Imperial College of Science, Technology and Medicine
Department of Computing

Diagrammatic Algebra for Equivariant Neural Network Architectures

Edward Pearce-Crump

Submitted in part fulfilment of the requirements for the degree of
Doctor of Philosophy in Computing of Imperial College London
October 2024

Abstract

Group equivariant neural networks is an area of deep learning that looks at how symmetries can be encoded in neural network architectures as an inductive bias. Historically, many group equivariant neural networks have been designed using either group convolutions or tensor power representations that have been decomposed into irreducible representations. Whilst these approaches have value, they come with drawbacks. In the former, a change of basis into the Fourier domain is often required, which can be computationally expensive. In the latter, the irreducible decomposition of a tensor power representation for most groups is unknown.

We instead establish weight tying as a fundamental paradigm for constructing group equivariant neural networks. We show that, for many important groups, including the symmetric, alternating, orthogonal, special orthogonal, and symplectic groups, we can determine the weight matrices that appear in group equivariant neural networks having tensor power representations of $\mathbb{R}^n$ as their layer spaces without needing to decompose them into irreducible representations. We study the combinatorics of set partition diagrams that are associated with each group to achieve a full characterisation of these weight matrices. We create a deeper structure for understanding the group equivariant neural networks themselves by developing a monoidal category theoretic framework and use it to construct a fast multiplication algorithm for the linear layer functions for four of these groups. We extend this framework to characterise the equivariant weight matrices for the automorphism group of a graph. Finally, we show that the equivariant neural network paradigm can be broadened to quantum groups, which often describe symmetries in non-commutative geometries. In particular, we use Woronowicz's version of Tannaka–Krein duality to derive Compact Matrix Quantum Group Equivariant Neural Networks. We hope that our work will inspire others to develop neural network architectures that encode different algebraic structures as an inductive bias.

Copyright

Statement of Originality

I declare that I am the sole author of this thesis, and that the work that is presented within is my own, unless otherwise stated.

Edward Pearce-Crump
Imperial College London
October 2024

Acknowledgements

I would like to express my gratitude to:

- My supervisor, Professor William J. Knottenbelt, for your guidance, support, and belief in me. I am particularly grateful that you encouraged me to pursue my curiosity without hesitation, which has resulted in me having the time of my life during these past four years.
- The Department of Computing at Imperial College London, for honouring me with the Doctoral Scholarship for Applied Research, and the anonymous donor, for supporting me with funding under this award.
- My examiners, Professor Alessio Lomuscio and Professor Haggai Maron, for their careful reading of my thesis and for the insightful discussion during my viva.
- The judging panel at G-Research, for recognising my work by awarding me with second prize in their Imperial College PhD Prize competition.
- Dr Shubhendu Trivedi, for championing my research. Your support came at a time when I needed the impetus to pursue my ideas further.
- Edward Lam, for your mentorship and for your long-standing belief in me.
- Chen Lin, for your friendship and for your genuine interest in my research. I look forward to our future collaborations.
- Mia Hulla, for your encouragement and for our engaging conversations, which I always enjoy.
- Julianna Bor and Joe Stacey, for your friendship, support, and advice. You have both made my overall experience much greater.
- To every teacher I have had, for nurturing my curiosity and for cultivating in me what I hope will be a lifelong passion for learning.
- My sister Alexandra and my brother Andrew, for being there for me over the years.
- My parents, Donald and Amanda. No words can convey the love I have for you. You always tell me to never give in, and I am made stronger by your unwavering belief in me. I hope that I will continue to make you both proud.

ad matrem amandam meam et patrem

'Drawing is not what one sees but what one can make others see.'

Edgar Degas (1834-1917)

Notation

Group Theory	
Symbol	**Description**
$GL(V)$	General linear group of invertible linear maps $V \to V$
$GL(n)$	General linear group of invertible $n \times n$ matrices
$G(n)$	Generic subgroup of $GL(n)$
$SL(n)$	Special linear group
$(\cdot, \cdot)$	Non-degenerate, symmetric bilinear form
$\langle \cdot, \cdot \rangle$	Non-degenerate, skew-symmetric bilinear form
$O(n)$	Orthogonal group
$SO(n)$	Special orthogonal group
$Sp(n)$	Symplectic group, $n = 2m$
S_n	Symmetric group
A_n	Alternating group
$U(n)$	Unitary group
H_n	Hyperoctahedral group
B_n	Bistochastic group
ρ	Representation ρ of a group
(V, ρ_V)	Representation ρ_V of a group, V a vector space
$\mathrm{Hom}_{G(n)}(V, W)$	Vector space of linear maps $V \to W$ that are equivariant to $G(n)$
$\mathrm{End}_{G(n)}(V)$	Same as $\mathrm{Hom}_{G(n)}(V, V)$
$((\mathbb{R}^n)^{\otimes k}, \rho_k)$	k-order tensor power representation of $G(n)$
$(((\mathbb{R}^n)^*)^{\otimes k}, (\rho_k^{-1})^\top)$	Contragredient representation for $((\mathbb{R}^n)^{\otimes k}, \rho_k)$

Set Partitions	
Symbol	**Description**
π	Set partition, typically of $\{1, \dots, l+k\}$
Π_{l+k}	Set of all set partitions of $\{1, \dots, l+k\}$
$\Pi_{l+k,n}$	Set of all set partitions of $\{1, \dots, l+k\}$ having at most n blocks
d_π	Set partition diagram; diagram basis element; (k, l)–partition diagram
D_π	Diagram basis matrix; image of a diagram basis element for S_n

x_π	Orbit basis diagram/element
X_π	Orbit basis matrix; image of an orbit basis element for S_n
β	Brauer partition, typically of $\{1, \dots, l+k\}$
d_β	Brauer diagram
E_β	Image of a Brauer diagram for $O(n)$
F_β	Image of a Brauer diagram for $Sp(n)$
α	$(l+k)\backslash n$–partition
d_α	$(l+k)\backslash n$–diagram
H_α	Image of an $(l+k)\backslash n$–diagram for $SO(n)$
$\left\{ {n \atop t} \right\}$	Stirling number of the second kind
$\mathrm{B}(n)$	n^{th} Bell number
$\mathrm{B}(n,k)$	k-restricted n^{th} Bell number
B_i	i^{th} block in a set partition π
$P_k^l(n)$	Partition vector space
$B_k^l(n)$	Brauer vector space
$D_k^l(n)$	Brauer–Grood vector space
det	Determinant map $(\mathbb{R}^n)^{\otimes n} \to \mathbb{R}$
λ_A	$\sum_{i \in A} \lambda_i$ for A an index set

Category Theory

Symbol	**Description**
$\mathcal{C}$	Category
$Ob(\mathcal{C})$	Collection of objects in a category $\mathcal{C}$
$\mathrm{Hom}_{\mathcal{C}}(X, Y)$	Collection of morphisms between objects X, Y in a category $\mathcal{C}$
$f : X \to Y$	A morphism between objects X, Y in $\mathcal{C}$; an arrow
$\circ$	Composition of morphisms
1_X	Identity morphism $X \to X$
$\mathcal{F} : \mathcal{C} \to \mathcal{D}$	Functor from a category $\mathcal{C}$ to a category $\mathcal{D}$
$\otimes$, $\otimes_{\mathcal{C}}$	Bifunctor for a monoidal category $\mathcal{C}$; tensor product
$\mathbb{1}$, $\mathbb{1}_{\mathcal{C}}$	Unit object of a monoidal category $\mathcal{C}$
Set	Category of sets
$\mathbf{Vect}_{\mathbb{R}}$	Category of vector spaces over $\mathbb{R}$
Group	Category of groups

$\mathbf{Mat}_{\mathbb{R}}$	Category of matrices with elements in $\mathbb{R}$
$\mathcal{P}(n)$	Partition category
$\mathcal{B}(n)$	Brauer category
$\mathcal{BG}(n)$	Brauer–Grood category
$\text{Rep}(G)$	Representation category for a group G
$\mathcal{C}(G(n))$	Category of tensor power representations for $G(n) \subseteq GL(n)$

Graph Theory

Symbol	Description
G	Graph
$V(G)$	Set of vertices for a graph G
$E(G)$	Set of edges for a graph G
A_G	Adjacency matrix for a graph G
$\overline{G}$	Complement of a graph G
K_n	Complete graph on n vertices
C_n	Cycle graph on n vertices
$\text{Aut}(G)$	Automorphism group of a graph G
$M^{k,l}$	Spider map
$\boldsymbol{H}$	Bilabelled graph
$[\boldsymbol{H}]$	Isomorphism class of the bilabelled graph $\boldsymbol{H}$
$\mathcal{G}$	Category of all bilabelled graphs
$X_{\boldsymbol{H}}^G$	G-homomorphism matrix of $[\boldsymbol{H}]$
$\mathcal{C}^G$	Category of all G-homomorphism matrices

General Topology

Symbol	Description
τ_X	Topology for a set X
(X, τ_X)	Topological space
d_X	Metric for a set X
(X, d_X)	Metric space
$B(x, \epsilon)$	Open ball of radius ϵ in a metric space (X, d_X) centred at a point $x \in X$
$\|\cdot\|$	Norm
$\|\cdot\|_2$	Euclidean Norm for $\mathbb{R}^n$ or $\mathbb{C}^n$

$\overline{V}$	Closure of a set V in a topological space (X, τ_X)
(x_n)	Sequence of points in a topological space (X, τ_X)

Compact Matrix Quantum Groups

Symbol	Description
A	Algebra
a^*	Involution of an element a in a *-algebra A
$\mathcal{B}(\mathcal{H})$	C^*-algebra of bounded linear operators on a Hilbert space $\mathcal{H}$
$M_n(\mathbb{C})$	C^*-algebra of $n \times n$ matrices with elements in $\mathbb{C}$
$M_n(A)$	C^*-algebra of $n \times n$ matrices with elements in a C^*-algebra A
$C(X)$	C^*-algebra of continuous functions $X \to \mathbb{C}$ for a compact Hausdorff space X
$\odot$	Algebraic tensor product
$\otimes_{\text{min}}$	Minimal tensor product
$C^*(E \mid R)$	Universal C^*-algebra for generators E and relations R
γ	Seminorm
$A = (G(n), u)$	Compact matrix quantum group with its n-dimensional fundamental representation u
Δ	Comultiplication map $A \to A \otimes_{\text{min}} A$
S_n^+	Symmetric quantum group
$O(n)^+$	Orthogonal quantum group
$U(n)^+$	Unitary quantum group
$\{\circ, \bullet\}$	Two-coloured set
w_k	Word of length k formed from the two-coloured set $\{\circ, \bullet\}$
$\text{FundRep}_{G(n)}$	Fundamental representation category for the compact matrix quantum group $G(n)$
$\mathcal{K}(n)$	Two-coloured category of partitions

Miscellaneous

Symbol	Description
$[n]$	The set $\{1, \ldots, n\}$
$[n]^k$	Cartesian product set $\{1, \ldots, n\}^k$
(I, J)	Tuple, typically $I \in [n]^l$ and $J \in [n]^k$

$E_{I,J}$	Matrix unit whose (I, J)-entry is 1, otherwise 0
e_I	If $I \in [n]^k$, then $e_I = e_{i_1} \otimes \cdots \otimes e_{i_k}$, a standard basis vector in $(\mathbb{R}^n)^{\otimes k}$
$\delta_{i,j}$	Kronecker delta
$\epsilon_{i,j}$	Value of $\langle e_i, e_j \rangle$ on the symplectic basis of $\mathbb{R}^n$, $n = 2m$
$\sum$	Sum
$\prod$	Product
λ	Scalar
$n!$	Factorial
$n!!$	Double Factorial
O	Big O notation
$\preceq$	Partial Ordering
$\mathrm{sgn}(\sigma)$	Sign of a permutation $\sigma \in S_n$
$\bigcup$	Set union

Contents

Chapter 1

Introduction

Anyone who has ever had the pleasure of teaching mathematics knows how hard it is to communicate the core ideas of a subject using only words and symbols. In essence, the teacher is attempting to transfer to their students the mental representations that they have developed using whatever languages they have to hand, be those natural or mathematical. These imperfect tools create many difficulties for students as they typically have to translate the words and symbols back into mental representations of their own in order to develop the intuition that is needed to understand a subject at a fundamental level. The teacher, however, has one additional tool that they can use: they can draw diagrams. Often, many of the difficulties that are caused by words and symbols alone can be overcome immediately once the right diagram or two is introduced. Moreover, for many subjects, just a few diagrams can be enough to encapsulate, and hence can be used to generate, most of the content that appears in an entire subject.

For example, when we are first taught complex numbers, we are asked to express complex numbers such as $z = \sqrt{2}e^{\frac{3\pi i}{4}}$ in the form $z = x + iy$ for some real numbers x and y. To defend against this oncoming attack, the uninitiated panic and start relying on formulae that they have learned by rote, such as $\sqrt{x^2 + y^2} = \sqrt{2}$ and $\tan^{-1}(\frac{y}{x}) = \frac{3\pi i}{4}$, hoping that this will solve the problem. Inevitably, following this approach results in many difficulties for the student. But with the right diagram, the solution becomes immediately apparent. In this case, by drawing a line of length $\sqrt{2}$ from the origin at an angle of $\frac{3\pi i}{4}$ from the positive x-axis on the Argand diagram, together with knowledge of the $1 : 1 : \sqrt{2}$ triangle — see Figure 1.1 — we immediately see that $\sqrt{2}e^{\frac{3\pi i}{4}}$ is the complex number is $-1 + i$. In fact, much of the material on complex numbers can be reduced to drawing them in the Argand diagram and then applying a few simple triangles that are taught in trigonometry.

The key point in general is that even though the diagrams are not necessarily rigorous, they lead to a level of understanding and intuition that cannot be attained through words and symbols alone.

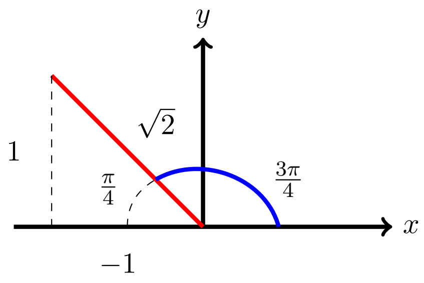


Figure 1.1: By drawing the complex number $\sqrt{2}e^{\frac{3\pi i}{4}}$ in the Argand diagram, it is easy to see that its expression in the form $x + iy$ is $-1 + i$.

In this thesis, we argue that our core ideas, which must be communicated formally using words and symbols, can be understood far more easily by studying just a few important diagrams. We might even be so bold as to say that the diagrams are the thesis, in that they encapsulate and can be used to generate most of the content that is presented. Specifically, we use the diagrams in three main ways: to encode algebraic structure, to represent a piece of computation, or to derive algebraic structure. These diagrams provide the necessary intuition that makes our results almost immediately apparent.

The starting point for our story can be found in 1897, when Ferdinand Georg Frobenius, a professor at the University of Berlin, published a landmark paper that laid the foundations for an entirely new area of mathematics. In his paper, titled *Über die Darstellung der endlichen Gruppen durch lineare Substitutionen* [*On the Representation of Finite Groups by Linear Substitutions*], Frobenius (1897) defined a representation of a group for the first time. Many mathematicians, such as the group theorist William Burnside (1897a,b, 1900a,b, 1904), made major contributions to the new representation theory. But it was Frobenius' student who reshaped the field into a form that is effectively the one we see today.

Issai Schur became a professor at the University of Berlin himself, and during his time there, he established the leading research centre for this new type of mathematics. In 1905, Schur formulated group representation theory in terms of matrices, which made the field more widely accessible (Schur, 1905). His most well-known contribution is the result known as Schur's Lemma, which characterises all of the possible equivariant linear maps between irreducible representations of a group.

However, he also published another result that is less well-known, one that began with his investigations into the polynomial representations of the general linear group $GL(n)$ in his doctoral thesis of 1901, titled *Über eine Klasse von Matrizen, die sich einer gegebenen Matrix zuordnen lassen* [*On a Class of Matrices That Can Be Assigned to a Given Matrix*] (Schur, 1901) . This result, which appeared in a paper of his in 1927, titled *Über die rationalen Darstellungen der allgemeinen linearen Gruppe*, [*On the Rational*

Representations of the General Linear Group] (Schur, 1927), rederived all of the results that appeared in his doctoral thesis of 1901, and today is known as Schur–Weyl duality after it was popularised by Hermann Weyl in his book of 1939 titled *The Classical Groups* (Weyl, 1939).

Schur used ideas from the representation theory of the symmetric group to obtain his duality. In 1900, Alfred Young introduced combinatorial objects that are now known as Young diagrams to represent all possible integer partitions of a natural number n (Young, 1900). These diagrams were then used by Frobenius (1903) to show that the irreducibles of the symmetric group S_n correspond bijectively with all possible integer partitions of n.

Schur–Weyl duality describes the irreducible representations of the general linear group $GL(n)$ in terms of integer partitions through dual actions of $GL(n)$ and S_k on the tensor power space $(\mathbb{R}^n)^{\otimes k}$. In particular, Schur established a one-to-one correspondence between the irreducibles of the general linear group $GL(n)$ and the irreducibles of the symmetric group S_k that appear in the decomposition of the tensor power space $(\mathbb{R}^n)^{\otimes k}$, namely

$$(\mathbb{R}^n)^{\otimes k} \cong \bigoplus_{\lambda \in \Lambda(k,n)} V_n^\lambda \otimes S_k^\lambda \tag{1.1}$$

In (1.1), the irreducibles V_n^λ of the general linear group $GL(n)$ are indexed by the same set of integer partitions λ of k into at most n parts that index irreducibles S_k^λ of the symmetric group S_k. It is this result that became known as Schur–Weyl duality. In the years since, a number of other Schur–Weyl dualities have been shown to exist (Brauer, 1937; Martin, 1990, 1994, 1996; Jones, 1994; Halverson and Ram, 2022). Some of them were established by Richard Brauer, a student of Schur's, who would go on to make a number of significant contributions of his own to representation theory. We discuss some of these Schur–Weyl dualities in the opening chapters of this thesis.

