On a Two-Way Street:
Inducing Parking Spaces to the Hyperoctahedral Group

Alexander Wilson

York University

with J. Carlos Martinez Mori and Pamela Estephania Harris


Alexander Wilson
awilson2@yorku.ca
Employment
York University, Postdoctoral Visitor.
Oberlin College, Visiting Assistant Professor.
Education
Dartmouth College.
Ph.D., Mathematics
Thesis Advisor: Rosa Orellana
Michigan State University.
Bachelor of Science in Advanced Math
Minor in German
Publications
Group Activities and Active Learning
One way I've been able to leverage my skills work-
with students individually is to introduce some group
the classroom. I believe I've been most
strategy while teaching introductory
and Oberlin. We have
to compare the areas of
imagine an optimization
and build flipbooks
in an animation corre-
erivative. These activ-
nation to students for
not the right intro-
provided space in my
students talking
Research
Teaching
Service
CV
Gallery
D from Dartmouth College advised by Rosa Orellana. My research is in
Mystery Set Operation
A tool for students to experiment with a set operation by providing two sets to test.
Student Work:
Research Statement
Alexander Wilson
My research is in the area of algebraic combinatorics. I am primarily interested in studyin
representations of groups and algebras using combinatorial objects like graphs and tableaux. Th
document includes subsections highlighting undergraduate projects and next steps for e
research project throughout.
Diagram Algebras
Centralizer algebras encode the symmetry of a representation. In studying these algeb
consider the following situation. A semisimple algebra A acts faithfully on a vector space
the endomorphisms of V which commute with the A action are denoted B := End_A(V)
is the centralizer algebra of A. In this case, there is a correspondence between i
of A and B. This phenomenon is often called Schur-Weyl duality after
in this form and Weyl who went on to generalized it to study
in the study of A's representations (e.g. tensorin
in B's representations (e.g. restricting
are connected to long-stan
including the Kronec

Outline

Section 1

Parking on a Two-Way Street

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like

When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like

When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like

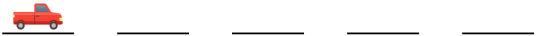
When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like



When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like

When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like

When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like

When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like

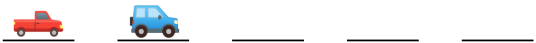
When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like



When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like

When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like

When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like

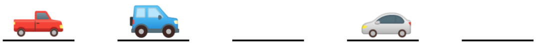
When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like



When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like

When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like

When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like

When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like

When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like

When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like

When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like

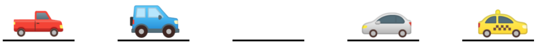
When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like



When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like

When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like

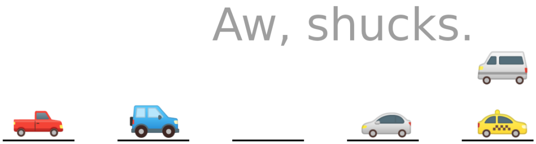
When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like



When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like

When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like

When all cars successfully park, we call the tuple a signed parking function.

Parking on a Two-Way Street

Given a tuple with positive and negative preferences, we define a parking process where

- positive cars drive left-to-right, and
- negative cars drive right-to-left.

For $\alpha = (1, 1, \overline{4}, 4, \overline{5})$, this looks like

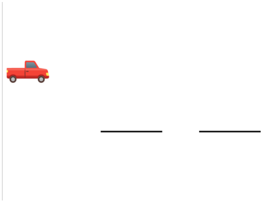
When all cars successfully park, we call the tuple a signed parking function.

Failing to Park

In this model, there are two ways for cars to fail to park:

Positive cars can drive off the right.

$$\alpha = (2, 2)$$



Negative cars can drive off the left.

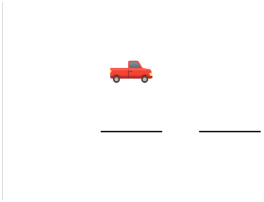
$$\alpha = (1, \overline{1})$$

Failing to Park

In this model, there are two ways for cars to fail to park:

Positive cars can drive off the right.

$$\alpha = (2, 2)$$



Negative cars can drive off the left.

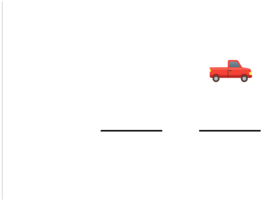
$$\alpha = (1, \bar{1})$$

Failing to Park

In this model, there are two ways for cars to fail to park:

Positive cars can drive off the right.

$$\alpha = (2, 2)$$



Negative cars can drive off the left.

