(Super)symmetric polynomials and the Pieri rule

Berta Hudak

Okinawa Institute of Science and Technology

March 2023

Motivation

Schur polynomials, denoted s_λ, are certain symmetric polynomials in n variables, indexed by partitions.
In representation theory, they are characters of irreducible represenations of the general linear group $\mathfrak{gl}_n$.
Schur polynomials form a basis for the space of all symmetric polynomials.

Motivation

Schur polynomials, denoted s_λ, are certain symmetric polynomials in n variables, indexed by partitions.
In representation theory, they are characters of irreducible represenations of the general linear group $\mathfrak{gl}_n$.
Schur polynomials form a basis for the space of all symmetric polynomials.

Any product of two Schur polynomials can be written as a sum of Schur polynomials with non-negative coefficients (Littlewood-Richardson rule):

$$s_\lambda s_\mu = \sum_\nu c^\nu_{\lambda,\mu} s_\nu.$$

Motivation

Schur polynomials, denoted s_λ, are certain symmetric polynomials in n variables, indexed by partitions.
In representation theory, they are characters of irreducible represenations of the general linear group $\mathfrak{gl}_n$.
Schur polynomials form a basis for the space of all symmetric polynomials.

Any product of two Schur polynomials can be written as a sum of Schur polynomials with non-negative coefficients (Littlewood-Richardson rule):

$$s_\lambda s_\mu = \sum_\nu c^\nu_{\lambda,\mu} s_\nu.$$

In this talk, we will present a special case of this multiplication when all coefficients are equal to 1.

Partitions and Young diagrams

A **partition** $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_m)$ of n is a non-increasing sequence of integers such that $\sum_{i=1}^{m} \lambda_i = n$. We write $\lambda \vdash n$ if λ is a partition of n.

Partitions and Young diagrams

A **partition** $\lambda = (\lambda_1, \lambda_2, \dots, \lambda_m)$ of n is a non-increasing sequence of integers such that $\sum_{i=1}^{m} \lambda_i = n$. We write $\lambda \vdash n$ if λ is a partition of n.

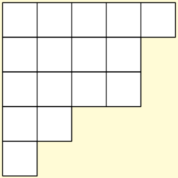
We can visualise partitions by drawing **Young diagrams**. A Young diagram corresponding to λ is a set of boxes such that in each row i we draw λ_i many boxes.

Partitions and Young diagrams

A **partition** $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_m)$ of n is a non-increasing sequence of integers such that $\sum_{i=1}^{m} \lambda_i = n$. We write $\lambda \vdash n$ if λ is a partition of n.

We can visualise partitions by drawing **Young diagrams**. A Young diagram corresponding to λ is a set of boxes such that in each row i we draw λ_i many boxes.

For example, take $n = 16$ and $\lambda = (5, 4, 4, 2, 1)$. The Young diagram is drawn below.



Tableaux

If we fill the boxes of the Young diagram with integers, we obtain a **tableau of shape** λ. We call a tableau semi-standard if entries weakly increase along the rows and strictly increase down the columns. If they also strictly increase along the rows, we say that the tableau is standard.

Tableaux

If we fill the boxes of the Young diagram with integers, we obtain a **tableau of shape** λ. We call a tableau semi-standard if entries weakly increase along the rows and strictly increase down the columns. If they also strictly increase along the rows, we say that the tableau is standard. Let $\lambda = (4, 3, 2, 2)$ and take

$$\mathtt{T}_1 = \begin{array}{|c|c|c|c|}\hline 2 & 3 & 1 & 4 \\ \hline 6 & 7 & 8 \\ \cline{1-3} 3 & 3 \\ \cline{1-2} 5 & 1 \\ \cline{1-2} \end{array} \qquad \mathtt{T}_2 = \begin{array}{|c|c|c|c|}\hline 1 & 2 & 3 & 6 \\ \hline 4 & 4 & 5 \\ \cline{1-3} 5 & 8 \\ \cline{1-2} 6 & 11 \\ \cline{1-2} \end{array} \qquad \mathtt{T}_3 = \begin{array}{|c|c|c|c|}\hline 1 & 2 & 3 & 4 \\ \hline 5 & 6 & 7 \\ \cline{1-3} 8 & 9 \\ \cline{1-2} 10 & 11 \\ \cline{1-2} \end{array}.$$

Then $\mathtt{T}_2$ is semi-standard and $\mathtt{T}_3$ is standard.