Schur, upon his election to the Berlin Academy in 1922, gave his initial address to the members of the Academy. In his speech, he said that "what fascinated me so extraordinarily in these investigations [into the representation theory of groups] was the fact that [...] there arose a new and fertile chapter of Algebra which is also important for Geometry and Analysis and which was distinguished by great beauty and perfection" (Curtis, 1999). To begin with, we show that the duality which bears his name makes for a surprising relationship between the representation theory that was first discovered over a hundred years ago by Frobenius and an important subdomain of deep learning that is known as group equivariant neural networks.

1.1 Motivation

Ironically, Frobenius is said to have been angered by any application of his pure mathematics to topics of a practical nature. So, undoubtedly, he would have been horrified to learn that symmetry, expressed through the representation theory that he initiated, has become one of the key principles for constructing neural networks.

Group equivariant neural networks is an area of deep learning that looks at the question of how best to learn from data that has an underlying symmetry (Cohen and Welling, 2016, 2017; Gilmer et al., 2017; Ravanbakhsh et al., 2017; Qi et al., 2017; Zaheer et al., 2017; Cohen et al., 2018; Kondor and Trivedi, 2018; Thomas et al., 2018; Maron et al., 2019a,b, 2020; Ravanbakhsh, 2020; Finzi et al., 2021; Villar et al., 2021; Yarotsky, 2022). Symmetries in mathematics are usually described in terms of groups. They often arise in problems that come from physics, where the data that is generated by a physical process has a certain type of symmetry that is inherent in the data itself. Examples include molecular structures (Thomas et al., 2018; Hy et al., 2019; Fuchs et al., 2020; Satorras et al., 2021), which have rotational and translation symmetry, or point clouds (Qi et al., 2017; Zaheer et al., 2017), which have permutation symmetry.

Formally, a group equivariant neural network for a group G is a neural network such that

1. each layer space is a representation of the group G, that is, a choice of vector space V and a group homomorphism $G \to GL(V)$, and
2. the maps between layer spaces that appear in the network are equivariant to G, that is, if $f : V \to W$ is a function between layer spaces V and W, then $f(g \cdot v) = g \cdot f(v)$ for all $g \in G$ and $v \in V$.

These networks guarantee that if we change the input to such a network by acting on it with an appropriate symmetry operation in G, then we know exactly what the output of the network will be. Consequently, these networks come with advantages such as requiring less training data to learn — since all symmetries on the input data will be learned automatically by the network — and they have fewer parameters overall, which makes them easier to train. The equivariance property, in particular, reduces the space of functions that can be learned. This usually results in simpler, more interpretable models that generalise better to unseen data.

The convolutional neural network (CNN) was one of the first neural networks to use equivariance in its construction (LeCun et al., 1989; Krizhevsky et al., 2012). A convolutional neural network is built using convolutional layers that are equivariant to translational symmetries, enabling it to learn effectively from data that has an underlying Euclidean or grid-like structure. The success of convolutional neural networks in these domains inspired the development of an entirely new field of deep learning, popularly known as *Geometric Deep Learning* (Bronstein et al., 2017, 2021). The primary goal of Geometric Deep Learning is to develop principled approaches for constructing neural network architectures that incorporate prior knowledge as inductive biases, particularly by exploiting symmetries that are inherent in a variety of domains, whether they be Euclidean or non-Euclidean. This framework provides a principled foundation for studying group equivariant neural network architectures. We now review the historical development of these architectures.

Given the success of convolutional neural networks and the fact that convolutions can be defined for other groups, it is natural that, in the years following the introduction of the convolutional neural network, several group equivariant neural networks were designed using group convolutions. Group Equivariant Convolutional Neural Networks (G-CNNs) (Cohen and Welling, 2016) generalised the notion of translation equivariance to discrete groups generated by translations, reflections, and rotations. Some important classes of equivariant convolutional neural networks were built upon G-CNNs: Steerable CNNs (Cohen and Welling, 2017; Worrall et al., 2017; Thomas et al., 2018; Weiler et al., 2018a,b; Weiler and Cesa, 2019) were designed to model symmetries in both Euclidean and non-Euclidean spaces, and Spherical CNNs (Cohen et al., 2017, 2018; Esteves et al., 2018; Kondor et al., 2018) were designed to learn from data on spherical surfaces by exploiting their rotational symmetries. In a complementary line of work, Kondor and Trivedi (2018) proved that neural networks are equivariant to the action of any compact group if and only if they use a general notion of convolution. In addition, Cohen et al. (2019a) gave a general theory of G-CNNs on homogeneous spaces. However, the desire to learn from graph-structured data led to the development of a number of different approaches for defining convolution on graphs, which is consistent with the challenge of applying convolution to non-Euclidean structures. Several authors developed spectral graph convolutional neural networks, which perform spectral convolutions on graphs by using a generalised Fourier basis that is derived from the eigenvectors of the graph Laplacian (Bruna et al., 2014; Defferrard et al., 2016; Kipf and Welling, 2017). An alternative approach for extending convolution to graphs involved designing functions that, for each node, aggregate information from its local neighborhood in order to update its feature vector (Duvenaud et al., 2015; Atwood and Towsley, 2016; Niepert et al., 2016; Hamilton et al., 2017; Velicković et al., 2018). Gilmer et al. (2017) introduced a message passing framework for learning from graphs and showed that many of these methods can be unified under it. This framework has since been widely adopted in the development of a number of important equivariant Graph Neural Network architectures (Satorras et al., 2021; Schütt et al., 2021; Batatia et al., 2022).

Despite the value in using group convolutions to create group equivariant neural networks, this approach has a number of drawbacks. As noted in Kondor and Trivedi (2018), it can be difficult sometimes to determine the correct notion of convolution to use since it depends on the domain in which the data lives. Moreover, most of these approaches require a change of basis either through the use of a Fourier transform (Cohen et al., 2017; Worrall et al., 2017; Cohen et al., 2018; Esteves et al., 2018; Kondor and Trivedi, 2018; Kondor et al., 2018) — in order to use that the Fourier transform of a convolution is the element-wise product of the individual Fourier transforms — or by a direct decomposition into irreducible representations of a group (Cohen and Welling, 2017; Thomas et al., 2018; Weiler et al., 2018a; Weiler and Cesa, 2019), if such a decomposition exists and can be found.

Another common approach for building group equivariant neural networks is to take the space of equivariant linear maps between layer spaces that appear in the network, decompose these layer spaces into their irreducible decompositions, and then apply Schur's Lemma to obtain the parameters. Given that no non-trivial equivariant linear maps $\mathbb{R}^n \to \mathbb{R}^m$ exist for some groups (Blum-Smith and Villar, 2023), the layer spaces that are typically used in this approach are tensor power spaces of $\mathbb{R}^n$, since tensors naturally appear throughout much of deep learning (Lim, 2021). The group is usually a subgroup of $GL(n)$, and it acts diagonally on each tensor power by matrix multiplication. Consequently, the equivariant linear maps under consideration are of the form $(\mathbb{R}^n)^{\otimes k} \to (\mathbb{R}^n)^{\otimes l}$ for some non-negative integers k and l. This approach has been successful, in part, for some groups, such as the special orthogonal group $SO(3)$ (Kondor, 2018; Kondor et al., 2018; Thomas et al., 2018; Dym and Maron, 2021), the proper orthochronous Lorentz group $SO(1,3)^+$ (Bogatskiy et al., 2020), and the special Euclidean group $SE(3)$ (Fuchs et al., 2020). However, the major drawback with this approach is that the irreducible decomposition of a tensor power representation for most groups is unknown (Blum-Smith and Villar, 2023). This is known as the Clebsch–Gordan problem. Even if the Clebsch–Gordan decomposition is known, applying it in practice in a neural network is computationally expensive, since it involves, at best, a change of basis into the Fourier basis at the input stage of the network and another change of basis at the output stage of the network.

Upon starting this period of research, the kernel of a new idea was being explored alongside the two main approaches for constructing group equivariant neural networks. This new idea involved tying weights within the weight matrices that appear in the linear layer functions of group equivariant neural networks. This approach emerged from the characterisation of permutation invariant and equivariant linear maps on first order tensor power spaces, that is, on data given in the form of a set (Qi et al., 2017; Zaheer et al., 2017). Ravanbakhsh et al. (2017) showed that, for linear maps from $\mathbb{R}^n$ to $\mathbb{R}^m$ that are equivariant to a finite subgroup of the symmetric group, their equivariance properties can be explained by the symmetries of their associated weight matrices, namely by relating each weight matrix to a bipartite graph between a set of n vertices and a set of m vertices. Hartford et al. (2018) introduced standard bases of permutation invariant and equivariant linear maps between high-order tensor power spaces that model interactions between sets of differing sizes, each of which is acted upon by a different permutation group. In a seminal work on equivariance relating to the symmetries of graphs and hypergraphs, Maron et al. (2019a) proceeded to characterise all permutation invariant and equivariant linear maps between high-order tensor power spaces of $\mathbb{R}^n$.

Given the drawbacks that are associated with the two main approaches, we ask ourselves the following question: can we establish weight tying as a fundamental paradigm for constructing group equivariant neural networks?

1.2 Objectives

The main objective is to derive a method that is grounded in proper mathematical theory for constructing group equivariant neural networks that does not use any group convolutions; that does not require any decomposition of the layer spaces into irreducible representations of a group; and that does not involve any changes of basis.

1.3 Contributions

We can divide our contributions into three overarching themes.

In our first major contribution, we achieve our main objective. We show that Schur–Weyl duality between a number of diagram algebras and their associated groups can be adapted to characterise all of the possible group equivariant neural networks whose layers are some tensor power of $\mathbb{R}^n$ for many important groups, including the symmetric, alternating, orthogonal, special orthogonal and symplectic groups. Crucially, we do not use any group convolutions; we do not decompose the layer spaces into irreducibles of the groups in question; and we do not require any changes of basis. In doing so, we show that set partition diagrams play a vital role in group equivariant neural networks.

The second major contribution of our work is the realisation that certain subsets of set partition diagrams can be given the structure of a strict $\mathbb{R}$–linear monoidal category. We establish strict $\mathbb{R}$–linear monoidal functor relationships between these partition categories and group representation categories for the tensor power layer spaces. We use these functors to obtain an algorithm for performing the linear equvariant layer functions for four groups — the symmetric, orthogonal, special orthogonal, and symplectic groups — that is exponentially faster compared with the standard matrix multiplication approach. We then extend the category theory framework to characterise all of the possible neural networks whose layers are some tensor power of $\mathbb{R}^n$ that are equivariant to the automorphism group of a graph by using bilabelled graph diagrams, of which set partition diagrams form a subset.

In the third major contribution of our work, we go far beyond our original objectives by constructing a whole new class of equivariant neural networks. These neural networks are equivariant to compact matrix quantum groups, which, unlike groups, are algebras that generalise the space of continuous functions on compact matrix groups. Where groups are used to describe symmetries in classical (commutative) geometries, compact matrix quantum groups are often used to describe symmetries in non-commutative geometries. In particular, we show that Woronowicz's version of Tannaka–Krein duality, a result that appears in the theory of quantum groups and in non-commutative geometry, can be used to construct this new class of neural networks which we term Compact Matrix Quantum Group Equivariant Neural Networks. We also show that, for the so-called easy compact matrix quantum groups, the weight matrices that appear in these neural networks can be characterised using either one- or two-coloured set partition diagrams.

1.4 Publications

The following is a list of publications that is the result of the work that was carried out during this period of research. We have noted the chapter number after each publication reference where that publication forms the basis of a chapter in this thesis.

Conference Publications

1. Edward Pearce-Crump. (2023a) Brauer's Group Equivariant Neural Networks. *Proceedings of the 40th International Conference on Machine Learning* (ICML 2023), PMLR 202:27461-27482. [Live Oral] [Chapter 5]

2. Edward Pearce-Crump. (2023b) How Jellyfish Characterise Alternating Group Equivariant Neural Networks. *Proceedings of the 40th International Conference on Machine Learning* (ICML 2023), PMLR 202:27483-27495. [Poster] [Chapter 4]

3. Edward Pearce-Crump and William J. Knottenbelt. (2024b) Graph Automorphism Group Equivariant Neural Networks. *Proceedings of the 41st International Conference on Machine Learning* (ICML 2024), PMLR 235:40051-40077. [Poster] [Chapter 8]

4. Edward Pearce-Crump. (2024) Connecting Permutation Equivariant Neural Networks and Partition Diagrams. *Proceedings of the 27th European Conference on Artificial Intelligence* (ECAI 2024), IOS Press Ebooks 392:1511-1518. [Live Oral] [Chapter 3]

Preprints

5. Edward Pearce-Crump. (2023c) Compact Matrix Quantum Group Equivariant Neural Networks. `arXiv:2311.06358`. [Chapter 9]

6. Edward Pearce-Crump and William J. Knottenbelt. (2024a) A Diagrammatic Approach to Improve Computational Efficiency in Group Equivariant Neural Networks. `arXiv:2412.10837`. [Chapters 6 and 7]

Other

7. Edward Pearce-Crump. (2022) A Multigraph Approach for Performing the Quantum Schur Transform. *Quantum Computing Theory in Practice* (QCTiP 2023) [Poster]

1.5 Thesis Outline

The remaining chapters are organised as follows.

Chapter 2: Fundamental Concepts

We introduce the groups that appear throughout this thesis, all of which are subgroups of the general linear group. We discuss the group equivariant neural networks in question and explain how the tensor power layer spaces are representations for each of these groups. We review some graph theory and introduce some fundamental concepts in category theory, namely categories and functors. We finish the chapter by defining some concepts from general topology that we will use in Chapter 9.

Chapter 3: Partition Algebra and the Symmetric Group

We show that an entirely different approach from the one that appears in Maron et al. (2019a) can be used to obtain a full characterisation of all of the possible permutation equivariant weight matrices that appear between any two layers that are tensor power spaces of $\mathbb{R}^n$. The starting point for our approach is Schur–Weyl duality between the partition algebra and the symmetric group that was simultaneously found by Martin (1990, 1994, 1996) and Jones (1994). We adapt the definition of the partition algebra by considering a partition vector space instead, and show how to construct a so-called orbit basis of set partition diagrams for this vector space. We use the orbit basis to obtain a basis for the space of permutation equivariant linear maps between any two tensor power spaces of $\mathbb{R}^n$, and, in doing so, characterise the permutation equivariant weight matrices that appear between any two tensor power spaces of $\mathbb{R}^n$.

Chapter 4: Jellyfish and the Alternating Group

We adapt the approach that is taken with the symmetric group to obtain a full characterisation of all of the possible alternating group equivariant weight matrices that appear between any two tensor power spaces of $\mathbb{R}^n$. To do this, we introduce a jellyfish diagram that represents the determinant map, which is applied to vectors that live in an n-order tensor power space of $\mathbb{R}^n$. The jellyfish is used to create two basis matrices from certain orbit basis diagrams that appear in the characterisation for the symmetric group. Whether an orbit basis diagram corresponds to one or two basis matrices depends on the number of subsets, known as blocks, that appear in the diagram.

Chapter 5: Brauer's Group Equivariant Neural Networks

We show how three more Schur–Weyl dualities can be used to obtain all of the possible weight matrices that appear in group equivariant neural networks for the orthogonal, special orthogonal and symplectic groups. We construct a spanning set for the space of equivariant linear maps between any two tensor power spaces of $\mathbb{R}^n$ for each of these groups. All three Schur–Weyl dualities were originally found by Brauer (1937), with an additional contribution made later by Grood (1999).

Chapter 6: Partition Categories

After the publication of Maron et al. (2019a) and the preprint of our paper Pearce-Crump (2024) on permutation equivariant weight matrices, Godfrey et al. (2023) found an alternative basis for the space of permutation equivariant linear maps between any two tensor power spaces of $\mathbb{R}^n$. We show that we can recover this alternative basis by using another basis of set partition diagrams, called the diagram basis, for the partition vector space that is involved. We then show that these set partition diagrams and the ones that appear in Chapter 5 can each be given the structure of a strict $\mathbb{R}$–linear monoidal category. For each of these partition categories, we obtain a full, strict $\mathbb{R}$–linear monoidal functor that maps to a category whose objects are the tensor power space representations of the group associated with the partition category. We emphasise that this additional structure has important implications for group equivariant neural networks, in that to work with any group equivariant weight matrix that comes from these functors, it is sufficient to work with the set partition diagrams themselves, as the functors imply that their vertices and connected components can be manipulated like strings.

Chapter 7: Algorithm for Performing Linear Equivariant Layer Functions

We use the strict $\mathbb{R}$–linear monoidal category structure for the orthogonal, special orthogonal, and symplectic groups to construct an algorithm for performing the equivariant linear layer functions that reduces the time complexity exponentially compared with a naive matrix multiplication approach. We also use the strict $\mathbb{R}$–linear monoidal category structure for the symmetric group to recover the algorithm that first appeared in Godfrey et al. (2023), except we obtain this algorithm using operations on set partition diagrams instead.

Chapter 8: Graph Automorphism Group Equivariant Neural Networks

Instead of studying set partitions, we now consider bilabelled graphs. In short, a (k, l)–bilabelled graph is a graph that comes with two tuples, one of length k and the other of length l, whose entries are taken from the vertex set of the graph (with repetitions amongst entries allowed). Writing the automorphism group of a graph G having some n vertices as $\text{Aut}(G)$, we use bilabelled graphs to give a full characterisation of all of the possible $\text{Aut}(G)$-equivariant weight matrices that appear between any two tensor power spaces of $\mathbb{R}^n$. This result is especially powerful when it is combined with Frucht's Theorem (Frucht, 1938). Frucht's Theorem states that every finite group is isomorphic to the automorphism group of a finite undirected graph. Hence, for any finite group, by constructing a graph having some n vertices such that its automorphism group is isomorphic to the group in question, we can determine the equivariant weight matrix that must appear between any two tensor power spaces of $\mathbb{R}^n$.