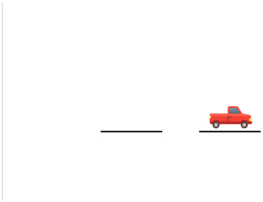
$$\alpha = (1, \bar{1})$$

Failing to Park

In this model, there are two ways for cars to fail to park:

Positive cars can drive off the right.

$$\alpha = (2, 2)$$



Negative cars can drive off the left.

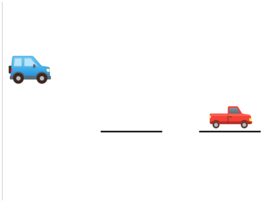
$$\alpha = (1, \bar{1})$$

Failing to Park

In this model, there are two ways for cars to fail to park:

Positive cars can drive off the right.

$$\alpha = (2, 2)$$



Negative cars can drive off the left.

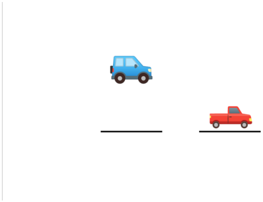
$$\alpha = (1, \bar{1})$$

Failing to Park

In this model, there are two ways for cars to fail to park:

Positive cars can drive off the right.

$$\alpha = (2, 2)$$



Negative cars can drive off the left.

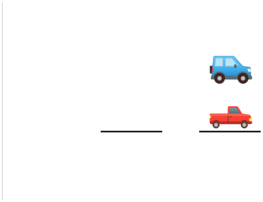
$$\alpha = (1, \bar{1})$$

Failing to Park

In this model, there are two ways for cars to fail to park:

Positive cars can drive off the right.

$$\alpha = (2,2)$$



Negative cars can drive off the left.

$$\alpha = (1,\bar{1})$$

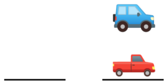
Failing to Park

In this model, there are two ways for cars to fail to park:

Positive cars can drive off the right.

$$\alpha = (2, 2)$$

Ah, I see.



Negative cars can drive off the left.

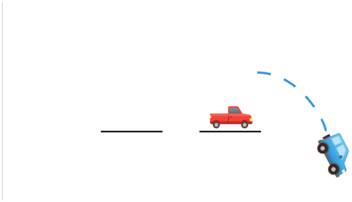
$$\alpha = (1, \bar{1})$$

Failing to Park

In this model, there are two ways for cars to fail to park:

Positive cars can drive off the right.

$$\alpha = (2, 2)$$



Negative cars can drive off the left.

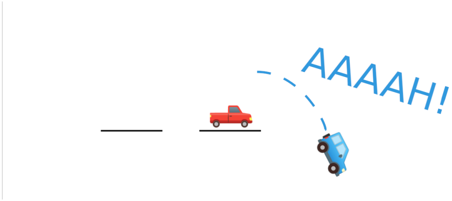
$$\alpha = (1, \bar{1})$$

Failing to Park

In this model, there are two ways for cars to fail to park:

Positive cars can drive off the right.

$$\alpha = (2, 2)$$



Negative cars can drive off the left.

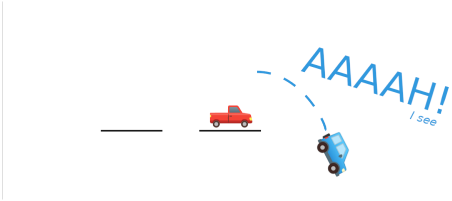
$$\alpha = (1, \overline{1})$$

Failing to Park

In this model, there are two ways for cars to fail to park:

Positive cars can drive off the right.

$$\alpha = (2, 2)$$



Negative cars can drive off the left.

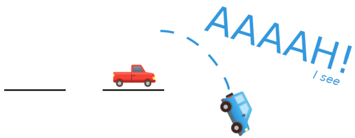
$$\alpha = (1, \overline{1})$$

Failing to Park

In this model, there are two ways for cars to fail to park:

Positive cars can drive off the right.

$$\alpha = (2, 2)$$



Negative cars can drive off the left.

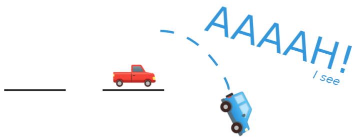
$$\alpha = (1, \bar{1})$$

Failing to Park

In this model, there are two ways for cars to fail to park:

Positive cars can drive off the right.

$$\alpha = (2, 2)$$



Negative cars can drive off the left.

