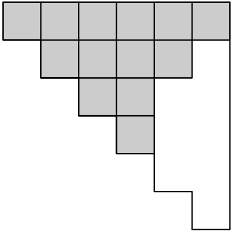
Crystallizing the Schur Q-functions

Maria Gillespie, University of California, Davis
Jake Levinson, University of Washington
Kevin Purbhoo, University of Waterloo

AMS Fall Western Sectional
Nov 4, 2017

Shifted tableaux

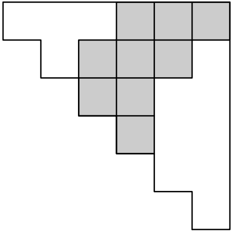
- **Shifted partitions:** Partitions with distinct parts; *i*th row shifted to the right *i* steps.



$\lambda = (6, 4, 2, 1)$

Shifted tableaux

- **Shifted partitions:** Partitions with distinct parts; *i*th row shifted to the right *i* steps.

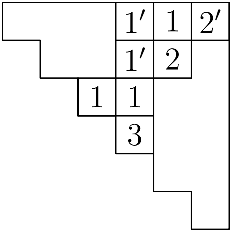


$$\lambda = (6, 4, 2, 1)$$
$$\mu = (3, 1)$$

- **Skew shape:** λ/μ

Shifted tableaux

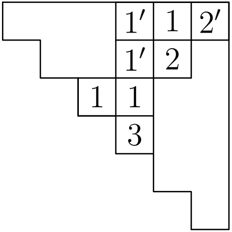
- **Shifted partitions:** Partitions with distinct parts; *i*th row shifted to the right *i* steps.



- **Skew shape:** λ/μ
- **Semistandard tableaux:** $1' < 1 < 2' < 2 < 3' < 3 < \cdots$ is alphabet, entries weakly increasing down and right. Primed letters can only repeat in columns and unprimed only in rows.

Shifted tableaux

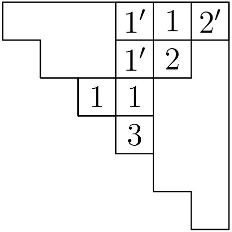
- **Shifted partitions:** Partitions with distinct parts; *i*th row shifted to the right *i* steps.



- **Skew shape:** λ/μ
- **Semistandard tableaux:** $1' < 1 < 2' < 2 < 3' < 3 < \cdots$ is alphabet, entries weakly increasing down and right. Primed letters can only repeat in columns and unprimed only in rows.
- **Canonical form:** First i or i' is always *unprimed* in reading order (read rows from bottom to top, $3111'21'12'$).

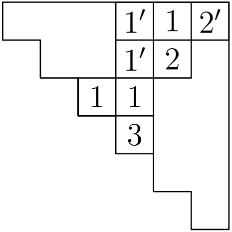
Shifted tableaux

- **Jeu de Taquin sliding:** Primed letters lose their primes when sliding into the diagonal.



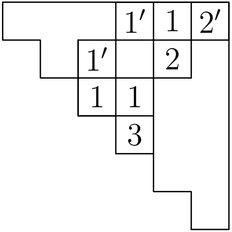
Shifted tableaux

- **Jeu de Taquin sliding:** Primed letters lose their primes when sliding into the diagonal.



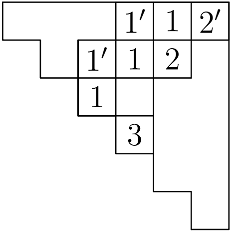
Shifted tableaux

- **Jeu de Taquin sliding:** Primed letters lose their primes when sliding into the diagonal.



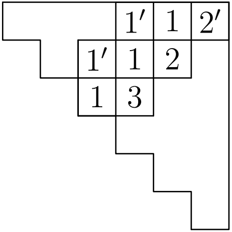
Shifted tableaux

- **Jeu de Taquin sliding:** Primed letters lose their primes when sliding into the diagonal.



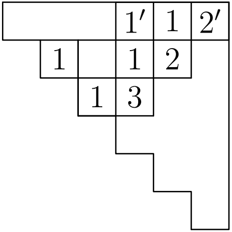
Shifted tableaux

- **Jeu de Taquin sliding:** Primed letters lose their primes when sliding into the diagonal.



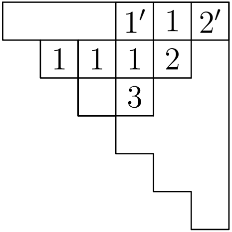
Shifted tableaux

- **Jeu de Taquin sliding:** Primed letters lose their primes when sliding into the diagonal.



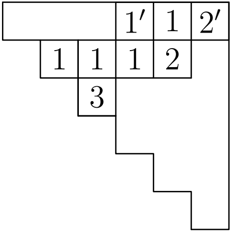
Shifted tableaux

- **Jeu de Taquin sliding:** Primed letters lose their primes when sliding into the diagonal.



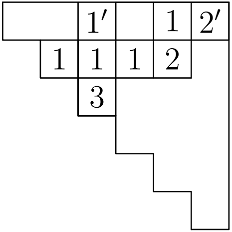
Shifted tableaux

- **Jeu de Taquin sliding:** Primed letters lose their primes when sliding into the diagonal.



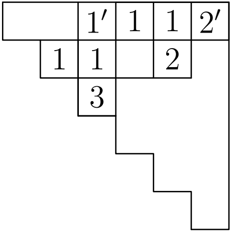
Shifted tableaux

- **Jeu de Taquin sliding:** Primed letters lose their primes when sliding into the diagonal.



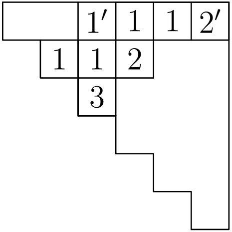
Shifted tableaux

- **Jeu de Taquin sliding:** Primed letters lose their primes when sliding into the diagonal.



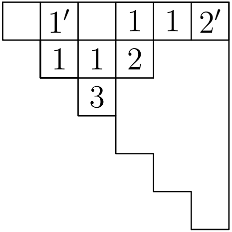
Shifted tableaux

- **Jeu de Taquin sliding:** Primed letters lose their primes when sliding into the diagonal.



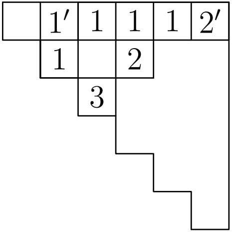
Shifted tableaux

- **Jeu de Taquin sliding:** Primed letters lose their primes when sliding into the diagonal.



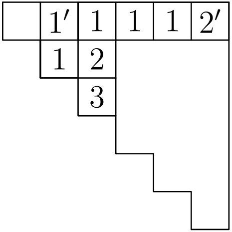
Shifted tableaux

- **Jeu de Taquin sliding:** Primed letters lose their primes when sliding into the diagonal.



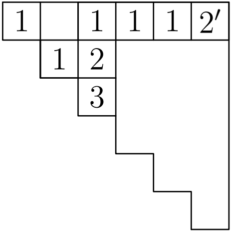
Shifted tableaux

- **Jeu de Taquin sliding:** Primed letters lose their primes when sliding into the diagonal.



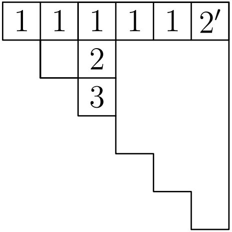
Shifted tableaux

- **Jeu de Taquin sliding:** Primed letters lose their primes when sliding into the diagonal.



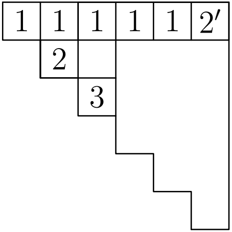
Shifted tableaux

- **Jeu de Taquin sliding:** Primed letters lose their primes when sliding into the diagonal.



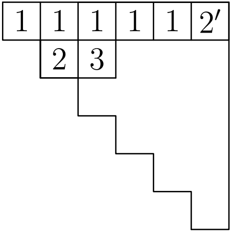
Shifted tableaux

- **Jeu de Taquin sliding:** Primed letters lose their primes when sliding into the diagonal.



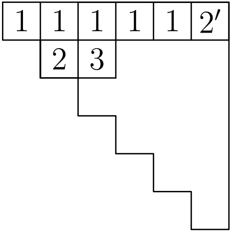
Shifted tableaux

- **Jeu de Taquin sliding:** Primed letters lose their primes when sliding into the diagonal.

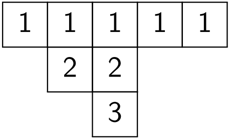


Shifted tableaux

- **Jeu de Taquin sliding:** Primed letters lose their primes when sliding into the diagonal.

