Symmetric Functions from Stanley

February 6, 2018

1 Symmetric Functions in General

Definition. Λ^N is the space of symmetric functions, which are formal power series $f(x) = \sum_\alpha c_\alpha x^\alpha$ where each α is a "weka composition" of n . $\Lambda = \oplus_{N=1}^{\infty} \Lambda$.

2 Partitions and Their Orderings

Definition. $\mu \subset \lambda$ means that the Young diagrams are subsets of each other

Definition. If $|\mu| = |\lambda|$ then we define $\mu \leq \lambda$ to mean:

$$\mu_1 + \ldots + \mu_k \leq \lambda_1 + \ldots + \lambda_k \text{ for every } k \geq 1$$

3 Monomial Symmetric Functions

Definition. For λ a Young diagram, define:

$$m_\lambda = \sum_\alpha x^\alpha$$

Where the sum ranges over DISTINCT permutations $\alpha = (\alpha_1, \ldots)$ of the entries of the vector $\lambda = (\lambda_1, \ldots)$. Another way to think about this is that α is a weak composition of $|\lambda|$of the same "type" as λ...i.e. the same numbers appear. You can also write this as:

$$m_\lambda = \sum x_{i_1}^{\lambda_1} x_{i_2}^{\lambda_2} \cdots x_{i_n}^{\lambda_n}$$

Where the i's count over all the ordred possible n element subsets of $\mathbb{Z}$, $i_1 < i_2 < \ldots$

Example. Have:

$$\begin{aligned} m_\emptyset &= 1 \\ m_{(1)} &= \sum x_i \\ m_{(1,1)} &= \sum_{i<j} x_i x_j \end{aligned}$$

It is easy to see that these form a basis for Λ, since every $f = \sum c_\alpha x^\alpha \in \Lambda^N$ has $f = \sum_{|\lambda|=n} c_\lambda m_\lambda$ (its just a rearranging of the terms to group the weak compositions

4 Elementary Symmetric Functions

Definition. The elementary symmetric functions e_n are given by:

$$e_n = m_{1^n} = \sum_{i_1 < \ldots < i_n} x_{i_1} \cdot \ldots \cdot x_{i_n}, n \geq 1$$

For example:

$$\begin{aligned} e_1 &= m_{(1)} = \sum x_i \\ e_2 &= m_{(11)} = \sum_{i<j} x_i x_j \\ e_3 &= m_{(111)} = \sum_{i<j<k} x_i x_j x_k \end{aligned}$$

We also define the convention $e_\lambda = e_{\lambda_1} e_{\lambda_2} \cdots e_{\lambda_n}$ when $\lambda = (\lambda_1, \ldots)$

Definition. If $A = (a_{ij})_{i,j\geq 1}$ is an integer matrix with finitely many nonzero entries and with row and column sums:

$$\begin{aligned} r_i &= \sum_j a_{ij} \\ c_j &= \sum_i a_{ij} \end{aligned}$$

then define the **row-sum vector** $\mathrm{row}(A)$ and **column-sum vector** $\mathrm{col}(A)$ by:

$$\begin{aligned} row(A) &= (r_1, r_2, \ldots) \\ col(A) &= (c_1, c_2, \ldots) \end{aligned}$$

Proposition. *(7.4.1.) Let $|\lambda| = n$ and let $\alpha = (\alpha_1, \alpha_2, \ldots)$ be a weak composition of n. Since $\{m_\lambda\}_{\lambda\in\mathbb{Y}}$ is a basis, we may write:*

$$e_\lambda(x) = \sum_{|\mu|=n} M_{\lambda\mu} m_\mu(x)$$

for some coefficients $M_{\lambda\mu}$. The claim is that the coefficient of $M_{\lambda\alpha}$ is equal to the number of $(0,1)-$matrices A (i.e. matrices with entries from {0,1}) that have $row(A) = \lambda$ and $col(A) = \alpha$.

Proof. Consider the matrix:

$$X = \begin{bmatrix} x_1 & x_2 & x_3 & \ldots \\ x_1 & x_2 & x_3 & \ldots \\ \vdots & \vdots & \vdots & \end{bmatrix}$$

To obtain a term of $e_\lambda = e_{\lambda_1} e_{\lambda_2} \cdots e_{\lambda_n}$,choose λ_1 entries from the first row, λ_2 entries from the second row etc. As each e_{λ_k} is the product of all the possible choices of λ_k things from row k, we see that all the terms in e_λ arise in this way. The set of such choices of elements from the rows of X is naturally in bijection with (0,1)-matrices A with the convetniton that we put a 1 at A_{ij} if we choose the element X_{ij} and a 0 at A_{ij} if we do not choose the element X_{ij}. Notice that the term arising is x^α if and only if the matrix A has $row(A) = \lambda$ and $col(A) = \alpha$. This bijection and this observation complete the proof. □

Corollary. *(7.4.2) Let $M_{\lambda\mu}$ be the coefficients above. Then $M_{\lambda\mu} = M_{\mu\lambda}$. i.e. its a symmetric matrix*

Proof. Taking the transpose of every counted matrix gives the result. □

Proposition. *(7.4.3.) We have:*

$$\begin{aligned} \prod_{i,j}(1 + x_i y_j) &= \prod_{i,j}\left(x_i^0 y_j^0 + x_i^1 y_j^1\right) \\ &= \sum_{\lambda,\mu} M_{\lambda\mu} m_\lambda(x) m_\mu(y) \\ &= \sum_\lambda m_\lambda(x) e_\lambda(y) \end{aligned}$$

Proof. The basic idea is that for fixed weak compositions α, β the coefficient of $x_1^{\alpha_1} x_2^{\alpha_2} \ldots y_1^{\beta_1} y_2^{\beta_2} \ldots = x^\alpha y^\beta$ appearing in $\prod_{i,j}\left(x_i^0 y_j^0 + x_i^1 y_j^1\right)$ is equal to the number of $0-1$ matrices that have α_i 1's in row i and β_j 1's in column j. Indeed, each term $x_i^0 y_j^0 + x_i^1 y_j^1$ corresponds to the i-jth entry of the matrix, and we get a multiplicative contribution of x_i for each choice of 1 in row i and a multiplicative contribution of y_j for each 1 in column j. On the other hand, for a fixed α, β, the number of such matrices is exactly $M_{\lambda\mu}$by our previous combinatorial description. Finally, we have only to group all the coefficients that appear from weak compositions x^α when α all have the same shape λ to see how the $m_\lambda' s$ apear:

$$\begin{aligned} \prod_{i,j}(1 + x_i y_j) &= \sum_{\alpha,\beta} M_{\alpha\beta} x^\alpha y^\beta \\ &= \sum_{\lambda,\mu} M_{\lambda\mu} \left(\sum_{\alpha\sim\lambda} x^\alpha\right)\left(\sum_{\beta\sim\mu} y^\beta\right) \\ &= \sum_{\lambda,\mu} M_{\lambda\mu} m_\lambda(x) m_\mu(y) \end{aligned}$$

The last equality follows by the identity $e_\lambda(x) = \sum_{|\mu|=n} M_{\lambda\mu} m_\mu(x)$ from the previous proposition. □

Remark. This identity gives another proof that $M_{\lambda\mu}$ is symmetric; interchanging $x \leftrightarrow y$ gives the same thing.

Theorem. *(7.4.4.) If $|\lambda| = |\mu| = n$ then $M_{\lambda\mu} = 0$ unless $\mu \leq \lambda'$while $M_{\lambda\lambda'} = 1$. This shows that the matrix relating the basis $\{m_\lambda\}$ to the basis $\{e_\lambda\}$ is upper triangular with 1's on the diagonal and is hence invertable. Consequently, we see that $\{e_\lambda\}$ is a basis for Λ.*

Remark. Recall that $\lambda \leq \mu$ is the seemingly strange relation $\mu_1 + \ldots + \mu_k \leq \lambda_1 + \ldots + \lambda_k$ for every $k \geq 1$. We see here why this is important.

Proof. We will show $M_{\lambda\mu} \neq 0 \implies \mu \leq \lambda$. Suppose that $M_{\lambda\mu} \neq 0$ by our combinatorial interpretation, there is at least one 0-1 matrix A with row sums λ and column sums μ. Let A' be the matrix that looks like A except all the 1-s have been left justified (i.e. we slide the 1's in each row to the left until we have a block of 1's ....this makes the collection of 1's look like a Young diagram) Because we haven't changed the row sums by doing this, we know $row(A') = \lambda$. Since the 1's in A' look like a Young diagram, $col(A') = \lambda'$. Now, doing our left justificiation operation, the only thing we did is move 1's to the left. Consequently, the number of 1's in the first k columns of A' is greater than or equal to the number of 1's in the first k columns of A. This is precisely the statemnet that for every k we have $\lambda'_k = col(A')_k \geq col(A_k) = \mu_k$ for each k. In other words, $\lambda \geq \mu$.

This argument also shows that there is precisely one (0,1) matrix A with column sums λ' and row sums λ, namely the left justied matrix above! Hence $M_{\lambda\lambda'} = 1$.