Chapter 9: Compact Matrix Quantum Group Equivariant Neural Networks

We broaden the scope of equivariant neural networks beyond classical symmetries by defining a new class of neural networks that encode, as an inductive bias, symmetries that are described by compact matrix quantum groups. Unlike groups, compact matrix quantum groups are algebras that generalise the space of continuous functions on compact matrix groups, and are often used to describe symmetries in non-commutative geometries. We derive the existence of this new class of equivariant neural networks, which we have termed Compact Matrix Quantum Group Equivariant Neural Networks, by using a version of Tannaka–Krein duality that was originally found by Woronowicz (1987, 1988, 1991). In particular, since every compact matrix quantum group comes with an associated fundamental representation category, we show that if we know what the fundamental representation category is for any compact matrix quantum group, then we can immediately characterise all of the weight matrices that appear in these neural networks. Crucially, we know what the fundamental representation category is for a large class of compact matrix quantum groups known as easy: they are characterised entirely by categories of certain set partitions (Banica and Speicher, 2009; Banica et al., 2010; Weber, 2013; Freslon and Weber, 2016; Raum and Weber, 2016; Tarrago and Weber, 2016; Weber, 2017; Tarrago and Weber, 2018; Gromada, 2020; Freslon, 2023). Moreover, we show that since compact matrix groups are compact matrix quantum groups, this new class of neural networks generalises and encompasses all compact matrix group equivariant neural networks. Furthermore, we obtain classifications of the weight matrices that appear in five compact matrix group equivariant neural networks that had not appeared in the machine learning literature before. In particular, they include classifications for the hyperoctahedral, bistochastic, and unitary groups.

Chapter 10: Conclusion

In the final chapter, we review the contributions that we have made in this thesis, discuss potential applications of our work, and suggest new avenues for research that we can explore in the future.

Chapter 2

Fundamental Concepts

We introduce the fundamental concepts that appear throughout this thesis. We begin by defining a number of groups, each of which is a subgroup of the general linear group. We discuss their tensor power representations since they will form the layers of neural networks that are equivariant to these groups. We then outline some fundamentals on graphs that will be used in Chapter 8, and introduce some of the key concepts from category theory that will appear in Chapter 6 onwards. We conclude this chapter by providing an overview of the main ideas in general topology that will be used in Chapter 9. We choose our field of scalars to be the real numbers $\mathbb{R}$ throughout, except for Chapter 9, where we will turn to the complex numbers $\mathbb{C}$ instead. Also, we let $[n]$ denote the set $\{1, \dots, n\}$.

2.1 The General Linear Group and its Subgroups

Groups are ubiquitous in mathematics and physics. They can be defined abstractly, but they also arise naturally, both in the study of symmetries of objects, where the group of symmetries for a given object consists of all transformations that leave that object invariant, and in the study of geometric transformations, which considers bijections on a set that preserves some sort of geometric structure, such as angles and distances. In this section, we introduce a number of groups, all of which can be considered as subgroups of a general linear group of invertible linear transformations.

The **general linear group**, $GL(V)$, is the group of invertible linear transformations from a vector space V to itself, with the group operation given by composition of linear transformations. We often choose $V = \mathbb{R}^n$, in which case we write $GL(V)$ as $GL(n)$. If we pick a basis for each copy of $\mathbb{R}^n$, then, for each linear map in $GL(n)$, we obtain its matrix representation in the bases of $\mathbb{R}^n$ that were chosen. $GL(n)$ then becomes a **matrix group** consisting of all invertible $n \times n$ matrices with entries in $\mathbb{R}$, with the group operation given by matrix multiplication.

The **special linear group**, $SL(n)$, is the subgroup of $GL(n)$ consisting of all invertible linear transformations from $\mathbb{R}^n$ to $\mathbb{R}^n$ whose determinant is $+1$.

We can associate to $\mathbb{R}^n$ one of the following two bilinear forms:

1. a non-degenerate, symmetric bilinear form $(\cdot,\cdot) : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$.
2. a non-degenerate, skew-symmetric bilinear form $\langle\cdot,\cdot\rangle : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$. In this case, n must be even, say $n = 2m$, as a result of applying Jacobi's Theorem. See Goodman and Wallach (2009, Section 1.1.2) for more details.

Then we can define the **orthogonal group** $O(n)$, the **special orthogonal group** $SO(n)$, and the **symplectic group** $Sp(n)$ as follows:

1. $O(n) := \{g \in GL(n) \mid (gx, gy) = (x, y) \text{ for all } x, y \in \mathbb{R}^n\}$
2. $SO(n) := O(n) \cap SL(n)$
3. $Sp(n) := \{g \in GL(n) \mid \langle gx, gy\rangle = \langle x, y\rangle \text{ for all } x, y \in \mathbb{R}^n\}$, for $n = 2m$.

It is easy to show that each of these groups are subgroups of $GL(n)$.

There are special bases of $\mathbb{R}^n$ with respect to each of the bilinear forms given above that enable us to consider these three groups as matrix groups.

Firstly, for the form $(\cdot,\cdot)$, we may assume that there is an ordered basis

$$B := \{e_1, \ldots, e_n\} \tag{2.1}$$

of $\mathbb{R}^n$, where e_i has a 1 in the i^{th} position, and a 0 elsewhere, that satisfies the relations

$$(e_i, e_j) = \delta_{i,j} \tag{2.2}$$

with respect to the form $(\cdot,\cdot)$. The basis B is called the **standard basis** of $\mathbb{R}^n$, and by (2.2), it is an orthonormal basis of $\mathbb{R}^n$.

Hence the form $(\cdot,\cdot)$ in the basis B is the Euclidean inner product

$$(x, y) = x^\top y \tag{2.3}$$

for all $x, y \in \mathbb{R}^n$, where x is the column vector $(x_1, \dots, x_n)^\top$ and y is the column vector $(y_1, \dots, y_n)^\top$ when expressed in the basis B.

Secondly, for the form $\langle \cdot,\cdot \rangle$, where $n = 2m$, we may assume that there is an ordered basis

$$\widetilde{B} \coloneqq \{e_1, e_{1'}, \dots, e_m, e_{m'}\} \tag{2.4}$$

of $\mathbb{R}^n$, where the i^{th} basis vector in the set has a 1 in the i^{th} position and a 0 elsewhere, that satisfies the relations

$$\langle e_\alpha, e_\beta \rangle = \langle e_{\alpha'}, e_{\beta'} \rangle = 0 \tag{2.5}$$

and

$$\langle e_\alpha, e_{\beta'} \rangle = -\langle e_{\alpha'}, e_\beta \rangle = \delta_{\alpha,\beta} \tag{2.6}$$

with respect to the form $\langle \cdot,\cdot \rangle$. The basis $\widetilde{B}$ is called the **symplectic basis** of $\mathbb{R}^n$, but it is, by definition, also standard.

Hence the form $\langle \cdot,\cdot \rangle$ in the basis $\widetilde{B}$ is the skew product

$$\langle x, y \rangle = \sum_{r=1}^{m} (x_r y_{r'} - x_{r'} y_r) = \sum_{i,j} \epsilon_{i,j} x_i y_j \tag{2.7}$$

for all $x, y \in \mathbb{R}^n$, where x is the column vector $(x_1, x_{1'}, \dots, x_m, x_{m'})^\top$ and y is the column vector $(y_1, y_{1'}, \dots, y_m, y_{m'})^\top$ when expressed in the basis $\widetilde{B}$, and

$$\epsilon_{i,j} \coloneqq \langle e_i, e_j \rangle \tag{2.8}$$

See Goodman and Wallach (2009, Lemma 1.1.2 and Lemma 1.1.5) respectively for more details.

We introduce two finite permutation groups that also appear in this thesis.

The **symmetric group** S_n is the group of all bijective functions from the set $[n]$ to itself, where the group operation is given by composition. The elements $\sigma \in S_n$ are called **permutations**. There are a number of equivalent ways of denoting the permutations in S_n, but for our purposes, we only need to consider **cycle notation**. For any permutation $\sigma \in S_n$, we take $1 \in [n]$ and calculate 1, $\sigma(1)$, $\sigma^2(1)$ etc. until we obtain the smallest power k such that $\sigma^k(1) = 1$. Such a value k must exist since $[n]$ is a

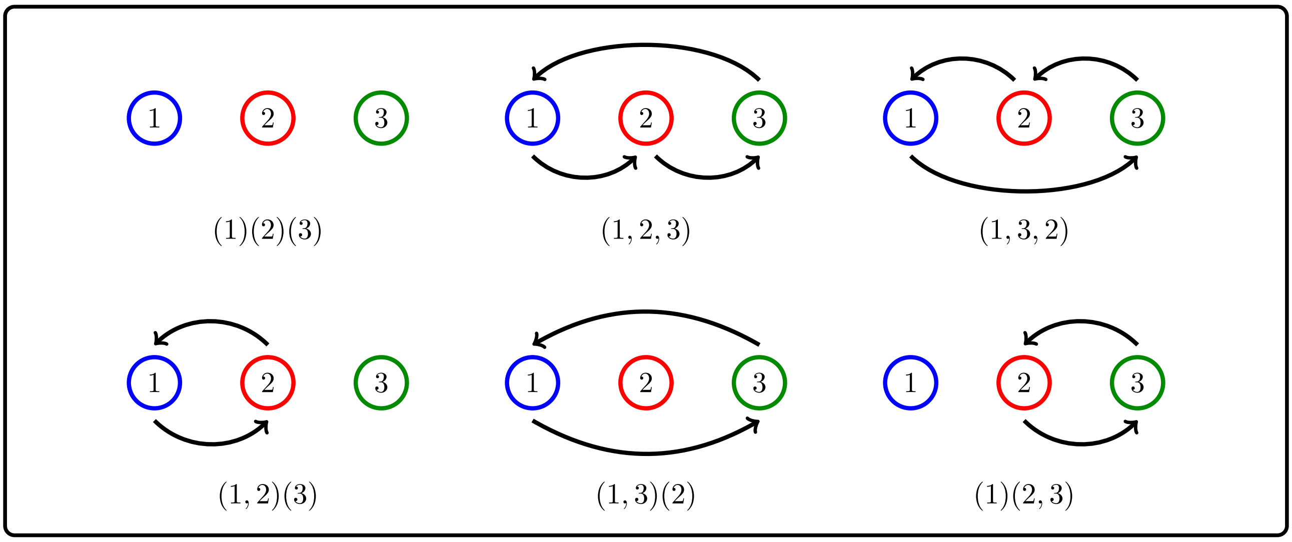


Figure 2.1: An illustration of the permutation action of S_3 on a set containing three objects.

finite set. From this sequence we obtain the cycle

$$(1, \sigma(1), \sigma^2(1), \ldots, \sigma^{k-1}(1)) \tag{2.9}$$

which can be read as $\sigma \in S_n$ maps 1 to $\sigma(1)$, $\sigma(1)$ to $\sigma^2(1)$, and so on. Next we pick another element in $[n]$ that is not in the cycle (2.9) and continue creating new cycles by picking any element in $[n]$ that is not in any of the previous cycles until all elements of $[n]$ have been exhausted. Finally, to express $\sigma \in S_n$ in cycle notation, we put the cycles that we have obtained side by side, in any order.

For example, if $\sigma \in S_6$ is given by

$$\sigma(1) = 3, \quad \sigma(2) = 4, \quad \sigma(3) = 2, \quad \sigma(4) = 1, \quad \sigma(5) = 6, \quad \sigma(6) = 5 \tag{2.10}$$

then σ can be expressed in cycle notation as

$$\sigma = (1, 3, 2, 4)(5, 6) \tag{2.11}$$

The symmetric group S_n can also be viewed as a group of matrices; however, to do this, we need to pick a basis of $\mathbb{R}^n$. If we choose the standard basis B of $\mathbb{R}^n$ that was given in (2.1), then the symmetric group S_n is the group of all $n \times n$ matrices having a single 1 in each row and column, and 0 otherwise. Consequently, the symmetric group S_n can be thought of as a subgroup of the matrix group $O(n)$ and hence a subgroup of the matrix group $GL(n)$, where the standard basis B of $\mathbb{R}^n$ has been chosen.

For more information on the symmetric group, see Sagan (2000).

In order to define the second group, we need a few more definitions. A **transposition** in the symmetric group S_n is a cycle containing exactly two elements. It is well-known that every permutation in S_n can be expressed as a composition of transpositions (Sagan, 2000). We say that a permutation in S_n is **even** if it can be expressed as a composition of an even number of transpositions, and a similar definition exists for **odd** permutations. From this, we can define the **sgn** function, which maps all even permutations in S_n to $+1$ and all odd permutations to -1. The **alternating group** A_n is the group of all even permutations in S_n, and it is a subgroup of S_n.

2.2 Group Actions and Group Representations

We now study representations of the groups that were introduced in the previous section. One reason why we consider representations of groups is that they make it possible for us to use linear algebra to gain insights into the groups themselves. But group representations can also be used to learn more about the vector spaces that appear in the representations. This latter viewpoint will be particularly important in the context of group equivariant neural networks, where we consider maps between vector spaces, each of which is a representation of a group.

Group actions can be used to generate group representations. Indeed, we often use the same language when referring to both concepts, but group actions are important in their own right. When a group acts on a set, the group permutes the elements of the set.

Formally, a group G is said to **act** on a set X if there is a map

$$\phi : G \times X \to X \tag{2.12}$$

such that, for all $x \in X$,

1. if e is the identity element of G, then $\phi(e, x) = x$, and
2. $\phi(g, \phi(h, x)) = \phi(gh, x)$ for all $g, h \in G$.

We often write $g \cdot x$ to refer to the action of $g \in G$ on $x \in X$.

When a group G acts on a set X, we can consider, for each element $x \in X$, its orbit under the action of G. The **orbit** of $x \in X$ is given by the set of elements in X that can be obtained from the action of

G on x, that is, the set $\{g \cdot x \in X \mid g \in G\}$. Note that the orbits partition X. We say that the action of G on X is **transitive** if and only if it has exactly one orbit. By contrast, the **stabilizer** of $x \in X$ consists of all $g \in G$ that fix x, that is, the set $\{g \in G \mid g \cdot x = x \in X\}$. The **Orbit–Stabilizer Theorem** states that the size of the orbit of $x \in X$ is equal to the index of the stabilizer of $x \in X$ in G.

We can turn group actions into group representations by giving the set the structure of a vector space, namely by letting its elements index a basis. However, we focus mostly on group representations because there are group representations that do not come from group actions and, typically, group representations form the foundation of group equivariant neural networks.

Formally, a **representation** of a group G is a choice of vector space V over $\mathbb{R}$ and a group homomorphism

$$\rho_V : G \to GL(V) \tag{2.13}$$

We focus on finite-dimensional vector spaces V that are some tensor power of $\mathbb{R}^n$. We often abuse our terminology by calling V a representation of G, even though the representation is technically the homomorphism ρ. When the homomorphism ρ needs to be emphasised alongside its vector space V, we will use the notation (V, ρ).

Consider the group $G = GL(n)$. Since we have a representation of $GL(n)$

$$\rho_1 : GL(n) \to GL(\mathbb{R}^n); \quad g \mapsto g \tag{2.14}$$

that is given by matrix multiplication on any vector in $\mathbb{R}^n$, then, for any positive integer k, the space $(\mathbb{R}^n)^{\otimes k}$ is also a representation of $GL(n)$, which we denote by ρ_k, where

$$\rho_k(g)(v_1 \otimes \cdots \otimes v_k) := gv_1 \otimes \cdots \otimes gv_k \tag{2.15}$$

for all $g \in GL(n)$ and for all vectors $v_i \in \mathbb{R}^n$. We call $(\mathbb{R}^n)^{\otimes k}$ the ***k*-order tensor power space** of $\mathbb{R}^n$.

Moreover, $(\mathbb{R}^n)^{\otimes k}$ is a representation for any subgroup $G(n)$ of $GL(n)$, which is obtained by restricting ρ_k to $G(n)$. We will also label this representation by ρ_k, with the context making it clear that it is the restriction of the $GL(n)$ representation.

Note in particular that, for any subgroup of the symmetric group S_n, itself viewed as a subgroup of $GL(n)$ having chosen the standard basis (2.1) of $\mathbb{R}^n$, the representation ρ_k maps a standard basis element of $(\mathbb{R}^n)^{\otimes k}$ of the form

$$e_I \coloneqq e_{i_1} \otimes e_{i_2} \otimes \cdots \otimes e_{i_k} \tag{2.16}$$

for all $I \coloneqq (i_1, i_2, \ldots, i_k) \in [n]^k$ to

$$e_{\sigma(I)} \coloneqq e_{\sigma(i_1)} \otimes e_{\sigma(i_2)} \otimes \cdots \otimes e_{\sigma(i_k)} \tag{2.17}$$

This form will be particularly useful when we consider group equivariant neural networks for the symmetric group S_n itself, the alternating group A_n, and any automorphism group of a graph G having n vertices, $\mathrm{Aut}(G)$.

2.3 Group Equivariant Neural Networks

We are going to define neural networks that are equivariant to a given group by choosing the layers to be the tensor power space representations that we introduced in the previous section, and the layer functions to be a composition of a linear and a non-linear map, each of which is equivariant to the group in question. The network itself will be equivariant since the composition of any number of equivariant maps is equivariant. A major benefit of constructing neural networks in this way can be stated as follows: if we change the input to the network by applying the appropriate tensor power space representation for a particular group element, then, by the equivariance property, we know exactly what the output of the network will be. Hence we encode the structure that is inherent in the group directly into the networks through its representations, which makes them more interpretable.