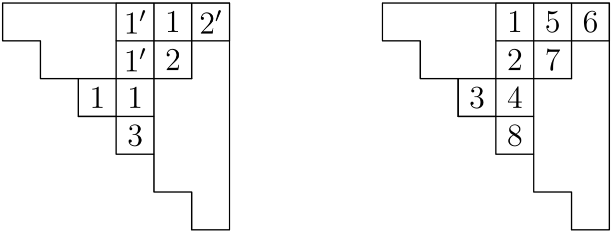


- **Highest weight**: Rectifies to shifted tableau with all *i*'s in *i*th row:



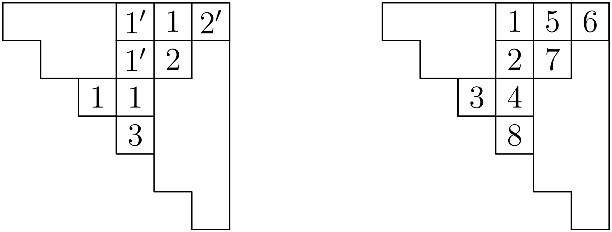
Standardization

- **Standardization order:** Break ties by reading order for unprimed entries, reverse reading order for primed entries



Standardization

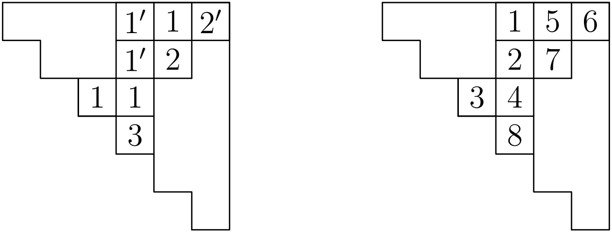
- **Standardization order:** Break ties by reading order for unprimed entries, reverse reading order for primed entries



- **Weight:** $\mathrm{wt}(T) = (m_1, m_2, \ldots)$ where m_i is the total number of i and i' entries in T. Above, weight is $(5, 2, 1)$.

Standardization

- **Standardization order:** Break ties by reading order for unprimed entries, reverse reading order for primed entries



- **Weight:** $\mathrm{wt}(T) = (m_1, m_2, \ldots)$ where m_i is the total number of i and i' entries in T. Above, weight is $(5, 2, 1)$.
- **Monomial weight:** $\mathbf{x}^{wt(T)} = x_1^{m_1} x_2^{m_2} \cdots$. Above, $x_1^5 x_2^2 x_3$.

Shifted Littlewood-Richardson Rule

- **Schur Q-functions:** Let $\ell(\mathrm{wt}\,T)$ be the number of nonzero entries in $\mathrm{wt}\,T$.

$$Q_{\lambda/\mu}(x_1, x_2, \ldots) = \sum_{T \in \mathrm{ShST}(\lambda/\mu)} 2^{\ell(\mathrm{wt}\,T)} \mathbf{x}^{\mathrm{wt}\,T}$$

Shifted Littlewood-Richardson Rule

- **Schur Q-functions:** Let $\ell(\mathrm{wt}\,T)$ be the number of nonzero entries in $\mathrm{wt}\,T$.

$$Q_{\lambda/\mu}(x_1, x_2, \ldots) = \sum_{T \in \mathrm{ShST}(\lambda/\mu)} 2^{\ell(\mathrm{wt}\,T)} \mathbf{x}^{\mathrm{wt}\,T}$$

- **Shifted Littlewood-Richardson rule:**

$$Q_{\lambda/\mu} = \sum f^{\lambda}_{\mu\nu} Q_{\nu}$$

where $f^{\lambda}_{\mu\nu}$ is the number of **highest weight** canonical shifted semistandard tableaux of shape λ/μ and weight ν.

Shifted Littlewood-Richardson Rule

- **Schur Q-functions:** Let $\ell(\mathrm{wt}\, T)$ be the number of nonzero entries in $\mathrm{wt}\, T$.

$$Q_{\lambda/\mu}(x_1, x_2, \ldots) = \sum_{T \in \mathrm{ShST}(\lambda/\mu)} 2^{\ell(\mathrm{wt}\, T)} \mathbf{x}^{\mathrm{wt}\, T}$$

- **Shifted Littlewood-Richardson rule:**

$$Q_{\lambda/\mu} = \sum f^{\lambda}_{\mu\nu} Q_{\nu}$$

where $f^{\lambda}_{\mu\nu}$ is the number of **highest weight** canonical shifted semistandard tableaux of shape λ/μ and weight ν.

- **Question:** Can we detect these highest weight skew shifted tableaux with crystal-like raising operators?

Shifted Littlewood-Richardson Rule

- **Schur Q-functions:** Let $\ell(\mathrm{wt}\, T)$ be the number of nonzero entries in $\mathrm{wt}\, T$.

$$Q_{\lambda/\mu}(x_1, x_2, \ldots) = \sum_{T \in \mathrm{ShST}(\lambda/\mu)} 2^{\ell(\mathrm{wt}\, T)} \mathbf{x}^{\mathrm{wt}\, T}$$

- **Shifted Littlewood-Richardson rule:**

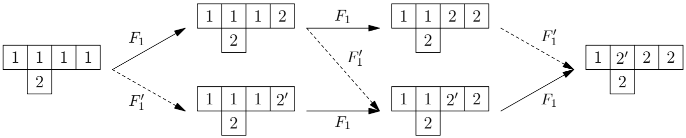
$$Q_{\lambda/\mu} = \sum f^{\lambda}_{\mu\nu} Q_{\nu}$$

where $f^{\lambda}_{\mu\nu}$ is the number of **highest weight** canonical shifted semistandard tableaux of shape λ/μ and weight ν.

- **Question:** Can we detect these highest weight skew shifted tableaux with crystal-like raising operators? (**Main result:** yes!)

Straight shapes, two letters

- Restrict to alphabet $\{1', 1, 2', 2\}$. Shape can have two rows:

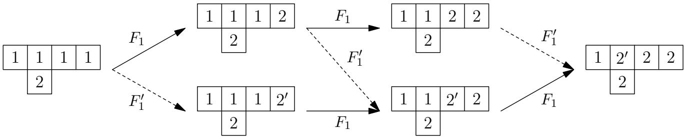


Or one row:

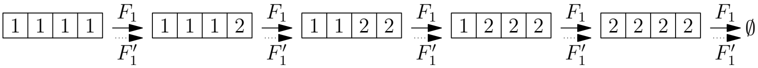
$$\begin{array}{|c|c|c|c|}\hline 1&1&1&1\\\hline\end{array} \overset{F_1}{\underset{F_1'}{\rightrightarrows}} \begin{array}{|c|c|c|c|}\hline 1&1&1&2\\\hline\end{array} \overset{F_1}{\underset{F_1'}{\rightrightarrows}} \begin{array}{|c|c|c|c|}\hline 1&1&2&2\\\hline\end{array} \overset{F_1}{\underset{F_1'}{\rightrightarrows}} \begin{array}{|c|c|c|c|}\hline 1&2&2&2\\\hline\end{array} \overset{F_1}{\underset{F_1'}{\rightrightarrows}} \begin{array}{|c|c|c|c|}\hline 2&2&2&2\\\hline\end{array} \overset{F_1}{\underset{F_1'}{\rightrightarrows}} \emptyset$$

Straight shapes, two letters

- Restrict to alphabet $\{1', 1, 2', 2\}$. Shape can have two rows:



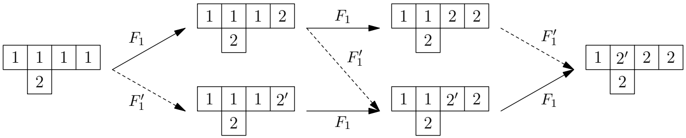
Or one row:



- Need two operators F_1, F_1' and their partial inverses E_1, E_1'.

Straight shapes, two letters

- Restrict to alphabet $\{1', 1, 2', 2\}$. Shape can have two rows:



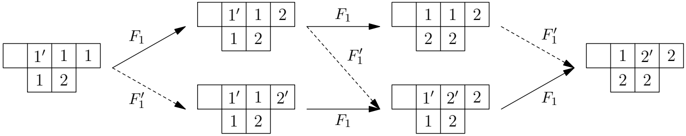
Or one row:

$$\boxed{1\,|\,1\,|\,1\,|\,1} \xrightarrow[F_1']{F_1} \boxed{1\,|\,1\,|\,1\,|\,2} \xrightarrow[F_1']{F_1} \boxed{1\,|\,1\,|\,2\,|\,2} \xrightarrow[F_1']{F_1} \boxed{1\,|\,2\,|\,2\,|\,2} \xrightarrow[F_1']{F_1} \boxed{2\,|\,2\,|\,2\,|\,2} \xrightarrow[F_1']{F_1} \emptyset$$

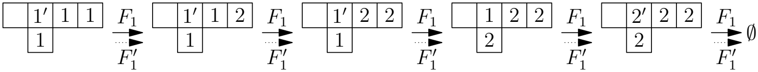
- Need two operators F_1, F_1' and their partial inverses E_1, E_1'.
- **Coplacticity**: Extend to skew shapes by applying outer slides.

Straight shapes, two letters

- Restrict to alphabet $\{1', 1, 2', 2\}$. Shape can have two rows:



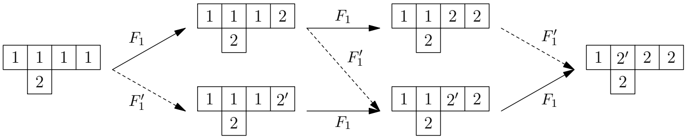
Or one row:



- Need two operators F_1, F_1' and their partial inverses E_1, E_1'.
- **Coplacticity**: Extend to skew shapes by applying outer slides.