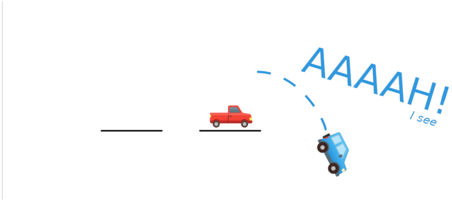
$$\alpha = (1, \bar{1})$$

Failing to Park

In this model, there are two ways for cars to fail to park:

Positive cars can drive off the right.

$$\alpha = (2, 2)$$



Negative cars can drive off the left.

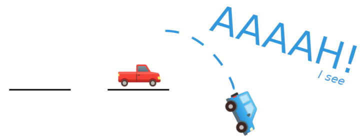
$$\alpha = (1, \overline{1})$$

Failing to Park

In this model, there are two ways for cars to fail to park:

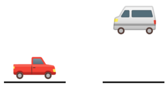
Positive cars can drive off the right.

$$\alpha = (2, 2)$$



Negative cars can drive off the left.

$$\alpha = (1, \overline{1})$$

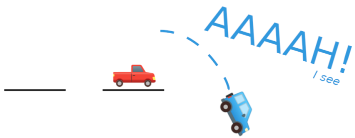


Failing to Park

In this model, there are two ways for cars to fail to park:

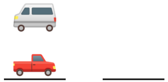
Positive cars can drive off the right.

$$\alpha = (2, 2)$$



Negative cars can drive off the left.

$$\alpha = (1, \bar{1})$$

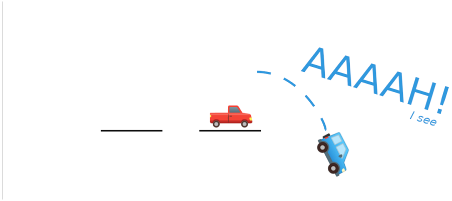


Failing to Park

In this model, there are two ways for cars to fail to park:

Positive cars can drive off the right.

$$\alpha = (2, 2)$$



Negative cars can drive off the left.

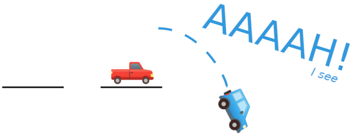
$$\alpha = (1, \bar{1})$$

Failing to Park

In this model, there are two ways for cars to fail to park:

Positive cars can drive off the right.

$$\alpha = (2, 2)$$



Negative cars can drive off the left.

$$\alpha = (1, \overline{1})$$

Failing to Park

In this model, there are two ways for cars to fail to park:

Positive cars can drive off the right.

$$\alpha = (2, 2)$$

Negative cars can drive off the left.

$$\alpha = (1, \overline{1})$$

Failing to Park

In this model, there are two ways for cars to fail to park:

Positive cars can drive off the right.

$$\alpha = (2, 2)$$

Negative cars can drive off the left.

$$\alpha = (1, \bar{1})$$

Section 2

Toggle Maps

Signatures

Given a signed parking function α, its signature Σ is the set of indices at which it has negative preferences. For example, the elements of SPF_2 arranged by signature are as follows.

Σ	α	Σ	α
$\emptyset$	$(1,1)$	$\{2\}$	$(2,\bar{2})$
	$(1,2)$		$(1,\bar{2})$
	$(2,1)$		$(2,\bar{1})$
$\{1\}$	$(\bar{1},1)$	$\{1,2\}$	$(\bar{2},\bar{2})$
	$(\bar{1},2)$		$(\bar{1},\bar{2})$
	$(\bar{2},1)$		$(\bar{2},\bar{1})$

Signatures

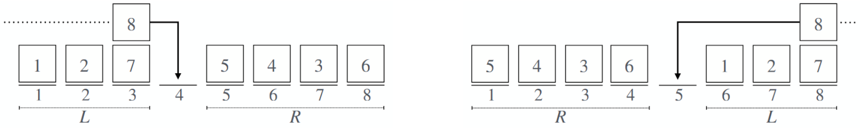
Given a signed parking function α, its signature Σ is the set of indices at which it has negative preferences. For example, the elements of SPF_2 arranged by signature are as follows.