We first need to define group equivariance formally. If (V, ρ_V) and (W, ρ_W) are two representations of a group G, then a function $\phi : V \to W$ is said to be **equivariant** to G if, for all $g \in G$ and $v \in V$,

$$\phi(\rho_V(g)[v]) = \rho_W(g)[\phi(v)] \tag{2.18}$$

or, alternatively, if the following diagram commutes, for all $g \in G$ and $v \in V$:

$$\begin{tikzcd} \boldsymbol{V} \arrow[r, "\rho_V(g)"] \arrow[d, "\phi"'] & \boldsymbol{V} \arrow[d, "\phi"] \\ \boldsymbol{W} \arrow[r, "\rho_W(g)"] & \boldsymbol{W} \end{tikzcd} \tag{2.19}$$

We denote the set of all *linear* maps between V and W that are equivariant to G by $\mathrm{Hom}_G(V, W)$. When $V = W$, we sometimes write this set as $\mathrm{End}_G(V)$. It can be shown that $\mathrm{Hom}_G(V, W)$ is a vector space over $\mathbb{R}$; see Segal (2014) for more details. A special case of equivariance is invariance, since if (W, ρ_W) is the 1-dimensional trivial representation of G, then (2.18) becomes, for all $g \in G$ and $v \in V$,

$$\phi(\rho_V(g)[v]) = \phi(v) \tag{2.20}$$

In this case we say that ϕ is **invariant** to G. It is worth noting that the sum, composition, and direct sum of two equivariant functions is equivariant, for appropriate domains and codomains of the individual functions.

Using the tensor power space representations of a subgroup $G(n)$ of $GL(n)$, having chosen the standard basis of $\mathbb{R}^n$, we can define a **group equivariant neural network** for $G(n)$ as follows: it is a function f_{NN} that is a composition of some L *layer functions*

$$f_{NN} \coloneqq f_L \circ \ldots \circ f_l \circ \ldots \circ f_1 \tag{2.21}$$

such that the l^{th} layer function is equivariant to $G(n)$ and is a function between tensor power space representations of $G(n)$

$$f_l : ((\mathbb{R}^n)^{\otimes k_{l-1}}, \rho_{k_{l-1}}) \rightarrow ((\mathbb{R}^n)^{\otimes k_l}, \rho_{k_l}) \tag{2.22}$$

where each tensor power space representation $((\mathbb{R}^n)^{\otimes k_l}, \rho_{k_l})$ of $G(n)$ was defined in (2.15).

Typically, each layer function (2.22) is itself a composition $f_l \coloneqq \sigma_l \circ \phi_l$ of a learnable, linear function

$$\phi_l : ((\mathbb{R}^n)^{\otimes k_{l-1}}, \rho_{k_{l-1}}) \rightarrow ((\mathbb{R}^n)^{\otimes k_l}, \rho_{k_l}) \tag{2.23}$$

that is equivariant to $G(n)$, together with a fixed, non-linear activation function

$$\sigma_l : ((\mathbb{R}^n)^{\otimes k_l}, \rho_{k_l}) \rightarrow ((\mathbb{R}^n)^{\otimes k_l}, \rho_{k_l}) \tag{2.24}$$

that is also equivariant to $G(n)$ and often acts pointwise on its inputs. Since we have chosen the standard basis of $\mathbb{R}^n$, the learnable, linear functions ϕ_l that are equivariant to $G(n)$ become **matrices** that have **weights** or **parameters** that can be **learned**. We call these matrices **weight matrices** or **group equivariant weight matrices** when we wish to emphasise the group itself. For certain groups $G(n)$, we initially focus on understanding what the **space of matrices** is between any two adjacent layers

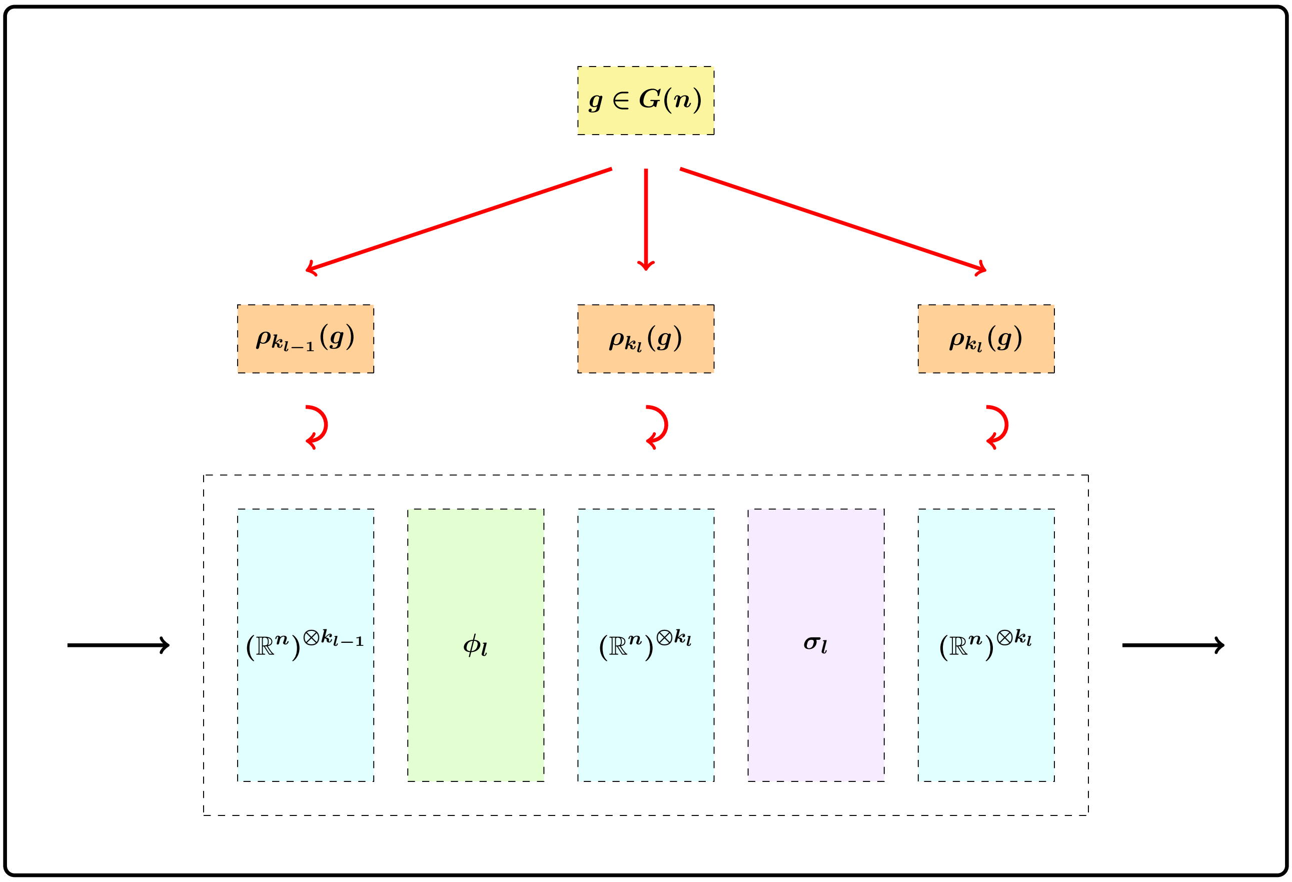


Figure 2.2: An illustration of the l^{th} layer function of a group equivariant neural network for $G(n)$. Each layer comes with a representation of $G(n)$, where the layer itself is some tensor power of $\mathbb{R}^n$. Each layer function is a composition of a learnable, linear function ϕ_l together with a fixed, non-linear activation function σ_l. As we have chosen the standard basis of $\mathbb{R}^n$, the learnable, linear function ϕ_l becomes a weight matrix containing weights that can be learned. Since the neural network is equivariant to $G(n)$, the structure in $G(n)$ is encoded directly into the network in the folllowing sense: if we change the input to the network by applying the linear transformation that comes from the representation for the initial layer for any group element g in $G(n)$, then, by applying the linear transformations that come from the representations for each of the other layers in the network for this particular group element at the same time, we know exactly what the output of the network will be.

that appear in these group equivariant neural networks. To do this, it is sufficient to find either a basis or a spanning set for the $\text{Hom}_{G(n)}$ space that is associated with any two adjacent layers. Note that if $G(n)$ is a compact matrix group, then we call the neural network a **compact matrix group equivariant neural network**.

We have chosen the layers to have a feature dimension of one and the layer functions to be without bias terms since it is easy to adapt our results to include **features** and **biases**.

In particular, to include features, if $\{M_\alpha \mid \alpha \in [p]\}$ is a spanning set of $\text{Hom}_{G(n)}((\mathbb{R}^n)^{\otimes k_{l-1}}, (\mathbb{R}^n)^{\otimes k_l})$

for some positive integer p, then

$$\{M_\alpha \otimes E_{i,j} \mid \alpha \in [p], i \in [d_l], j \in [d_{l-1}]\} \tag{2.25}$$

is a spanning set of $\mathrm{Hom}_{G(n)}((\mathbb{R}^n)^{\otimes k_{l-1}} \otimes \mathbb{R}^{d_{l-1}}, (\mathbb{R}^n)^{\otimes k_l} \otimes \mathbb{R}^{d_l})$ for some positive integers d_{l-1} and d_l.

Moreover, since the addition of two functions that are equivariant to $G(n)$ is also equivariant to $G(n)$, we can include bias terms by adding to each linear layer function of the form (2.23) a bias function $\beta_l : ((\mathbb{R}^n)^{\otimes k_{l-1}}, \rho_{k_{l-1}}) \to ((\mathbb{R}^n)^{\otimes k_l}, \rho_{k_l})$ that is both equivariant to $G(n)$ and constant on all elements of $(\mathbb{R}^n)^{\otimes k_{l-1}}$. Hence, for some constant element $c \in (\mathbb{R}^n)^{\otimes k_l}$, $\beta_l(v) = c$ for all $v \in (\mathbb{R}^n)^{\otimes k_{l-1}}$. Since β_l is equivariant to $G(n)$, we see that

$$c = \rho_{k_l}(g)c \tag{2.26}$$

for all $g \in G(n)$. Given that any $c \in (\mathbb{R}^n)^{\otimes k_l}$ satisfying (2.26) can be viewed as an element of $\mathrm{Hom}_{G(n)}((\mathbb{R}^n)^{\otimes 0}, (\mathbb{R}^n)^{\otimes k_l})$, to find the matrix form of c, all we need to do is find a spanning set for $\mathrm{Hom}_{G(n)}((\mathbb{R}^n)^{\otimes 0}, (\mathbb{R}^n)^{\otimes k_l})$.

2.4 Graph Theory

In Section 2.1, we defined the general linear group $GL(n)$ and a number of its subgroups. There also exist a class of subgroups of the symmetric group S_n that come from graphs having n vertices. Hence these subgroups of S_n can also be viewed as matrix subgroups of $GL(n)$. We begin by recalling some of the fundamentals of graph theory in order to define this class of groups.

A **graph** G is a tuple $(V(G), E(G))$ of sets, where $V(G)$ is a set of vertices for G and $E(G)$ is a subset of unordered pairs of elements from $V(G) \times V(G)$ denoting the undirected edges between the vertices of G. We include the possibility that the graph has loops; however, we only allow at most one loop per vertex. If G is a graph having n vertices, then the **adjacency matrix** of G, denoted by A_G, is the $n \times n$ matrix whose (i, j)-entry is 1 if vertex i is adjacent to vertex j in G, and is 0 otherwise. Note that A_G is a symmetric matrix, because G has undirected edges, and the (i, i)-entry is 1 in A_G if and only if G has a loop at vertex i. The **complement** of a graph G, denoted by $\overline{G}$, is the graph having the same vertex set as G and the same loops as G, but now distinct vertices are adjacent in $\overline{G}$ if and only if they are not adjacent in G.

For example, the complete graph on n vertices, K_n, is the loopless graph where every vertex is adjacent

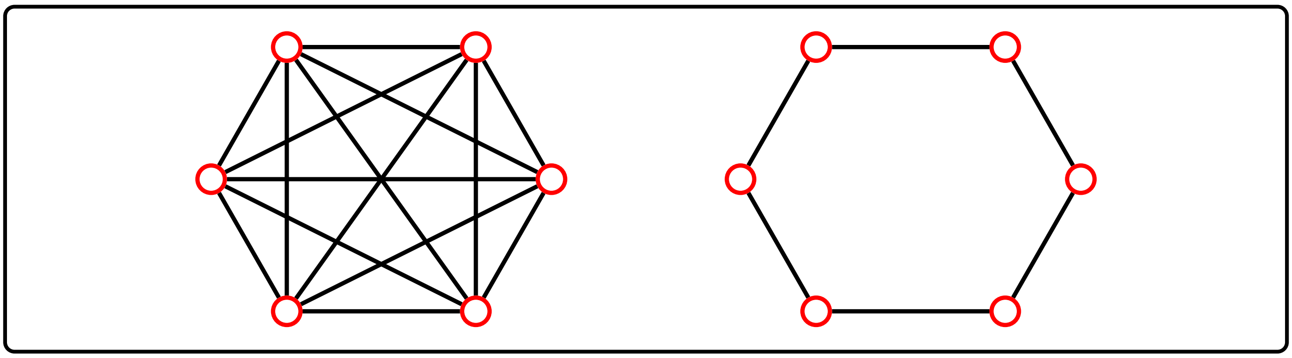


Figure 2.3: The graph depicted on the left is the complete graph on six vertices, K_6. The graph depicted on the right is the cycle graph on six vertices, C_6.

to every other vertex, and the cycle graph on n vertices, C_n, is the loopless graph where every vertex $i \in [n]$ is adjacent to $j = i \pm 1 \mod n$, where $j \in [n]$. The complete graph on six vertices and the cycle graph on six vertices are shown in Figure 2.3.

We can also consider maps between graphs. Let H and G be graphs. A **graph homomorphism** from H to G is a function $\phi : V(H) \to V(G)$ such that if i is adjacent to j in H, then $\phi(i)$ is adjacent to $\phi(j)$ in G. A **graph isomorphism** from H to G is a graph homomorphism from H to G that is also a bijection. Consequently, an **automorphism** of a graph G is a graph isomorphism from G to G. The **set of all automorphisms** of G, written Aut(G), can be shown to be a **group**.

If G is a graph having n vertices, then Aut(G) is, in fact, a subgroup of S_n. Specifically, if $\sigma \in S_n$, then it is easy to show, viewing S_n as a subgroup of $GL(n)$, that

$$\sigma \in \text{Aut}(G) \iff \sigma A_G = A_G \sigma \tag{2.27}$$

It is also clear that Aut$(G) \cong$ Aut$(\overline{G})$.

For example, the automorphism group of the complete graph K_n on n vertices, Aut(K_n), is the symmetric group S_n. Consequently, the automorphism group of the edgeless graph having n vertices, Aut$(\overline{K_n})$, is also the symmetric group S_n. Similarly, the automorphism group of the cycle graph C_n on n vertices, Aut(C_n), is isomorphic to the dihedral group D_n of order $2n$, and the automorphism group of two copies of the complete graph on two vertices, Aut$(2K_2)$, is isomorphic to the dihedral group D_4 of order 8.

For more information on graphs, see any standard book on graph theory, such as Bollobas (1998).

2.5 Category Theory

A number of different branches of mathematics across algebra, geometry, and topology consist of structures and concepts whose foundations are similar. Category theory was introduced by Eilenberg and Mac Lane in the 1940s in a paper titled *General Theory of Natural Equivalences* (Eilenberg and Mac Lane, 1945) to provide a common language for talking about these structures and for creating new connections between them. Informally, category theory is concerned with abstracting away the specific details of objects and focuses instead on the relationships between them. These relationships are called morphisms. A collection of objects and a collection of morphisms between objects (satisfying some additional conditions) form a category. We can perform operations with morphisms, such as (vertically) composing them, to form new morphisms between objects. However, we are interested not only in the relationships within a category but also in how relationships are preserved across different categories. They are described by functors. We outline some of the basics on category theory to help the reader understand some of the terminology that is used in Chapter 6 onwards. In particular, we will consider the relationships between different layer functions that appear in the group equivariant neural networks that were defined in Section 2.3 at the category level. For more information on category theory, see any standard reference, such as Mac Lane (1998); Leinster (2014) or Riehl (2017).

We begin by defining the first fundamental concept of category theory. A **category** $\mathcal{C}$ consists of

- a *collection* of **objects** $Ob(\mathcal{C})$
- for every $X, Y \in Ob(\mathcal{C})$, a *collection* $\mathrm{Hom}_{\mathcal{C}}(X, Y)$ of **morphisms** from X to Y
- for every $X, Y, Z \in Ob(\mathcal{C})$, an associative composition rule

$$\circ : \mathrm{Hom}_{\mathcal{C}}(Y, Z) \times \mathrm{Hom}_{\mathcal{C}}(X, Y) \to \mathrm{Hom}_{\mathcal{C}}(X, Z) \tag{2.28}$$

- and, for every $X \in Ob(\mathcal{C})$, there is an identity morphism $1_X \in \mathrm{Hom}_{\mathcal{C}}(X, X)$ such that every morphism $f \in \mathrm{Hom}_{\mathcal{C}}(Y, X)$ satisfies $1_X \circ f = f$ and every morphism $g \in \mathrm{Hom}_{\mathcal{C}}(X, Z)$ satisfies $g \circ 1_X = g$.

We write $f : X \to Y$ for a morphism in $\mathrm{Hom}_{\mathcal{C}}(X, Y)$. X is said to be the **domain** of f and Y is said to be the **codomain** of f.

We list some examples of categories below. Note that if we are unable to write the name of a category using a calligraphic font, then we write it using a bold font.

- **Set** has sets as its objects, and functions, with specified domain and codomain, as its morphisms.
- $\mathbf{Vect}_{\mathbb{R}}$ is the category of vector spaces over $\mathbb{R}$ and the $\mathbb{R}$–linear maps between them.
- **Group** is the category of groups and the group homomorphisms between them.

