A presentation about

**Young Tableaux & Hook Length**

on

December 5, 2012

by

& B u b b a
S t
e
p
h
Presentation Outline

1. Introduction: History, Definitions, Applications
2. Bijective Proof
3. Probabilistic Proof
4. Questions
1900: What would eventually be called Young tableaux were introduced by Alfred Young (1873-1940), a mathematician at Cambridge University.

1928: Young tableaux actually named (by Herman Weyl).


1954-present: Several attempts to develop a satisfying and simple proof of the hook length formula.

1979: Probabilistic proof by Green, Nijenhuis, and Wilf.

1982: Bijective proof by Franzblau and Zeilberger.

2012: Steph and Bubba conduct an amazing presentation about Young Tableaux and the hook length formula.
A Young diagram is a collection of boxes in left-justified rows, where each row has the same or shorter length than the one above it. The number of rows and boxes in each row represents a unique partition (a.k.a. shape) $\lambda$ of $n$. A Young diagram is essentially the same as a Ferrers diagram, which uses dots instead of boxes.

**Example:**

$n = 12$, $\lambda = \{5, 3, 3, 1\}$
A Young tableau is obtained by filling in the boxes of the Young diagram with entries 1, 2, \ldots n. A tableau is called “standard” if the entries across each row and down each column are increasing.

<table>
<thead>
<tr>
<th>1</th>
<th>3</th>
<th>4</th>
<th>6</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>5</td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(This is just one of 3168 possible examples of a SYT for the this particular partition/shape \( \lambda \).)

Question: In which boxes can the number \( n = 12 \) appear in an SYT of this shape?
The hook length $h_{ij}$ of a box $(i, j)$ is the number of boxes that are in the same row $i$ to the right of it plus the number of boxes in the same column $j$ below it, plus one (for the box itself).
The hook length $h_{ij}$ of a box $(i, j)$ is the number of boxes that are in the same row $i$ to the right of it plus the number of boxes in the same column $j$ below it, plus one (for the box itself).
The hook length formula says that the number of standard Young tableaux, \( f_\lambda \), representing a partition \( \lambda \) of \( n \) is \( n! \) divided by the product of all of the hook lengths \( h_{ij} \):

\[
f_\lambda = \frac{n!}{\prod h_{ij}}
\]

Question: How many possible SYT are there for \[
\begin{array}{cccc}
\_ & \_ & \_ & \_ \\
\_ & \_ & \_ & \_ \\
\_ & \_ & \_ & \_ \\
\end{array}
\]?

\[
f_\lambda = \frac{n!}{\prod h_{ij}} = \frac{12!}{(9)(7)(5)(2)(1)(5)(3)(2)(4)(2)(1)(1)} = 3168.
\]
Applications of (Standard) Young Tableaux

- Partitions.
- Cool number puzzles.
- Proving combinatorial formulas.
- **Representation theory**.
- Algebraic geometry.
- Telephone numbers (number of distinct standard Young tableaux on $n$ entries, count the number of ways that $n$ subscribers to a telephone system can be linked in pairs, applications in graph theory).
A presentation of Franzblau and Zeilberger’s (1982) Bijective Proof
Franzblau and Zeilberger presented their bijective proof of the hook length formula in 1982.

To prove $f_\lambda = \frac{n!}{\prod h_{ij}}$, they equivalently showed that $n! = f_\lambda \cdot \prod h_{ij}$.

Their bijection is between the set of all possible Young tableau and the set of tuples with entries drawn from the set of all possible standard Young tableau and a set with cardinality $\prod h_{ij}$.

$$Y \mapsto (S, P),$$

where $Y$ is a Young tableau, $S$ is a standard Young tableau and $P$ is a pointer tableau, each of shape $\lambda$. 

**The General Idea**
The Pointer Tableaux

A “pointer” tableau of shape $\lambda$ is a Young diagram whose entry in box $(i, j)$ is any of the positions in hook $(i, j)$.

For example, consider box $(2, 1)$ of $\lambda = \{5, 3, 3, 1\}$.

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$(2, 1)$</td>
<td>●</td>
<td>●</td>
<td></td>
</tr>
<tr>
<td>●</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>●</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>R3</td>
</tr>
</tbody>
</table>

Notice that the top left position of the hook could have been labelled C1 or R1. Because the upcoming algorithm manipulates columns first and then rows, we’re giving columns the primacy in labeling.
A sample pointer tableau of shape $\lambda$ could be

\[
\begin{array}{cccc}
R3 & R3 & C1 & C2 & C1 \\
C1 & R2 & C1 \\
C1 & C2 & C1 \\
C1 & \\
\end{array}
\]

Notice that the entries are not unique.