Row insertion

The algorithm **row insertion** takes a tableau $\mathtt{T}$ and inserts a positive integer x into it, resulting in a new tableau, denoted $\mathtt{T} \leftarrow x$.

1. If $x \geq i$ for all entries i in the first row of $\mathtt{T}$, add x to the end of the first row.
2. If not, find the leftmost i in the first row of $\mathtt{T}$ such that $i > x$.
3. Place x into the place of i and take i out of the tableau.
4. Repeat the process in the second row with i.
5. Keep going until the bumped entry can be placed at the end of the row it is bumped into or until it is bumped at the bottom, in which case it forms a new row of length 1.

Row insertion

The algorithm **row insertion** takes a tableau T and inserts a positive integer x into it, resulting in a new tableau, denoted $\mathtt{T} \leftarrow x$.

1. If $x \geq i$ for all entries i in the first row of T, add x to the end of the first row.
2. If not, find the leftmost i in the first row of T such that $i > x$.
3. Place x into the place of i and take i out of the tableau.
4. Repeat the process in the second row with i.
5. Keep going until the bumped entry can be placed at the end of the row it is bumped into or until it is bumped at the bottom, in which case it forms a new row of length 1.

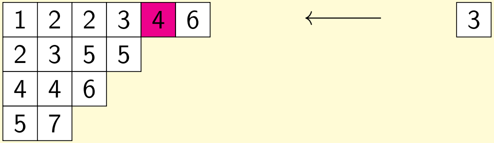
1	2	2	3	4	6
2	3	5	5		
4	4	6			
5	7				

$\longleftarrow$ 3

Row insertion

The algorithm **row insertion** takes a tableau T and inserts a positive integer x into it, resulting in a new tableau, denoted $\mathtt{T} \leftarrow x$.

1. If $x \geq i$ for all entries i in the first row of T, add x to the end of the first row.
2. If not, find the leftmost i in the first row of T such that $i > x$.
3. Place x into the place of i and take i out of the tableau.
4. Repeat the process in the second row with i.
5. Keep going until the bumped entry can be placed at the end of the row it is bumped into or until it is bumped at the bottom, in which case it forms a new row of length 1.



Row insertion

The algorithm **row insertion** takes a tableau T and inserts a positive integer x into it, resulting in a new tableau, denoted $\texttt{T} \leftarrow x$.

1. If $x \geq i$ for all entries i in the first row of T, add x to the end of the first row.
2. If not, find the leftmost i in the first row of T such that $i > x$.
3. Place x into the place of i and take i out of the tableau.
4. Repeat the process in the second row with i.
5. Keep going until the bumped entry can be placed at the end of the row it is bumped into or until it is bumped at the bottom, in which case it forms a new row of length 1.

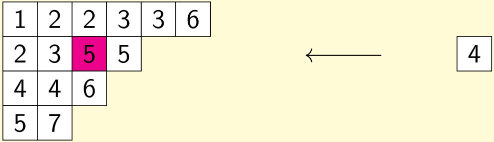
1	2	2	3	3	6
2	3	5	5		
4	4	6			
5	7				

$\longleftarrow$ 4

Row insertion

The algorithm **row insertion** takes a tableau T and inserts a positive integer x into it, resulting in a new tableau, denoted $\mathtt{T} \leftarrow x$.

1. If $x \geq i$ for all entries i in the first row of T, add x to the end of the first row.
2. If not, find the leftmost i in the first row of T such that $i > x$.
3. Place x into the place of i and take i out of the tableau.
4. Repeat the process in the second row with i.
5. Keep going until the bumped entry can be placed at the end of the row it is bumped into or until it is bumped at the bottom, in which case it forms a new row of length 1.



Row insertion

The algorithm **row insertion** takes a tableau T and inserts a positive integer x into it, resulting in a new tableau, denoted $\mathtt{T} \leftarrow x$.

1. If $x \geq i$ for all entries i in the first row of T, add x to the end of the first row.
2. If not, find the leftmost i in the first row of T such that $i > x$.
3. Place x into the place of i and take i out of the tableau.
4. Repeat the process in the second row with i.
5. Keep going until the bumped entry can be placed at the end of the row it is bumped into or until it is bumped at the bottom, in which case it forms a new row of length 1.