These are all examples of **concrete** categories, which are categories where the objects have sets underlying them and the morphisms are functions between these underlying sets. There also exist **abstract** categories where this phenomenon does not occur: the standard example is the category $\mathbf{Mat}_{\mathbb{R}}$ whose objects are positive integers and whose morphisms from n to m is the set of $m \times n$ matrices with values in $\mathbb{R}$. In particular, this means that the morphisms in a category are not always functions. Another commonly used word for a morphism is an **arrow**.

From categories we can obtain subcategories. A **subcategory** $\mathcal{D}$ of a category $\mathcal{C}$ consists of a subcollection of the objects in $\mathcal{C}$ and a subcollection of the morphisms in $\mathcal{C}$ such that

- if $X \in Ob(\mathcal{D})$, then the identity morphism 1_X in $\mathrm{Hom}_{\mathcal{C}}(X, X)$ is also in $\mathrm{Hom}_{\mathcal{D}}(X, X)$,
- for $X, Y \in Ob(\mathcal{C})$, if $f \in \mathrm{Hom}_{\mathcal{D}}(X, Y)$, then $X, Y \in Ob(\mathcal{D})$, and
- if $f \in \mathrm{Hom}_{\mathcal{D}}(X, Y)$ and $g \in \mathrm{Hom}_{\mathcal{D}}(Y, Z)$, then $g \circ f \in \mathrm{Hom}_{\mathcal{D}}(X, Z)$.

A subcategory $\mathcal{D}$ of a category $\mathcal{C}$ is said to be **full** if, for every $X, Y \in Ob(\mathcal{D})$, every morphism f in $\mathrm{Hom}_{\mathcal{C}}(X, Y)$ is also a morphism in $\mathrm{Hom}_{\mathcal{D}}(X, Y)$.

We note that in the definition of a category, the objects and the morphisms between objects are collections. This is to avoid Russell's paradox, namely that there is no set of all sets. In particular, the collection of objects that appear in every example of a concrete category that we listed is not a set. However, we can avoid this issue by only considering categories that are **locally small**: these are categories where the collection of morphisms between any pair of objects in the category is a set. For a pair of objects, the set of morphisms between them is sometimes called a **Hom-set**. In particular, each Hom-set is an object in the category **Set**.

We can also create a new category as the product of two individual categories. Suppose that $\mathcal{C}$ and $\mathcal{D}$ are two categories. Then the **product category** $\mathcal{C} \times \mathcal{D}$ is the category whose

- objects are ordered pairs (X, Y), where X is an object of $\mathcal{C}$ and Y is an object of $\mathcal{D}$
- morphisms from (X_1, Y_1) to (X_2, Y_2) are pairs of morphisms (f, g), where $f : X_1 \to X_2$ is a morphism in $\mathcal{C}$ and $g : Y_1 \to Y_2$ is a morphism in $\mathcal{D}$

such that composition of morphisms is defined componentwise by composition in $\mathcal{C}$ and composition in $\mathcal{D}$ and the identity morphism for each object (X, Y) in $\mathcal{C} \times \mathcal{D}$ is $(1_X, 1_Y)$.

We now look at the second fundamental concept of category theory. Just as there is a notion of a map between sets, there is also a notion of a "map" between categories, which is given by the following definition.

Suppose that $\mathcal{C}$ and $\mathcal{D}$ are two categories. Then a **functor** $\mathcal{F} : \mathcal{C} \to \mathcal{D}$ consists of

- a map $\mathcal{F} : Ob(\mathcal{C}) \to Ob(\mathcal{D})$, and
- for each pair of objects X, Y in $\mathcal{C}$, a map

$$\mathcal{F}_{X,Y} : \mathrm{Hom}_{\mathcal{C}}(X, Y) \to \mathrm{Hom}_{\mathcal{D}}(\mathcal{F}(X), \mathcal{F}(Y)) \tag{2.29}$$

  that maps the morphism $f : X \to Y$ in $\mathcal{C}$ to the morphism $\mathcal{F}(f) : \mathcal{F}(X) \to \mathcal{F}(Y)$ in $\mathcal{D}$ such that it preserves both the composition rule and the identity morphisms in each category; that is, for morphisms $f : X \to Y$, $g : Y \to Z$ in $\mathcal{C}$, we have that

$$\mathcal{F}(g \circ_{\mathcal{C}} f) = \mathcal{F}(g) \circ_{\mathcal{D}} \mathcal{F}(f) \tag{2.30}$$

  and, for all $X \in Ob(\mathcal{C})$, we have that

$$\mathcal{F}(1_X) = 1_{\mathcal{F}(X)} \tag{2.31}$$

We list some examples of functors below.

- the functor **Group** $\to$ **Set**, which associates to each group G its underlying set and to each group homomorphism the underlying set map.
- the functor **Set** $\to$ $\mathbf{Vect}_{\mathbb{R}}$, which associates to each set the vector space spanned by the elements of the set, and, for a given set map, the associated linear map is the one that is given by extending linearly on the elements of the set.

Finally, a functor $\mathcal{F} : \mathcal{C} \to \mathcal{D}$ is said to be **full** if for every $X, Y \in Ob(\mathcal{D})$, the function

$$\mathcal{F}_{X,Y} : \mathrm{Hom}_{\mathcal{C}}(X, Y) \to \mathrm{Hom}_{\mathcal{D}}(\mathcal{F}(X), \mathcal{F}(Y)) \tag{2.32}$$

is surjective. For example, given a subcategory $\mathcal{D}$ of a category $\mathcal{C}$, the inclusion functor $\mathcal{D} \hookrightarrow \mathcal{C}$ is full.

2.6 General Topology

To finish this chapter, we need to define some concepts from general topology — also known as point set topology — that will be used in Chapter 9. General topology provides an abstract mathematical framework for studying the properties of a wide range of mathematical structures. The mathematical structures in general topology are the so-called topological spaces, namely, sets containing a family of subsets that satisfy some nice properties, together with the so-called continuous functions, which are the structure-preserving maps between topological spaces. The family of subsets that is chosen for a set is called a topology, and a set can have different topologies assigned to it. In this way the notion of a continuous function depends on the topology that is assigned to the domain and the codomain of the function. Our treatment is only an overview of the main ideas and results: we will mostly state them without proof and so we refer the reader to any standard reference, such as Bourbaki (1995) or Munkres (2018), for more details.

We define the first key concept of general topology. A **topology** on a set X is a family of subsets τ_X of X such that

1. Both the empty set and X are elements of τ_X;
2. Any union of elements of τ_X is an element of τ_X;
3. Any intersection of finitely many elements of τ_X is an element of τ_X.

We call a set X together with a topology τ_X on X a **topological space**, written as a pair (X, τ_X). When the topology of a topological space (X, τ_X) is understood, we often just refer to the topological space as X. We sometimes call the elements of X **points**. We call the elements of τ_X the **open sets** of (X, τ_X). A subset of (X, τ_X) is said to be **closed** if its complement is an open set. Note that by changing the topology that is associated with a set, the definition of the open sets typically changes.

We said that the maps of interest between topological spaces are the continuous functions, which are defined as follows. A function $f : (X, \tau_X) \to (Y, \tau_Y)$ between topological spaces is said to be **continuous** if f is a function on the underlying sets such that if U is an open set of Y, then its preimage $f^{-1}(U)$ is an open set of X. One can show using this definition that f is continuous if and only if the preimage of a closed set in Y is a closed set in X. The continuous functions are important because they preserve certain properties of topological spaces, many of which will be defined in what follows.

One useful notion that helps to define topologies on sets is that of a **base**, or sometimes **basis**. A base B for a topological space (X, τ_X) is a collection of open sets in τ_X such that every open set can be written as a union of elements in B. We say that B **generates the topology** τ_X.

An important class of topological spaces are **metric spaces**, which come with a notion of distance between any pair of points in the set. A metric space is a set X together with a function $d_X : X \times X \to \mathbb{R}$, called a **metric**, such that the following axioms hold:

1. $d_X(x, y) \geq 0$ for all $x, y \in X$ and $d_X(x, y) = 0$ if and only if $x = y$;
2. $d_X(x, y) = d_X(y, x)$ for all $x, y \in X$;
3. $d_X(x, y) \leq d_X(x, z) + d_X(z, y)$ for all $x, y, z \in X$.

A metric space is often denoted by the pair (X, d_X) or simply by X when the metric d_X is understood. Some of the most important sets in a metric space are the **open balls**. For any point x in a metric space (X, d_X) and for any real number $\epsilon > 0$, the open ball of radius ϵ around x, written $B(x, \epsilon)$, is defined to be the set $\{y \in X \mid d_X(x, y) < \epsilon\}$. When a metric space is given a topology that is induced by the metric, known as the **metric topology**, then the metric space together with the metric topology becomes a topological space. The metric topology is generated by the base that is given by all of the open balls that are defined by the metric.

A particularly important class of metric spaces are the **normed vector spaces**, which are commonly referred to simply as **normed spaces**. To define them, we first need to define the concept of a **norm** on a vector space. A norm on a complex vector space X is a function $\|\cdot\| : X \to \mathbb{R}$ such that

1. $\|x\| \geq 0$ for all $x \in X$ and if $\|x\| = 0$ then $x = 0$;
2. $\|\lambda x\| \leq |\lambda| \, \|y\|$ for all $x \in X$ and $\lambda \in \mathbb{C}$;
3. $\|x + y\| \leq \|x\| + \|y\|$ for all $x, y \in X$;

If X is an algebra, then a norm on X must also satisfy the submultiplicativity axiom, namely

4. $\|xy\| \leq \|x\| \, \|y\|$ for all $x, y \in X$

A normed space is a complex vector space (or algebra) X that is equipped with a norm $\|\cdot\| : X \to \mathbb{R}$. Normed spaces X become metric spaces under the metric that is induced by the norm, namely

$$d_X(x, y) := \|x - y\| \text{ for all } x, y \in X \tag{2.33}$$

In particular, one can show that d_X satisfies the axioms of a metric on X.

One nice property for a topological space to have is the **Hausdorff property**. The Hausdorff property defines a notion of separation between points in a topological space that will be particularly useful when we consider limits of sequences below. To define the Hausdorff property, we first need the definition of a **neighbourhood**. A neighbourhood of a point p in a topological space (X, τ_X) is a subset V of X such that V contains an open set U that contains p, that is, $p \in U \subseteq V \subseteq X$. Note that V does not necessarily need to be an open set itself. We say that a topological space (X, τ_X) is **Hausdorff** if, for any two distinct points $x, y \in X$, there exists a neighbourhood V of x and a neighbourhood W of y such that V and W are disjoint. In particular, one can show that every metric space (X, d_X) is Hausdorff.

We now look at the **closure** of a set in a topological space (X, τ_X). Every subset V of a topological space (X, τ_X) has a closure which is denoted by $\overline{V}$: it is the intersection of all closed sets in (X, τ_X) that contain V. This definition is not particularly helpful, so to make it more useful we need the following definition of a **limit point**. For a subset V of a topological space (X, τ_X), a limit point of V is a point $x \in X$ such that every neighbourhood of x contains a point of V that is not x. One can then show that the closure of V is the union of V and the set containing all limit points of V. We say that a subset V of a topological space (X, τ_X) is **dense** in X if its closure is X, that is, $\overline{V} = X$. Moreover, one can show that the closure of a closed set is the set itself. Hence if V is dense in X and V is closed, then $V = X$. All of these definitions apply immediately in the case where X is a metric space.

We continue now with metric spaces and consider sequences of points in a metric space. A sequence (x_n) in a metric space (X, d_X) is said to be **Cauchy** if for all $\epsilon > 0$, there exists a positive integer N such that for all $m, n \geq N$, $d_X(x_m, x_n) < \epsilon$. A sequence (x_n) in a metric space (X, d_X) is said to **converge** to a limit $x \in X$ if for all $\epsilon > 0$, there exists a positive integer N such that for all $n \geq N$, $d_X(x_x, x) < \epsilon$. In particular, one can show that if a sequence converges in a metric space, then the limit must be unique. This is a direct consequence of a metric space being Hausdorff. Moreover, continuous functions preserve the convergence of sequences, in that if $f : (X, d_X) \to (Y, d_Y)$ is a continuous function between metric spaces and (x_n) is a sequence that converges to $x \in X$, then the sequence $(f(x_n))$ in Y converges to $f(x) \in Y$. Particularly nice metric spaces are those that are said to be complete: a metric space (X, d_X) is **complete** if every Cauchy sequence in (X, d_X) converges. In particular, a normed space that is complete with the metric induced by the norm is known as a **Banach algebra**, which is the starting point for the material that appears in Chapter 9. One further useful result is that a subset of a complete metric space is complete if and only if it is closed.

A notable point is that closed sets have an alternative characterisation when X is a metric space. Firstly, note that if (x_n) is a sequence that lives in a subset V of X that converges, then its limit x is not necessarily in V (but it is in X). However, one can show that a subset V of a metric space (X, d_X) is **closed** if and only if every sequence (x_n) in V converges to a limit in V.

We said that complete metric spaces are particularly nice: they allow us to solve problems iteratively using Cauchy sequences, since we know that they have a limit. However, even if a metric space is incomplete, one can create a complete metric space from it in a process known as **completion**. We first need the following definition. A map $i : (X, d_X) \to (Y, d_Y)$ is said to be an **isometry** if it satisfies

$$d_Y(i(x_1), i(x_2)) = d_X(x_1, x_2) \tag{2.34}$$

for all $x_1, x_2 \in X$. A surjective isometry between metric spaces is called an **isomorphism of metric spaces** or simply an **isomorphism**. Hence, we have that a metric space $(\tilde{X}, d_{\tilde{X}})$ is called the completion of a metric space (X, d_X) if the following conditions are satisfied:

- there is an isometry $i : X \to \tilde{X}$
- the image $i(X)$ is dense in $\tilde{X}$, and
- the space $(\tilde{X}, d_{\tilde{X}})$ is complete.

In fact, one can show that every metric space has a unique completion, up to isomorphism. In particular, every incomplete metric space can be completed.

One of the most important concepts in general topology is **compactness**, which is a notion of finiteness for topological spaces. In general topology, compact topological spaces play an analogous role to finite dimensional vector spaces in the theory of vector spaces or to finite sets in the theory of sets.

The motivation for compactness of a topological space X comes from wanting to understand the conditions that X needs to satisfy such that any continuous map $f : X \to \mathbb{C}$ is bounded. Recall that a function $f : X \to \mathbb{C}$ is **bounded** if there exists a non-negative real number M such that $|f(x)| \leq M$ for all $x \in X$.

One way in which a continuous function $f : X \to \mathbb{C}$ will be bounded is if it is bounded on a finite number of subsets of X whose union is X itself. This provides sufficient motivation for the following definitions. If (X, τ_X) is a topological space, then a **cover** of X is a family $(U_i)_{i \in I}$ of subsets of X such that $\bigcup_{i \in I} U_i = X$. The cover is **finite** if the indexing set I is finite, and the cover is **open** if U_i is open for each $i \in I$. If $(U_i)_{i \in I}$ is a cover of X, then we say that $(U_j)_{j \in J}$ is a subcover of $(U_i)_{i \in I}$ if $J \subseteq I$ and $(U_j)_{j \in J}$ is itself a cover of X. Consequently, a topological space (X, τ_X) is **compact** if every open cover of X has a finite subcover. Hence, every continuous function $f : X \to \mathbb{C}$ on a compact topological space X is bounded. We will study $C(X)$, the space of continuous functions $f : X \to \mathbb{C}$, in the case where X is a compact Hausdorff space in Chapter 9. It is also worth noting that every finite topological space is compact since any open cover must already be finite.

One can also show that every closed subspace of a compact space is compact, where the subspace is given the **subspace topology**: if V is a subspace of a topological space (X, τ_X), then the subspace topology τ_V on V is defined by $\{U \cap V \mid U \in \tau_X\}$. Although it is not true that every compact subspace of a compact space is closed, we do have that every compact subspace of a Hausdorff space is closed. Moreover, if (X, τ_X) is compact and $f : (X, \tau_X) \to (Y, \tau_Y)$ is a continuous function, then $f(X)$ is compact. And if (X, τ_X) is compact and $f : X \to \mathbb{C}$ is a continuous function, then $f(X)$ is bounded. For metric spaces (X, d_X), one can show that compactness is equivalent to sequential compactness, which means that every sequence in X has a convergent subsequence. Furthermore, if the metric space is a Euclidean space, say $\mathbb{R}^n$ or $\mathbb{C}^n$, then one can show that a subspace of a Euclidean space is compact if and only if that subspace is closed and bounded.

Some of the most important compact topological spaces that we will consider are **compact groups**. Firstly, a **topological group** is a topological space (G, τ_G) that has a group structure defined on it such that the group operations of multiplication and inversion

$$(g, h) \mapsto gh, \quad g \mapsto g^{-1} \tag{2.35}$$

are both continuous functions. A topological group is called a compact group if the underlying topological space is compact. Note that we always choose our topological groups to be Hausdorff, and so this property is always assumed when we refer to a topological group. Note also that, in the theory of topological groups, a subgroup of a topological group is always assumed to be closed. In particular, a closed subgroup of a compact group is a compact group.

Two of the most important compact groups that we will consider are matrix groups that were defined in Section 2.1: the orthogonal group $O(n)$ and the special orthogonal group $SO(n)$, expressed in the standard basis B that was defined in (2.1). To show that $O(n)$ is compact, it is enough to show that it is closed and bounded, since $O(n)$ is a subset of the Euclidean space $\mathbb{R}^{n\times n}$. Given that one can show that singleton sets in a metric space are closed and that polynomials are continuous functions, $O(n)$ is closed since it is the preimage of the singleton set $\{I\}$ under the continuous map $f : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}$ that is given by $M \mapsto M^\top M$. $O(n)$ is bounded since the Euclidean norm of an orthogonal $n \times n$ matrix is $\sqrt{n}$. Moreover, $SO(n)$ is closed in $O(n)$ since the determinant function is polynomial in the entries of the matrix. Hence $SO(n)$ is compact. Moreover, the symmetric group S_n, and for that matter, any finite group, is compact.