The total number of possible pointer tableaux is $\prod h_{ij}$. 
The Algorithm - An Overview

Let \( \phi : \) represent our bijection from \{Young tableaux\} to \{standard Young tableaux\} × \{pointer tableaux\}. Let \( Y \) be a Young tableau and \( \phi(Y) = (S, P) \), where \( S \) is a standard Young tableau and \( P \) is a pointer tableau, all of shape \( \lambda \).

1. Starting from the RIGHTMOST column of \( Y \), we will add columns from right to left to \( S \), sorting the rows to be increasing as we go. At the same time, we update \( P \), again from right to left, to reflect the row sorting.

2. After adding a column, if the columns are not increasing, we will switch entries between rows, automatically sorting the rows as we go. At the same time, we update \( P \) to reflect the entry switching between rows.

3. At the end of each column add, we should always have a standard Young tableau, with partitions less than or equal to the partitions of \( \lambda \).
Let $\lambda = \{3, 2, 2, 1\}$.
Suppose $Y =$

\[
\begin{array}{ccc}
8 & 7 & 1 \\
4 & 3 \\
2 & 6 \\
5 \\
\end{array}
\]

$\phi(Y)$:

Column 1:

\[
\begin{array}{c}
1 \\
7 \\
3 \\
6 \\
\end{array} \mapsto \begin{array}{c}
1 \\
17 \\
3 \\
6 \\
\end{array} \quad \text{C1}
\]

Column 2:

\[
\begin{array}{c}
17 \\
3 \\
6 \\
\end{array} \mapsto \begin{array}{c}
17 \\
3 \\
6 \\
\end{array} \quad \text{C2C1}
\]
The Algorithm - Column 3

Suppose $Y = \begin{bmatrix} 8 & 7 & 1 \\ 4 & 3 \\ 2 & 6 \\ 5 \end{bmatrix}$

$\phi(Y)$:

$\begin{bmatrix} 8 & \rightarrow & 1 & 7 & 8 & C3C2C1 \\ 4 & \rightarrow & 3 & 4 & C2C1 \\ 2 & \rightarrow & 2 & 6 & C1C1 \\ 5 & \rightarrow & 5 & \rightarrow & C1 \end{bmatrix}$

Column 3:
Before we finish with column 3, we need to adjust $S$ so that the columns of $S$ are increasing.

1. Identify which box entries are “out of order” - defined as smaller than the one above it.
2. Let $x$ denote the smallest of the out-of-order box entries.
3. Exchange $x$ with the entry most recently added to the row above it.
4. Update pointers:
   - $x$’s switching partner’s pointer is its new position.
   - $x$’s new pointer is
     \[
     \begin{cases} 
     R(u + 1) & \text{if } x \text{ is from the most recently switched row}, \\
     R2 & \text{if } x \text{'s switching partner moves into the position of the most recently added to the row}, \\
     x\text{'s old pointer} & \text{otherwise}.
     \end{cases}
     \]
### The Algorithm - Where It Gets Hinky, cont.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>C3</th>
<th></th>
<th></th>
<th></th>
<th>R3</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7</td>
<td>8</td>
<td></td>
<td>1</td>
<td>3</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td></td>
<td>R2</td>
<td>2</td>
<td>8</td>
<td></td>
<td>C2</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td></td>
<td>C1</td>
<td>4</td>
<td>6</td>
<td></td>
<td>C1</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td>C1</td>
<td>5</td>
<td></td>
<td></td>
<td>C1</td>
<td></td>
</tr>
</tbody>
</table>

→

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>R3</th>
<th></th>
<th></th>
<th></th>
<th>R3</th>
<th>C2</th>
<th>C1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>7</td>
<td></td>
<td>1</td>
<td>3</td>
<td>7</td>
<td></td>
<td>C2</td>
<td>C1</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td></td>
<td>C2</td>
<td>2</td>
<td>6</td>
<td></td>
<td>C1</td>
<td></td>
<td>C1</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td></td>
<td>C1</td>
<td>4</td>
<td>8</td>
<td></td>
<td>C2</td>
<td></td>
<td>C1</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td>C1</td>
<td>5</td>
<td></td>
<td></td>
<td>C1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

→ end.
Let $\psi$ represent a map from \{standard Young tableaux\} $\times$ \{pointer tableaux\} to \{Young tableaux\}. Starting with a specific SYT $S$ and a pointer tableau $P$, let $\psi$ be defined as the following procedure:

1. Read the pointer tableau $P$ from left to right.

2. Assess whether any entries in $S$ were switched between rows.

3. Identify the possible switched entries between rows.

4. Select the largest entry to switch back, and switch it with an entry from the row below it, using the pointer table as a guide.