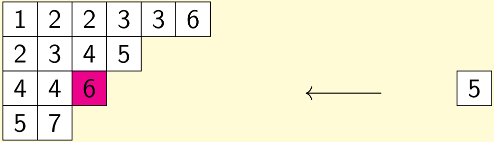
1	2	2	3	3	6
2	3	4	5		
4	4	6			
5	7				

$\longleftarrow$ 5

Row insertion

The algorithm **row insertion** takes a tableau T and inserts a positive integer x into it, resulting in a new tableau, denoted $\mathtt{T} \leftarrow x$.

1. If $x \geq i$ for all entries i in the first row of T, add x to the end of the first row.
2. If not, find the leftmost i in the first row of T such that $i > x$.
3. Place x into the place of i and take i out of the tableau.
4. Repeat the process in the second row with i.
5. Keep going until the bumped entry can be placed at the end of the row it is bumped into or until it is bumped at the bottom, in which case it forms a new row of length 1.



Row insertion

The algorithm **row insertion** takes a tableau T and inserts a positive integer x into it, resulting in a new tableau, denoted $\mathtt{T} \leftarrow x$.

1. If $x \geq i$ for all entries i in the first row of T, add x to the end of the first row.
2. If not, find the leftmost i in the first row of T such that $i > x$.
3. Place x into the place of i and take i out of the tableau.
4. Repeat the process in the second row with i.
5. Keep going until the bumped entry can be placed at the end of the row it is bumped into or until it is bumped at the bottom, in which case it forms a new row of length 1.

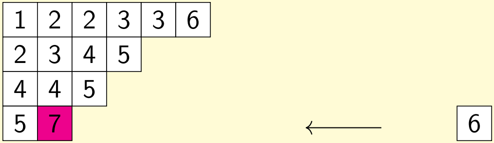
1	2	2	3	3	6
2	3	4	5		
4	4	5			
5	7				

$\longleftarrow$ 6

Row insertion

The algorithm **row insertion** takes a tableau T and inserts a positive integer x into it, resulting in a new tableau, denoted $\mathtt{T} \leftarrow x$.

1. If $x \geq i$ for all entries i in the first row of T, add x to the end of the first row.
2. If not, find the leftmost i in the first row of T such that $i > x$.
3. Place x into the place of i and take i out of the tableau.
4. Repeat the process in the second row with i.
5. Keep going until the bumped entry can be placed at the end of the row it is bumped into or until it is bumped at the bottom, in which case it forms a new row of length 1.



Row insertion

The algorithm **row insertion** takes a tableau T and inserts a positive integer x into it, resulting in a new tableau, denoted $\mathtt{T} \leftarrow x$.

1. If $x \geq i$ for all entries i in the first row of T, add x to the end of the first row.
2. If not, find the leftmost i in the first row of T such that $i > x$.
3. Place x into the place of i and take i out of the tableau.
4. Repeat the process in the second row with i.
5. Keep going until the bumped entry can be placed at the end of the row it is bumped into or until it is bumped at the bottom, in which case it forms a new row of length 1.

1	2	2	3	3	6
2	3	4	5		
4	4	5			
5	6				

$\longleftarrow$ 7

Row insertion

The algorithm **row insertion** takes a tableau T and inserts a positive integer x into it, resulting in a new tableau, denoted $\texttt{T} \leftarrow x$.

1. If $x \geq i$ for all entries i in the first row of T, add x to the end of the first row.
2. If not, find the leftmost i in the first row of T such that $i > x$.
3. Place x into the place of i and take i out of the tableau.
4. Repeat the process in the second row with i.
5. Keep going until the bumped entry can be placed at the end of the row it is bumped into or until it is bumped at the bottom, in which case it forms a new row of length 1.

1	2	2	3	3	6
2	3	4	5		
4	4	5			
5	6				
7					

Row bumping lemma

Assume that we successively insert two integers a and b (in this order) into some T. We label the resulting new boxes in T by B_a and B_b. Then we have the following:

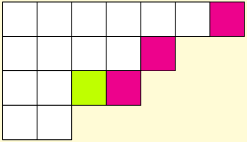
- if $a \leq b$, B_a is strictly left of and weakly below B_b,
- if $b < a$, B_b is weakly left of and strictly below B_a.