Chapter 3

Partition Algebra and the Symmetric Group

The most natural class of group equivariant neural networks to consider first are those that are equivariant to the symmetric group S_n, since a large number of machine learning tasks use data that has an underlying permutation symmetry. The use cases for permutation equivariant neural networks range from standard examples such as learning from sets (Qi et al., 2017; Zaheer et al., 2017) and graphs (Maron et al., 2019a; Thiede et al., 2020; Morris et al., 2022) through to predicting dynamics of objects (Guttenberg et al., 2016), modelling composition in natural language (Gordon et al., 2020), and even in designing auctions that maximise expected revenues in economics (Rahme et al., 2021).

Existing work on permutation equivariant neural networks has focused on two main areas: designing networks that encode permutation symmetries for specific tasks, and creating more general permutation equivariant functions that can be used to perform learning. For the former, Qi et al. (2017) constructed a permutation equivariant neural network to learn from point clouds. Zaheer et al. (2017) developed a permutation equivariant neural network to learn from sets of data. Hartford et al. (2018) modelled interactions between sets of elements using a permutation equivariant neural network. For the latter, Ravanbakhsh et al. (2017) generalised the contribution of Zaheer et al. (2017) by showing that a parameter sharing scheme can be used to establish, amongst other discrete symmetries, permutation equivariance for linear functions between two different sets of indices. Hy et al. (2019) considered higher-order relations between sets of indices instead, namely tensor power spaces of $\mathbb{R}^n$, and showed that a number of operations on these spaces are permutation equivariant. Maron et al. (2019a) then studied the problem of classifying all of the linear permutation equivariant and invariant neural network layer functions on tensor power spaces of $\mathbb{R}^n$, with their motivation coming from learning relations between the nodes of graphs. They characterised all of the learnable, linear, permutation equivariant layer functions from a k-order tensor power space of $\mathbb{R}^n$ to an l-order tensor power space of $\mathbb{R}^n$ in the practical cases (specifically, when $n \geq k + l$). Their method used equalities involving Kronecker products to obtain a number of fixed point equations which they then solved to find a basis, in tensor form, for the layer functions under consideration. Pan and Kondor (2022) went on to establish a method for organising the computation of the layer functions that appeared in Maron et al. (2019a). Finzi et al. (2021) developed a numerical algorithm to calculate the weight matrices for permutation and other group equivariant neural networks for small values of n, k and l.

In this chapter, we show that an entirely different approach from the one that appears in Maron et al. (2019a) can be used to obtain a full characterisation of all of the possible permutation equivariant weight matrices that appear between any two tensor power spaces of $\mathbb{R}^n$. The starting point for our approach is Schur–Weyl duality between the symmetric group and the partition algebra, a result that commonly appears in the algebraic combinatorics and representation theory literature (Martin, 1990, 1994, 1996; Jones, 1994; Halverson and Ram, 2005; Benkart et al., 2017; Benkart and Halverson, 2019a,b). We describe this Schur–Weyl duality in more detail in the next section. Duality itself appears in many areas of mathematics and physics as a concept for understanding one object through two different viewpoints (Atiyah, 2007). We show that the weight matrices, which have permutation symmetry — the first viewpoint — can be obtained analytically through a so-called partition vector space consisting of combinatorial diagrams that partition sets into disjoint subsets — the second viewpoint.

The content of this chapter is modelled on the paper:

Edward Pearce-Crump. (2024) Connecting Permutation Equivariant Neural Networks and Partition Diagrams. *Proceedings of the 27th European Conference on Artificial Intelligence* (ECAI 2024), IOS Press Ebooks 392:1511-1518. [Live Oral]

3.1 Schur–Weyl Duality

At the very beginning of this thesis, we introduced Schur–Weyl duality between the general linear group $GL(n)$ and the symmetric group S_k, which was discovered by Schur (1927). This Schur–Weyl duality describes the irreducible representations of the general linear group $GL(n)$ in terms of integer partitions that index the irreducible representations of the symmetric group S_k through dual actions of $GL(n)$ and S_k on the tensor power space $(\mathbb{R}^n)^{\otimes k}$.

However, the Schur–Weyl duality that is the focus of this chapter is the one that exists between the symmetric group S_n and the partition algebra $P_k^k(n)$ that was simultaneously found by Martin (1990, 1994, 1996) and Jones (1994). Before stating what this Schur–Weyl duality is, we need to define the partition algebra. To do this, we require the following two definitions.

Definition 3.1. A **set partition** π of $[2k]$ is a partition of the set $[2k]$ into a number of disjoint subsets. We call the subsets of π **blocks**.

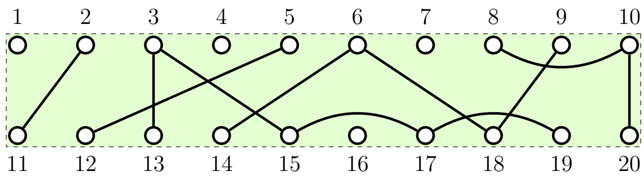
Definition 3.2. We define a **diagram** d_π from each set partition π of $[2k]$ that has two rows of vertices and edges between vertices such that there are

1. k white vertices on the top row, labelled left to right by $1, \ldots, k$
2. k white vertices on the bottom row, labelled left to right by $k+1, \ldots, 2k$, and
3. the edges between the vertices correspond to the connected components of the set partition π that indexes the diagram.

Example 3.3. If π is the set partition of [20] given by

$$\pi := \{1 \mid 2, 11 \mid 3, 13, 15, 17, 19 \mid 4 \mid 5, 12 \mid 6, 9, 14, 18, \mid 7 \mid 8, 10, 20 \mid 16\} \tag{3.1}$$

then the diagram d_π is



(3.2)

Consequently, we have that

Definition 3.4. The **partition algebra** $P_k^k(n)$ is the $\mathbb{R}$–linear span of the set of diagrams d_π indexed by all of the set partitions π of $[2k]$ (together with an algebra product that we omit for brevity).

Similar to Schur's 1927 version, Schur–Weyl duality between the symmetric group S_n and the partition algebra $P_k^k(n)$ describes a one-to-one correspondence between the irreducibles of the symmetric group S_n and the irreducibles of the partition algebra $P_k^k(n)$ that appear in the decomposition of the tensor power space $(\mathbb{R}^n)^{\otimes k}$, namely

$$(\mathbb{R}^n)^{\otimes k} \cong \bigoplus_{\lambda \in \Lambda(n)} S_n^\lambda \otimes Z_{k,n}^\lambda \tag{3.3}$$

Here, $\Lambda(n)$ is the set of all integer partitions of n, S_n^λ is an irreducible of the symmetric group S_n and $Z_{k,n}^\lambda$ is an irreducible of the partition algebra $P_k^k(n)$.

This Schur–Weyl duality was obtained by Jones (1994) through a surjective map from the partition algebra $P_k^k(n)$ onto $\mathrm{End}_{S_n}((\mathbb{R}^n)^{\otimes k})$. We describe this map in what follows as we adapt it, and hence Schur–Weyl duality, to characterise all of the possible weight matrices that can appear in permutation equivariant neural networks where the layers are some tensor power of $\mathbb{R}^n$.

3.2 Characterisation of Permutation Equivariant Linear Layer Functions

We consider group equivariant neural networks of the form defined in Section 2.3 where the group $G(n) = S_n$. We refer to them as **permutation equivariant neural networks** in what follows. Recall from Section 2.2 that, since S_n can be viewed as a subgroup of $GL(n)$ having chosen the standard basis of $\mathbb{R}^n$, the representation ρ_k for a given element $\sigma \in S_n$ maps a standard basis element of $(\mathbb{R}^n)^{\otimes k}$ of the form

$$e_I \coloneqq e_{i_1} \otimes e_{i_2} \otimes \cdots \otimes e_{i_k} \tag{3.4}$$

for all $I \coloneqq (i_1, i_2, \ldots, i_k) \in [n]^k$ to

$$e_{\sigma(I)} \coloneqq e_{\sigma(i_1)} \otimes e_{\sigma(i_2)} \otimes \cdots \otimes e_{\sigma(i_k)} \tag{3.5}$$

Our goal is to calculate all of the weight matrices that can appear between any two layers of the permutation equivariant neural networks in question. It is enough to construct a basis of matrices for $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$, by viewing it as a subspace of $\mathrm{Hom}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ and choosing the standard basis of $\mathbb{R}^n$, since any weight matrix will be a weighted linear combination of these basis matrices.

To construct such a basis, we begin by introducing the following vector spaces that are adapted from the definition of the partition algebra that was given in Section 3.1.

3.2.1 The Partition Vector Space, $P_k^l(n)$

Instead of considering the set $[2k]$, we now look at the set $[l+k]$. As before, we can create a set partition of $[l+k]$ by partitioning it into a number of disjoint subsets, which we also call blocks. Let Π_{l+k} be the set of all set partitions of $[l+k]$. It will also be useful to define the set $\Pi_{l+k,n}$, which is the subset of Π_{l+k} consisting of all set partitions of $[l+k]$ having at most n blocks.

As the number of set partitions in Π_{l+k} having exactly t blocks is the Stirling number of the second kind $\left\{ {l+k \atop t} \right\}$, we see that the number of elements in Π_{l+k} is equal to $\mathrm{B}(l+k)$, the $(l+k)^{\text{th}}$ Bell number, and that the number of elements in $\Pi_{l+k,n}$ is therefore equal to

$$\sum_{t=1}^{n} \begin{Bmatrix} l+k \\ t \end{Bmatrix} = \mathrm{B}(l+k,n) \tag{3.6}$$

the n-restricted $(l+k)^{\text{th}}$ Bell number.

Example 3.5. If $l = 4$ and $k = 5$, then

$$\pi := \{1,2,5,7 \mid 3,4,8 \mid 6 \mid 9\} \tag{3.7}$$

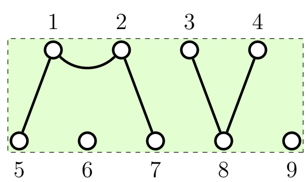
is a set partition in Π_{4+5} with 4 blocks. Hence $\pi \in \Pi_{4+5,n}$ for all $n \geq 4$.

Similar to the partition algebra, we can form a vector space from the $\mathbb{R}$–linear span of a set of diagrams d_π, except this time they are indexed by the elements π of Π_{l+k}. Each diagram d_π in the set has two rows of vertices and edges between vertices, except now there are

1. l white vertices on the top row, labelled left to right by $1, \ldots, l$
2. k white vertices on the bottom row, labelled left to right by $l+1, \ldots, l+k$, and
3. the edges between the vertices correspond to the connected components of the set partition π that indexes the diagram.

As a result, d_π represents the equivalence class of all diagrams with connected components equal to the blocks of π. We call this vector space the **partition vector space**, and denote it by $P_k^l(n)$. By construction, it has dimension $\mathrm{B}(l+k)$. We call the basis described here the **diagram basis**.

Example 3.6. Continuing on from Example 3.5, we see that the diagram d_π corresponding to the set partition π given in (3.7) is



(3.8)

Remark 3.7. It is clear that if we set $l = k$, then we obtain the partition algebra $P_k^k(n)$ that was given in Definition 3.4.

3.2.2 The Orbit Basis of $P_k^l(n)$

We can construct another basis of $P_k^l(n)$ that we will use in what follows to obtain the basis of matrices for $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$.

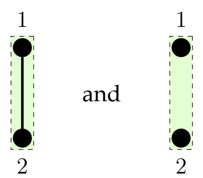
First, we define a **partial ordering** on the set partitions in Π_{l+k}, denoted by $\preceq$, which states that, for all $\pi_1, \pi_2 \in \Pi_{l+k}$, $\pi_1 \preceq \pi_2$ if every block of π_1 is contained in a block of π_2. Then we can define a set of elements in $P_k^l(n)$ that are indexed by the set partitions in Π_{l+k}, $B_O := \{x_\pi \mid \pi \in \Pi_{l+k}\}$, with respect to the diagram basis as

$$d_\pi = \sum_{\pi \preceq \theta} x_\theta \tag{3.9}$$

To see why the set B_O forms a basis of $P_k^l(n)$, first, we form an ordered set of set partitions of Π_{l+k} by ordering the set partitions by the number of blocks that they have from smallest to largest, with any arbitrary ordering allowed for a pair of set partitions that have the same number of blocks. Call this set S_{l+k}. Then, because the square matrix that maps elements of the diagram basis to linear combinations of the set B_O — whose rows and columns are indexed (in order) by the ordered set S_{l+k} — is unitriangular by (3.9), it is therefore invertible, and so we get that B_O forms a basis of $P_k^l(n)$. We call B_O the **orbit basis** of $P_k^l(n)$.

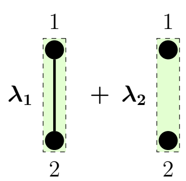
We draw elements of the orbit basis with black vertices, whereas we draw elements of the diagram basis with white vertices.

Example 3.8. The orbit basis of $P_1^1(n)$ consists of the two elements



(3.10)

Hence, for some values of $\lambda_1, \lambda_2 \in \mathbb{R}$, any element of $P_1^1(n)$ can be expressed as



(3.11)

For more details on the orbit basis, specifically for how to express an orbit basis element as a linear combination of diagram basis elements, see Benkart and Halverson (2019a,b).

3.2.3 $P_k^l(n)$ and a Basis of $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$

In this section, we show how the weight matrices that appear in the permutation equivariant neural networks in question are related to the partition vector space $P_k^l(n)$, namely by establishing a bijective correspondence between a basis of matrices for the vector space of S_n-equivariant linear maps from $(\mathbb{R}^n)^{\otimes k}$ to $(\mathbb{R}^n)^{\otimes l}$, expressed in the standard basis of $\mathbb{R}^n$, and certain orbit basis diagrams that appear in $P_k^l(n)$.

We begin by establishing the following bijective correspondence.

Proposition 3.9. *The basis elements of* $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ *are in bijective correspondence with the orbits coming from the action of* S_n *on the* $(l+k)$*-fold Cartesian product set* $[n]^{l+k}$.

Proof. As a result of choosing the standard basis for each copy of $\mathbb{R}^n$ that appears in the vector space of all linear maps from $(\mathbb{R}^n)^{\otimes k}$ to $(\mathbb{R}^n)^{\otimes l}$, this vector space has a standard basis of matrix units

$$\{E_{I,J}\}_{I\in[n]^l, J\in[n]^k} \tag{3.12}$$

where $E_{I,J}$ has a 1 in the (I,J) position and is 0 elsewhere.

Hence, for any standard basis element $e_P \in (\mathbb{R}^n)^{\otimes k}$, we see that

$$E_{I,J}e_P = \delta_{J,P}e_I \tag{3.13}$$

and so, for any linear map $f : (\mathbb{R}^n)^{\otimes k} \to (\mathbb{R}^n)^{\otimes l}$, expressing f in the basis of matrix units as

$$f = \sum_{I\in[n]^l}\sum_{J\in[n]^k} f_{I,J}E_{I,J} \tag{3.14}$$

we get that

$$f(e_P) = \sum_{I\in[n]^l} f_{I,P}e_I \tag{3.15}$$

Consequently, given that $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ is a subspace of $\mathrm{Hom}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$, we have that

f is an S_n-equivariant linear map if and only if, for all $\sigma \in S_n$ and standard basis vectors $e_J \in (\mathbb{R}^n)^{\otimes k}$,

$$f(\rho_k(\sigma)[e_J]) = \rho_l(\sigma)[f(e_J)] \tag{3.16}$$

(3.16) holds if and only if

$$\sum_{I \in [n]^l} f_{I,\sigma(J)} e_I = \sum_{I \in [n]^l} f_{I,J} e_{\sigma(I)} \tag{3.17}$$

which is true if and only if

$$f_{\sigma(I),\sigma(J)} = f_{I,J} \tag{3.18}$$

for all $\sigma \in S_n$, $I \in [n]^l$ and $J \in [n]^k$.

Therefore, concatenating the pair $I \in [n]^l, J \in [n]^k$ into a single element $(I, J) \in [n]^{l+k}$, (3.18) tells us that the basis elements of $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ are in bijective correspondence with the orbits coming from the action of S_n on $[n]^{l+k}$, where $\sigma \in S_n$ acts on the pair (I, J) by

$$\sigma(I, J) \coloneqq (\sigma(I), \sigma(J)) \tag{3.19}$$

However, since S_n acts on $[n]$ transitively, we get that the action of S_n on $[n]^{l+k}$ gives a set of orbits that completely partition the set $[n]^{l+k}$. $\square$

We now show how the orbits relate to the partition vector space $P_k^l(n)$.

Proposition 3.10. *The orbits that come from the action of S_n on $[n]^{l+k}$ are in bijective correspondence with the orbit basis diagrams x_π of $P_k^l(n)$ that have at most n blocks.*

Proof. Consider an orbit coming from the action of S_n on $[n]^{l+k}$. We can define the bijection in question on a class representative (I, J) of the orbit as follows.

Replacing momentarily the elements of J by $i_{l+p} \coloneqq j_p$ for all $p \in [k]$, so that

$$(I, J) = (i_1, i_2, \dots, i_l, j_1, j_2, \dots, j_k) \tag{3.20}$$

$$= (i_1, i_2, \dots, i_l, i_{l+1}, i_{l+2}, \dots i_{l+k}) \tag{3.21}$$

then, for indices x, $y \in [l + k]$, we define the bijection by

$$i_x = i_y \iff x, y \text{ are in the same block of } \pi \tag{3.22}$$

We see that the left hand side of (3.22) is checking for an equality on the elements of $[n]$, whereas the right hand side is separating the elements of $[l+k]$ into blocks, hence there must be at most n such blocks.

Moreover, the bijection given in (3.22) is independent of the choice of class representative, since

$$i_x = i_y \iff \sigma(i_x) = \sigma(i_y) \text{ for all } \sigma \in S_n \tag{3.23}$$

This gives us the desired result. □

Combining Propositions 3.9 and 3.10, we obtain the following key result.