Straight shapes, two letters

- Restrict to alphabet $\{1', 1, 2', 2\}$. Shape can have two rows:



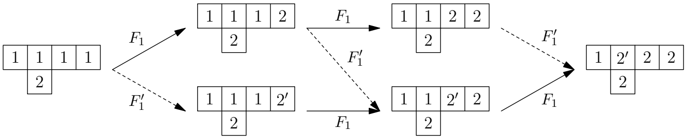
Or one row:

$$\boxed{1\,|\,1\,|\,1\,|\,1} \xrightarrow[F_1']{F_1} \boxed{1\,|\,1\,|\,1\,|\,2} \xrightarrow[F_1']{F_1} \boxed{1\,|\,1\,|\,2\,|\,2} \xrightarrow[F_1']{F_1} \boxed{1\,|\,2\,|\,2\,|\,2} \xrightarrow[F_1']{F_1} \boxed{2\,|\,2\,|\,2\,|\,2} \xrightarrow[F_1']{F_1} \emptyset$$

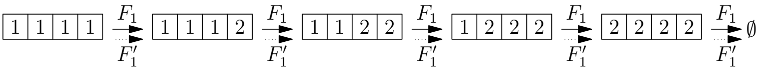
- Need two operators F_1, F_1' and their partial inverses E_1, E_1'.
- **Coplacticity**: Extend to skew shapes by applying outer slides.
- **General operators:** F_i, F_i', E_i, E_i' act on the strip of $i', i, (i+1)', i+1$ letters, by JDT rectifying, applying the appropriate operator, and unrectifying.

Straight shapes, two letters

- Restrict to alphabet $\{1', 1, 2', 2\}$. Shape can have two rows:



  Or one row:

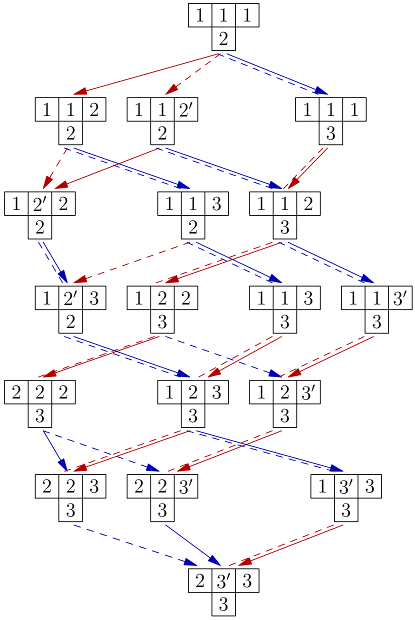


- Need two operators F_1, F_1' and their partial inverses E_1, E_1'.
- **Coplacticity**: Extend to skew shapes by applying outer slides.
- **General operators:** F_i, F_i', E_i, E_i' act on the strip of $i', i, (i+1)', i+1$ letters, by JDT rectifying, applying the appropriate operator, and unrectifying.
- Highest weight iff killed by all raising operators E_i, E_i'.

Crystal-like structure

- “Crystal graph” for $i = 1, 2$:

$$\boxed{\color{red}{F_1} \quad F_1' \qquad \color{blue}{F_2} \quad F_2'}$$



Crystal-like structure

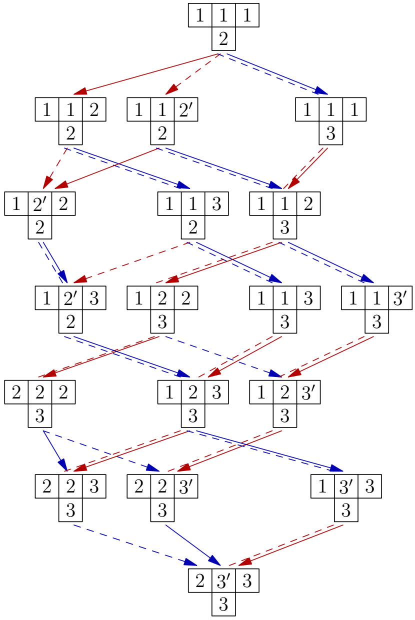
- "Crystal graph" for $i = 1, 2$:

$$\boxed{\color{red}{F_1} \quad \color{red}{F_1'} \qquad \color{blue}{F_2} \quad \color{blue}{F_2'}}$$

- Characters are Schur Q-functions:

$$\sum_{T \text{ in crystal}} 2^{\ell(wt(T))} x^{\mathrm{wt}\, T} = Q_\lambda$$

Graph structure implies Q_λ is symmetric.



Crystal-like structure

- "Crystal graph" for $i = 1, 2$:

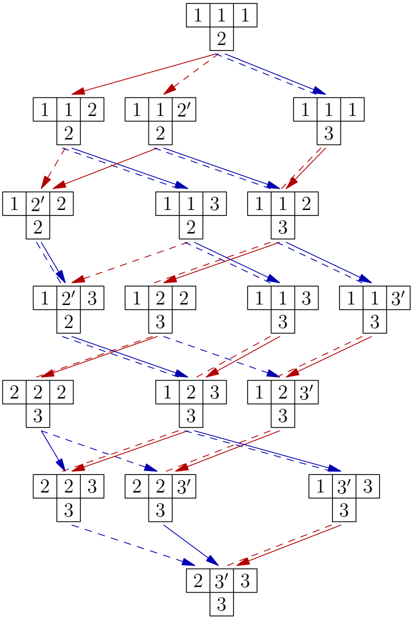
$$\boxed{F_1 \quad F_1' \qquad F_2 \quad F_2'}$$

- Characters are Schur Q-functions:

$$\sum_{T \text{ in crystal}} 2^{\ell(wt(T))} x^{\mathrm{wt}\,T} = Q_\lambda$$

  Graph structure implies Q_λ is symmetric.

- Connected components for skew shapes give LR rule for $Q_{\lambda/\mu}$.

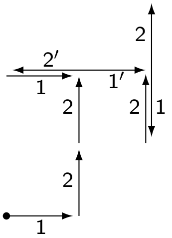


Lattice walks of words

- **Walk** of $w = w_1 w_2 \cdots w_n \in \{1', 1, 2', 2\}^n$ is a lattice walk in first quadrant from $(x_0, y_0) = (0, 0)$ to (x_n, y_n), with w_i labeling the step $(x_i, y_i) \to (x_{i+1}, y_{i+1})$. Directions:

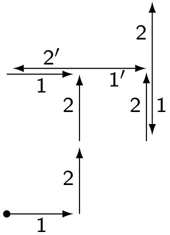
$\xrightarrow{1'}$	$\xrightarrow{1}$	$\uparrow 2'$	$\uparrow 2$	if $x_i y_i = 0$
$\xrightarrow{1'}$	$\downarrow 1$	$\xleftarrow{2'}$	$\uparrow 2$	if $x_i y_i \neq 0$

- **Example:** The walk of $1222'11'122$ looks like:



Properties of lattice walks (G., Levinson, Purbhoo)

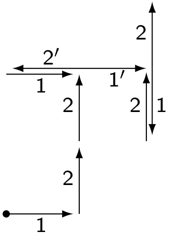
- **Rectification:** Endpoint (x_n, y_n) tells much about $\mathrm{rect}(w)$:



1	1	1	1	2	2	2
	2	2				

Properties of lattice walks (G., Levinson, Purbhoo)

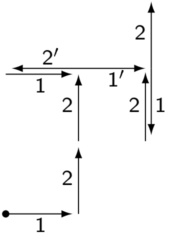
- **Rectification:** Endpoint (x_n, y_n) tells much about $\mathrm{rect}(w)$:
    - Shape is $((n + x_n + y_n)/2, (n - x_n - y_n)/2)$.



1	1	1	1	2	2	2
	2	2				

Properties of lattice walks (G., Levinson, Purbhoo)

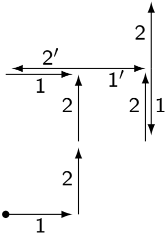
- **Rectification:** Endpoint (x_n, y_n) tells much about $\mathrm{rect}(w)$:
  - Shape is $((n + x_n + y_n)/2, (n - x_n - y_n)/2)$.
  - Weight is $((n + x_n - y_n)/2, (n - x_n + y_n)/2)$.



1	1	1	1	2	2	2
	2	2				

Properties of lattice walks (G., Levinson, Purbhoo)

- **Rectification:** Endpoint (x_n, y_n) tells much about $\mathrm{rect}(w)$:
  - Shape is $((n + x_n + y_n)/2, (n - x_n - y_n)/2)$.
  - Weight is $((n + x_n - y_n)/2, (n - x_n + y_n)/2)$.

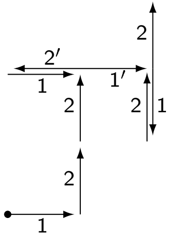


1	1	1	1	2	2	2
	2	2				

- **Highest weight:** A word w with letters in $\{1', 1, 2', 2\}$ has $E_1(w) = E_1'(w) = \varnothing$ iff its walk ends on the x-axis.

Properties of lattice walks (G., Levinson, Purbhoo)

- **Rectification:** Endpoint (x_n, y_n) tells much about $\mathrm{rect}(w)$:
  - Shape is $((n + x_n + y_n)/2, (n - x_n - y_n)/2)$.
  - Weight is $((n + x_n - y_n)/2, (n - x_n + y_n)/2)$.



1	1	1	1	2	2	2
	2	2				

- **Highest weight:** A word w with letters in $\{1', 1, 2', 2\}$ has $E_1(w) = E_1'(w) = \varnothing$ iff its walk ends on the x-axis.
- Proofs via **Knuth equivalence:** An elementary shifted Knuth move (Sagan, Worley) does not change the endpoint of the walk.

The operation F_1 on words

- Let $w \in \{1', 1, 2', 2\}^n$. An **F-critical substring** of w is a substring of any of the types and locations below.