Finally, for any n look at the set of Young diagrams $Par(n) = \{\lambda : |\lambda| = n\}$. Enumerate these so that $\lambda^1 \leq \lambda^2 \leq \ldots \leq \lambda^{p(n)}$ and so that $\left(\lambda^{p(n)}\right)' \leq \ldots \leq (\lambda^1)'$ (Stanley claims its easy to see why you can do this...) In this basis the matrix $M_{\lambda\mu}$ is upper triangular with 1's on the diagonal and is hence invertable. □

5 Complete Homogeneous Symmetric Functions

Definition. The **complete homogeeneous symmetric function** $h_\lambda$are given by the formulas:

$$h_n = \sum_{|\lambda|=n} m_\lambda = \sum_{i_1 \leq \ldots \leq i_n} x_{i_1} \ldots x_{i_n}$$

We also define the convention:

$$h_\lambda = h_{\lambda_1} h_{\lambda_2} \ldots h_{\lambda_n}$$

Proposition. *(7.5.1.) Let $|\lambda| = n$ and write:*

$$h_\lambda = \sum_{|\mu|=n} N_{\lambda\mu} m_\mu$$

Then $N_{\lambda\mu}$ is equal to the number of $\mathbb{N}-$matrices (i.e. matrices with entries from $\mathbb{N}$) that have row sums $row(A) = \lambda$ and column sums $col(A) = \mu$.

Proof. This is exactly analagous to proposition 7.4.1. □

Proposition. *(7.5.2.) Let $N_{\lambda\mu}$ be given by (7.7.) then $N_{\lambda\mu} = N_{\mu\lambda}$ i.e. the transition matrix between $\{m_\lambda\}$ and $\{h_\lambda\}$ is symmetric.*

Remark. Suppose we plug in $x_1 = x_2 = \ldots = x_n = 1$ and $x_{n+1} = x_{n+2} = \ldots = 0$. Denote this by $f(1^n)$. Notice then that:

$$\begin{aligned} e_k(1^n) &= \sum_{i_1 < \ldots < i_n \leq n} 1 = \binom{n}{k} \\ h_k(1^n) &= \sum_{i_1 \leq \ldots \leq i_n \leq n} 1 = \left(\binom{n}{k}\right) \end{aligned}$$

Where $\left(\binom{n}{k}\right)$ is the mutlichoose, the number of ways you can create an ice cream cone with k scoops from n flavors with multiple flavors allowed. (By the "star and stripes" way of looking at it, $\left(\binom{n}{k}\right) = \binom{n+k-1}{k}$)

Proposition. *(7.5.3.) We have:*

$$\begin{aligned} \prod_{i,j} (1 - x_i y_j) &= \prod_{i,j} \left(x_i^0 y_j^0 + x_i^1 y_j^1 + x_i^2 y_j^2 + \ldots\right) \\ &= \sum_{\lambda,\mu} N_{\lambda\mu} m_\lambda(x) m_\mu(y) \\ &= \sum_{\lambda} m_\lambda(x) h_\lambda(y) \end{aligned}$$

Proof. The proof is exactly analagous to the proof of Proposition 7.4.3. with $\mathbb{N}$ matrices instead of $0-1$ matrices. □

Remark. You might expect the next thing to be a theorem showing that h_λ is indeed a basis for Λ. However, this is not as starightforward as before. Instead we will define a certain map that takes $h'_\lambda s$ to e_λ's that will make the result apparent.

6 An Involution

Definition. An algebra homomorphism on Λ is defined uniqeuly by its values on any basis. Define $\omega : \Lambda \to \Lambda$ by specifiying that $\omega(e_n) = h_n$ for all n. Since ω presevers multiplication, we must have that $\omega(h_\lambda) = e_\lambda$ too.

Theorem. *(7.6.1.) The homomorphism ω is an involution, i.e. $\omega^2 = Id$ or equivalently $\omega(h_n) = e_n$.*

Proof. Define, $H(t)$ and $E(t)$ which will be elements of Λ for each t:

$$\begin{aligned} H(t) &= \sum_{n\geq 0} h_n t^n \\ E(t) &= \sum_{n\geq 0} e_n t^n \end{aligned}$$

Its easy to verify using the definitions of e_n and h_n that:

$$\begin{aligned} H(t)(x) &= \prod_n \left(1 + x_n t + x_n t^2 + \ldots\right) = \prod_n (1 - x_n t)^{-1} \\ E(t)(x) &= \prod_n (1 + x_n t) \end{aligned}$$

Hence $H(t)E(-t) = 1$ and equating the coefficients on both sides of the formal power series gives us that:

$$0 = \sum_{i=0}^{n} (-1)^i e_i h_{n-i}$$

Conversly, if we have some u_i's with $\sum(-1)^i u_i h_{n-i} = 0$ then we could conclude that the generating function $U(t)$ for the u_i's has $H(t)U(-t) = 1$ and from there we know that U and E have the same generating function so it must be that $u_n = e_n$.

Apply ω to this identity to get that:

$$0 = \sum_{i=0}^{n} (-1)^i h_i \omega(h_{n-i})$$

And hence $\omega(h_n) = u_n = e_n$ for each n by the above discussion! □

Corollary. *(7.6.2.) The set $\{h_\lambda : |\lambda| = n\}$ is a basis for Λ^n.*

Proof. The endomorphism $\omega : \Lambda \to \Lambda$ defined by $\omega(e_n) = h_n$ is invertible (we know this since $\omega^2 = Id$ i.e. $\omega^{-1} = \omega$) and hence is an autmorophism of Λ. Since e_n is a basis (we can see that its matrix in the right basis is upper triangular with 1's on the diagonal....see Thm 7.4.4.) hence h_n is a basis too. □

7 Power sum Symmetric Functions

Definition. The **power sum symmetric functions** are defined by:

$$p_n = m_{(n)} = \sum_i x_i^n$$

With the convention:

$$p_\lambda = p_{\lambda_1} p_{\lambda_2} \ldots p_{\lambda_n} \text{ for } \lambda = (\lambda_1, \ldots \lambda_n)$$

Proposition. *Let $\lambda = (\lambda_1, \ldots, \lambda_\ell)$ with $|\lambda| = n$ and $\ell = \ell(\lambda)$. Write p_λ in the basis $\{m_\mu\}$ with coefficients $R_{\lambda\mu}$. I.e. define $R_{\lambda\mu}$ by the relation:*

$$p_\lambda = \sum_{|\mu|=n} R_{\lambda\mu} m_\mu$$

We now describe the coefficients $R_{\lambda\mu}$. For fixed λ, μ, let $k = \ell(\mu)$. Then $R_{\lambda\mu}$ is equal to the number of ordered partitions $\pi = (B_1, \ldots, B_k)$ of the set $[\ell] = \{1, 2, \ldots, \ell\}$ such that:

$$\mu_j = \sum_{i \in B_j} \lambda_i \;\; 1 \leq j \leq k$$

Remark. Here's a way to think about this. Imagine the Young diagrams are made out of blocks and are sitting in front of you. Treat each row of the Young diagram λ as being a collection of glued together boxes. The game is to distribute these rows into $k = \ell(\mu)$ groups so that the number of boxes in each group is equal to μ_j. Once you do this, you can then place all the boxes in the row next to each other and create the Young diagram μ out of the boxes.

For example, μ and λ must have the same total number of blocks for this to be even possible.

Proof. $R_{\lambda\mu}$ is the coefficient of $x^\mu = x_1^{\mu_1} \cdot x_2^{\mu_2} \ldots$ in the product $p_\lambda = \left(\sum x_i^{\lambda_i}\right)\left(\sum x_i^{\lambda_2}\right)\ldots$. (By symmetry this is the same as the coefficient in front of $x^{\pi(\mu)}$ for any permutaion π....this is the reason we can write it as a sum of monomial symmetric functions) To obtain the monomial x^μ in the expansion of this product, we choose a term $x_{i_j}^{\lambda_j}$from each factor $p_{\lambda_j} = \sum x_i^{\lambda_j}$ in such a way that the chosen factors give us a x^μ i.e. $\prod_j x_{i_j}^{\lambda_j} = x^\mu$. This happens only when for each k we have $x_k^{\mu_k} = \prod_{\{j:i_j=k\}} x_{i_j}^{\lambda_j}$ or in other words $\mu_k = \sum_{\{j:i_j=k\}} \lambda_j$ so this defines our partion. The whole thing works in reverse too....given such an ordered partion $B_1, \ldots B_k$ we know exactly how to choose the factors. □

Corollary. *(7.7.2) Let $R_{\lambda\mu}$ be as above. Then $R_{\lambda\mu} = 0$ unless $\lambda \leq \mu$ while the diagonal terms $R_{\lambda\lambda}$have:*

$$R_{\lambda\lambda} = \prod_i m_i(\lambda)!$$

Where $m_i(\lambda)$ is the number of rows of length i from λ. This shows that the $p_\lambda$are a basis for Λ.