Row bumping lemma

Assume that we successively insert two integers a and b (in this order) into some `T`. We label the resulting new boxes in `T` by B_a and B_b. Then we have the following:

- if $a \leq b$, B_a is strictly left of and weakly below B_b,
- if $b < a$, B_b is weakly left of and strictly below B_a.

If $a \leq b$ and B_a is the green node, then the possible positions of B_b are highlighted in pink.

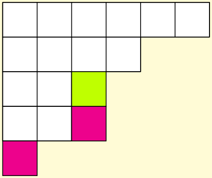


Row bumping lemma

Assume that we successively insert two integers a and b (in this order) into some `T`. We label the resulting new boxes in `T` by B_a and B_b. Then we have the following:

- if $a \leq b$, B_a is strictly left of and weakly below B_b,
- if $b < a$, B_b is weakly left of and strictly below B_a.

Similarly, if $a > b$ and B_a is the green node, then the possible positions of B_b are highlighted in pink.



Symmetric polynomials

Let $\mathbf{x} = (x_1, x_2, \ldots, x_\ell)$ be a set of variables. The monomial $x_1^{a_1} x_2^{a_2} \ldots x_\ell^{a_\ell}$ is said to have degree n if $\sum_i a_i = n$.

Symmetric polynomials

Let $\mathbf{x} = (x_1, x_2, \ldots, x_\ell)$ be a set of variables. The monomial $x_1^{a_1} x_2^{a_2} \ldots x_\ell^{a_\ell}$ is said to have degree n if $\sum_i a_i = n$.
We fix a partition $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_m)$ of n where $m \leq \ell$
The **monomial symmetric polynomial corresponding to** λ is given by

$$m_\lambda = m_\lambda(\mathbf{x}) = \sum x_1^{\lambda_1} \ldots x_\ell^{\lambda_\ell}$$

where the sum is over all distinct monomials having exponents λ_i.

Symmetric polynomials

Let $\mathbf{x} = (x_1, x_2, \ldots, x_\ell)$ be a set of variables. The monomial $x_1^{a_1} x_2^{a_2} \ldots x_\ell^{a_\ell}$ is said to have degree n if $\sum_i a_i = n$.
We fix a partition $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_m)$ of n where $m \leq \ell$
The **monomial symmetric polynomial corresponding to** λ is given by

$$m_\lambda = m_\lambda(\mathbf{x}) = \sum x_1^{\lambda_1} \ldots x_\ell^{\lambda_\ell}$$

where the sum is over all distinct monomials having exponents λ_i.

For example, if $\lambda = (2, 1)$ and $\mathbf{x} = (x_1, x_2, x_3)$, then

$$m_{(2,1)} = x_1^2 x_2 + x_1^2 x_3 + x_1 x_2^2 + x_1 x_3^2 + x_2^2 x_3 + x_2 x_3^2.$$

The n**th elementary symmetric polynomial** $(n \leq \ell)$ is given by

$$e_n = m_{(n)} = \sum_{i_1 < \cdots < i_n} x_{i_1} \dots x_{i_n}$$

and the n**th complete homogeneous symmetric polynomial** is given by

$$h_n = \sum_{\lambda \vdash n} m_{(1^n)} = \sum_{i_1 \leq \cdots \leq i_n} x_{i_1} \dots x_{i_n}.$$

The n**th elementary symmetric polynomial** $(n \leq \ell)$ is given by

$$e_n = m_{(n)} = \sum_{i_1 < \cdots < i_n} x_{i_1} \ldots x_{i_n}$$

and the n**th complete homogeneous symmetric polynomial** is given by

$$h_n = \sum_{\lambda \vdash n} m_{(1^n)} = \sum_{i_1 \leq \cdots \leq i_n} x_{i_1} \ldots x_{i_n}.$$

For example, if n=4 and $\mathbf{x} = (x_1, x_2, x_3, x_4, x_5)$, then

$$e_4 = x_1x_2x_3x_4 + x_1x_2x_3x_5 + x_1x_2x_4x_5 + x_1x_3x_4x_5 + x_2x_3x_4x_5,$$
$$h_4 = x_1^4 + x_2^4 + \cdots + x_1^3x_4 + x_1^3x_3 + \cdots + x_1x_2x_4x_5 + \cdots .$$

The polynomial e_n is the sum of all square-free monomials of degree n.

Schur polynomials

To each $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_m)$, we can associate another important symmetric polynomial $s_\lambda(x_1, \ldots, x_\ell)$ called the Schur polynomial.