Theorem 3.11. *For all non-negative integers l, k and positive integers n, the basis elements of the vector space $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ are in bijective correspondence with the orbit basis diagrams x_π in $P_k^l(n)$ having at most n blocks, and so*

$$\dim \mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l}) = \mathrm{B}(l+k, n) \tag{3.24}$$

where $\mathrm{B}(l+k, n)$ is the n-restricted $(l+k)^{th}$ Bell number.

Example 3.12. Suppose that $l = k = 1$, and let $n = 4$. It is clear from (3.19) that the action of S_4 on $[4]^{1+1}$ partitions the set into precisely two orbits. From (3.23), it is sufficient to choose $(1, 1)$ to be the class representative of the first orbit and $(1, 2)$ to be the class representative of the second orbit. (3.22) tells us that the set partition corresponding to the first orbit must be $\pi_1 := \{1, 2\}$ whereas the set partition corresponding to the second orbit must be $\pi_2 := \{1 \mid 2\}$. The orbit basis diagrams that correspond to π_1 and π_2 first appeared in Example 3.8. By Theorem 3.11, the basis matrices for the space of S_4-equivariant linear maps from $\mathbb{R}^4$ to $\mathbb{R}^4$ correspond bijectively with these orbit basis diagrams, hence there are two of them. We show how to calculate the basis matrices in Example 3.20.

3.2.4 Permutation Equivariant Weight Matrices

We can go further than Theorem 3.11 and obtain the basis matrices of $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ themselves from the orbit basis diagrams in $P_k^l(n)$ having at most n blocks. In doing so, we show how to construct all of the weight matrices that can appear between any two tensor power layers of the permutation equivariant neural networks in question.

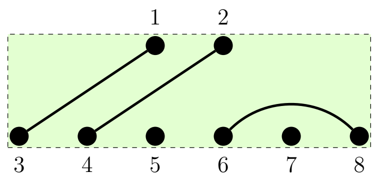
To obtain the basis matrices, we first need to define a procedure for labelling the blocks of an orbit basis diagram x_π in $P_k^l(n)$ having at most n blocks.

Definition 3.13. Let x_π be an orbit basis diagram in $P_k^l(n)$ having at most n blocks. Denote the number of blocks in x_π by t.

We obtain a **block labelling** for x_π by letting B_1 be the block that contains the number $1 \in [l+k]$, and iteratively letting B_j, for $1 < j \leq t$, be the block that contains the smallest number in $[l+k]$ that is not in $B_1 \cup B_2 \cup \cdots \cup B_{j-1}$.

We can represent the block labelling for x_π in two equivalent forms. The first form is an $(l+k)$-length tuple (I_π, J_π) with elements in $[n]$, where the length of I_π is l, the length of J_π is k, and the p^{th} entry is the label of the block that contains vertex p. The second form is a diagram which is obtained by relabelling each vertex in the orbit basis diagram x_π with the label of the block containing that vertex. We will see that this particular form is very useful in what follows as it highlights the structure of the blocks and their labels in the block labelling for x_π.

Example 3.14. Suppose that we have the orbit basis diagram x_π



(3.25)

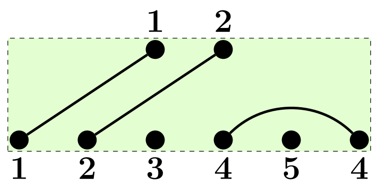
corresponding to the set partition

$$\pi = \{1, 3 \mid 2, 4 \mid 5 \mid 7 \mid 6, 8\} \tag{3.26}$$

Here $l = 2$ and $k = 6$. Suppose that $n = 5$. Then the blocks of x_π are labelled, in left-to-right order, as B_1, B_2, B_3, B_5, B_4. Hence, the block labelling for x_π is

$$(I_\pi, J_\pi) = (1, 2, 1, 2, 3, 4, 5, 4) \tag{3.27}$$

an element of $[5]^{2+6}$, or, in diagram form,



(3.28)

We see how the blocks and their labels have been made clear by using the diagram form of the block labelling for x_π.

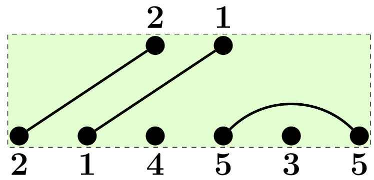
The diagram form of the block labelling is very nice for another reason: we can easily construct a matrix unit in the vector space of all linear maps from $(\mathbb{R}^n)^{\otimes k}$ to $(\mathbb{R}^n)^{\otimes l}$) from it. This matrix unit is simply E_{I_π, J_π}, where I_π is the top row of the diagram form of the block labelling and J_π is the bottom row of the diagram form of the block labelling.

Moreover, by acting on the block labelling (I_π, J_π) with S_n, we obtain an orbit for the S_n action on $[n]^{l+k}$ with (I_π, J_π) as the class representative. Denote this orbit by $O((I_\pi, J_\pi))$. The beauty of the diagram form for the block labelling is that it shows explicitly how all of the elements (I, J) in this orbit are precisely all of the possible labellings of the blocks of x_π! Hence, we have that

Proposition 3.15. *$O((I_\pi, J_\pi))$ is equal to*

$$\left\{(I,J) \in [n]^{l+k} \;\middle|\; i_x = i_y \textit{ if and only if } x, y \textit{ are in the same block of } \pi \right\} \tag{3.29}$$

Example 3.16. Continuing on from Example 3.14, the matrix unit that we obtain from the diagram form of the block labelling given in (3.28) is $E_{(1,2|1,2,3,4,5,4)}$. Moreover, we see that



(3.30)

is in the orbit of (3.28) as a result of relabelling the blocks of (3.25), or, more formally, by applying the permutation $(12)(345)$ in S_5 to the block labels of (3.28). In particular, we obtain the matrix unit $E_{(2,1|2,1,4,5,3,5)}$ from (3.30), which is a linear map from $(\mathbb{R}^5)^{\otimes 6}$ to $(\mathbb{R}^5)^{\otimes 2}$.

The reason for defining the block labelling of an orbit basis diagram x_π having at most n blocks in $P_k^l(n)$ is that we can use it to construct a basis element of $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ as follows: obtaining all of the elements (I, J) that appear in $O((I_\pi, J_\pi))$, and noting that we can form the matrix unit $E_{I,J}$ from each element, we can define X_π to be

$$X_\pi := \sum_{(I,J) \in O((I_\pi, J_\pi))} E_{I,J} \tag{3.31}$$

We see that X_π is a basis element of $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ by (3.18).

Put simply, to obtain a basis element X_π of $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$, we have added together the matrix units that come from all of the possible labellings of the blocks of x_π.

Consequently, we can define the following linear map to make clear the connection between the partition vector space $P_k^l(n)$ and all of the weight matrices that can appear in a permutation equivariant neural network between the layers $(\mathbb{R}^n)^{\otimes k}$ and $(\mathbb{R}^n)^{\otimes l}$.

Definition 3.17. For all non-negative integers l, k and positive integers n, we define a surjective map

$$\Gamma_{k,n}^l : P_k^l(n) \to \mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l}) \tag{3.32}$$

on the orbit basis of $P_k^l(n)$ as follows, and extend linearly:

$$\Gamma_{k,n}^l(x_\pi) := \begin{cases} X_\pi & \text{if } \pi \text{ has } n \text{ or fewer blocks} \\ 0 & \text{if } \pi \text{ has more than } n \text{ blocks} \end{cases} \tag{3.33}$$

In the case where $k = l$, (3.32) is the map that Jones (1994) used to obtain Schur–Weyl duality between the symmetric group and the partition algebra. Hence we have adapted Schur–Weyl duality to characterise the weight matrices that appear in any permutation equivariant neural network where the layers are some tensor power of $\mathbb{R}^n$.

We summarise our results with the following two theorems.

Theorem 3.18. *For all non-negative integers l, k and positive integers n, we have that*

$$\{X_\pi \mid \pi \textit{ is a set partition of } [l+k] \textit{ having at most } n \textit{ blocks}\} \tag{3.34}$$

is a basis of $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$.

Theorem 3.19 (Permutation Equivariant Weight Matrices). *For all non-negative integers l, k and positive integers n, the weight matrix W that appears in an S_n-equivariant linear layer function from $(\mathbb{R}^n)^{\otimes k}$ to $(\mathbb{R}^n)^{\otimes l}$ must be of the form*

$$W = \sum_{\pi \in \Pi_{l+k,n}} \lambda_\pi X_\pi \tag{3.35}$$

for $\mathrm{B}(l+k, n)$ *many weights* $\lambda_\pi \in \mathbb{R}$.

3.2.5 A Note on the Relationship between n, k and l

In classifying the weight matrices that can appear in permutation equivariant neural networks, it is important to note that there is a relationship between n, k and l. In particular, the number of weights in the weight matrix can depend on n.

If $n \geq l+k$, we see that the map $\Gamma^l_{k,n}$ is an isomorphism of vector spaces. This is because an orbit basis diagram in $P^l_k(n)$ can have at most $l+k$ blocks, and so, in this case, there are no orbit basis diagrams with more than n blocks. Consequently, the number of weights in the weight matrix does not depend on n.

However, if $n < l+k$, then the map $\Gamma^l_{k,n}$ is *not* an isomorphism of vector spaces. Indeed, in this case, the kernel of this map is non-trivial, of dimension $\mathrm{B}(l+k) - \mathrm{B}(l+k,n)$, since it is the $\mathbb{R}$–linear span of the orbit basis diagrams in $P^l_k(n)$ having more than n blocks. Consequently, the number of weights in the weight matrix *does* depend on n.

This improves upon the result that appears in Maron et al. (2019a). Although this relationship was first mentioned in the Appendix of Finzi et al. (2021), we wish to highlight this point here because a number of papers that we have read in the machine learning literature on this topic assume that the number of weights in the weight matrix is independent of n in all cases. This becomes more important when l, k are large and n is small, since the dimension of the kernel becomes very large relative to the actual number of weights in the weight matrix. For more information on the kernel of $\Gamma^l_{k,n}$, see Benkart and Halverson (2019a).

3.2.6 General Procedure and Examples

In the teal-coloured box, we provide a general procedure for how to calculate the weight matrix that appears in a permutation equivariant neural network from the layer space $(\mathbb{R}^n)^{\otimes k}$ to the layer space $(\mathbb{R}^n)^{\otimes l}$ so that our results will be accessible to the general machine learning practitioner. In the blue-coloured box, we describe a procedure for how to calculate the (I, J)-entry of each basis matrix that appears in the overall weight matrix. This method is powerful because each (I, J)-entry of X_π can be calculated simply by placing the I indices on the top row of the orbit basis diagram x_π and the J indices on the bottom row of x_π and seeing whether the blocks of the diagram are consistently and distinctly labelled.

We finish this chapter by giving a number of examples that display the simplicity and power of our method for calculating the permutation equivariant weight matrix between any two tensor power spaces of $\mathbb{R}^n$.

A General Procedure for Calculating the Weight Matrix of an S_n-Equivariant Linear Layer Function from $(\mathbb{R}^n)^{\otimes k}$ to $(\mathbb{R}^n)^{\otimes l}$.

Perform the following steps:

1. Calculate all of the set partitions π of $\{1, \dots, l+k\}$ that have at most n blocks.
2. Express each set partition π as an orbit basis diagram x_π in $P_k^l(n)$.
3. Obtain each basis matrix X_π that is associated with its orbit basis diagram x_π, either by calculating the orbit $O((I_\pi, J_\pi))$ and adding together the matrix units that are indexed by the elements in the orbit or by calculating each (I, J)-entry of X_π using the procedure given in the blue-coloured box.
4. Attach a weight $\lambda_\pi \in \mathbb{R}$ to each matrix X_π.
5. Finally, calculate $\sum \lambda_\pi X_\pi$ to give the overall weight matrix.

Consequently, all of the orbit basis diagrams in $P_k^l(n)$ having at most n blocks determine the weight matrix of an S_n-equivariant linear layer function from $(\mathbb{R}^n)^{\otimes k}$ to $(\mathbb{R}^n)^{\otimes l}$.

A Procedure for Calculating the (I, J)-entry of each Permutation Equivariant Basis Matrix X_π from $(\mathbb{R}^n)^{\otimes k}$ to $(\mathbb{R}^n)^{\otimes l}$.

Assume that x_π is an orbit basis diagram in $P_k^l(n)$ having at most n blocks.

Perform the following steps:

1. Place the indices I on the top row of x_π and the indices J on the bottom row of x_π.
2. If all of the vertices in each block in x_π have been overlaid with the same number, and no two blocks have had their vertices overlaid with the same number, then the (I, J)-entry of X_π is 1, otherwise it is 0.

Example 3.20. Suppose that we would like to find the weight matrix for an S_4-equivariant linear layer function from $\mathbb{R}^4$ to $\mathbb{R}^4$. Note that $l = k = 1$ and $n = 4$.

To calculate this weight matrix, we follow the procedure that is given in the teal-coloured box. First, we need to calculate all of the set partitions of $[1+1]$ having at most 4 blocks. They are $\pi_1 = \{1, 2\}$ and $\pi_2 = \{1 \mid 2\}$. Next, we express each of these set partitions as an orbit basis diagram in $P_1^1(4)$. These diagrams appeared in Example 3.8. Now we apply the map $\Gamma_{1,4}^1$ to each of these orbit basis diagrams to obtain the basis matrices X_{π_1} and X_{π_2}. Figure 3.1 gives examples of how to calculate the (I, J)-entries for both of these matrices using the procedure that is given in the blue-coloured box. In particular, we see that the $(1, 1)$-entry of X_{π_1} is 1, since the only block in x_{π_1} is consistently labelled (by 1), whereas the $(2, 4)$-entry of X_{π_1} is 0, since the only block in x_{π_1} is inconsistently labelled ($2 \neq 4$).

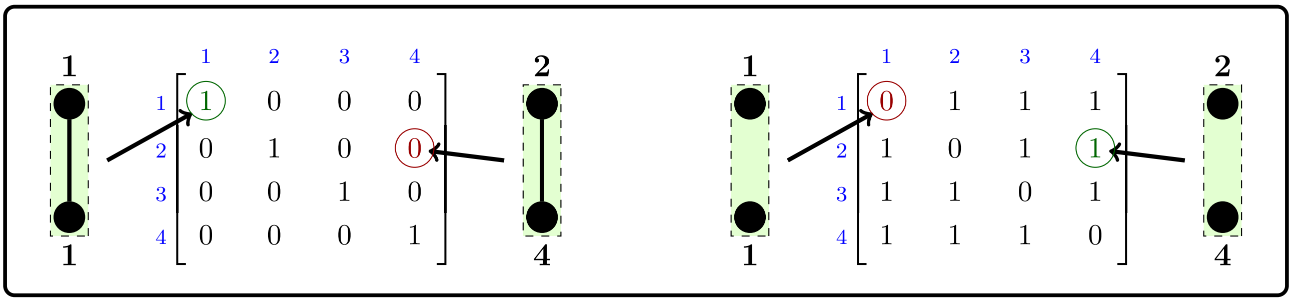


Figure 3.1: We obtain the two basis matrices whose weighted linear combination gives all of the possible weight matrices that can appear in an S_4-equivariant neural network from $\mathbb{R}^4$ to $\mathbb{R}^4$. We obtain these matrices from the orbit basis diagrams in $P_1^1(4)$ that have at most 4 blocks. For each orbit basis diagram, to calculate the (I, J)-entry of its associated basis matrix, we place the I-tuple on the top row of the diagram and the J-tuple on the bottom row of the diagram, and see if they consistently label the diagram's blocks such that no two blocks have the same label. If the labelling is consistent, then we put a 1 in the (I, J)-entry of the matrix, otherwise 0.

The procedure is the same for X_{π_2}. We see that only the diagonal entries of X_{π_2} are zero since the two blocks in x_{π_2} must be distinctly labelled.

Finally, we multiply each matrix by a weight, namely λ_1 and λ_2, respectively, and then add the two matrices together to obtain the overall weight matrix. Hence, the weight matrix for an S_4-equivariant linear layer function from $\mathbb{R}^4$ to $\mathbb{R}^4$ is of the form

$$\begin{array}{c c} & \begin{matrix} 1 & 2 & 3 & 4 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \end{matrix} & \begin{bmatrix} \lambda_1 & \lambda_2 & \lambda_2 & \lambda_2 \\ \lambda_2 & \lambda_1 & \lambda_2 & \lambda_2 \\ \lambda_2 & \lambda_2 & \lambda_1 & \lambda_2 \\ \lambda_2 & \lambda_2 & \lambda_2 & \lambda_1 \end{bmatrix} \end{array} \tag{3.36}$$

for weights $\lambda_1, \lambda_2 \in \mathbb{R}$.

It is not hard to see that the weight matrix for an S_n-equivariant linear layer function from $\mathbb{R}^n$ to $\mathbb{R}^n$ is an $n \times n$ matrix, with the diagonal entries given by the weight λ_1 and the off-diagonal entries given by the weight λ_2.

Example 3.21. Continuing on from Example 3.14, we see that the $(1, 2 \mid 1, 2, 3, 4, 5, 4)$-entry of the weight matrix for an S_5-equivariant linear layer function from $(\mathbb{R}^5)^{\otimes 6}$ to $(\mathbb{R}^5)^{\otimes 2}$ will be λ_π, a parameter that corresponds to the set partition π given in (3.26).

This is because the diagram given in (3.28), where the orbit basis diagram corresponding to π has had the top row overlaid with the indices of $I = (1, 2)$ and the bottom row overlaid with the indices of

$J = (1, 2, 3, 4, 5, 4)$, satisfies condition 2 of the blue-coloured box, namely that the indices consistently and distinctly label the blocks of x_π.