Σ	α	Σ	α
$\emptyset$	$(1,1)$	$\{2\}$	$(2,\bar{2})$
	$(1,2)$		$(1,\bar{2})$
	$(2,1)$		$(2,\bar{1})$
$\{1\}$	$(\bar{1},1)$	$\{1,2\}$	$(\bar{2},\bar{2})$
	$(\bar{1},2)$		$(\bar{1},\bar{2})$
	$(\bar{2},1)$		$(\bar{2},\bar{1})$

Well, that divided nicely... 🤔

Toggle map

We define a family of involutions τ_i on the set SPF_n that toggle the direction that car i is driving. That is, the map τ_i toggles whether i is in the signature of α.



Enumeration

Using these toggle maps, we know that the signed parking functions for each possible signature are equinumerous.

Theorem (Martinez Mori, Harris, W.)

The number of signed parking functions of length n is given by

$$|\mathrm{SPF}_n| = 2^n |\mathrm{PF}_n| = 2^n (n+1)^{n-1}.$$

Section 3

Parking Spaces

Classical Parking Spaces

There are 16 parking functions in PF_3. The symmetric group S_3 acts by rearranging the preference vector.

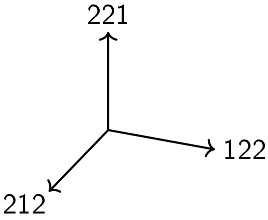
111					
122	212	221			
113	131	311			
112	121	211			
123	132	213	231	312	321

Classical Parking Spaces

There are 16 parking functions in PF_3. The symmetric group S_3 acts by rearranging the preference vector.

111					
122	212	221			
113	131	311			
112	121	211			
123	132	213	231	312	321

We can understand the structure of this action in a more refined way by turning each orbit into a vector space.

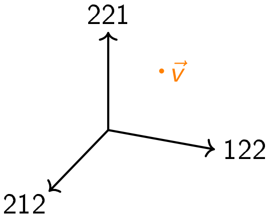


Classical Parking Spaces

There are 16 parking functions in PF_3. The symmetric group S_3 acts by rearranging the preference vector.

111					
122	212	221			
113	131	311			
112	121	211			
123	132	213	231	312	321

We can understand the structure of this action in a more refined way by turning each orbit into a vector space.



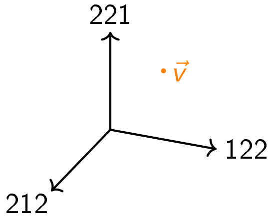
- $\vec{v} = 0.5(122) + 0(212) + 0.7(221)$

Classical Parking Spaces

There are 16 parking functions in PF_3. The symmetric group S_3 acts by rearranging the preference vector.

111					
122	212	221			
113	131	311			
112	121	211			
123	132	213	231	312	321

We can understand the structure of this action in a more refined way by turning each orbit into a vector space.



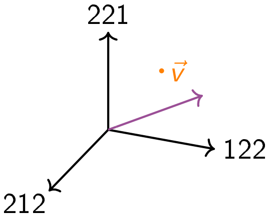
- $\vec{v} = 0.5(122) + 0(212) + 0.7(221)$
- $(1\,2).\vec{v} = 0(122) + 0.5(212) + 0.7(221)$

Classical Parking Spaces

There are 16 parking functions in PF_3. The symmetric group S_3 acts by rearranging the preference vector.

111					
122	212	221			
113	131	311			
112	121	211			
123	132	213	231	312	321

We can understand the structure of this action in a more refined way by turning each orbit into a vector space.



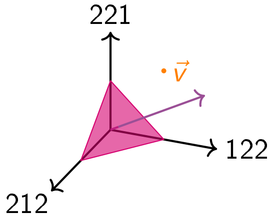
- $\vec{v} = 0.5(122) + 0(212) + 0.7(221)$
- $(1\,2).\vec{v} = 0(122) + 0.5(212) + 0.7(221)$
- $V = \{a(122) + a(212) + a(221) : a \in \mathbb{C}\}$

Classical Parking Spaces

There are 16 parking functions in PF_3. The symmetric group S_3 acts by rearranging the preference vector.

111					
122	212	221			
113	131	311			
112	121	211			
123	132	213	231	312	321

We can understand the structure of this action in a more refined way by turning each orbit into a vector space.