Type	Substring	Starting Location	Transformation
1F	$1(1')^*2'$	$y = 0$ or $y = 1,\ x \geqslant 1$	$2'(1')^*2$
2F	$1(2)^*1'$	$x = 0$ or $x = 1,\ y \geqslant 1$	$2'(2)^*1$
3F	1	$y = 0$	2
4F	$1'$	$x = 0$	$2'$
5F	1 or $2'$	$x = 1,\ y \geqslant 1$	$\varnothing$

The operation F_1 on words

- Let $w \in \{1', 1, 2', 2\}^n$. An **F-critical substring** of w is a substring of any of the types and locations below.

Type	Substring	Starting Location	Transformation
1F	$1(1')^*2'$	$y = 0$ or $y = 1,\ x \geqslant 1$	$2'(1')^*2$
2F	$1(2)^*1'$	$x = 0$ or $x = 1,\ y \geqslant 1$	$2'(2)^*1$
3F	1	$y = 0$	2
4F	$1'$	$x = 0$	$2'$
5F	1 or $2'$	$x = 1,\ y \geqslant 1$	$\varnothing$

- **Final substring** is the F-critical substring $w_i \cdots w_j$ with largest j.

The operation F_1 on words

- Let $w \in \{1', 1, 2', 2\}^n$. An **F-critical substring** of w is a substring of any of the types and locations below.

Type	Substring	Starting Location	Transformation
1F	$1(1')^*2'$	$y = 0$ or $y = 1,\ x \geqslant 1$	$2'(1')^*2$
2F	$1(2)^*1'$	$x = 0$ or $x = 1,\ y \geqslant 1$	$2'(2)^*1$
3F	1	$y = 0$	2
4F	$1'$	$x = 0$	$2'$
5F	1 or $2'$	$x = 1,\ y \geqslant 1$	$\varnothing$

- **Final substring** is the F-critical substring $w_i \cdots w_j$ with largest j.
- $F_1(w)$: Replace $w_i \cdots w_j$ with its transformation.

The operation F_1 on words

- Let $w \in \{1', 1, 2', 2\}^n$. An **F-critical substring** of w is a substring of any of the types and locations below.

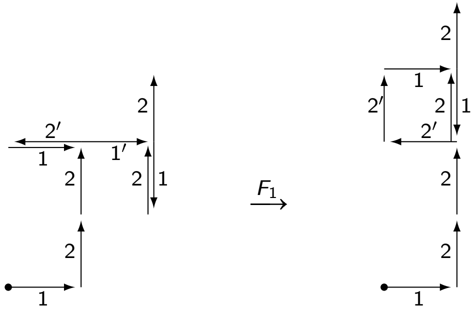
Type	Substring	Starting Location	Transformation
1F	$1(1')^*2'$	$y = 0$ or $y = 1$, $x \geqslant 1$	$2'(1')^*2$
2F	$1(2)^*1'$	$x = 0$ or $x = 1$, $y \geqslant 1$	$2'(2)^*1$
3F	1	$y = 0$	2
4F	$1'$	$x = 0$	$2'$
5F	1 or $2'$	$x = 1$, $y \geqslant 1$	$\varnothing$

- **Final substring** is the F-critical substring $w_i \cdots w_j$ with largest j.
- $F_1(w)$: Replace $w_i \cdots w_j$ with its transformation.
- If no F-critical substrings, $F_1(w) = \varnothing$.

Example

Type	Substring	Starting Location	Transformation
1F	$1(1')^*2'$	$y=0$ or $y=1,\ x \geqslant 1$	$2'(1')^*2$
2F	$1(2)^*1'$	$x=0$ or $x=1,\ y \geqslant 1$	$2'(2)^*1$
3F	1	$y=0$	2
4F	$1'$	$x=0$	$2'$
5F	1 or $2'$	$x=1,\ y \geqslant 1$	$\varnothing$

The word $w = 1222'11'122$ has a type 2F substring at $11'$, and this is its final F-critical substring. Thus $F_1(w) = 1222'2'1122$.



The operation E_1 on words

- Let $w \in \{1', 1, 2', 2\}^n$. An **E-critical substring** of w is a substring of any of the types and locations below.

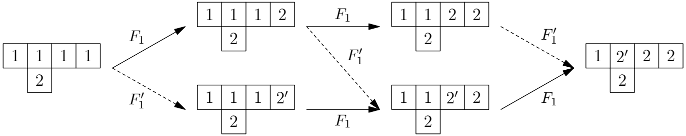
Type	Substring	Starting Location	Transformation
1E	$2'(2)^*1$	$x = 0$ or $x = 1,\ y \geqslant 1$	$1(2)^*1'$
2E	$2'(1')^*2$	$y = 0$ or $y = 1,\ x \geqslant 1$	$1(1')^*2'$
3E	$2'$	$y = 0$	$1'$
4E	2	$x = 0$	1
5E	1 or $2'$	$y = 1,\ x \geqslant 1$	$\varnothing$

- **Final substring** is the E-critical substring $w_i \cdots w_j$ with largest i, breaking ties by largest j.
- $E_1(w)$ defined by applying the appropriate transformation to the final E-critical substring of w.
- If there are no E-critical substrings we define $E_1(w) = \varnothing$.

Properties of E_1 and F_1 (G., Levinson, Purbhoo.)

Theorem. The operators E_1 and F_1 are:

- **Defined on tableaux:** Applying E_1 or F_1 to the reading word of a skew shifted semistandard tableau preserves semistandardness of the entries.
- **Agree with diagram on straight shapes:**



- **Coplactic:** E_1 and F_1 commute with all sequences of inner or outer JDT slides. (Difficult!)
- **Partial inverses:** $E_1(T) = T'$ if and only if $F_1(T') = T$.

Primed operators E_1' and F_1'

- $E_1'(w)$ is defined by changing the last $2'$ in w to a 1 if this does not change the standardization. Otherwise $E_1'(w) = \varnothing$.
- $F_1'(w)$ is defined by changing the last 1 in w to a $2'$ if this does not change the standardization. Otherwise $F_1'(w) = \varnothing$.
- Two maximal F_1' chains:

$$12211' \xrightarrow{F_1'} 1222'1' \xrightarrow{F_1'} \varnothing$$

$$1111'1' \xrightarrow{F_1'} 1121'1' \xrightarrow{F_1'} 1221'1' \xrightarrow{F_1'} 22211' \xrightarrow{F_1'} 2222'1 \xrightarrow{F_1'} 2222'2' \xrightarrow{F_1'} \varnothing$$

Primed operators E_1' and F_1'

- $E_1'(w)$ is defined by changing the last $2'$ in w to a 1 if this does not change the standardization. Otherwise $E_1'(w) = \varnothing$.
- $F_1'(w)$ is defined by changing the last 1 in w to a $2'$ if this does not change the standardization. Otherwise $F_1'(w) = \varnothing$.
- Two maximal F_1' chains:

$$12211' \xrightarrow{F_1'} 1222'1' \xrightarrow{F_1'} \varnothing$$

$$1111'1' \xrightarrow{F_1'} 1121'1' \xrightarrow{F_1'} 1221'1' \xrightarrow{F_1'} 22211' \xrightarrow{F_1'} 2222'1 \xrightarrow{F_1'} 2222'2' \xrightarrow{F_1'} \varnothing$$

- **Theorem.** The operations E_1' and F_1' are:
    - Coplactic and well-defined on skew shifted tableaux.
    - Partial inverses: if $E_1'(T) = T'$ then $F_1'(T') = T$.
    - Have chains of length 2 unless the rectification shape has one row; in the latter case they coincide with E_1 and F_1.

Properties

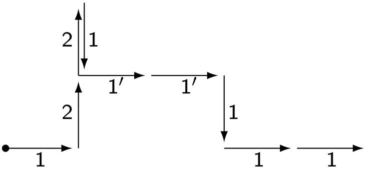
- E'_1, E_1, F'_1, F_1 all commute with each other.

Properties

- E_1', E_1, F_1', F_1 all commute with each other.
- E_1 and E_1' move the endpoint of the walk by $(1, -1)$, F_1 and F_1' by $(-1, 1)$. Example of repeated F_1 followed by one F_1':

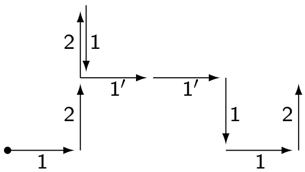
Properties

- E_1', E_1, F_1', F_1 all commute with each other.
- E_1 and E_1' move the endpoint of the walk by $(1, -1)$, F_1 and F_1' by $(-1, 1)$. Example of repeated F_1 followed by one F_1':



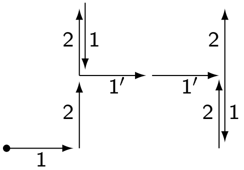
Properties

- E_1', E_1, F_1', F_1 all commute with each other.
- E_1 and E_1' move the endpoint of the walk by $(1, -1)$, F_1 and F_1' by $(-1, 1)$. Example of repeated F_1 followed by one F_1':



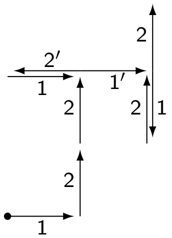
Properties

- E_1', E_1, F_1', F_1 all commute with each other.
- E_1 and E_1' move the endpoint of the walk by $(1, -1)$, F_1 and F_1' by $(-1, 1)$. Example of repeated F_1 followed by one F_1':