Remark. Recall that $\lambda \leq \mu$ is the seemingly strange relation $\lambda_1 + \ldots + \lambda_k \leq \mu_1 + \ldots + \mu_k$ for every $k \geq 1$. We see here why this is important.

Proof. The diagonal formula $R_{\lambda\lambda}$ is clear because the only way to create the partition $B_1, \ldots, B_k$ is to permute the rows of the same length amonst themselves. (Any other arangement is impossible since if you don't place the longest row into the longest row then you would have to place it in a shorter row....in which it doesn't fit!).

We will show $R_{\lambda\mu} \neq 0 \implies \lambda \leq \mu$. We prove the inequalities $\lambda_1 + \ldots + \lambda_r \leq \mu_1 + \ldots + \mu_r$ for any r. If $R_{\lambda\mu} \neq 0$ we find a fixed arangment $B_1, \ldots, B_k$ with $\mu_j = \sum_{i \in B_j} \lambda_i$. (This arrangement B will be crucial in showing the inequalities) For a fixed r, conider the subset $B_{i_1}, \ldots B_{i_l}$which are the blocks that contiain all the rows $\lambda_1, \lambda_2, \ldots, \lambda_r$. The number of blocks i_l here is at most r. Have then:

$$\mu_{i_1} + \ldots + \mu_{i_l} = \sum_{i \in B_{i_1} \cup B_{i_2} \cup \ldots \cup B_{li}} \lambda_i \geq \lambda_1 + \ldots + \lambda_r$$

But $\mu_{i_1} + \ldots + \mu_{i_l} \leq \mu_1 + \ldots + \mu_r$ since the RHS is the r longest rows and the LHS consists of at most r rows from μ. So these two inequalities together give $\mu_1 + \ldots + \mu_r \geq \lambda_1 + \ldots + \lambda_r$ as desired.

We will now see how the involution ω acts on the functions p_λ. □

Definition. For a Young diagram $\lambda = 1^{m_1} 2^{m_2} \ldots$, define:

$$z_\lambda = 1^{m_1} m_1! 2^{m_2} m_2! \ldots$$

Remark. Recall that you can think of λ as the cycle type of a permutation. If π is a permutation of cycle type λ, then z_λ is the number of permutations μ that commute with π . (The set of such μ is called the centralizer Z_π) This is easiest to see using the fact that cycle structure plays nice with conjugation. if you fix one permutation of the cycle type, and then apply conjugation. If $\pi = (1,2,3)(4)(5)$ then $\sigma\pi\sigma^{-1} = (\sigma(1), \sigma(2), \sigma(3))(\sigma(4))(\sigma(5))$. Now ask yourself how many ways there are to choose a σ to get back to π....σ can swap cylces of the same length amonst themselves ($m_i!$ ways) or it can do rearrangements in each individual cycle of length i in i ways (for a total of i^{m_i} possibilities).

Remark. The number of cycles that are conjugate to a given cycle π of type λ is $n!/z_\lambda$, since the number of such cycles is equal to the number of cosets the centralizer cuts S_n into. Since two permutations are conjugate if and only if they have the same cycle type, we see that $n!/z_\lambda$ is exactly the number of permuations of cycle type λ.

Definition. Define the "sign homomorphism" by:

$$\epsilon_\lambda = (-1)^{m_2 + m_4 + \ldots} = (-1)^{n - \ell(\lambda)}$$

This is $+1$ if permutations π of cycle type $\lambda$are even permutations and -1 if they are odd permutations.

Proposition. *(7.7.4.) We have:*

$$\begin{aligned} \prod_{i,j} (1 - x_i y_j)^{-1} &= \exp\left(\sum_{n \geq 1} \frac{1}{n} p_n(x) p_n(y)\right) \\ &= \sum_\lambda \frac{1}{z_\lambda} p_\lambda(x) p_\lambda(y) \end{aligned}$$

And:

$$\begin{aligned} \prod_{i,j} (1 + x_i y_j) &= \exp\left(\sum_{n \geq 1} \frac{1}{n} (-1)^{n-1} p_n(x) p_n(y)\right) \\ &= \sum_\lambda \frac{1}{z_\lambda} \epsilon_\lambda p_\lambda(x) p_\lambda(y) \end{aligned}$$

Proof. Its a manipulation of power series:

$$\begin{aligned}
\log\left(\prod_{i,j}(1-x_iy_j)^{-1}\right) &= \sum_{i,j}\log\left[(1-x_iy_j)^{-1}\right] \\
&= -\sum_{i,j}\log\left[(1-x_iy_j)\right] \\
&= \sum_{i,j}\sum_{n\geq 1}\frac{1}{n}x_i^ny_j^n \\
&= \sum_{n\geq 1}\frac{1}{n}\left(\sum_i x_i^n\right)\left(\sum_j y_j^n\right) \\
&= \sum_{n\geq 1}\frac{1}{n}p_n(x)p_n(y)
\end{aligned}$$

The fact thatexp $\left(\sum_{n\geq 1}\frac{1}{n}p_n(x)p_n(y)\right) = \sum_\lambda \frac{1}{z_\lambda}p_\lambda(x)p_\lambda(y)$ is a fact about exponential generating functions and the interpretation of $n!z_\lambda^{-1}$as the number of permutations of cycle type λ.

The fact from the theory of exponential generating functions we use is as follows. If $f:\mathbb{N}\to K$ is given, and we define$h:\mathbb{N}\to K$ by first choosing any set S, and then let $h(n) = h(|[n]|) = \sum_{\pi\in S_n([n])} f(|C_1|)f(|C_2|)\dots f(|C_k|)$ where C_i's are the cycles of the permutation π, then we will have that the exponential generating $E_h(x) = \sum_{n\geq 1} h(n)\frac{x^n}{n!} = \exp\left(\sum_{n\geq 1} f(n)\frac{x^n}{n}\right)$. (See chapter 5 of Stanley for more on this)

In our case, we put $f(n) = p_n(y)$ so then $h(n) = \sum_{\pi\in S_n([n])} p_{|C_1|}(y)\dots p_{|C_k|}(y) = \sum_{\pi\in S_n[n]} p_\pi(y) = \sum_\lambda \frac{n!}{z_\lambda}p_\lambda(y)$.

The proof of the other formula is analagous. □

Proposition. *The previous proposition can be interpreted to show that:*

$$\omega p_\lambda = \epsilon_\lambda p_\lambda$$

Proposition. *We have:*

$$\begin{aligned}
h_n &= \sum_{|\lambda|=n} z_\lambda^{-1}p_\lambda \\
e_n &= \sum_{|\lambda|=n} \epsilon_\lambda z_\lambda^{-1}p_\lambda
\end{aligned}$$

8 Specializations

Definition. (7.8.1.) A **specialization** is a homomorphism $\varphi:\Lambda\to R$ where R is some commutative algebra with an identity. Ex. $R=\mathbb{R}$. We require $\varphi(1_\Lambda) = 1_R$.

Example. Substituting $x_1 = a_1, x_2 = a_2, \dots, x_n = a_n$ and $x_k = 0, k\geq n+1$ where a_i are real numbers (as opposed to the x_i's which are formal variables) maps the symmetric functions Λ to numbers. Or if you like you can map to Λ_n , the symmetric functions with only n variables.

Example. Another important specialization is the specialization $ps_q(f) = f(1,q,q^2,\dots)$

Example. The **exponential specialization** is the one parameter family of specializations defined by its action on the power sum symmetric functions $p_1\to t$ and $p_k\to 0$ for all $k\geq 2$. This is denotes by $ex_t(f)$.

Proposition. *We have:*

$$ex(f) = \sum_{n\geq 0}\left([x_1x_2\dots x_n]f\right)\frac{t^n}{n!}$$

Where $[x_1\dots x_n]f$ denotes the coefficient of this monomial in f.

Proof. Check that $ex(f)$ has $p_1\to t$ and $p_k\to 0$. □

Example. Have: $ex_t\left(\prod_i \frac{1}{1-x_i}\right) = e^t$ and $ex_t\left(\prod_{i<j}\frac{1}{1-x_ix_j}\right) = e^{\frac{t^2}{2}}$.

Remark. The exponential specialization arises in some sense as a limit of the "plugging in" specialization:

$$f\left((1-q)t, q(1-q)t, q^2(1-q)t, \dots\right)$$

In the limit $q \to 1$. Indeed the action of this pluging in on p_n is:

$$\begin{aligned} p_1 &\to \sum_k q^k(1-q)t = t \\ p_n &\to \sum_k \left(q^k(1-q)t\right)^n = t^n(1-q)^n \sum q^{kn} = t^n(1-q)^n \frac{1}{1-q^n} \to 0 \text{ as } q \to 1 \end{aligned}$$

Remark. Suppose you have a specialization ρ. If you define $\tilde{\rho}(f) = \rho(\omega f)$ then that is a specialization too!