5. Return to step 2, because step 4 may have changed the $S$ significantly.

6. When the leftmost column of $P$ indicates no more row switches, remove entries from $S$ as indicated by $P$. These are the new leftmost column of $\psi(S, P)$. Delete the leftmost column of $P$.

7. Iterate this process until $S$ has been dismantled.
The Reverse Algorithm - An Example

$\psi((S, P))$: Indicators that there has been a row switch:

- There exists an $R_x$ in the leftmost column of $P$ and the entry below it is in the form of $C_z$, or
- There is some $C_x$ directly above some $C_y$ such that $x < y$ in the leftmost column of $P$ and the entry below it is in the form of $C_z$.

Contenders for switching are the entries in $S$ in the $C_z$ position. Pick the entry with the largest value.

Let $(S, P) =$

\[
\begin{array}{ccc}
1 & 3 & 7 \\
2 & 6 & \\
4 & 8 \\
5 & \\
\end{array}
\begin{array}{ccc}
R3 & C2 & C1 \\
C1 & C1 \\
C2 & C1 \\
C1 \\
\end{array}
\]
The Reverse Algorithm - An Example, cont.

ψ((S, P)):

- Look at the diagonal pairs of entries, starting with the pair above and to the left of 8. Pick the top entry of the first pair that the higher entry is larger than the lower entry. If there isn’t one, select the entry in the row above 8 in the first column. This is 8’s switching partner.

- Update pointers: 8’s switching partner’s new pointer is
  \[
  \begin{cases} 
  R(u + 1) & \text{if 8’s switching partner is from an } Ru \text{ row, } u \neq 2 \\
  \text{8’s switching partner’s old position} & \text{if 8’s switching partner is from an } R2 \text{ row} \\
  \text{8’s old pointer} & \text{otherwise.}
  \end{cases}
  \]

8’s pointer is its new position.
The Reverse Algorithm - Skipping Ahead

ψ((S, P)): After two more row switches, we arrive at:

\[
\begin{array}{ccc}
1 & 7 & 8 \\
3 & 4 & \\
2 & 6 & \\
5 & \\
\end{array}
\begin{array}{ccc}
C3 & C2 & C1 \\
C2 & C1 & \\
C1 & C1 & \\
C1 & \\
\end{array}
\rightarrow
\begin{array}{c}
8 \\
4 \\
2 \\
5 \\
\end{array}
\]

\[
\begin{array}{ccc}
1 & 7 & \\
3 & \\
6 & \\
\end{array}
\begin{array}{cc}
C2 & C1 \\
C1 & \\
C1 & \\
\end{array}
\rightarrow
\begin{array}{cc}
8 & 7 \\
4 & 3 \\
2 & 6 \\
\end{array}
\]

\[
\begin{array}{ccc}
8 & 7 & 1 \\
4 & 3 & \\
2 & 6 & \\
\end{array}
\rightarrow
\begin{array}{c}
5 \\
\end{array}
\]
Given a bijection between the set of Young tableau to the cartesian product of standard Young tableau and pointer tableau:

\[ \{YT\} \leftrightarrow \{SYT\} \times \{PT\} \]

\[ |\{YT\}| = |\{SYT\} \times \{PT\}| \]

\[ = |\{SYT\}| \cdot |\{PT\}| \]

\[ n! = f_\lambda \cdot \prod h_{ij} \]

\[ \frac{n!}{\prod h_{ij}} = f_\lambda. \]
A presentation of Greene, Nijenhuis, and Wilf’s (1979) Probabilistic Proof
In 1979, Green, Nijenhuis, and Wilf used probabilistic methods in an elegant proof of the hook length formula.

Their proof is based upon an algorithm that produces any standard Young tableau (SYT) of shape $\lambda$ with uniform probability $\prod h_{ij}/n!$. In this way, the number of SYT of $\lambda$ (i.e. the value of the hook length formula) is the reciprocal of the probability.