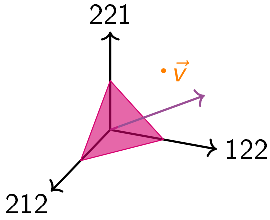
- $\vec{v} = 0.5(122) + 0(212) + 0.7(221)$
- $(1\,2).\vec{v} = 0(122) + 0.5(212) + 0.7(221)$
- $V = \{a(122) + a(212) + a(221) : a \in \mathbb{C}\}$
- $W = \{a(122) + b(212) + c(221) : a + b + c = 0\}$

Classical Parking Spaces

There are 16 parking functions in PF_3. The symmetric group S_3 acts by rearranging the preference vector.

111						
122	212	221	$\cong$	V	$\oplus$	W
113	131	311				
112	121	211				
123	132	213	231	312	321	

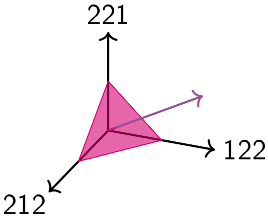
We can understand the structure of this action in a more refined way by turning each orbit into a vector space.



- $\vec{v} = 0.5(122) + 0(212) + 0.7(221)$
- $(1\,2).\vec{v} = 0(122) + 0.5(212) + 0.7(221)$
- $V = \{a(122) + a(212) + a(221) : a \in \mathbb{C}\}$
- $W = \{a(122) + b(212) + c(221) : a + b + c = 0\}$

Classical Parking Spaces

The irreducible representations of S_n are indexed by partitions $\lambda \vdash n$ and are denoted S^λ. For example,



- $V \cong S^{(3)}$
- $W \cong S^{(2,1)}$

As an S_3-representation,

$$\mathbb{C}PF_3 \cong \left(S^{(3)}\right)^{\oplus 5} \oplus \left(S^{(2,1)}\right)^{\oplus 5} \oplus \left(S^{(1,1,1)}\right)$$

Frobenius Character

The Frobenius character is a map

$$\begin{aligned} \mathrm{ch} : \{S_n\text{-representations}\} &\to \mathbb{C}\{s_\lambda : \lambda \vdash n\} \\ S^\lambda &\mapsto s_\lambda \end{aligned}$$

where we call the vector space spanned by the s_λ the space of symmetric functions.
For example,

$$\begin{aligned} \mathbb{C}PF_3 &\cong \left(S^{(3)}\right)^{\oplus 5} \oplus \left(S^{(2,1)}\right)^{\oplus 5} \oplus \left(S^{(1,1,1)}\right) \\ \mathrm{ch}(\mathbb{C}PF_3) &= 5s_{(3)} + 5s_{(2,1)} + s_{(1,1,1)} \end{aligned}$$

(This way of rewriting things may seem a little silly now, but it will make some things much easier to handle later)

Frobenius Character

There is another basis h_λ of symmetric functions for which the Frobenius character of parking functions has a nice expansion:

$$\mathrm{ch}(\mathbb{C}PF_n) = \sum_{\lambda \vdash n} \mathrm{Krew}(\lambda) h_\lambda$$

where for $\lambda = \{1^{m_1}, 2^{m_2}, \ldots\}$ with k parts,

$$\mathrm{Krew}(\lambda) = \frac{1}{n+1} \binom{n+1}{n+1-k, m_1, m_2, \ldots}$$

is the number of noncrossing set partitions of $\{1, 2, \ldots, n\}$ whose blocks have sizes given by λ.

Section 4

A Hyperoctahedral Action

Hyperoctahedral Group

The hyperoctahedral group has three main flavors.

Combinatorial: Signed permutations

Algebraic: Generators and relations

Geometric: Type B Weyl group

Because of the geometric flavor, we denote the hyperoctahedral group by B_n.

Signed Permutations

Let $\langle n \rangle = \{-n, -(n-1), \ldots, -1, 1, 2, \ldots, n\}$. A bijection $f : \langle n \rangle \to \langle n \rangle$ is a signed permutation if $f(-i) = -f(i)$ for all i.

An example of such a signed permutation in one-line notation is

$$21\bar{3}5\bar{4}.$$

Generators and Relations

The group B_n is generated by symbols

- s_i for $1 \leq i \leq n-1$ *(these are the simple transpositions)*
- t_i for $1 \leq i \leq n$ *(these toggle whether the ith position is negative)*

The s_i have the same relations as the simple transpositions in S_n, and the additional relations involving the t_i are

$$t_i t_j = \begin{cases} 1 & \text{if } i = j \\ t_j t_i & \text{if } i \neq j \end{cases}$$

$$t_i s_j = s_j t_{s_j(i)}$$

Hyperoctahedral Action

It turns out that the toggle maps τ_i act very much like the generators t_i:

Lemma (Martinez Mori, Harris, W.)