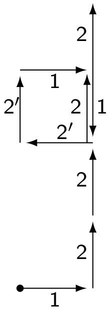
Properties

- E_1', E_1, F_1', F_1 all commute with each other.
- E_1 and E_1' move the endpoint of the walk by $(1, -1)$, F_1 and F_1' by $(-1, 1)$. Example of repeated F_1 followed by one F_1':



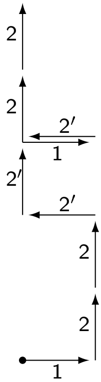
Properties

- E_1', E_1, F_1', F_1 all commute with each other.
- E_1 and E_1' move the endpoint of the walk by $(1, -1)$, F_1 and F_1' by $(-1, 1)$. Example of repeated F_1 followed by one F_1':



Properties

- E_1', E_1, F_1', F_1 all commute with each other.
- E_1 and E_1' move the endpoint of the walk by $(1, -1)$, F_1 and F_1' by $(-1, 1)$. Example of repeated F_1 followed by one F_1':



Application: Type B Schubert curves

- **Orthogonal Grassmannian** $\mathrm{OG}(2n+1, n)$: the type B analog of $\mathrm{Gr}(n, k)$

Application: Type B Schubert curves

- **Orthogonal Grassmannian** $\mathrm{OG}(2n+1,n)$: the type B analog of $\mathrm{Gr}(n,k)$
- Can be defined as the variety of n-dimensional *isotropic* (self-orthogonal) subspaces V of $\mathbb{C}^{2n+1}$ with respect to the symmetric inner form $\langle (a_i),(b_j)\rangle = \sum a_i b_{2n+1-i}$.

Application: Type B Schubert curves

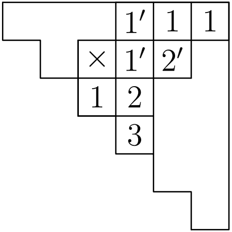
- **Orthogonal Grassmannian** $\mathrm{OG}(2n+1, n)$: the type B analog of $\mathrm{Gr}(n, k)$
- Can be defined as the variety of n-dimensional *isotropic* (self-orthogonal) subspaces V of $\mathbb{C}^{2n+1}$ with respect to the symmetric inner form $\langle (a_i), (b_j) \rangle = \sum a_i b_{2n+1-i}$.
- Schubert varieties $\Omega_\lambda(\mathcal{F})$ defined for shifted partitions λ in the $n \times n$ staircase.

Application: Type B Schubert curves

- **Orthogonal Grassmannian** $\mathrm{OG}(2n+1, n)$: the type B analog of $\mathrm{Gr}(n, k)$
- Can be defined as the variety of n-dimensional *isotropic* (self-orthogonal) subspaces V of $\mathbb{C}^{2n+1}$ with respect to the symmetric inner form $\langle (a_i), (b_j) \rangle = \sum a_i b_{2n+1-i}$.
- Schubert varieties $\Omega_\lambda(\mathcal{F})$ defined for shifted partitions λ in the $n \times n$ staircase.
- **Schubert curves:** certain 1-dimensional intersections of Schubert varieties

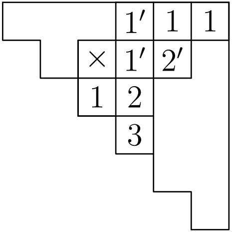
Schubert curves in the Orthogonal Grassmannian

- Real Schubert curves have a natural smooth covering of $\mathbb{RP}^1$, monodromy operator given by a certain operation on highest weight skew tableau with a marked inner corner:



Schubert curves in the Orthogonal Grassmannian

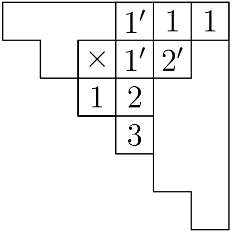
- Real Schubert curves have a natural smooth covering of $\mathbb{RP}^1$, monodromy operator given by a certain operation on highest weight skew tableau with a marked inner corner:



- Monodromy operator:
  1. **Rectify**, with $\boxtimes = 0$
  2. **Slide** the $\boxtimes$ to an outer corner with an outer JDT slide
  3. **Unrectify** to the original shape, with $\boxtimes = \infty$
  4. **Slide** the $\boxtimes$ back to an inner corner

Schubert curves in the Orthogonal Grassmannian

- Real Schubert curves have a natural smooth covering of $\mathbb{RP}^1$, monodromy operator given by a certain operation on highest weight skew tableau with a marked inner corner:

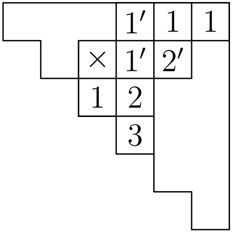


- Monodromy operator:
  1. **Rectify**, with $\boxtimes = 0$
  2. **Slide** the $\boxtimes$ to an outer corner with an outer JDT slide
  3. **Unrectify** to the original shape, with $\boxtimes = \infty$
  4. **Slide** the $\boxtimes$ back to an inner corner
- Operators F_i, F_i' give us a new easier rule that avoids rectification!

Thank You!

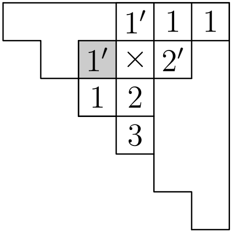
Local rule (G., J. Levinson and K. Purbhoo)

- Local rule for steps $1-3$, without rectifying:
  - **Phase 1.** Switch $\boxtimes$ with the nearest $1'$ *after* it in reading order, if one exists, and then with the nearest 1 *before* it in reading order. Do the same for $2'$ and 2, and so on until there is no i' after it for some i.



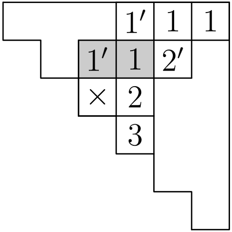
Local rule (G., J. Levinson and K. Purbhoo)

- Local rule for steps $1-3$, without rectifying:
  - **Phase 1.** Switch $\boxtimes$ with the nearest $1'$ *after* it in reading order, if one exists, and then with the nearest 1 *before* it in reading order. Do the same for $2'$ and 2, and so on until there is no i' after it for some i.



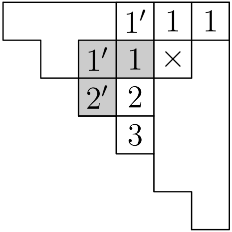
Local rule (G., J. Levinson and K. Purbhoo)

- Local rule for steps $1-3$, without rectifying:
  - **Phase 1.** Switch $\boxtimes$ with the nearest $1'$ *after* it in reading order, if one exists, and then with the nearest 1 *before* it in reading order. Do the same for $2'$ and 2, and so on until there is no i' after it for some i.



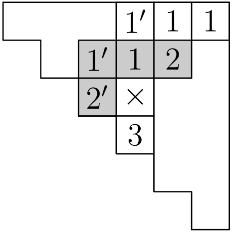
Local rule (G., J. Levinson and K. Purbhoo)

- Local rule for steps $1-3$, without rectifying:
  - **Phase 1.** Switch $\boxtimes$ with the nearest $1'$ *after* it in reading order, if one exists, and then with the nearest 1 *before* it in reading order. Do the same for $2'$ and 2, and so on until there is no i' after it for some i.



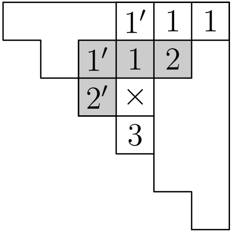
Local rule (G., J. Levinson and K. Purbhoo)

- Local rule for steps $1-3$, without rectifying:
  - **Phase 1.** Switch $\boxtimes$ with the nearest $1'$ *after* it in reading order, if one exists, and then with the nearest 1 *before* it in reading order. Do the same for $2'$ and 2, and so on until there is no i' after it for some i. At this point go to Phase 2.



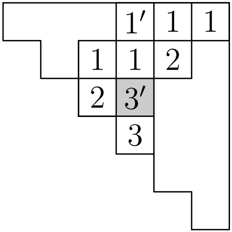
Local rule (G., J. Levinson and K. Purbhoo)

- Local rule for steps $1 - 3$, without rectifying:
  - **Phase 1.** Switch $\boxtimes$ with the nearest $1'$ *after* it in reading order, if one exists, and then with the nearest 1 *before* it in reading order. Do the same for $2'$ and 2, and so on until there is no i' after it for some i. At this point go to Phase 2.
  - **Phase 2.** Replace the $\boxtimes$ with i' and apply $F_i, F_{i+1}, \ldots$ in that order until only one entry is changing. Then replace that entry with $\boxtimes$.



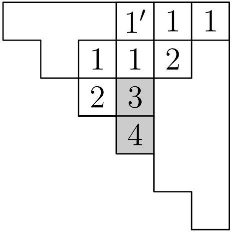
Local rule (G., J. Levinson and K. Purbhoo)

- Local rule for steps $1-3$, without rectifying:
  - **Phase 1.** Switch $\boxtimes$ with the nearest $1'$ *after* it in reading order, if one exists, and then with the nearest 1 *before* it in reading order. Do the same for $2'$ and 2, and so on until there is no i' after it for some i. At this point go to Phase 2.
  - **Phase 2.** Replace the $\boxtimes$ with i' and apply $F_i, F_{i+1}, \ldots$ in that order until only one entry is changing. Then replace that entry with $\boxtimes$.