Schur polynomials

To each $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_m)$, we can associate another important symmetric polynomial $s_\lambda(x_1, \ldots, x_\ell)$ called the Schur polynomial. These polynomials have an easy definition using tableaux:

$$s_\lambda(\mathbf{x}) = s_\lambda(x_1, \ldots, x_\ell) = \sum \mathbf{x}^{\mathtt{T}}$$

where the sum is taken over all monomials coming from semi-standard $\mathtt{T}$ of shape λ filled with numbers from 1 to ℓ.

Schur polynomials

To each $\lambda = (\lambda_1, \lambda_2, \dots, \lambda_m)$, we can associate another important symmetric polynomial $s_\lambda(x_1, \dots, x_\ell)$ called the Schur polynomial. These polynomials have an easy definition using tableaux:

$$s_\lambda(\mathbf{x}) = s_\lambda(x_1, \dots, x_\ell) = \sum \mathbf{x}^{\mathtt{T}}$$

where the sum is taken over all monomials coming from semi-standard $\mathtt{T}$ of shape λ filled with numbers from 1 to ℓ.

Continuing with the previous example, if

$$\mathtt{T}_2 = \begin{array}{|c|c|c|c|}\hline 1 & 2 & 3 & 6 \\ \hline 4 & 4 & 5 \\ \cline{1-3} 5 & 8 \\ \cline{1-2} 6 & 11 \\ \cline{1-2}\end{array}, \qquad \text{then } \mathbf{x}^{\mathtt{T}_2} = x_1 x_2 x_3 x_4^2 x_5^2 x_6^2 x_8 x_{11}.$$

Schur polynomials

To each $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_m)$, we can associate another important symmetric polynomial $s_\lambda(x_1, \ldots, x_\ell)$ called the Schur polynomial. These polynomials have an easy definition using tableaux:

$$s_\lambda(\mathbf{x}) = s_\lambda(x_1, \ldots, x_\ell) = \sum \mathbf{x}^{\mathtt{T}}$$

where the sum is taken over all monomials coming from semi-standard $\mathtt{T}$ of shape λ filled with numbers from 1 to ℓ.

Continuing with the previous example, if

$\mathtt{T}_2 =$

1	2	3	6
4	4	5	
5	8		
6	11		

, then $\mathbf{x}^{\mathtt{T}_2} = x_1 x_2 x_3 x_4^2 x_5^2 x_6^2 x_8 x_{11}$.

Observe, that if $\lambda = (n)$ (a single row) or $\lambda = (1^n)$ (a single column), then

$$s_{(n)}(\mathbf{x}) = h_n(\mathbf{x}) \qquad \text{and} \qquad s_{(1^n)}(\mathbf{x}) = e_n(\mathbf{x}).$$

Pieri rule

The following is an immediate consequence of the row bumping lemma:

Corollary (Pieri rule)

Using the insertion process from before, we obtain the following formulas:

$$s_\lambda s_{(n)} = \sum_\mu s_\mu$$

where the sum is taken over all μ's that are obtained from λ by adding n nodes, with no two in the same column; and

$$s_\lambda s_{(1^n)} = \sum_\mu s_\mu.$$

where the sum is taken over all μ's that are obtained from λ by adding n nodes, with no two in the same row.

Example of the Pieri rule

Take $\lambda = (4, 3, 2, 2)$ and $\nu = (3)$. Then using the formula from the previous slide, we see that

$$s_\lambda s_\nu = s_{(4,3,2,2)} s_{(3)} = \sum_\mu s_\mu$$

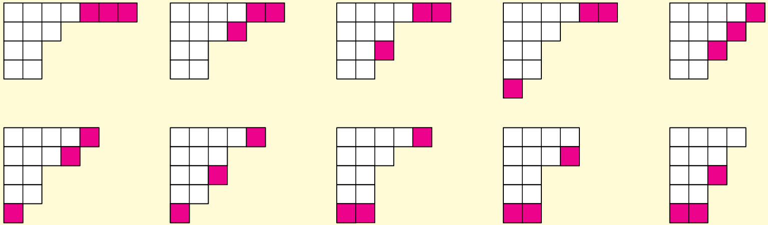
where μ is obtained from λ by adding 3 boxes, no two in the same column.