For $\alpha \in \mathrm{SPF}_n$ *and* $1 \leq i, j \leq n$,

$$\tau_i(\tau_j(\alpha)) = \begin{cases} \alpha & \textit{if } i = j \\ \tau_j(\tau_i(\alpha)) & \textit{if } i \neq j \end{cases}.$$

Based on this fact, we can use the toggle maps to lift the usual action of S_n on PF_n to an action of B_n on SPF_n.

Hyperoctahedral Action

Let $\alpha = (1, 3, \overline{3}, 4, \overline{5})$. If we want to act on α by the simple transposition s_3, we do the following:

$$\begin{array}{c} (1, 3, \overline{3}, 4, \overline{5}) \\ \Big\downarrow{\scriptstyle \tau_3} \\ (3, 1, 1, 4, \overline{5}) \\ \Big\downarrow{\scriptstyle s_3} \\ (3, 1, 4, 1, \overline{5}) \\ \Big\downarrow{\scriptstyle \tau_4} \\ (1, 4, 2, \overline{4}, \overline{5}) \end{array}$$

In general, to apply a permutation, we: (i) use toggles to clear out the signs, (ii) apply the permutation on the positions of the preferences, and (iii) use toggles to reintroduce signs.

Frobenius Character of SPF_n

Definition

Given $\mu \vdash a$ and $\nu \vdash b$ with $a + b = n$, define the signed Kreweras number $\mathrm{Krew}(\mu, \nu)$ to be

$$\mathrm{Krew}(\mu, \nu) = \sum_{\alpha,\beta} \mathrm{Krew}(\alpha + \beta)$$

where the sum is over weak compositions α and β such that $\mathrm{sort}(\alpha) = \mu$, $\mathrm{sort}(\beta) = \nu$, and $(\alpha + \beta) \vdash n$.

Frobenius Character of SPF_n

Definition

Given $\mu \vdash a$ and $\nu \vdash b$ with $a + b = n$, define the signed Kreweras number $\mathrm{Krew}(\mu, \nu)$ to be

$$\mathrm{Krew}(\mu, \nu) = \sum_{\alpha, \beta} \mathrm{Krew}(\alpha + \beta)$$

where the sum is over weak compositions α and β such that $\mathrm{sort}(\alpha) = \mu$, $\mathrm{sort}(\beta) = \nu$, and $(\alpha + \beta) \vdash n$.

The Frobenius character of a hyperoctahedral representation lies in a space spanned by two copies of symmetric functions: $\{s_\lambda(\mathrm{x})\}$ and $\{s_\lambda(\mathrm{y})\}$.

Frobenius Character of SPF_n

Definition

Given $\mu \vdash a$ and $\nu \vdash b$ with $a + b = n$, define the signed Kreweras number $\mathrm{Krew}(\mu, \nu)$ to be

$$\mathrm{Krew}(\mu, \nu) = \sum_{\alpha, \beta} \mathrm{Krew}(\alpha + \beta)$$

where the sum is over weak compositions α and β such that $\mathrm{sort}(\alpha) = \mu$, $\mathrm{sort}(\beta) = \nu$, and $(\alpha + \beta) \vdash n$.

The Frobenius character of a hyperoctahedral representation lies in a space spanned by two copies of symmetric functions: $\{s_\lambda(\mathrm{x})\}$ and $\{s_\lambda(\mathrm{y})\}$.

Theorem (Martinez Mori, Harris, W.)

The Frobenius character of the signed parking space is given by

$$\mathrm{ch}(\mathbb{C}\mathrm{SPF}_n) = \sum_{|\mu| + |\nu| = n} \mathrm{Krew}(\mu, \nu) h_\mu(\mathrm{x}) h_\nu(\mathrm{y}).$$

Some Open Questions

- Characterize the tuples in SPF_n by inequalities.
- An interpretation for $\mathrm{Krew}(\lambda, \mu)$ in terms of *(type B?)* noncrossing set partitions *(currently, we interpret them in terms of certain decorated Dyck paths)*.

Thank you!

Signed Dyck Paths

Definition

A signed Dyck path is a Dyck path whose up-steps are labeled with a sign $+$ or $-$ and an up-step with a positive sign cannot immediately follow an up-step with a negative sign.