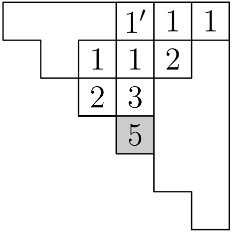
Local rule (G., J. Levinson and K. Purbhoo)

- Local rule for steps $1-3$, without rectifying:
  - **Phase 1.** Switch $\boxtimes$ with the nearest $1'$ *after* it in reading order, if one exists, and then with the nearest 1 *before* it in reading order. Do the same for $2'$ and 2, and so on until there is no i' after it for some i. At this point go to Phase 2.
  - **Phase 2.** Replace the $\boxtimes$ with i' and apply $F_i, F_{i+1}, \ldots$ in that order until only one entry is changing. Then replace that entry with $\boxtimes$.



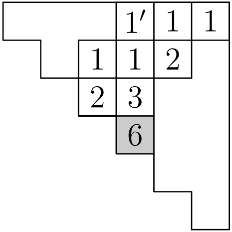
Local rule (G., J. Levinson and K. Purbhoo)

- Local rule for steps $1-3$, without rectifying:
  - **Phase 1.** Switch $\boxtimes$ with the nearest $1'$ *after* it in reading order, if one exists, and then with the nearest 1 *before* it in reading order. Do the same for $2'$ and 2, and so on until there is no i' after it for some i. At this point go to Phase 2.
  - **Phase 2.** Replace the $\boxtimes$ with i' and apply $F_i, F_{i+1}, \ldots$ in that order until only one entry is changing. Then replace that entry with $\boxtimes$.



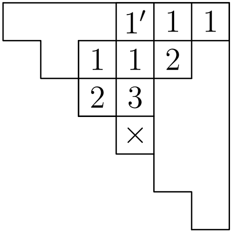
Local rule (G., J. Levinson and K. Purbhoo)

- Local rule for steps $1 - 3$, without rectifying:
  - **Phase 1.** Switch $\boxtimes$ with the nearest $1'$ *after* it in reading order, if one exists, and then with the nearest 1 *before* it in reading order. Do the same for $2'$ and 2, and so on until there is no i' after it for some i. At this point go to Phase 2.
  - **Phase 2.** Replace the $\boxtimes$ with i' and apply $F_i, F_{i+1}, \ldots$ in that order until only one entry is changing. Then replace that entry with $\boxtimes$.



Local rule (G., J. Levinson and K. Purbhoo)

- Local rule for steps $1-3$, without rectifying:
  - **Phase 1.** Switch $\boxtimes$ with the nearest $1'$ *after* it in reading order, if one exists, and then with the nearest 1 *before* it in reading order. Do the same for $2'$ and 2, and so on until there is no i' after it for some i. At this point go to Phase 2.
  - **Phase 2.** Replace the $\boxtimes$ with i' and apply $F_i, F_{i+1}, \ldots$ in that order until only one entry is changing. Then replace that entry with $\boxtimes$.



Larger Phase 2 example

						1
					1	2
			×	3		
	1	1				
	2					
	3					
2	4					

Larger Phase 2 example

$$
\begin{array}{ccccccc}
 & & & & & & \boxed{1} \\
 & & & & & \boxed{1} & \boxed{2} \\
 & & & \boxed{1'} & \boxed{3} & & \\
 & \boxed{1} & \boxed{1} & & & & \\
 & \boxed{2} & & & & & \\
 & \boxed{3} & & & & & \\
\boxed{2} & \boxed{4} & & & & &
\end{array}
$$

Larger Phase 2 example

$$
\begin{array}{ccccccc}
 & & & & & & 1 \\
 & & & & & 1 & 2 \\
 & & & 1 & 3 & & \\
 & 1 & 2' & & & & \\
 & 2 & & & & & \\
 & 3 & & & & & \\
2 & 4 & & & & &
\end{array}
$$

Larger Phase 2 example

$$
\begin{array}{ccccccc}
 & & & & & & \boxed{1} \\
 & & & & & \boxed{1} & \boxed{2} \\
 & & & \boxed{1} & \boxed{3} & & \\
 & \boxed{1} & \boxed{2} & & & & \\
 & \boxed{3'} & & & & & \\
 & \boxed{3} & & & & & \\
\boxed{2} & \boxed{4} & & & & &
\end{array}
$$

Larger Phase 2 example

$$
\begin{array}{ccccccc}
 & & & & & & \boxed{1} \\
 & & & & & \boxed{1} & \boxed{2} \\
 & & & \boxed{1} & \boxed{3} & & \\
 & \boxed{1} & \boxed{2} & & & & \\
 & \boxed{3} & & & & & \\
 & \boxed{4'} & & & & & \\
\boxed{2} & \boxed{4} & & & & &
\end{array}
$$

Larger Phase 2 example

						1
					1	2
			1	3		
	1	2				
	3					
	4					
2	5					

Larger Phase 2 example

						1
					1	2
			1	3		
	1	2				
	3					
	4					
2	6					

Larger Phase 2 example

						1
					1	2
			1	3		
	1	2				
	3					
	4					
2	7					

Larger Phase 2 example

						1
					1	2
			1	3		
	1	2				
	3					
	4					
2	$\times$					

Application: Schubert curves

- **Grassmannian**: $\mathrm{Gr}(n, k)$ is the variety of k-dimensional subspaces of $\mathbb{C}^n$.

Application: Schubert curves

- **Grassmannian**: $\mathrm{Gr}(n, k)$ is the variety of k-dimensional subspaces of $\mathbb{C}^n$.
- **Schubert varieties:** Certain subvarieties $\Omega_\lambda(\mathcal{F})$ where λ fits in a $k \times (n-k)$ rectangle $\ydiagram{3,3}$ and $\mathcal{F}$ is a complete flag.

Application: Schubert curves

- **Grassmannian**: $\mathrm{Gr}(n, k)$ is the variety of k-dimensional subspaces of $\mathbb{C}^n$.
- **Schubert varieties:** Certain subvarieties $\Omega_\lambda(\mathcal{F})$ where λ fits in a $k \times (n-k)$ rectangle and $\mathcal{F}$ is a complete flag.
- **Schubert curve:** A one-dimensional intersection of Schubert varieties.

Application: Schubert curves

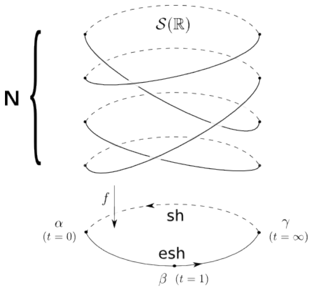
- **Grassmannian**: $\mathrm{Gr}(n, k)$ is the variety of k-dimensional subspaces of $\mathbb{C}^n$.
- **Schubert varieties:** Certain subvarieties $\Omega_\lambda(\mathcal{F})$ where λ fits in a $k \times (n-k)$ rectangle and $\mathcal{F}$ is a complete flag.
- **Schubert curve:** A one-dimensional intersection of Schubert varieties.
- Special Schubert curves: Three partitions α, β, γ with $|\alpha| + |\beta| + |\gamma| = k(n-k) - 1$. Define

$$\boxed{S = S(\alpha, \beta, \gamma) = \Omega_\alpha(\mathcal{F}_0) \cap \Omega_\beta(\mathcal{F}_1) \cap \Omega_\gamma(\mathcal{F}_\infty)}$$

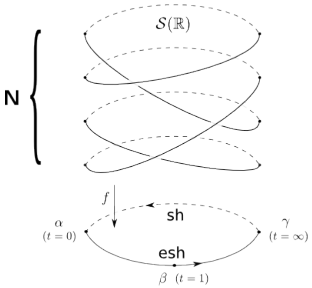
where the flag $\mathcal{F}_t$ is the maximally tangent flag at t of the *rational normal curve* in $\mathbb{P}^{n-1}$:

$$(1 : t) \mapsto (1 : t : t^2 : \cdots : t^{n-1})$$

Real geometry of S

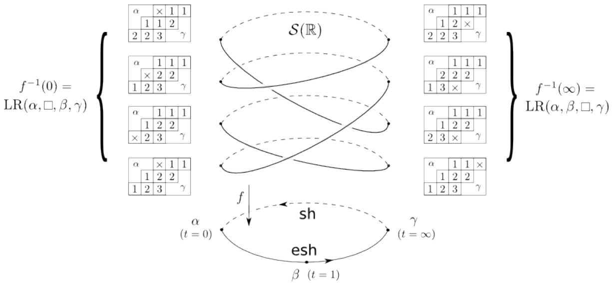


Real geometry of S



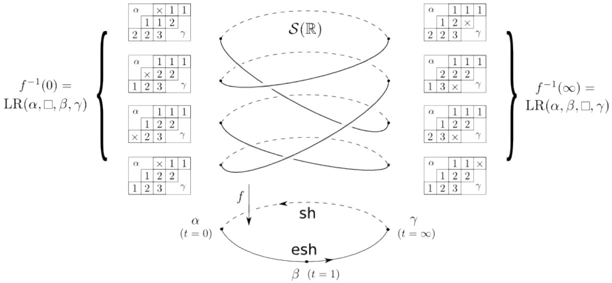
Theorem(s). (Levinson, Speyer.) There is a degree-N map $f : S \to \mathbb{P}^1$ that makes $S(\mathbb{R})$ a smooth covering of the circle $\mathbb{RP}^1$, with finite fibers of size $N = c^{\ydiagram{3,3}}_{\alpha,\square,\beta,\gamma}$.