Example of the Pieri rule

Take $\lambda = (4, 3, 2, 2)$ and $\nu = (3)$. Then using the formula from the previous slide, we see that

$$s_\lambda s_\nu = s_{(4,3,2,2)} s_{(3)} = \sum_\mu s_\mu$$

where μ is obtained from λ by adding 3 boxes, no two in the same column. We see that the sum consists of the Schur polynomials corresponding to the following tableaux:



Supersymmetric polynomials

Let $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_m)$ be a partition of n and $\mathbf{x} = (x_1, x_2, \ldots, x_\ell)$ and $\mathbf{y} = (y_1, y_2, \ldots, y_k)$ two sets of variables.

Supersymmetric polynomials

Let $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_m)$ be a partition of n and $\mathbf{x} = (x_1, x_2, \ldots, x_\ell)$ and $\mathbf{y} = (y_1, y_2, \ldots, y_k)$ two sets of variables.

A polynomial $p(\mathbf{x}, \mathbf{y})$ is called **supersymmetric** if it is symmetric in both $\mathbf{x}$ and $\mathbf{y}$ and upon substitution $x_1 = t$ and $y_1 = -t$ the resulting expression is independent of t.

Supersymmetric polynomials

Let $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_m)$ be a partition of n and $\mathbf{x} = (x_1, x_2, \ldots, x_\ell)$ and $\mathbf{y} = (y_1, y_2, \ldots, y_k)$ two sets of variables.

A polynomial $p(\mathbf{x}, \mathbf{y})$ is called **supersymmetric** if it is symmetric in both $\mathbf{x}$ and $\mathbf{y}$ and upon substitution $x_1 = t$ and $y_1 = -t$ the resulting expression is independent of t.

For example, $x_1 + x_2 + x_3 + y_1 + y_2$ is supersymmetric (in $3 + 2$ variables).

Supersymmetric Schur polynomials

Characters of the Lie superalgebra $\mathfrak{gl}_{(m|n)}$.
Just like Schur polynomials, we can read them off from semi-standard supertableaux:

$$S_\lambda(\mathbf{x}/\mathbf{y}) = \sum \mathbf{x}^{\mathtt{T}}\mathbf{y}^{\mathtt{T}}.$$

Supersymmetric Schur polynomials

Characters of the Lie superalgebra $\mathfrak{gl}_{(m|n)}$.
Just like Schur polynomials, we can read them off from semi-standard supertableaux:

$$S_\lambda(\mathbf{x}/\mathbf{y}) = \sum \mathbf{x}^{\mathtt{T}}\mathbf{y}^{\mathtt{T}}.$$

Entries in a supertableau have even or odd parity such that

$$1' < 2' < 3' < \cdots < k < 1 < 2 < \cdots < \ell.$$

Odd (primed) entries must strictly increase along the rows and even (unprimed) entries must strictly increase down the columns.

Supersymmetric Schur polynomials

Characters of the Lie superalgebra $\mathfrak{gl}_{(m|n)}$.
Just like Schur polynomials, we can read them off from semi-standard supertableaux:

$$S_\lambda(\mathbf{x}/\mathbf{y}) = \sum \mathbf{x}^{\mathtt{T}}\mathbf{y}^{\mathtt{T}}.$$

Entries in a supertableau have even or odd parity such that

$$1' < 2' < 3' < \cdots < k < 1 < 2 < \cdots < \ell.$$

Odd (primed) entries must strictly increase along the rows and even (unprimed) entries must strictly increase down the columns.

For example, if $\lambda = (6, 4, 4, 3, 2)$ and

$$\mathtt{T} = \begin{array}{|c|c|c|c|c|c|} \hline 1' & 2' & 3' & 1 & 1 & 3 \\ \hline 1' & 3' & 4' & 2 \\ \cline{1-4} 2' & 3' & 1 & 3 \\ \cline{1-4} 1 & 2 & 2 \\ \cline{1-3} 2 & 4 \\ \cline{1-2} \end{array}, \qquad \text{then } \mathbf{x}^{\mathtt{T}} = y_1^2 y_2^2 y_3^3 y_4 x_1^4 x_2^4 x_3^2 x_4.$$

Is there an easy way to calculate $S_\lambda S_{(n)}$ and $S_\lambda S_{(1^n)}$?

Is there an easy way to calculate $S_\lambda S_{(n)}$ and $S_\lambda S_{(1^n)}$?

Let's try to row-insert an odd number.