Theorem. *The only Schur positivie specializations that take $\rho(s_\lambda) \geq 0$ for every λ are:*
1. *Plugging in non-negative numbers*
2. *First applying the involution ω, and then plugging in non-negative numbers.*
3. *The exponential specialization.*

Proof. I don't know the proof! This is stated as a fact in the Borodin integrable probability paper. □

9 A Scalar Product

We now define a scalar product $\langle \cdot, \cdot \rangle$ on Λ that plays nice with the bases of functions we have been looking at so far. It is defined by:

Definition. The scalar product is defined by:

$$\langle m_\lambda, h_\mu \rangle = \delta_{\lambda\mu}$$

Since this definition does not make it clear that its really an inner product (it has to be symmetric and postive definite) the first thing we do is prove these facts. We take advantage of the relations between the bases p_n, h_n, e_n we have been working on so far to do this!

Proposition. *(7.9.1.) The scalar product is symmetric $\langle f, g \rangle = \langle g, f \rangle$*

Proof. Can check by the coefficients $N_{\lambda\mu}$ that write $h_\lambda$in the $m_\mu$basis that:

$$\langle h_\lambda, h_\mu \rangle = \left\langle \sum_\nu N_{\lambda\nu} m_\nu, h_\mu \right\rangle = N_{\lambda\mu}$$

Since $N_{\lambda\mu}$ is symmetric (recall its combinatorial interpretation), we have $\langle h_\lambda, h_\mu \rangle = \langle h_\mu, h_\lambda \rangle$ as desired. □

Definition. Two bases $\{u_i\}$ and $\{v_j\}$ are called **dual bases** if $\langle u_i, v_j \rangle = \delta_{ij}$. By the definition of $\langle \cdot, \cdot \rangle$ here, we see that $\{m_\lambda\}$ and $\{h_\lambda\}$are dual bases for Λ.

Lemma. *(7.9.2.) $\{u_\lambda\}$ and $\{v_\lambda\}$ are dual bases for Λ under this inner product if and only if:*

$$\sum_\lambda u_\lambda(x) v_\lambda(y) = \prod_{i,j} (1 - x_i y_j)^{-1}$$

Proof. Write ,$m_\lambda = \sum_\rho \zeta_{\lambda\rho} u_\rho$ and $h_\mu = \sum_\nu \eta_{\mu\nu} v_\nu$ for coefficients ζ and η. By def'n of $\langle \cdot, \cdot \rangle$:

$$\delta_{\lambda\mu} = \langle m_\lambda, h_\mu \rangle = \sum_{\rho,\nu} \zeta_{\lambda\rho} \eta_{\mu\nu} \langle u_\rho, v_\nu \rangle$$

Hence u, v is a dual basis if and only if, $\zeta\eta = Id$ as matrices. But this is the case if and only if we have that $\prod_{i,j}(1 - x_i y_j)^{-1} = \sum_\lambda m_\lambda h_\lambda = \sum u_\rho v_\nu$. □

Proposition. *(7.9.3.) We have:*

$$\langle p_\lambda, p_\mu \rangle = z_\lambda \delta_{\lambda\mu}$$

Proof. By a previous proposition, and the preceding lemma, we see that $\{p_\lambda\}$ and $\left\{\frac{1}{z_\mu} p_\mu\right\}$ are dual bases. □

Remark. Notice that if you want to turn p_λ into an orthonormal basis, you have to normalize by $\sqrt{z_\lambda}$, which is a really nasty coefficient.

Corollary. *(7.9.4.) The scalar product $\langle \cdot, \cdot \rangle$ is positive definite i.e. $\langle f, f \rangle \geq 0$ for all $f \in \Lambda$ with equality iff $f = 0$.*

Proof. Write f in the basis p_λ as $f = \sum c_\lambda p_\lambda$ and then observe that:

$$\langle f, f \rangle = \sum c_\lambda^2 z_\lambda \geq 0$$

with equality iff $c_\lambda = 0$ for all λ that happens iff $f \equiv 0$ since p_λ is a basis. □

10 The Cominatorial Definition of the Schur Functions

In this section we will construct a fifth basis, whose elements s_λ are called the **Schur functions**. The definition is a lot weirder than that for the other basis we have seen so far, but these have very nice properties that we will see. For example, they will be an orthonormal basis for Λ with the inner product $\langle \cdot, \cdot \rangle$. (In fact this is one way they are obviously nicer that then p_λ's they are an o.n.b with integer coefficients...no $\sqrt{z_\lambda}$s floating around at all.) These have lots of connections to other brances of mathematics too.

Definition. A **semistandard Young tableux (SSYT)** of shape λ is a Young diagram where you draw a number in each box so that the numbers are:

-Weakly increasing in each row

-Strictly increasing in each column.

The Young diagram λ that the numbers are put in is called the **shape** of the Young tableux, $\lambda = \mathrm{sh}(T)$. The collection of numbers, in the sense of "how many 1's in there, how many 2's in the tableux etc. is called the **type** of the diagram.

Example. Here's one of shape $\lambda = (6, 5, 3, 3)$

$$T = \begin{array}{cccccc} 1 & 1 & 1 & 3 & 4 & 4 \\ 2 & 4 & 4 & 5 & 5 & \\ 5 & 5 & 7 & & & \\ 6 & 6 & 9 & & & \end{array}$$

This diagram is of type $(3, 1, 1, 4, 4, 1, 1, 0, 2)$ since there are 3 $1's$ appearing, 1 2 appearing etc.

Definition. For a Young tableau T of type $\mathrm{type}(T) = \alpha = (\alpha_1, \alpha_2, \ldots)$ one defines:

$$x^T := x_1^{\alpha_1(T)} \cdot x_2^{\alpha_2(T)} \cdot \ldots$$

In our above example:

$$x^T = x_1^3 x_2 x_3 x_4^4 x_5^4 x_6 x_7 x_9^2$$

Definition. A **semistandard Young tableux (SSYT) of skew shape** λ/μ is one where you fill in the shape left by the set difference of a Young diagram λ take away a Young diagram μ. For example, here is one of type $(6, 5, 3, 3)/(3, 1)$:

$$\begin{array}{cccccc} & & & 3 & 4 & 4 \\ & 1 & 4 & 7 & 7 & \\ 2 & 2 & 6 & & & \\ 3 & 8 & 8 & & & \end{array}$$

We similarly define the type for these skew Young tableuxs and x^T accordingly.

Definition. (7.10.1) Let λ/μ be a skew shape. The **skew Schur function** $s_{\lambda/\mu} = s_{\lambda/\mu}(x)$ of shape λ/μ in the variables $(x_1, x_2, \ldots)$ is the formal power series:

$$s_{\lambda/\mu}(x) = \sum_T x^T$$

where the sum ranges over all SSYT's of skew shape λ/μ. If $\mu = \emptyset$ then we refer to this one as s_λ.

Theorem. *(7.10.2.) For any skew shape λ/μ, the skew Schur function $s_{\lambda/\mu}$ is a symmetric function.*

Proof. It suffices to show that $s_{\lambda/\mu}$ is invarient under the operation of swapping to adjacent variables $x_k \leftrightarrow x_{k+1}$. (This is becuase such swaps generate the symmetric group). There is a combinatorial way to see that given any SSYT T it is possible to create another SSYT T' by rearranging a little bit so that the number of boxes with a k in T and the number of boxes with $k+1$ in T are swapped for T'. □

Definition. If $|\lambda| = n$ is a partition of n and α is weak composition of n (i.e. 0's are allowed in α, $\sum \alpha_i = n$, α is not increasing or anything special) Define $K_{\lambda\alpha}$ to be the number of SSYT's of shape λ and type α. These are called the **Kostka numbers**. By the definition of the Schur functions, we have:

$$s_\lambda = \sum_\alpha K_{\lambda\alpha} x^\alpha$$

Where α is summed over all weak compositions of n. By the fact that the Schur functions are actually symmetric, we have in fact:

$$s_\lambda = \sum_{|\mu|=n} K_{\lambda\mu} m_\mu$$

More generally, the skew Kostka numbers are defined as you would excpet $K_{\lambda/\backslash nu\mu,\alpha}$is the number of SSYTs of shape λ/ν and of type α. Have:

$$s_{\lambda/\nu} = \sum_{|\mu|=n} K_{\lambda/\nu,\mu} m_\mu$$

Remark. There is no simple formula for $K_{\lambda\mu}$ and it is likely that such a formula does not exist.

Definition. Let λ be a partition with $|\lambda| = n$. Define f^λ by:

$$f^\lambda := K_{\lambda,1^n}$$

This is the number of SSYT's of shape λ that have exactly the numbers $\{1, 2, \ldots, n\}$ in the boxes. Hence each number appears once, and so both the rows and columns are strictly increasing. Such a Young tableux is called a **standard Young Tableau (SYT)** of shape λ.

The number f^λ has several nice combinatorial interpretations:

Proposition. *(7.10.3.) f^λ counts the number of objects belows (we prove these by making bijections to SYT's)*

a) Chains of partitions: $\emptyset = \lambda^0 \subset \lambda^1 \subset \ldots \subset \lambda^n$ so that λ^{n+1} is obtained from λ^n by attaching a single extra box in an exposed corner.