Real geometry of S



- (Fiber over 0) $\leftrightarrow$ Tableaux of shape γ^c/α with one **inner** corner $\times$ and the rest a highest weight tableau of weight β.
- (Fiber over ∞) $\leftrightarrow$ Tableaux of shape γ^c/α with one **outer** corner $\times$ and the rest a highest weight tableau of weight β.

Real geometry of S

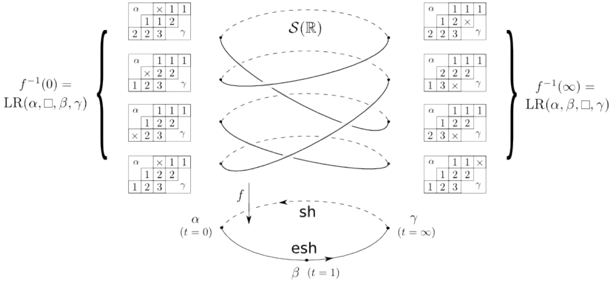


- The arcs of $S(\mathbb{R})$ covering $\mathbb{R}_-$ and $\mathbb{R}_+$ respectively induce the *shuffling* and *evacuation-shuffling* bijections sh and esh:

$$\mathrm{LR}(\alpha, \square, \beta, \gamma) \underset{\mathrm{sh}}{\overset{\mathrm{esh}}{\rightleftarrows}} \mathrm{LR}(\alpha, \beta, \square, \gamma)$$

- **Monodromy operator:** $\omega = \mathrm{sh} \circ \mathrm{esh}$. Cycles of ω correspond to connected components of $S(\mathbb{R})$.

Real geometry of S



- The arcs of $S(\mathbb{R})$ covering $\mathbb{R}_-$ and $\mathbb{R}_+$ respectively induce the *shuffling* and *evacuation-shuffling* bijections sh and esh:

$$\mathrm{LR}(\alpha, \square, \beta, \gamma) \underset{\mathrm{sh}}{\overset{\mathrm{esh}}{\rightleftarrows}} \mathrm{LR}(\alpha, \beta, \square, \gamma)$$

- **Monodromy operator:** $\omega = \mathrm{sh} \circ \mathrm{esh}$. Cycles of ω correspond to connected components of $S(\mathbb{R})$.

Shuffling

- **Shuffling**, or **JDT**: Do an outer jeu de taquin slide with the $\boxtimes$ as the empty square to get an element of $\mathrm{LR}(\alpha, \square, \beta, \gamma)$.

α			1	1	1
		1	2	2	2
	2	3	3		
1	3	4	4	γ	
3	4	5	$\times$		

Shuffling

- **Shuffling**, or **JDT**: Do an outer jeu de taquin slide with the $\boxtimes$ as the empty square to get an element of $\mathrm{LR}(\alpha, \square, \beta, \gamma)$.

			1	1	1
		1	2	2	2
	2	3	3		
1	3	4	4		
3	4	5	$\times$		

Shuffling

- **Shuffling**, or **JDT**: Do an outer jeu de taquin slide with the $\boxtimes$ as the empty square to get an element of $\mathrm{LR}(\alpha, \square, \beta, \gamma)$.

			1	1	1
		1	2	2	2
	2	3	3		
1	3	4	4		
3	4	$\times$	5		

Shuffling

- **Shuffling**, or **JDT**: Do an outer jeu de taquin slide with the $\boxtimes$ as the empty square to get an element of $\mathrm{LR}(\alpha, \square, \beta, \gamma)$.

			1	1	1
		1	2	2	2
	2	3	3		
1	3	$\times$	4		
3	4	4	5		

Shuffling

- **Shuffling**, or **JDT**: Do an outer jeu de taquin slide with the $\boxed{\times}$ as the empty square to get an element of $\mathrm{LR}(\alpha, \square, \beta, \gamma)$.

			1	1	1
		1	2	2	2
	2	$\times$	3		
1	3	3	4		
3	4	4	5		

Shuffling

- **Shuffling**, or **JDT**: Do an outer jeu de taquin slide with the $\boxtimes$ as the empty square to get an element of $\mathrm{LR}(\alpha, \square, \beta, \gamma)$.

			1	1	1
		1	2	2	2
	$\times$	2	3		
1	3	3	4		
3	4	4	5		

Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.

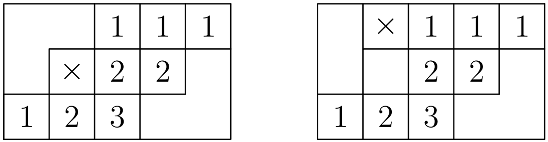
		1	1	1
	$\times$	2	2	
1	2	3		

T

		1	1	1
	$\times$	2	2	
1	2	3		

Evacuation-shuffling

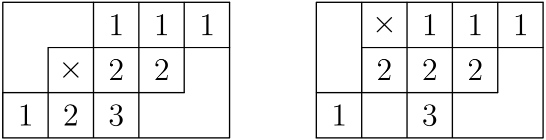
- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.



T

Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.



T

Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.

		1	1	1
	$\times$	2	2	
1	2	3		

T

	$\times$	1	1	1
	2	2	2	
1	3			

Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.

		1	1	1
	$\times$	2	2	
1	2	3		

T

	$\times$	1	1	1
	2	2	2	
1	3	1		

Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.

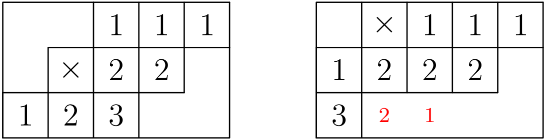
		1	1	1
	$\times$	2	2	
1	2	3		

T

	$\times$	1	1	1
1	2	2	2	
	3	1		

Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.



T

Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.

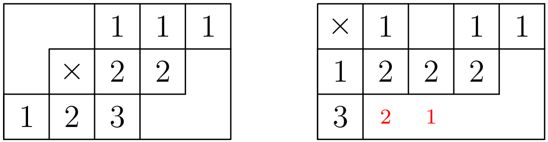
		1	1	1
	$\times$	2	2	
1	2	3		

T

$\times$		1	1	1
1	2	2	2	
3	2	1		

Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.



T

Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.

		1	1	1
	$\times$	2	2	
1	2	3		

T

$\times$	1	1		1
1	2	2	2	
3	2	1		

Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.
- **Shuffling**

		1	1	1
	$\times$	2	2	
1	2	3		

T

$\times$	1	1	1	3
1	2	2	2	
3	2	1		

Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.
- **Shuffling**

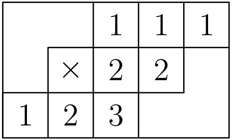
		1	1	1
	$\times$	2	2	
1	2	3		

T

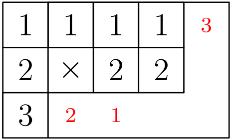
1	1	1	1	3
$\times$	2	2	2	
3	2	1		

Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.
- **Shuffling**



T



Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.
- **Shuffling**

		1	1	1
	$\times$	2	2	
1	2	3		

T

1	1	1	1	3
2	2	$\times$	2	
3	2	1		

Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.
- **Shuffling**
- **Un-rectification:** Treat $\times$ as largest entry.

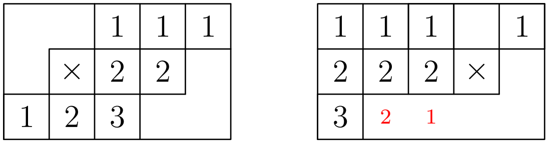
		1	1	1
	$\times$	2	2	
1	2	3		

T

1	1	1	1	3
2	2	2	$\times$	
3	2	1		

Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.
- **Shuffling**
- **Un-rectification:** Treat $\times$ as largest entry.



T

Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.
- **Shuffling**
- **Un-rectification:** Treat $\times$ as largest entry.

		1	1	1
	$\times$	2	2	
1	2	3		

T

1	1		1	1
2	2	2	$\times$	
3	2	1		

Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.
- **Shuffling**
- **Un-rectification:** Treat $\times$ as largest entry.

		1	1	1
	$\times$	2	2	
1	2	3		

T

1		1	1	1
2	2	2	$\times$	
3	2	1		

Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.
- **Shuffling**
- **Un-rectification:** Treat $\times$ as largest entry.

		1	1	1
	$\times$	2	2	
1	2	3		

T

	1	1	1	1
2	2	2	$\times$	
3	2	1		

Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.
- **Shuffling**
- **Un-rectification:** Treat $\times$ as largest entry.

		1	1	1
	$\times$	2	2	
1	2	3		

T

	1	1	1	1
2	2	2	$\times$	
	3	1		

Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.
- **Shuffling**
- **Un-rectification:** Treat $\times$ as largest entry.

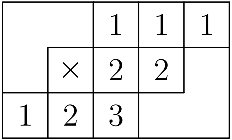
		1	1	1
	$\times$	2	2	
1	2	3		

T

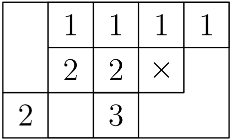
	1	1	1	1
	2	2	$\times$	
2	3	1		

Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.
- **Shuffling**
- **Un-rectification:** Treat $\times$ as largest entry.



T



Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.
- **Shuffling**
- **Un-rectification:** Treat $\times$ as largest entry.