$1'$	$2'$	$3'$	1	1	3
$1'$	$3'$	$4'$	2		
$2'$	$3'$	1	3		
1	2	2			
2	4				

$\longleftarrow$ $\boxed{2'}$

Is there an easy way to calculate $S_\lambda S_{(n)}$ and $S_\lambda S_{(1^n)}$?

Let's try to row-insert an odd number.

$1'$	$2'$	$3'$	1	1	3	$\longleftarrow$	$2'$
$1'$	$3'$	$4'$	2				
$2'$	$3'$	1	3				
1	2	2					
2	4						

Is there an easy way to calculate $S_\lambda S_{(n)}$ and $S_\lambda S_{(1^n)}$?

Let's try to row-insert an odd number.

$1'$	$2'$	$2'$	1	1	3
$1'$	$3'$	$4'$	2		
$2'$	$3'$	1	3		
1	2	2			
2	4				

$\longleftarrow$ $\boxed{3'}$

We see that simple row-insertion does not preserve the semi-standardness of our tableau.

Mixed insertion

In his paper, Muth described a new ‘mixed’ insertion process, called ε-insertion, which takes into account the parities of the entries in $\mathtt{T}$, by defining ε to be even or odd.

Mixed insertion

In his paper, Muth described a new ‘mixed’ insertion process, called ε-insertion, which takes into account the parities of the entries in T, by defining ε to be even or odd.

Two different algorithms:

Mixed insertion

In his paper, Muth described a new ‘mixed’ insertion process, called ε-insertion, which takes into account the parities of the entries in `T`, by defining ε to be even or odd.

Two different algorithms:

- even-insertion: even numbers are inserted in the next row down and odd numbers are inserted in the next column to the right,

Mixed insertion

In his paper, Muth described a new 'mixed' insertion process, called ε-insertion, which takes into account the parities of the entries in $\mathtt{T}$, by defining ε to be even or odd.

Two different algorithms:

- even-insertion: even numbers are inserted in the next row down and odd numbers are inserted in the next column to the right,
- odd-insertion: odd numbers are inserted in the next row down and even numbers are inserted in the next column to the right.

Mixed insertion

In his paper, Muth described a new 'mixed' insertion process, called ε-insertion, which takes into account the parities of the entries in $\mathtt{T}$, by defining ε to be even or odd.

Two different algorithms:

- even-insertion: even numbers are inserted in the next row down and odd numbers are inserted in the next column to the right,
- odd-insertion: odd numbers are inserted in the next row down and even numbers are inserted in the next column to the right.

In order to preserve the semi-standardness of $\mathtt{T}$, we modify the second step in row-insertion. Recall that

2. If not, find the leftmost i in the first row of $\mathtt{T}$ such that $i > x$.

Mixed insertion

In his paper, Muth described a new ‘mixed’ insertion process, called ε-insertion, which takes into account the parities of the entries in $\mathtt{T}$, by defining ε to be even or odd.

Two different algorithms:

- even-insertion: even numbers are inserted in the next row down and odd numbers are inserted in the next column to the right,
- odd-insertion: odd numbers are inserted in the next row down and even numbers are inserted in the next column to the right.

In order to preserve the semi-standardness of $\mathtt{T}$, we modify the second step in row-insertion. Recall that

2. If not, find the leftmost/uppermost i in the first row/column of $\mathtt{T}$ such that $i > x$ (even-ins.) or $i \geq x$ (odd-ins.).

Example of even-insertion

Under even insertion, even numbers are inserted in the next row down and odd numbers are inserted in the next column to the right, and we are looking for entries i such that $i > x$.

$1'$	$2'$	$3'$	1	1	3
$1'$	$3'$	1	2		
$2'$	$3'$	2	3		
1	3	3			
2					

$\uparrow$

$1'$

Example of even-insertion

Under even insertion, even numbers are inserted in the next row down and odd numbers are inserted in the next column to the right, and we are looking for entries i such that $i > x$.