(Pf: the SYT tells you the order to add the boxes in)

b Linear Extensions: I don't understand this one.....

c) Ballot sequences: Sequences of n possible votes in which n votes can be cast in an election for candidates $A_1, A_2, \ldots$ so that for all i A_i recieves λ_i votes and so on and so that A_i never trails A_{i+1} as the votes come in one by one.

(Pf: If the k-th voter votes for A_i put a k in the i-th row. Another way to see it is to biject to the chain of partitions from a): adding a box to row i counts as a vote for A_i)

d) Lattice Permutations: Sequences $a_1 \ldots a_n$ in which each i occurs λ_i times and such that in any left subsequnece $a_1 \ldots a_j$ the number of i's is at least as great as the number of $i+1$'s for all i. (These are called lattice permuations)

(Pf: These are clearly the same as ballot sequences)

e) Lattice Paths: n dimensional "up-right" paths from $(0, 0, \ldots 0)$ to $(\lambda_1, \lambda_2, \ldots \lambda_n)$ that stay in the region $x_1 \geq x_2 \geq \ldots \geq x_\ell \geq 0$.

(Pf: The lattice permutation tells you which order to take the steps in)

All five of these can be generalized to the skew case λ/μ for SYT of skew shape λ/μ. (E.g. Instead of starting at the empty set for the chains of partitions, one starts at the partition μ, the Ballot sequences start with some votes already cast, the lattice paths start at $(\mu_1, \ldots \mu_n)$ rather than $(0, 0, \ldots, 0)$.)

Here is one more combinatorial object that is bijective to **SSYT**'s.

Definition. A **Gelfand-Tsetlin pattern** aka **a complete branching** is a triangular array G of non-negative i.e.:

$$\begin{array}{c} a_{11}\ a_{12}\ \ldots\ a_{1n} \\ a_{22}\ \ldots\ a_{2n} \\ a_{33} \ldots a_{3n} \end{array}$$

We write these so that a_{22} sits one level down but in between a_{11} and a_{12}. More generally a_{ij} sits between $a_{i-1,j-1}$ and $a_{i-1,j}$. We also assert that the a_{ij}'s must satisfy the inequalities $a_{ij} \leq a_{i+1,j+1} \leq a_{i,j+1}$. In other words the rows are weakly increasing and each element lies weakly between its two parents above it. This explains the reason for writing it in the above pattern.

If you read each row backwards, since $a_{i,n} > a_{i,k-1} > \ldots > a_{i,i}$ you can think of each row as a partition $\lambda^i = (a_{i,n}, \ldots, a_{i,i})$. The condition on the "interlacing" means that the resulting Young diagrams are shriking $\lambda^1 \supset \lambda^2 \supset \ldots \supset \lambda^n$. You can make a SSYT from an INCREASING sequence of Young diagrams in the obvious way: if $\lambda_1 \subset \lambda_2 \subset \ldots \subset \lambda_n$ then put the number 1 in all the elements of λ_1, the number 2 in everything in $\lambda_2 \backslash \lambda_1$ etc. Hence if we just read the rows of the Gelfand-Tstelin pattern from bottom to top, and think of them as Young diagrams, we see that they are give us a SSYT.

There is a proposition here about "reverse SSYTs" where the rows are decreasing in rows and columns rather than increasing....I'm going to skip it for now.

Proposition. *(7.10.5) Suppse that μ and λ are partitions with $|\mu| = |\lambda|$ and $K_{\lambda\mu} \neq 0$. Then $\mu \leq \lambda$. Moreover, $K_{\lambda\lambda} = 1$*

Proof. Suppose $K_{\lambda\mu} \neq 0$. Then there is at least one SSYT of shape λ and type μ. We will show directly now that $\mu_1 + \ldots + \mu_k \leq \lambda_1 + \ldots + \lambda_k$. We first claim that any part $T_{ij} = k$ of the SSYT can only appear on rows $i \leq k$ i.e. before the k-th row reading from fattest row to narrowest. This is clear because SSYT are strictly decreasing in the columns, so if a k appears before the $k-$th row, we are in a jam at the top $1 \leq T_{1j} < T_{2j} < \ldots < T_{ij} = k$ is impossible for $i > k$. This shows that all the parts $\{1, 2, \ldots, k\}$ for all appear in the first k rows. The number of the parts $\{1, 2, \ldots, k\}$ in the SSYT is exactly $\mu_1 + \ldots + \mu_k$ since $\mu$is the type of the SSYT. The size of the first k rows is exactly $\lambda_1 + \ldots + \lambda_k$. Hence, since all these parts fit into these rows we must have $\mu_1 + \ldots + \mu_k \leq \lambda_1 + \ldots + \lambda_k$, as desired.

If $\mu = \lambda$ then we have equality at every stage. This means that the numbers $\{1, 2, \ldots, k\}$ not only fit into the first k rows, they actually take up ALL the room on the first k rows. By going inductivly starting at $k = 1$, we see that the i-th row can consist only of $i's$ in each box. Hence there is only 1 SSYT possible with $\mu = \lambda$. $\square$

Corollary. *(7.10.6.) The Schur functions s_λ with $|\lambda| = n$ for a basis for Λ^n . This shows that s_λ forms a basis for Λ. In fact, the transformation matrix $K_{\lambda\mu}$ which expresses the $s'_\lambda s$ in terms of the $m'_\mu s$ with respect to any linear ordering of Par(n) is lower triangular with 1's on the main diagonal.*

11 The RSK Algorithm

11.1 Some motivation

The RSK algorithm is a bijection between $\mathbb{N}$-matrices $A = (a_{ij})$ whose entries are non-negative numbers and pairs of SSYT (P, Q) with the following remarkable property:

$$\begin{aligned} type(P) &= col(A) \\ type(Q) &= row(A) \end{aligned}$$

Recall that $col(A)$ is the vector of column sums of A while $row(A)$ is the vector of row sums of A. The type of SSYT is the vector $(m_1, m_2, m_3, \ldots)$ where m_1 is the number of 1's in the SSYT, m_2 is the number of 2's in the SSYT etc.

This will be the key to relating the Schur functions (which are defined in terms of SSYTs and their types) to the other familys of symmetric functions (we have seen already that they have relationships to certain $\mathbb{N}$-matrices)

Remark. $\mathbb{N}-$matrices are naturally in bijection with so called **generalized permutations** which are written in two line notation:

$$\sigma = \begin{pmatrix} i_1 & i_2 & \ldots & i_m \\ j_1 & j_2 & \ldots & j_m \end{pmatrix}$$

Where the top line is weakly increasing $i_1 \leq \ldots \leq i_m$, and the bottom line is weakly increasing in blocks of equal top line...i.e. if $i_r = i_s$ and $r < s$ then $j_r \leq j_s$. The natural bijection is that the a_{ij} entry in the matrix A corresponds to the number of times the pair $\binom{i}{j}$ appears in the generalized permutation.

Notice that ORDINARY permutations are a type of generalized permutation with $i_1 = 1, i_2 = 2, \ldots$etc. and each element from $\{1, 2 \ldots, n\}$ appears excactly once on the list $\{j_1, j_2, \ldots, j_m\}$. This means that the corresponding matrix A has exactly one entry in each row and each column....it is a permutation matrix! The row sums and column sums are hence the vectors $(1, 1, 1, \ldots)$ and we see that the resulting SSYTs from the RSK algorithm will actually be SYT.

11.2 Definitions

Definition. The basic operation of the RSK algorithm consists of **row insertion** $P \leftarrow k$ of a postive integer k into a non-skew SSYT $P = (P_{ij})$ of type λ. The operation $P \leftarrow k$ is defined as follows. We always try to insert k into the topmost row λ_1 of P . If we can just add it on and increase λ_1 by one box, $\lambda_1 \to \lambda_1 + 1$ and still keep the "weakly increasing along rows" property of the SSYT then we do so. (This works if $k \geq P_{1\lambda_1}$.) If we ever manage to place something at the end of a row, we STOP and are done the insertion.

Otherwise, (if $k < P_{1\lambda_1}$) we will have to insert k somewhere in the middle of the first row. We insert it at the first point where we can do so legally, still obeying the "weakly increasing along rows" property. At this point, we SWAP the number in the box (which is some number $> k$) at this point with the number k. We are left with this extra number floating around which we try to insert into the next row. If it legally fits on the end we stop! Otherwise we find somewhere in the middle to swap it in and continue with the next row and so on.

