$$\begin{aligned}\sum_{k=0}^{m} q^{\min(k,m-k)}\left(q^k+q^{m-k}-1\right) &= \sum_{k=0}^{\frac{m+1}{2}} q^k\left(q^k+q^{m-k}-1\right)+\sum_{k=\frac{m+1}{2}+1}^{m} q^{m-k}\left(q^k+q^{m-k}-1\right)\\ &= 2\sum_{k=0}^{\frac{m+1}{2}} q^k\left(q^k+q^{m-k}-1\right)\\ &= 2q^m-2\left[\frac{m+1}{2}\right]+2\sum_{k=0}^{\frac{m+1}{2}} q^{2k}\end{aligned}$$

- μ has a valley and $i<j\leq\frac{m+1}{2}$

$$\begin{aligned}(q-1)^2q^iq^{m-j-1}\sum_{k=i}^{j-1} q^{\min(k,m-k)} &= (q-1)^2q^iq^{m-j-1}\left[j-i\right]q^i\\ &= (q-1)q^iq^{m-j-1}\left(q^{j-i}-1\right)q^i\\ &= (q-1)q^iq^{m-j-1}\left(q^j-q^i\right)\end{aligned}$$

- μ has a valley and $i<\frac{m+1}{2}<j$

$$\begin{aligned}(q-1)^2q^iq^{m-j-1}\sum_{k=i}^{j-1} q^{\min(k,m-k)} &= (q-1)^2q^iq^{m-j-1}\left(\left[\frac{m+1}{2}-i\right]q^i+\left[j-\frac{m+1}{2}\right]q^{m-j+1}\right)\\ &= (q-1)q^iq^{m-j-1}\left(\left(q^{\frac{m+1}{2}-i}-1\right)q^i+\left(q^{j-\frac{m+1}{2}}-1\right)q^{m-j+1}\right)\\ &= (q-1)q^iq^{m-j-1}\left(q^{\frac{m+1}{2}}-q^i+q^{\frac{m+1}{2}}-q^{m-j+1}\right)\end{aligned}$$

- μ has a valley and $\frac{m+1}{2}\leq i<j$

$$\begin{aligned}(q-1)^2q^iq^{m-j-1}\sum_{k=i}^{j-1} q^{\min(k,m-k)} &= (q-1)^2q^iq^{m-j-1}\left[j-i\right]q^{m-j+1}\\ &= (q-1)q^iq^{m-j-1}\left(q^{j-i}-1\right)q^{m-j+1}\\ &= (q-1)q^iq^{m-j-1}\left(q^{m-i+1}-q^{m-j+1}\right)\end{aligned}$$

Putting these pieces together, we get the probability function described for odd m in Theorem 4.3.6.

Next, we address the case where m is even. Note that the calculation is the same for the numerator if $i<j\leq\frac{m}{2}$ and $\frac{m}{2}\leq i<j$ as it is in the corresponding case for an odd m. Because of this we omit these cases.

- Denominator

$$\begin{aligned} q^m \sum_{k=0}^{m} q^{\min(k,m-k)} &= q^m \left(\sum_{k=0}^{m} q^{\min(k,m-k)} + q^{\frac{m}{2}} - q^{\frac{m}{2}} \right) \\ &= q^m \left(\sum_{k=0}^{\frac{m}{2}} q^k + \sum_{k=\frac{m}{2}}^{m} q^{m-k} - q^{\frac{m}{2}} \right) \\ &= q^m \left(2 \left[\frac{m}{2} \right] - q^{\frac{m}{2}} \right) \end{aligned}$$

- μ has no valleys

$$\begin{aligned} \sum_{k=0}^{m} q^{\min(k,m-k)} \left(q^k + q^{m-k} - 1 \right) &= \sum_{k=0}^{\frac{m}{2}} q^k \left(q^k + q^{m-k} - 1 \right) + \sum_{k=\frac{m}{2}}^{m} q^{m-k} \left(q^k + q^{m-k} - 1 \right) \\ &\quad - q^{\frac{m}{2}} \left(q^{\frac{m}{2}} + q^{\frac{m}{2}} - 1 \right) \\ &= 2 \sum_{k=0}^{\frac{m+1}{2}} q^k \left(q^k + q^{m-k} - 1 \right) - q^{\frac{m}{2}} \left(q^{\frac{m}{2}} + q^{\frac{m}{2}} - 1 \right) \\ &= 2q^m - 2 \left[\frac{m}{2} \right] + 2 \sum_{k=0}^{\frac{m}{2}} q^{2k} - 2q^m + q^{\frac{m}{2}} \\ &= q^{\frac{m}{2}} - 2 \left[\frac{m}{2} \right] + 2 \sum_{k=0}^{\frac{m}{2}} q^{2k} \end{aligned}$$

- μ has a valley and $i < \frac{m}{2} < j$

$$\begin{aligned} (q-1)^2 q^i q^{m-j-1} \sum_{k=i}^{j-1} q^{\min(k,m-k)} &= (q-1)^2 q^i q^{m-j-1} \left(\left[\frac{m}{2} - i \right] q^i + \left[j - \frac{m}{2} \right] q^{m-j+1} \right) \\ &= (q-1) q^i q^{m-j-1} \left(\left(q^{\frac{m}{2}-i} - 1 \right) q^i + \left(q^{j-\frac{m}{2}} - 1 \right) q^{m-j+1} \right) \\ &= (q-1) q^i q^{m-j-1} \left(q^{\frac{m}{2}+1} - q^i + q^{\frac{m}{2}} - q^{m-j+1} \right) \end{aligned}$$

Finally, we can express

$$\sum_{k=0}^{n-1} q^{2k}$$

in terms of the q-binomial. This gives us the following equality

$$\begin{aligned}
\begin{bmatrix} n \\ 2 \end{bmatrix} - q \begin{bmatrix} n-1 \\ 2 \end{bmatrix} &= \frac{1}{(q-1)^2} \frac{(q^n-1)(q^{n-1}-1) - q(q^{n-1}-1)(q^{n-2}-1)}{[2]} \\
&= \frac{1}{(q-1)^2} \left(q^{2n-1} - q^{2n-2} - q + 1\right) \\
&= \frac{(q-1)(q^{2n-2}-1)}{(q-1)^2[2]} \\
&= \frac{\left((q^2)^{n-1}-1\right)}{q^2-1} \\
&= \sum_{k=0}^{n-1} q^{2k}.
\end{aligned}$$