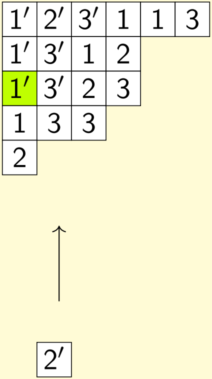
$1'$	$2'$	$3'$	1	1	3
$1'$	$3'$	1	2		
$2'$	$3'$	2	3		
1	3	3			
2					

$\uparrow$

$1'$

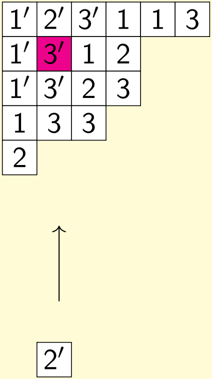
Example of even-insertion

Under even insertion, even numbers are inserted in the next row down and odd numbers are inserted in the next column to the right, and we are looking for entries i such that $i > x$.



Example of even-insertion

Under even insertion, even numbers are inserted in the next row down and odd numbers are inserted in the next column to the right, and we are looking for entries i such that $i > x$.



Example of even-insertion

Under even insertion, even numbers are inserted in the next row down and odd numbers are inserted in the next column to the right, and we are looking for entries i such that $i > x$.

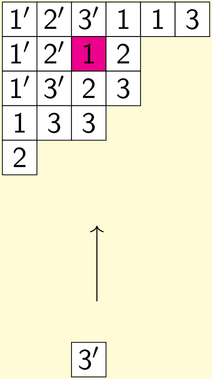
$1'$	$2'$	$3'$	1	1	3
$1'$	**$2'$**	1	2		
$1'$	$3'$	2	3		
1	3	3			
2					

$\uparrow$

$3'$

Example of even-insertion

Under even insertion, even numbers are inserted in the next row down and odd numbers are inserted in the next column to the right, and we are looking for entries i such that $i > x$.



Example of even-insertion

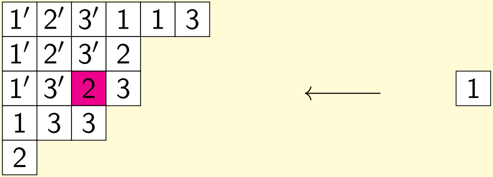
Under even insertion, even numbers are inserted in the next row down and odd numbers are inserted in the next column to the right, and we are looking for entries i such that $i > x$.

$1'$	$2'$	$3'$	1	1	3
$1'$	$2'$	$3'$	2		
$1'$	$3'$	2	3		
1	3	3			
2					

$\longleftarrow$ 1

Example of even-insertion

Under even insertion, even numbers are inserted in the next row down and odd numbers are inserted in the next column to the right, and we are looking for entries i such that $i > x$.



Example of even-insertion

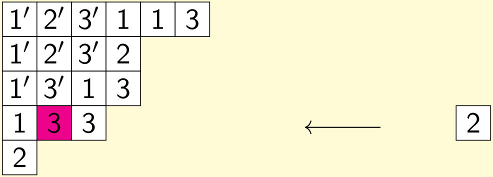
Under even insertion, even numbers are inserted in the next row down and odd numbers are inserted in the next column to the right, and we are looking for entries i such that $i > x$.

$1'$	$2'$	$3'$	1	1	3
$1'$	$2'$	$3'$	2		
$1'$	$3'$	1	3		
1	3	3			
2					

$\longleftarrow$ 2

Example of even-insertion

Under even insertion, even numbers are inserted in the next row down and odd numbers are inserted in the next column to the right, and we are looking for entries i such that $i > x$.



Example of even-insertion

Under even insertion, even numbers are inserted in the next row down and odd numbers are inserted in the next column to the right, and we are looking for entries i such that $i > x$.

$1'$	$2'$	$3'$	1	1	3
$1'$	$2'$	$3'$	2		
$1'$	$3'$	1	3		
1	2	3			
2					

$\longleftarrow$ 3

Example of even-insertion

Under even insertion, even numbers are inserted in the next row down and odd numbers are inserted in the next column to the right, and we are looking for entries i such that $i > x$.

$1'$	$2'$	$3'$	1	1	3
$1'$	$2'$	$3'$	2		
$1'$	$3'$	1	3		
1	2	3			
2	3				

Result

Corollary (The Pieri formulas for supersymmetric Schur polynomials)

Following the ε-insertion as defined above, we obtain the following formulas:

$$S_\lambda S_{(n)} = \sum_{\mu} S_\mu$$

where the sum is taken over all μ's that are obtained from λ by adding n nodes, with no two in the same column; and

$$S_\lambda S_{(1^n)} = \sum_{\mu} S_\mu.$$

where the sum is taken over all μ's that are obtained from λ by adding n nodes, with no two in the same row.