Here is some brief pseudocode for the operation of $P \leftarrow k$:

```
#Intialization:
num_to_be_inserted = k
current_row = 1

#Main code:
LOOP:
        If num_to_be_inserted fits at the end of current_row:
                Increase length of current row by 1
                Put k in this new box
                STOP
        Else:
                Find the first spot that num_to_be_inserted fits.
                value_right_before_it_fits = the value to the right of the first spot it fits
                #This is always \geq num_to_be_inserted
                Replace the number in the box here with num_to_be_inserted
                num_to_be_inserted = value_right_before_it_fits
                current_row += 1
```

The RSK algorithm maps an $\mathbb{N}-$matrix A to a pair of SSYTs of the same shape P, Q. P is called the **insertion tableau** and Q is called the **recording tableu**. The RSK acts like each entry a_{ij} of the matrix A represents a_{ij} tokens at the location i and j, and the algorithm "eats" each token one at a time, placing the token into P, Q in some way, until there are none left. The algorithm starts at the top-left entry of A, and then it moves from left to right along the first row of A, then from left to right along the second row of A and so on. [It is therefore convenient to map A to the generalized permutation ω_A first...this just converts the matrix A into a "list of tokens" that we will encounter]. When the RSK algorithm eats a token, the COLUMN of the token gets INSERTED into the insertion tableux P, acording to $P \leftarrow j$ and the ROW of the token gets put into RECORDING tableu Q at the SAME location that the final entry of P was made at (so that P, Q always have the same shape). Here is the psuedo-code for $A \xrightarrow{RSK} (P, Q)$. For convenience, we

assume that A has already been converted to a generalized permutation $\binom{i_1 i_2 \ldots i_n}{j_1 j_2, \ldots j_n}$ where n is the total number of tokens. (This is just a list of all the coordinates of the tokens from A that one encounters in the order one encouters them):

Initialize $P(0) = \emptyset, Q(0) = \emptyset$.

For $t = 0$ to $n-1$:

$P(t+1) = P(t) \leftarrow j_{t+1}$

$Q(t+1) = Q(t) \cup \{i_{t+1}\}$ inserted at the location of the last operation done in the P diagram...so that P and Q always have the same shape.

The outpuyt is the final pair $(P(t), Q(t))$

Remark. It is clear from the algorithm that:

$$\begin{aligned} type(P) &= col(A) \\ type(Q) &= row(A) \end{aligned}$$

Since we insert the list of column coords of A into P one by one and we insert the list of row coords of A into Q one by one. One has to verify that indeed P and Q defined this way are still Young diagrams...I'm going to skip this step.

Theorem. *The RSK algorithm is a bijection between $\mathbb{N}-$matrices $A = (a_{ij})$ of finite support and pairs (P, Q) of SSYT of the same shape.*

Proof. It suffices to show that it has an inverse. First check the following property (this comes up in verifying that P, Q is a SSYT): "Equal elements of Q are inserted in left to right order: elements to the right were inserted after elements to the left". Also, since Q records the row-coordinates, which are increasing in the algorithm, larger numbers are always inserted after lower numbners. Using these two propeties, we can deduce the order the elements of Q were inserted. (e.g. the largest-right most element was inserted last. Other than the one just considered, the largest-right-most element was inserted second last etc.). This automatically gives us top line of the generalized permutation. Since the P's and Q's have the same shape at every step, this means we know the order that things were inserted into the P matrix as well. From here we can work backwords to figure out what was inserted at each step. If the inserting thing is on the top row, then thats what was inserted. If it is on the second row, then you know something was bumped from the first row. Figure out where the bumped element came from, and the thing in its spot is the element that had been inserted. If the the inserted thing is on the third row, figure out where it came from on the second row, then figure out where that came from on the first row. The thing in its spot is the element you want! Continuing in this way gives us the generalized permuation back. $\square$

Example. For $A = \begin{bmatrix} 1 & 0 & 2 \\ 0 & 2 & 0 \\ 1 & 1 & 0 \end{bmatrix}$ $\omega_A = \binom{1\ 1\ 1\ 2\ 2\ 3\ 3}{1\ 3\ 3\ 2\ 2\ 1\ 2}$ and we will have $P = \begin{array}{llll} 1 & 1 & 2 & 2 \\ 2 & 3 & & \\ 3 & & & \end{array}$, $Q = \begin{array}{llll} 1 & 1 & 1 & 3 \\ 2 & 2 & & \\ 3 & & & \end{array}$. Here is the step-by-step process of how $\omega_A \xrightarrow{RSK} (P, Q)$ and then how $(P, Q) \xrightarrow{RSK} \omega_A$

<u>Encoding $\omega_A \xrightarrow{RSK} (P, Q)$:</u>

The item inserted is bolded. The insertion path in the P tableux (the path of what is bumped) is underlined.

t	i_t, j_t	$P(t)$	$Q(t)$
1	$(1,1)$	$\underline{\mathbf{1}}$	$\mathbf{1}$
2	$(1,3)$	$1\ \underline{\mathbf{3}}$	$1\ \mathbf{1}$
3	$(1,3)$	$1\ 3\ \underline{\mathbf{3}}$	$1\ 1\ \mathbf{1}$
4	$(2,2)$	$1\ \underline{2}\ 3$ $\underline{\mathbf{3}}$	$1\ 1\ 1$ $\mathbf{2}$
5	$(2,2)$	$1\ 2\ \underline{2}$ $3\ \underline{\mathbf{3}}$	$1\ 1\ 1$ $2\ \mathbf{2}$
6	$(3,1)$	$1\ \underline{1}\ 2$ $\underline{2}\ 3$ $\underline{\mathbf{3}}$	$1\ 1\ 1$ $2\ 2$ $\mathbf{3}$
7	$(3,2)$	$1\ 1\ 2\ \underline{\mathbf{2}}$ $2\ 3$ 3	$1\ 1\ 1\ \mathbf{3}$ $2\ 2$ 3

<u>Decoding $(P, Q) \xrightarrow{RSK} \omega_A$:</u>

Step 1:

From reading the entries of Q in order sorted from highest to lowest and (amongst ties) from right to left, we know the order of the final thing that was placed. The last thing to get put into P follows the same order. Using subscripts to denote which step of the process they were added in, we can write:

$$P = \begin{array}{cccc} 1_1 & 1_2 & 2_3 & 2_7 \\ 2_4 & 3_5 & & \\ 3_6 & & & \end{array}, \quad Q = \begin{array}{cccc} 1_1 & 1_2 & 1_3 & 3_7 \\ 2_4 & 2_5 & & \\ 3_6 & & & \end{array}$$

Reading from the Q matrix, this automatically gives us the top row of the generalized permutation $(1, 1, 1, 2, 2, 3, 3)$.

Step 2:

We now work backwards on the P tableux starting with the things added in at the last stage and working our way backwards. The thing in bold is the thing that we know was the last opertation in the insertion (we read this information off from the Q tableux.) We work out the insertion path (underlined). The thing on the top row is the element we want. (has a hat)

t	$P(t)$	Logic	i_t
7	$\begin{array}{cccc} 1 & 1 & 2 & \widehat{\underline{\mathbf{2}}} \\ 2 & 3 & & \\ 3 & & & \end{array}$	In the top row, so the 2 in the top is i_t	2
6	$\begin{array}{cccc} 1 & \widehat{\underline{1}} & 2 & \\ \underline{2} & 3 & & \\ \underline{\mathbf{3}} & & & \end{array}$	In the third row, so the 3 in the third row was bumped from the second row. It must have been where the underlined 2 currently is, then bumped down. The 2 in the second row was bumped from the first row by the 1 there.	1
5	$\begin{array}{ccc} 1 & 2 & \hat{\underline{2}} \\ 3 & \underline{\mathbf{3}} & \end{array}$	The 3 must have been bumped from the first row to the second row. The underlined/hatted 2 on the first row must have been the element that bumped the 3 down.	2
4	$\begin{array}{ccc} 1 & \hat{\underline{2}} & 3 \\ \underline{\mathbf{3}} & & \end{array}$	The 3 was bumped from the first row to the second row, the hatted 2 must have been the element that bumped it down.	2
3	$\begin{array}{ccc} 1 & 3 & \hat{\underline{\mathbf{3}}} \end{array}$	On first row is easy.	3
2	$\begin{array}{cc} 1 & \hat{\underline{\mathbf{3}}} \end{array}$	On first row is easy.	3
1	$\begin{array}{c} \hat{\underline{\mathbf{1}}} \end{array}$	On first row is easy.	1

Remark. One thing that you can see from the RSK algorithm is that only the relative order of the entries is ever used....the insertion subroutine checks whether the entry to be inserted is $\geq$ or $\leq$ other entries but does not otherwise use the fact that numbers being entered are numbers. For this reason, there are some ways you can relabel the numbers in the generalized permutation being fed into the RSK algorithm that will simply lead to the same pair of SSYTs with some numbers relabeled (i.e the shape won't change, and the structure of "which number is placed where" won't change either). Here is an example of this called "standardization":

Lemma. *In the usual RSK algorithm, each token from the matrix A is thought of being labeled with a tuple (i, j) indicating which row and which column of the matrix A that it is in. The RSK algorithm then places the row-coords on the tokens into the Q matrix and the column-coords in the P matrix.*

Define the row-first-order for tokens from the matrix A to be the order one gets by the order they come in if you read starting on the first row from left to right, then the second row from left to right etc. Define the column-first-order for tokens from the matrix A by the order they come if you start in the first column and read from top to bottom, then second column from top to bottom etc. (If there is more than one token in a single row column, the go through them in any order being consistent with the choice for both the row ordering and the column ordering. In this way if two tokens are in the same spot, their row-order and column-order will both differ by 1.)