The bulk of their proof involves developing an expression for the probability of reaching a given corner square in an iteration of the algorithm.

The resulting expression is then related to the hook length formula, and a probabilistic interpretation followed by an inductive step shows that the hook length formula is indeed true for all SYT.
The Algorithm for Producing a Random SYT

1. Fix shape $\lambda \vdash n$.
2. Pick square $(a, b)$ with probability $1/n$. Let $(i, j) := (a, b)$.
3. While $(i, j)$ is not a corner square:
   a. Pick a square $(i', j')$ in the hook $H_{i,j} - \{(i, j)\}$ with probability $1/(h_{ij} - 1)$.
   b. Define a new $(i, j) := (i', j')$.
4. When a corner square $(\alpha, \beta)$ is reached, give it the label $n$.
5. Go back to step 2 with $\lambda := \lambda - \{(\alpha, \beta)\}$ and $n := n - 1$. Repeat until all squares of $\lambda$ are labeled.

Above: Example iteration from $\lambda_{15}$ to $\lambda_{14}$ by removing square $(3, 4)$ (in other words, square $(3, 4)$ becomes labeled by the value 15 in the final SYT). The iteration follows the path $(1, 2) \rightarrow (3, 2) \rightarrow (3, 3) \rightarrow (3, 4)$. The squares in the corresponding hooks at each stage are marked by $\times$’s.
The Proof.... (Slide 1/5)

Probability of terminating at a given corner square:
Generalizing the algorithm in terms of $k$, the $k$th iteration begins at a random square $(a, b)$ in $\lambda_k$ and terminates at a corner square $(\alpha_k, \beta_k)$, removing it to create $\lambda_{k-1}$. The corner square $(\alpha_k, \beta_k)$ corresponds to the square labeled $k$ in the final SYT.

The probability of terminating at a given corner square is computed by summing the conditional probabilities with respect to the first square chosen (chosen with probability $1/k$), and then, for each such first square, summing over all possible paths to reach the desired corner square, i.e.,

$$P((\alpha_k, \beta_k) | \lambda_k) = \frac{1}{k} \sum_{(a,b)} \sum_{\text{paths}} P((a, b) \text{ to } (\alpha_k, \beta_k) | \lambda_k).$$  \hspace{1cm} (1)

Note: from here on out I will remove the $k$ indices and just consider the case $n$. 

\[ \downarrow \]
Consider the paths to a corner square as projections:
To sum the probabilities of all possible paths, Green, Nijenhuis, and Wilf decomposed the paths into horizontal and vertical projections in row $\alpha$ and column $\beta$ of the terminating corner square.

Summing over all possible projections, the probability of reaching the corner square $(\alpha, \beta)$ is the product of the probabilities $1/(h_{i\beta} - 1)$ and $1/(h_{\alpha j} - 1)$ of reaching $(\alpha, \beta)$ from any of the squares in the projections, weighted by the respective hook lengths $h_{i\beta}$ and $h_{\alpha j}$ of the squares. The projection trick allows (1) to be re-expressed as

$$P((\alpha, \beta)|\lambda) = \frac{1}{n} \prod_{1 \leq i < \alpha} \frac{h_{i\beta}}{h_{i\beta} - 1} \prod_{1 \leq j < \beta} \frac{h_{\alpha j}}{h_{\alpha j} - 1}. \quad (2)$$
Observation on the ratio of hook length formulas:
Now comes a key observation: the ratio of the *proposed* hook length formula applied to $\lambda - \{(\alpha, \beta)\}$ (where the hook lengths of all squares in row $\alpha$ and column $\beta$ are reduced by 1 and the size of the SYT is reduced from $n$ to $n - 1$) to the proposed hook length formula applied to $\lambda$ gives a result identical to the expression for $P(\alpha, \beta)$ in (2). That is,

$$
\frac{f_{\lambda - \{\alpha, \beta\}}}{f_{\lambda}} = \frac{(n-1)!}{(\prod h_{ij})_{\lambda_{\alpha,\beta}}} = \frac{1}{n} \frac{\prod_{1 \leq i < \alpha} h_{i\beta} - 1}{h_{i\beta}} \frac{\prod_{1 \leq j < \beta} h_{\alpha j} - 1}{h_{\alpha j}}.
$$

(3)

In other words, the probability of reaching/labeling corner square $(\alpha, \beta)$ is equal to the ratio of the number of SYT of $\lambda - \{(\alpha, \beta)\}$ to number of SYT of $\lambda$. 
How we get (2) and (3) (and (2)=(3)) from a Young diagram:

(2) Summing over all projections to reach corner square:

\[ P((\alpha, \beta)|\lambda) \]

Example: \( \lambda_{15}, (\alpha, \beta) = (3, 4) \)

\[ \rightarrow \quad P((\alpha, \beta)|\lambda) = \frac{f_{\lambda-\{\alpha,\beta\}}}{f_{\lambda}} = \frac{1}{n} \prod_{1 \leq i < \alpha} \frac{h_{i\beta}}{h_{i\beta} - 1} \prod_{1 \leq j < \beta} \frac{h_{\alpha j}}{h_{\alpha j} - 1} \]

\[ = \frac{1}{15} \left( \frac{h_{14}}{h_{14} - 1} \right) \left( \frac{h_{24}}{h_{24} - 1} \right) \left( \frac{h_{31}}{h_{31} - 1} \right) \left( \frac{h_{32}}{h_{32} - 1} \right) \left( \frac{h_{33}}{h_{33} - 1} \right) = \frac{1}{15} \left( \frac{4}{3} \right) \left( \frac{2}{1} \right) \left( \frac{5}{4} \right) \left( \frac{4}{3} \right) \left( \frac{2}{1} \right) = \frac{16}{27} \]
Inductive conclusion:
Pulling together the equivalence of (2) and (3) and noting that the sum over a probabilistic expression is 1, we can write

$$1 = \sum_{\alpha, \beta} P((\alpha, \beta)|\lambda) = \sum_{\alpha, \beta} \frac{f_{\lambda-\{\alpha, \beta\}}}{f_{\lambda}}. \tag{4}$$

Thus, the number of SYT of \(\lambda\) is equal to the number of all possible SYT of size \(n - 1\) formed by removing a corner square from \(\lambda\). Hence, if we assume that the hook length formula can tell us the number of SYT of size \(n - 1\) formed by removing a square from \(\lambda\), then (4) indicates that we can then determine the number of SYT of \(\lambda\). It follows by induction that the hook length formula holds true for all SYT, completing the proof.
Illustrating the Probabilistic Method with an Example

Let’s consider the partition $\lambda = \{3, 1\}$:

```
      
```

Fill in the hook lengths (note: this is not a SYT):

```
  4 2 1  
  1  
```

Then the hook length formula tells us that there are three possible SYT for $\lambda = \{3, 1\}$:

$$f_\lambda = \frac{n!}{\prod h_{ij}} = \frac{4!}{(4)(2)(1)(1)} = 3.$$ 

The three SYT for $\lambda = \{3, 1\}$ are:

```
1 2 3 4
1 2 4 3
1 3 4 2
```
Let’s calculate the probability of one of the three SYTs:

\[
P \left( \begin{array}{ccc} 1 & 3 & 4 \\ 2 & & \end{array} \right) = P \left( \begin{array}{ccc} \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \\ \bullet & & \end{array} \right) \times P \left( \begin{array}{ccc} \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \\ \bullet & & \end{array} \right) \times P \left( \begin{array}{ccc} \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \\ \bullet & & \end{array} \right) \times P (\Box | \Box)
\]

\[
= \left[ P \left( \begin{array}{ccc} \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \\ \bullet & & \end{array} \right) + P \left( \begin{array}{ccc} \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \\ \bullet & & \end{array} \right) \right] \times \left[ P \left( \begin{array}{ccc} \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \\ \bullet & & \end{array} \right) + P \left( \begin{array}{ccc} \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \\ \bullet & & \end{array} \right) \right] \times \left[ P \left( \begin{array}{ccc} \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \\ \bullet & & \end{array} \right) + P \left( \begin{array}{ccc} \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \\ \bullet & & \end{array} \right) \right] \times P (\Box | \Box)
\]

\[
= \frac{1}{4} \times \frac{1}{4} \times \frac{1}{3} \times \frac{1}{1} = \frac{1}{4 \times 3 \times 2} \times \frac{1}{1} = \frac{1}{3}
\]

Indeed, the probability is 1/3, since each SYT for a given \( \lambda \) is equally likely. It follows that the probability of each SYT can be written as the reciprocal of the hook length formula, or \( P(SYT) = 1/f_\lambda = \prod h_{ij}/n! \).


G-C. Rota, "Indiscrete Thoughts," Birkhauser, Boston, 1997. (colorful portrayal of some of the great scientific personalities from 1950 to 1990)