		1	1	1
	$\times$	2	2	
1	2	3		

T

	1	1	1	1
		2	$\times$	
2	2	3		

Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.
- **Shuffling**
- **Un-rectification:** Treat $\times$ as largest entry.

		1	1	1
	$\times$	2	2	
1	2	3		

T

		1	1	1
	1	2	$\times$	
2	2	3		

$\mathrm{esh}(T)$

Evacuation-shuffling

- Conjugation of shuffling by JDT rectification.
- **Rectification:** Treat $\times$ as 0.
- **Shuffling**
- **Un-rectification:** Treat $\times$ as largest entry.

		1	1	1
	$\times$	2	2	
1	2	3		

T

		1	1	1
	1	2	$\times$	
2	2	3		

$\mathrm{esh}(T)$

- **Shuffle again** to compute $\omega = \mathrm{sh} \circ \mathrm{esh}$:

ωT:

		$\times$	1	1
	1	1	2	
2	2	3		

Local rule for evacuation-shuffling

- Recall: esh consists of rectifying, shuffling, and un-rectifying.
- (G., Levinson.) Local rule, without rectifying: Start at $i = 1$.

						1	1	1
			$\times$	1	1	2	2	
			1	2	2	3		
			3	3	4			
		4	4					
2	3	5						

Local rule for evacuation-shuffling

- Recall: esh consists of rectifying, shuffling, and un-rectifying.
- (G., Levinson.) Local rule, without rectifying: Start at $i = 1$.
  - **Phase 1.** Switch $\boxtimes$ with the nearest i *prior* to it in reading order, if one exists. Increment i by 1 and repeat.

						1	1	1
			$\times$	1	1	2	2	
			1	2	2	3		
			3	3	4			
		4	4					
2	3	5						

Local rule for evacuation-shuffling

- Recall: esh consists of rectifying, shuffling, and un-rectifying.
- (G., Levinson.) Local rule, without rectifying: Start at $i = 1$.
  - **Phase 1.** Switch $\boxtimes$ with the nearest i *prior* to it in reading order, if one exists. Increment i by 1 and repeat.

						1	1	1
			1	1	1	2	2	
			$\times$	2	2	3		
			3	3	4			
		4	4					
2	3	5						

Local rule for evacuation-shuffling

- Recall: esh consists of rectifying, shuffling, and un-rectifying.
- (G., Levinson.) Local rule, without rectifying: Start at $i = 1$.
  - **Phase 1.** Switch $\boxtimes$ with the nearest i *prior* to it in reading order, if one exists. Increment i by 1 and repeat.

						1	1	1
			1	1	1	2	2	
			2	2	2	3		
			3	3	4			
		4	4					
$\times$	3	5						

Local rule for evacuation-shuffling

- Recall: esh consists of rectifying, shuffling, and un-rectifying.
- (G., Levinson.) Local rule, without rectifying: Start at $i = 1$.
  - **Phase 1.** Switch $\boxtimes$ with the nearest i *prior* to it in reading order, if one exists. Increment i by 1 and repeat.
    If the $\boxtimes$ precedes all of the i's in reading order, go to Phase 2.

						1	1	1
			1	1	1	2	2	
			2	2	2	3		
			3	3	4			
		4	4					
$\times$	3	5						

Local rule for evacuation-shuffling

- Recall: esh consists of rectifying, shuffling, and un-rectifying.
- (G., Levinson.) Local rule, without rectifying: Start at $i = 1$.
  - **Phase 1.** Switch $\boxtimes$ with the nearest i *prior* to it in reading order, if one exists. Increment i by 1 and repeat.
    If the $\boxtimes$ precedes all of the i's in reading order, go to Phase 2.
  - **Phase 2.** Replace the $\boxtimes$ with i and apply $F_i, F_{i+1}, \ldots$ in that order until only one entry is changing. Then replace that entry with $\boxtimes$.

						1	1	1
			1	1	1	2	2	
			2	2	2	3		
			3	3	4			
		4	4					
$\times$	3	5						

Local rule for evacuation-shuffling

- Recall: esh consists of rectifying, shuffling, and un-rectifying.
- (G., Levinson.) Local rule, without rectifying: Start at $i = 1$.
    - **Phase 1.** Switch $\boxtimes$ with the nearest i *prior* to it in reading order, if one exists. Increment i by 1 and repeat.
      If the $\boxtimes$ precedes all of the i's in reading order, go to Phase 2.
    - **Phase 2.** Replace the $\boxtimes$ with i and apply $F_i, F_{i+1}, \ldots$ in that order until only one entry is changing. Then replace that entry with $\boxtimes$.

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

Local rule for evacuation-shuffling

- Recall: esh consists of rectifying, shuffling, and un-rectifying.
- (G., Levinson.) Local rule, without rectifying: Start at $i = 1$.
  - **Phase 1.** Switch $\boxtimes$ with the nearest i *prior* to it in reading order, if one exists. Increment i by 1 and repeat.
    If the $\boxtimes$ precedes all of the i's in reading order, go to Phase 2.
  - **Phase 2.** Replace the $\boxtimes$ with i and apply $F_i, F_{i+1}, \ldots$ in that order until only one entry is changing. Then replace that entry with $\boxtimes$.

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

Local rule for evacuation-shuffling

- Recall: esh consists of rectifying, shuffling, and un-rectifying.
- (G., Levinson.) Local rule, without rectifying: Start at $i = 1$.
  - **Phase 1.** Switch $\boxtimes$ with the nearest i *prior* to it in reading order, if one exists. Increment i by 1 and repeat.
    If the $\boxtimes$ precedes all of the i's in reading order, go to Phase 2.
  - **Phase 2.** Replace the $\boxtimes$ with i and apply $F_i, F_{i+1}, \ldots$ in that order until only one entry is changing. Then replace that entry with $\boxtimes$.

$$
\begin{array}{ccccccccc}
 & & & & & & 1 & 1 & 1 \\
 & & & 1 & 1 & 1 & 2 & 2 & \\
 & & & 2 & 2 & 2 & 3 & & \\
 & & & 3 & 3 & 5 & & & \\
 & & 4 & 4 & & & & & \\
3 & 4 & 5 & & & & & &
\end{array}
$$

Local rule for evacuation-shuffling

- Recall: esh consists of rectifying, shuffling, and un-rectifying.
- (G., Levinson.) Local rule, without rectifying: Start at $i = 1$.
  - **Phase 1.** Switch $\boxtimes$ with the nearest i *prior* to it in reading order, if one exists. Increment i by 1 and repeat.
    If the $\boxtimes$ precedes all of the i's in reading order, go to Phase 2.
  - **Phase 2.** Replace the $\boxtimes$ with i and apply $F_i, F_{i+1}, \ldots$ in that order until only one entry is changing. Then replace that entry with $\boxtimes$.

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

Local rule for evacuation-shuffling

- Recall: esh consists of rectifying, shuffling, and un-rectifying.
- (G., Levinson.) Local rule, without rectifying: Start at $i = 1$.
  - **Phase 1.** Switch $\boxtimes$ with the nearest i *prior* to it in reading order, if one exists. Increment i by 1 and repeat.
    If the $\boxtimes$ precedes all of the i's in reading order, go to Phase 2.
  - **Phase 2.** Replace the $\boxtimes$ with i and apply $F_i, F_{i+1}, \ldots$ in that order until only one entry is changing. Then replace that entry with $\boxtimes$.

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

Local rule for evacuation-shuffling

- Recall: esh consists of rectifying, shuffling, and un-rectifying.
- (G., Levinson.) Local rule, without rectifying: Start at $i = 1$.
  - **Phase 1.** Switch $\boxtimes$ with the nearest i *prior* to it in reading order, if one exists. Increment i by 1 and repeat.
    If the $\boxtimes$ precedes all of the i's in reading order, go to Phase 2.
  - **Phase 2.** Replace the $\boxtimes$ with i and apply $F_i, F_{i+1}, \ldots$ in that order until only one entry is changing. Then replace that entry with $\boxtimes$.

$$
\begin{array}{ccccccccc}
 & & & & & & 1 & 1 & 1 \\
 & & & 1 & 1 & 1 & 2 & 2 & \\
 & & & 2 & 2 & 2 & 3 & & \\
 & & & 3 & 3 & \times & & & \\
 & & 4 & 4 & & & & & \\
3 & 4 & 5 & & & & & &
\end{array}
$$

Geometric consequences (G., Levinson)

- **Connections to K-theory:** Pechenik and Yong's "genomic tableaux" that are used to compute in $K(\mathrm{Gr}(n,k))$ appear naturally as steps in the algorithm.

Geometric consequences (G., Levinson)

- **Connections to K-theory:** Pechenik and Yong's "genomic tableaux" that are used to compute in $K(\mathrm{Gr}(n, k))$ appear naturally as steps in the algorithm.
- Schubert curves can have **arbitrarily high arithmetic genus** (connected ω-orbits with many genomic tableaux appearing).

Geometric consequences (G., Levinson)

- **Connections to K-theory:** Pechenik and Yong's "genomic tableaux" that are used to compute in $K(\mathrm{Gr}(n,k))$ appear naturally as steps in the algorithm.
- Schubert curves can have **arbitrarily high arithmetic genus** (connected ω-orbits with many genomic tableaux appearing).
- Schubert curves can have **arbitrarily many connected components**, and in fact can be a disjoint union of arbitrarily many copies of $\mathbb{P}^1$ (when all tableaux of the given shape and content are fixed by ω).