Here is an example where the tokens from the matrix A have been labeled by (r,c) for their row and column ordering:

$$A = \begin{bmatrix} 2 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 3 & 0 \end{bmatrix} \to \begin{bmatrix} \begin{cases}(1,1)\\(2,2)\end{cases} & & (3,8) \\ & (4,4) & (5,9) \\ (6,3) & \begin{cases}(7,5)\\(8,6)\\(9,7)\end{cases} & \end{bmatrix}$$

If one runs the RSK algorithm and instead of feeding in the coordinate (row,column) as usual for the tokens, one feeds in the tuples (row-first-ordering,column-first-ordering) then the same Young diagram is produced, in the sense that the same tokens are inserted in the same spots (of course now they are labeled differently).

Proof. If (i,j) are the coordinates of a token and (r,c) are the row-first and column-first orderings then for two token labeled with a subscript 1 and subscript 2 repsectivly, we have $i_1 < i_2 \iff r_1 < r_2$ and $j_1 < j_2 \iff c_1 < c_2$ (this is just because the row/column orderings look at the rows/columns in order). This means that all the comparisons that arise in the insertion part of the RSK algorithm will be identical for both sets of tuples! (The only slightly subtle thing is how ties are handled due to the possibility of there being multiple tokens in one spot....this turns out to be ok because when the RSK algorithm essentially treats any ties as if the thing b □

Remark. Notice that the generlazied permutation corresponding to these row-orderings and column-orderings is actually just an ordinary permutation (i.e its a legit element from S_n)....for instance its easy to see that the top row is $(1\ 2\ \dots\ n)$.

If you plot this "standardized version" of a generalized permuation, it "looks like" the generalized permutation in some sense...especially with respect to the relative orderings of poitns. For example:

$$\begin{bmatrix} 1 & . & 2 \\ . & 2 & . \\ 1 & 1 & . \end{bmatrix} \to \begin{bmatrix} 1 & . & . & . & . & . & . \\ . & . & . & . & . & 1 & . \\ . & . & . & . & . & . & 1 \\ . & . & 1 & . & . & . & . \\ . & . & . & 1 & . & . & . \\ . & 1 & . & . & . & . & . \\ . & . & . & . & 1 & . & . \end{bmatrix}$$

Blocks of a given size in the original matrix get mapped to diagonal blocks of the form $\begin{bmatrix} 1 & & \\ & \ddots & \\ & & 1 \end{bmatrix}$ in the new matrix. Things that are in the same row or column are shifted over by a row so that there is only 1 thing left in each row or column and so that you can still travel on "down-right" between the same points.

12 Some Consequences of the RSK Algorithm

One immediate consequence is the important **Cauchy identity** for the Schur functions:

Theorem. *(7.12.1) We have:*

$$\prod_{i,j}(1-x_iy_j)^{-1} = \sum_{\lambda} s_\lambda(x)s_\lambda(y)$$

Proof. As we have seen before, the coefficient in front of the monomial $x^\alpha y^\beta$ on the left hand side is equal to the number of $\mathbb{N}$ matrices with row-sum vector α and column-sum vectorβ.

On the other hand, the coefficient in front of the monomial $x^\alpha y^\beta$on the right hand side is equal to the the number of pairs of SSYT (P,Q) such that $type(P) = \alpha$ and $type(Q) = \beta$. (This is just from the definition of the Schur functions)

By the RSK algorith, since every $\mathbb{N}$ matrix is in bijection with a pair of SSYTs whose types are the same respectivly as the row and column vectors, these are the same! □

Corollary. *(7.12.2.) The Schur functions are an orthonormal basis for Λ. ie.e $\langle s_\lambda, s_\mu\rangle = \delta_{\lambda\mu}$ with the inner product defined earlier.*

Proof. We checked that v,u are dual bases iff $\sum v_k(x)u_k(y) = \prod(1-x_iy_j)^{-1}$. □

Corollary. *(7.12.3) For $|\mu| = n$ and $|\nu| = n$ we have:*

$$\sum_{|\lambda|=n} K_{\lambda\mu}K_{\lambda\nu} = N_{\nu\mu} = \langle h_\mu, h_\nu\rangle$$

Proof. The LHS is the number of pairs (P,Q) of SSYT of shape the same shape of type μ and type ν resp. The RHS is the number of A matrices with column-sums μ and row-sums ν. These are the same by the RSK bijection.

Alternativly, you can take the coefficients of $x^\mu y^\nu$ on both sides of the equality $\prod_{i,j}(1-x_iy_j)^{-1} = \sum_\lambda s_\lambda(x)s_\lambda(y)$ to see this is true. □

Corollary. *Have:*

$$h_\mu = \sum_\lambda K_{\lambda\mu} s_\lambda$$

Proof. Skip for now! □

Corollary. *Have:*

$$h_{1^n} = \sum_{|\lambda|=n} f^\lambda s_\lambda$$

Proof. Since $K_{\lambda,1^n}$ is the definition of f^λ. □

Corollary. *Have:*

$$\sum_{|\lambda|=n} (f^\lambda)^2 = n!$$

Proof. This can be seen directly from RSK since the LHS counts the number of pairs of Standard Young Tableuxs of the same size, while the right hand side counts the number of permuations of n elements. □

13 Symmetry of the RSK Algorithm

The P matrix and the Q matrix in the RSK algorithm seem to play very different roles...but actually they are symmetric from a certain point of view.

Theorem. *(7.13.1.) Let A be an $\mathbb{N}-$matrix and suppose that $A \xrightarrow{RSK} (P,Q)$. Then $A^T \xrightarrow{RSK} (Q,P)$.*

Corollary. *If $A = A^T$ the $A \xrightarrow{RSK} (P,P)$*

To prove this we first need some lemmas:

Definition. Suppose that A is an $\mathbb{N}$ matrix (or equivalently $w_A = \binom{u_1 \dots u_n}{v_1 \dots v_n}$) is a generalized permutation. We define the **inversion partial order on** A **or on** w_A by $\binom{a}{b} < \binom{c}{d}$ if $a < c$ and $b < d$. Equivalently, $\binom{a}{b} < \binom{c}{d}$ is there is an up right path from the token at (a,b) to the token at (c,d). We denote this partially ordered set $I(A)$

Remark. Notice the number of incomparable elements of $I(A)$ is the number of incersions of the permutation, hence the terminology "inversion poset".

Lemma. *The map $\varphi : I(A) \to I(A^T)$ by $\varphi\left(\binom{a}{b}\right) \to \binom{b}{a}$ is an isomorphism of posets. i.e. $x < y$ in A if and only if $\varphi(x) < \varphi(y)$ in A^T.*

Proof. Easy? □

Definition. Take $I = I(A)$. Define I_1 to be the set of minimal elements, i.e. the elements x for whcih there is no y with $y < x$. (This will be first "line" in the geometric RSK). Define I_2 to be the set which are minimal for $I - I_2$ (this will be the second "line" in the geometric RSK). Define $I_3, \dots I_4$ etc in this way. Notice that the elemets in each I_k are incomparable. (We call such sets of incomparable elements "anti-chains") We can hence label them in such a way that:

$$I_k = (u_{k1}, v_{k1}), (u_{k2}, v_{k2}), \dots, (u_{kn_k}, v_{kn_k})$$

So that the x-coordinates $u_{k1} < u_{k2} < \dots < u_{kn_k}$ are INCREASING and the y-coordinates $v_{k1} > v_{k2} > \dots > v_{kn_k}$ is DECREASING.

Lemma. *(7.13.4) Let $I_1, \dots I_d$ be the anti-chains above. Let $A \xrightarrow{RSK} (P,Q)$. Then the first row of P is:*

$$v_{1n_1}, v_{2n_2}, \dots, v_{dn_d}$$

(I..e it is the collection of the y-coords for the LAST element of each of the I_l's)
The first row of Q is $u_{11}, u_{21}, \dots, u_{d1}$. (i.e. it s the collection of x-coords for the FIRST element of each of the I_k's)

Proof. The proof follows by induction. Use the fact that something will bump somethign from the top row of P only if it is either 1) in the same antichain by later in the order of that antichain 2) in a later antichain. Things from Q are never bumped, and the row coordinate is only added to the end if the coresponding element is in a later antichain. □

13.1 Geometric Construction of the RSK algorithm

Suppose that we apply the RSK algorithm to a legit permutation matrix (i.e. not a generalized one). (we have seen how to reduce an $\mathbb{N}$ matrix to a permutation matrix by relabeling the entries in such a way so that the RSK algorithm doesn't change...so this can always be assumed.)

Definition. Suppose that A is permutation matrix. We think of the entries of A as either having a token in them or not. We will now recusivly define, for each coordinate (i,j) in A a Young diagram. This is called th **growth diagram** $\mathcal{G}_\omega$ for a permutation ω. The following picture summerizes the rules:

The claim is that the reading the sequence of Young diagrams on the top row will give the P tableux and the reading the sequence of Young diagrams on the rightmost column will give the Q tableux (recall that a tableux corresponds to an increasing sequence of Young diagrams)

I'm going to skip the proof of this for now.

14 The Dual RSK Algorithm

The dual RSK algorithm sends $(0,1)$-matrices to pairs P, Q of the same shape where Q is still a SSYT but instead of P being a SSYT, P^T is a SSYT. The algorithm is exactly the same as the normal RSK alogrihtm, except that when inserting into the P matrix, we bump down TIES. (i.e. in the normal RSK algorith, when we were comparing a new element to an older one, we counted the new one as "larger". This meant that we could add it to the end or rows in the case of ties. In the dual RSK algorithm, we count the new element as "smaller", so we can't add it to the end of rows in the case of ties...it bumps in that case. You can see right away that this makes every row strictly increasing.)

This is a bijection in the same way the RSK algorithm is:

This leads to the dual Cauchy identity:

Theorem. *(7.14.3)We have:*

$$\prod_{i,j}(1+x_iy_j) = \sum_{\lambda} s_{\lambda'}(x)s_{\lambda}(y)$$

Proof. The coefficient of $x^{\alpha}y^{\beta}$on the LHS is equal to the number of $(0,1)$ matrices A of row type α and column type β. The coefficient of $x^{\alpha}y^{\beta}$ on the RHS is equal to the number of pairs of SSYT $\tilde{P}$ and Q so that $shape(\tilde{P}^T) = shape(Q)$ and $type(P) = \alpha$ and $type(Q) = \beta$. By the dual RSK algorithm these are in bijection to each other! □

Lemma. *(7.14.4) Let ω_y be the involution we have been discussing applied to the y variables only. Then:*

$$\omega_y\left(\prod \frac{1}{1-x_iy_j}\right) = \prod(1+x_iy_j)$$

Proof. Have:

$$\begin{aligned}
\omega_y\left(\prod \frac{1}{1-x_iy_j}\right) &= \omega_y\left(\sum_{\lambda} m_{\lambda}(x)h_{\lambda}(y)\right) \\
&= \sum_{\lambda} m_{\lambda}(x)e_{\lambda}(y) \\
&= \prod(1+x_iy_j)
\end{aligned}$$

□

Theorem. *For every λ we have $\omega s_{\lambda} = s_{\lambda'}$*

Proof. Have:

$$\begin{aligned}
\sum_{\lambda} s_{\lambda}(x)s_{\lambda'}(y) &= \prod(1+x_iy_j) \\
&= \omega_y\left(\prod \frac{1}{1-x_iy_j}\right) \\
&= \omega_y\left(\sum_{\lambda} s_{\lambda}(x)s_{\lambda}(y)\right) \\
&= \sum_{\lambda} s_{\lambda}(x)\omega_y(s_{\lambda}(y))
\end{aligned}$$

So since s_{λ} is a basis, it must be that $\omega s_{\lambda} = s_{\lambda'}$ as desired. □

15 The Classical Definition of the Schur Functions

For $\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{N}^n$ define $x^{\alpha} = x_1^{\alpha_1}, \ldots x_n^{\alpha_n}$ in the usual way. For any permuation $\omega \in S_n$ define $\omega(x^{\alpha}) = x^{\alpha(\omega)} = x_1^{\alpha_{\omega(1)}} \ldots x_n^{\alpha_{\omega(n)}}$. Now define:

$$a_{\alpha} = \sum_{\omega \in S_n} \operatorname{sgn}(\omega) \cdot \omega(x^{\alpha})$$

By the permuation expansion, this is a determiniant:

$$a_{\alpha} = \det\left(x_i^{\alpha_j}\right)_{i,j=1}^{n}$$

Notice that if α has $\alpha_i = \alpha_j$ for any $i \neq j$ then this determinant is 0 since the matrix has a repeated column.

Suppose that λ is a partition now. Define $\delta = (n-1, n-2, \ldots, 0)$ now so that $\lambda + \delta$ is STRICTLY increasing, rather than just weakly increasing. Look at $a_{\lambda+\delta}$ now. If any $x_i = x_j$ for $i \neq j$ then this matrix has a repeated row so we know that it vanishes. For this reason we know that $a_{\delta} = \Delta(x) = \prod(x_i - x_j)$ the vandermonde determinant divides $a_{\alpha+\lambda}$. For this reason an object of interest is:

$$\frac{a_{\lambda+\delta}}{a_{\delta}}$$

Proposition. *We have for any n that:*

$$\frac{a_{\lambda+\delta}}{a_\delta}(x_1,\dots x_n) = \frac{\det\left(x_i^{\lambda_j+n-j}\right)_{i,j}}{\det\left(x_i^{n-j}\right)_{i,j}} = s_\lambda(x_1,\dots x_n)$$

Proof. Skip for now!

Actually lets skip the rest of this section too. □

16 The Jacobi-Trudi Identity

Theorem. *Let $\lambda = (\lambda_1,\dots,\lambda_n)$ and $\mu = (\mu_1,\dots,\mu_n)$ then:*

$$s_{\lambda/\mu} = \det\left(h_{\lambda_i-\mu_j-i+j}\right)_{1\le i,j\le n}$$

Proof. We will demonstrate that $s_{\lambda/\mu}$can be written as a sum of weights over a set of non-crossing paths π, $\sum_{\pi\in\Pi_{NC}}\prod w(\pi)$ where the weight of each path is given by $h_{\lambda_i-\mu_j-i+j}$. The LGV theorem then applies to give us the above determinantal formula for $s_{\lambda/\mu}$. (Remark its actually a REVERSE SSYT with decreassing columns and rows....)

Indeed, we set up a bijection from SSYT's of shape $\lambda\backslash\mu$ to noncrossing paths on the $\mathbb{Z}^2$ plane. Here is an example:

$$\begin{array}{cccccc} . & . & 4 & 2 & 2 & \\ 3 & 2 & 1 & 1 & . & \leftrightarrow \\ 1 & . & . & . & . & \end{array}$$

C:/Users/Nica/Downloads/LGV Jacobi Trudi.jpg

Each row of the SSYT corresponds to RIGHT-DOWN path in $\mathbb{Z}^2$ that starts at (λ_j+n-j,∞) and ends at $(\mu_j+n-j,0)$. Notice that number of horizontal steps in each path is $\lambda_j-\mu_j$ which is the widht of the j-th row in the skew tableux $\lambda\backslash\mu$. The weight of the path is the product of x^k at each location k where it takes a horizontal step. The fact that the paths are non-interesecting makes sure that the the diagram is SSYT (the fact that its a down-right path ensures that the columns are decreasing and the fact that the columns are decreasing because they dont interesect (slide the k-th path over by k to recover the columns of the SSYT).

Notice that weight of a path from λ_i+n-i to μ_j+n-j is given exactly by $h_{(\lambda_i+n-i)-(\mu_j+n-j)} = h_{\lambda_i-\mu_j-i+j}$ since you choose indices $i_1\le i_2\le\dots\le i_n$ to be the heights at which to take horizontal steps.

By the LGV theorem, the result follows! □

Corollary. *Have:*

$$s_{\lambda/\mu} = \det\left(e_{\lambda'_i-\mu'_j-i+j}\right)_{i,j=1}^n$$

Proof. the map ω take $s_{\lambda/\mu}$ to $s_{\lambda'/\mu'}$and so:

$$\begin{aligned} s_{\lambda/\mu} &= \omega\left(s_{\lambda'/\mu'}\right) \\ &= \omega\left(\det\left(h_{\lambda'_i-\mu'_j-i+j}\right)\right) \\ &= \det\left(\omega\left(h_{\lambda'_i-\mu'_j-i+j}\right)\right) \\ &= \det\left(e_{\lambda'_i-\mu'_j-i+j}\right)_{i,j=1}^n \end{aligned}$$

□

Lets apply the exponential specialization ex to the Jacobi-Trudi identity. We have that $ex(s_\lambda) = \sum[x_1,\dots x_n]s_\lambda\frac{t^n}{n!} = \sum|\{\#SSYT$ of typ $\frac{t^{|\lambda|}}{\lambda!}f^\lambda$ where $f^\lambda$is the number of standard Young tableuxs of shape λ.

On the other hand, $ex(h_m) = \sum[x_1\dots x_n]h_m\frac{t^n}{n!} = 1\frac{t^m}{m!} = \frac{t^m}{m!}$ we get:

$$\begin{aligned} ex\left(s_\lambda\right) &= \det\left(ex\left(h_{\lambda_i-i+j}\right)\right)_{i,j} \\ f^\lambda\frac{t^{|\lambda|}}{\lambda!} &= \det\left(\frac{t^{\lambda_i-i+j}}{(\lambda_i-i+j)!}\right) = t^{|\lambda|}\det\left(\frac{1}{(\lambda_i-i+j)!}\right) \end{aligned}$$