Moreover, referring back to Example 3.16, we see that the $(2, 1 \mid 2, 1, 4, 5, 3, 5)$-entry of the same weight matrix will also be λ_π, since this is related to $(1, 2 \mid 1, 2, 3, 4, 5, 4)$ by the permutation $(12)(345)$ in S_5.

Example 3.22. We now give an example where the number of weights in a permutation equivariant weight matrix is not the full Bell number $\mathrm{B}(l + k)$. Suppose that we would like to find the weight matrix for an S_2-equivariant linear layer function from $(\mathbb{R}^2)^{\otimes 2}$ to $(\mathbb{R}^2)^{\otimes 2}$. In this case, $l = k = 2$ and $n = 2$.

To calculate this weight matrix, we again follow the procedure that is given in the teal-coloured box. We first need to calculate all of the set partitions of $[2+2]$ having at most 2 blocks. There are $\mathrm{B}(4, 2) = 8$ of them, and they are shown in the left hand column of Figure 3.2. Note, in particular, that there are not $\mathrm{B}(4) = 15$ of them, which implies that the map $\Gamma^2_{2,2}$ has a kernel. This is what we expected, since $n \ngeq l + k$.

Next, we apply the map $\Gamma^2_{2,2}$ to each of the eight orbit basis diagrams to obtain eight basis matrices $X_{\pi_1}, \ldots, X_{\pi_8}$. These matrices are also given in Figure 3.2. We multiply each matrix X_{π_i} by a weight λ_i, and then finally add them all together.

Hence, the weight matrix for an S_2-equivariant linear layer function from $(\mathbb{R}^2)^{\otimes 2}$ to $(\mathbb{R}^2)^{\otimes 2}$ is of the form

$$\begin{array}{c} \\ 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array}
\begin{array}{c} \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\
\left[\begin{array}{cc|cc}
\lambda_1 & \lambda_3 & \lambda_2 & \lambda_6 \\
\lambda_5 & \lambda_8 & \lambda_7 & \lambda_4 \\
\hline
\lambda_4 & \lambda_7 & \lambda_8 & \lambda_5 \\
\lambda_6 & \lambda_2 & \lambda_3 & \lambda_1
\end{array}\right] \end{array} \tag{3.37}$$

for weights $\lambda_1, \ldots, \lambda_8 \in \mathbb{R}$.

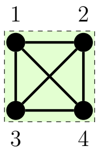
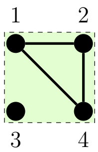
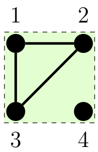
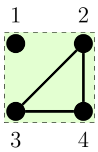
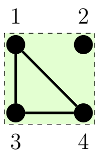
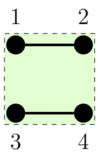
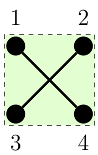
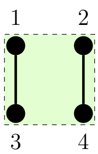
Orbit Basis Element x_π	Set Partition π	Block Labelling $(I_\pi \mid J_\pi)$	Basis Matrix X_π
	$\{1,2,3,4\}$	$(1,1 \mid 1,1)$	$\begin{array}{cc} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc\|cc} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right] \end{array}$
	$\{1,2,4 \mid 3\}$	$(1,1 \mid 2,1)$	$\begin{array}{cc} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc\|cc} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{array}\right] \end{array}$
	$\{1,2,3 \mid 4\}$	$(1,1 \mid 1,2)$	$\begin{array}{cc} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc\|cc} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right] \end{array}$
	$\{1 \mid 2,3,4\}$	$(1,2 \mid 2,2)$	$\begin{array}{cc} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc\|cc} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ \hline 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right] \end{array}$
	$\{1,3,4 \mid 2\}$	$(1,2 \mid 1,1)$	$\begin{array}{cc} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc\|cc} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{array}\right] \end{array}$
	$\{1,2 \mid 3,4\}$	$(1,1 \mid 2,2)$	$\begin{array}{cc} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc\|cc} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{array}\right] \end{array}$
	$\{1,4 \mid 2,3\}$	$(1,2 \mid 2,1)$	$\begin{array}{cc} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc\|cc} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right] \end{array}$
	$\{1,3 \mid 2,4\}$	$(1,2 \mid 1,2)$	$\begin{array}{cc} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc\|cc} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right] \end{array}$

Figure 3.2: The calculations of the eight basis matrices from the subset of orbit basis elements in $P_2^2(2)$ that determine the weight matrix for an S_2-equivariant linear layer function from $(\mathbb{R}^2)^{\otimes 2}$ to $(\mathbb{R}^2)^{\otimes 2}$.

Chapter 4

Jellyfish and the Alternating Group

The most natural group to consider after the symmetric group S_n is the alternating group A_n. The alternating group is an index two subgroup of the symmetric group that contains only the even permutations. Alternating group symmetry has proven to be particularly useful when learning from spherical image data that has been discretely represented on an icosahedron (Zhang et al., 2019); in constructing convolutional neural networks on an icosahedron (Cohen et al., 2019b); and in estimating polynomials that are invariant to the action of the alternating group (Kicki et al., 2020).

Similar to the previous chapter, we obtain a full characterisation of all of the possible **alternating group equivariant weight matrices** that appear between any two tensor power spaces of $\mathbb{R}^n$. As in the symmetric group case, we find a basis of matrices for the linear layer functions between any two tensor power spaces, having chosen the standard basis of $\mathbb{R}^n$, except they are now equivariant to the alternating group A_n. Our approach is similar to the one that we presented in the previous chapter, in that we look to relate orbit basis diagrams in the partition vector space $P_k^l(n)$ having at most n blocks to alternating group equivariant linear maps from $(\mathbb{R}^n)^{\otimes k}$ to $(\mathbb{R}^n)^{\otimes l}$ by considering orbits that come from the action of the alternating group A_n on $[n]^{l+k}$. However, since the group is now a subgroup of the symmetric group S_n, the relationship between orbit basis diagrams and equivariant linear maps is not necessarily one-to-one. To establish the true relationship between them, we use a concept that was first introduced by Comes (2020), namely, so-called **jellyfish**. In showing how their results can be applied for the purpose of learning from data that has an underlying alternating group symmetry, we simplify their approach and provide proofs that are more accessible to the machine learning community.

The content of this chapter is modelled on the paper:

Edward Pearce-Crump. (2023b) How Jellyfish Characterise Alternating Group Equivariant Neural Networks. *Proceedings of the 40th International Conference on Machine Learning* (ICML 2023), PMLR 202:27483-27495. [Poster]

4.1 Characterisation of Alternating Group Equivariant Linear Layer Functions

Recall from Section 2.2 that $(\mathbb{R}^n)^{\otimes k}$ is a representation of A_n, where the representation for a given element $\sigma \in A_n$ maps a standard basis element e_I in $(\mathbb{R}^n)^{\otimes k}$ to $e_{\sigma(I)}$ in $(\mathbb{R}^n)^{\otimes k}$ and is extended linearly, as in (2.17).

We begin with the following proposition which says that, by restricting to the alternating group, the bijective correspondence between equivariant basis matrices and orbits has not changed.

Proposition 4.1. *The basis elements of* $\mathrm{Hom}_{A_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ *are in bijective correspondence with the orbits coming from the action of* A_n *on the* $(l+k)$*-fold Cartesian product set* $[n]^{l+k}$.

Proof. The proof is almost identical to the one for Proposition 3.9, except we replace S_n with A_n. $\square$

However, what has changed in restricting to the alternating group is that the A_n orbits on $[n]^{l+k}$, and consequently the basis elements of $\mathrm{Hom}_{A_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$, are not necessarily in bijective correspondence with the orbit basis diagrams in $P_k^l(n)$ having at most n blocks. This is because some of the S_n orbits on $[n]^{l+k}$ — all of which are in bijective correspondence with the orbit basis diagrams in $P_k^l(n)$ having at most n blocks by Proposition 3.10 — become the disjoint union of more than one A_n orbit on $[n]^{l+k}$. Consequently, we say that an S_n orbit on $[n]^{l+k}$ **splits** if it is the disjoint union of more than one A_n orbit on $[n]^{l+k}$. Later, we sometimes say that an orbit basis diagram having at most n blocks splits, with the bijection between S_n orbits on $[n]^{l+k}$ and orbit basis diagrams having at most n blocks being understood. Our first task is to identify which S_n orbits on $[n]^{l+k}$ split and which do not.

Proposition 4.2. *Let* x_π *be a orbit basis diagram in* $P_k^l(n)$ *having at most n blocks, and consider its corresponding* S_n *orbit on* $[n]^{l+k}$, $O_{S_n}((I_\pi, J_\pi))$.

- *If the number of blocks in* x_π *is less than or equal to* $n-2$*, then* $O_{S_n}((I_\pi, J_\pi))$ *does not split.*
- *Otherwise,* $O_{S_n}((I_\pi, J_\pi))$ *splits into a disjoint union of exactly two* A_n *orbits.*

Proof. Consider the case where x_π has n blocks. Choose some arbitrary element (I_α, J_α) in the orbit $O_{S_n}((I_\pi, J_\pi))$, and consider $\mathrm{Stab}_{A_n}((I_\alpha, J_\alpha))$, the stabilizer of (I_α, J_α) under the action of A_n. As (I_α, J_α) is a labelling of the blocks for x_π, this implies that all of the elements in $[n]$ must appear in the tuple (I_α, J_α). Therefore, the only element of A_n that fixes the entries of (I_α, J_α) is the identity element; that is, $\text{Stab}_{A_n}((I_\alpha, J_\alpha)) = \{e_{A_n}\}$. Hence, by the Orbit–Stabilizer Theorem, we have that $|O_{A_n}((I_\alpha, J_\alpha))| = \frac{n!}{2}$. Since $|O_{S_n}((I_\pi, J_\pi))| = n!$ in this case, by another application of the Orbit–Stabilizer Theorem, and because (I_α, J_α) was an arbitary element of $O_{S_n}((I_\pi, J_\pi))$, we see that $O_{S_n}((I_\pi, J_\pi))$ splits into a disjoint union of exactly two A_n orbits.

Similarly, for the case where x_π has $n-1$ blocks, we see that $\text{Stab}_{A_n}((I_\alpha, J_\alpha)) = \{e_{A_n}\}$, and so, by the same argument, $O_{S_n}((I_\pi, J_\pi))$ splits into a disjoint union of exactly two A_n orbits.

Finally, if the number of blocks t in x_π is less than or equal to $n-2$, we have that $\text{Stab}_{A_n}((I_\alpha, J_\alpha)) \cong A_{n-t}$, and so, by the Orbit–Stabilizer Theorem, we see that $|O_{A_n}((I_\alpha, J_\alpha))| = |O_{S_n}((I_\pi, J_\pi))|$. Consequently, $O_{S_n}((I_\pi, J_\pi))$ does not split. □

Remark 4.3. The case for $t \leq n-2$ in Theorem 4.2 was proven independently in Maron et al. (2019b).

From Proposition 4.2, we immediately obtain the following two theorems.

Theorem 4.4. *For all non-negative integers l, k and positive integers n, if x_π is an orbit basis diagram in $P_k^l(n)$ having at most $n-2$ blocks, then x_π corresponds bijectively to a basis element of* $\text{Hom}_{A_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$. *Otherwise, x_π corresponds to two basis elements of* $\text{Hom}_{A_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$.

Theorem 4.5. *For all non-negative integers l, k and positive integers n, we have that*

$$\dim \text{Hom}_{A_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l}) = \sum_{t=1}^{n-2} \left\{ {l+k \atop t} \right\} + 2 \left\{ {l+k \atop n-1} \right\} + 2 \left\{ {l+k \atop n} \right\} \tag{4.1}$$

In other words, Theorem 4.4 tells us that, having chosen the standard basis for $\mathbb{R}^n$, finding a basis for $\text{Hom}_{A_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ is very similar to finding a basis for $\text{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$, since this amounts once again to considering all of the orbit basis diagrams in $P_k^l(n)$ having at most n blocks.

Indeed, if the number of blocks in an orbit basis diagram x_π is at most $n-2$, then we see that X_π as given in (3.31) is a basis element of $\text{Hom}_{A_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$, since, in this case, the S_n orbit on $[n]^{l+k}$ does not split, by Theorem 4.2.

The question remains as to how to take an orbit basis diagram x_π having either $n-1$ or n blocks and use it to obtain the two basis elements of $\text{Hom}_{A_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ that it corresponds to. Said differently, we would like to take such an orbit basis diagram x_π and identify the two A_n orbits that its corresponding S_n orbit on $[n]^{l+k}$, $O_{S_n}((I_\pi, J_\pi))$, splits into.

4.1.1 Enter: the Jellyfish

Comes (2020) originally came up with the idea of using **jellyfish** to identify the two A_n orbits; here, however, our choice of exposition is rather different in that we do not use any category theory to obtain the important results. In the process, we supply proofs that are more accessible and are easier to understand than those that were given in their paper.

We begin by defining the following map.

Definition 4.6. For any positive integer n, we define the **determinant map**

$$\det : (\mathbb{R}^n)^{\otimes n} \to \mathbb{R} \tag{4.2}$$

on the standard basis of $(\mathbb{R}^n)^{\otimes n}$ by

$$e_I = e_{i_1} \otimes \cdots \otimes e_{i_n} \mapsto \begin{vmatrix} e_{i_1} & \dots & e_{i_n} \end{vmatrix} \tag{4.3}$$

and extend linearly, where the right hand side of (4.3) is the determinant of an $n \times n$ matrix.

It is important to highlight that the determinant map (4.2) is different from the determinant operator on an $n \times n$ matrix. In particular, the determinant map is a linear map on $(\mathbb{R}^n)^{\otimes n}$ whereas the determinant operator on $n \times n$ matrices is not linear. Its usefulness in obtaining basis matrices is derived from the following lemma.

Lemma 4.7. *The determinant map is an element of* $\mathrm{Hom}_{A_n}((\mathbb{R}^n)^{\otimes n}, \mathbb{R})$. *However, it is not an element of* $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes n}, \mathbb{R})$.

Proof. It is clear that the determinant map is a linear map.

Then, for any $\sigma \in S_n$, we have that

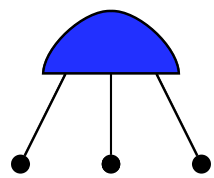
$$\det(\rho_n(\sigma)[e_I]) = \mathrm{sgn}(\sigma)\det(e_I) \tag{4.4}$$

since a transposition in S_n corresponds to swapping two columns of the $n \times n$ matrix.

As $\mathrm{sgn}(\sigma) = 1$ for all $\sigma \in A_n$, (4.4) shows that the determinant map is an element of $\mathrm{Hom}_{A_n}((\mathbb{R}^n)^{\otimes n}, \mathbb{R})$. It is enough to see that for any odd permutation in S_n, (4.4) implies that the determinant map is not an element of $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes n}, \mathbb{R})$. □

For each positive integer n, we represent the determinant map by a diagram that has a single row of n vertices, each of which is attached to a blue head. We will call this diagram a **jellyfish**, for obvious reasons.

Example 4.8. If $n = 3$, the jellyfish has the form



(4.5)

and it corresponds to the A_3-equivariant determinant map $(\mathbb{R}^3)^{\otimes 3} \to \mathbb{R}$.

We introduce the following useful lemma.

Lemma 4.9. *The determinant map applied to a standard basis vector of* $(\mathbb{R}^n)^{\otimes n}$ *gives either* $-1, 0$ *or* $+1$.

Proof. Let $e_I = e_{i_1} \otimes \cdots \otimes e_{i_n}$ be a standard basis vector of $(\mathbb{R}^n)^{\otimes n}$. If $i_x = i_y$ for some $1 \leq x \neq y \leq n$, then det$(e_I) = 0$ since two columns in the $n \times n$ matrix will be the same. Otherwise, $\{i_1, \ldots, i_n\}$ is some permutation of $[n]$. Defining $e_{[n]} \coloneqq e_1 \otimes \cdots \otimes e_n$, we know that det$(e_{[n]}) = +1$. Since, in this case, $e_I = \rho_n(\sigma)e_{[n]}$ for some $\sigma \in S_n$, we can use (4.4) to calculate det(e_I), giving a result of ± 1. □

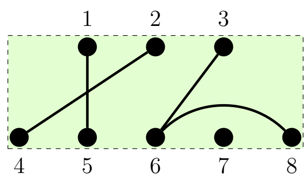
Remark 4.10. We can view Lemma 4.9 in another way; namely, that the determinant map splits basis vectors of $(\mathbb{R}^n)^{\otimes n}$ into disjoint classes. Let us call these classes $-1, 0$ and $+1$.

Consequently, Remark 4.10 suggests that the determinant map is a possible candidate function for identifying the A_n orbits that the S_n orbit $O_{S_n}((I_\pi, J_\pi))$ splits into, where π is a set partition of $[l+k]$ having either $n-1$ or n blocks.

However, since the determinant map is a map $(\mathbb{R}^n)^{\otimes n} \to \mathbb{R}$, and the elements of $O_{S_n}((I_\pi, J_\pi))$ are elements of $[n]^{l+k}$, to use the determinant map to try to identify the A_n orbits in $O_{S_n}((I_\pi, J_\pi))$, it would be useful to create a map $g_\pi : (\mathbb{R}^n)^{\otimes l+k} \to (\mathbb{R}^n)^{\otimes n}$ that corresponds bijectively with the set partition π, since such a map would project standard basis elements of $(\mathbb{R}^n)^{\otimes l+k}$ onto, at the very least, a linear combination of basis elements of $(\mathbb{R}^n)^{\otimes n}$. We could then use the splitting property of the determinant map on such linear combinations to try to split the elements of $O_{S_n}((I_\pi, J_\pi))$ into different classes. However, in order to use the three classes given in Remark 4.10, we cannot choose any map $(\mathbb{R}^n)^{\otimes l+k} \to (\mathbb{R}^n)^{\otimes n}$. We will see that with a clever choice of g_π, the determinant map then splits the elements of $O_{S_n}((I_\pi, J_\pi))$ into the ± 1 classes and sends all other elements of $[n]^{l+k}$ to the 0 class.

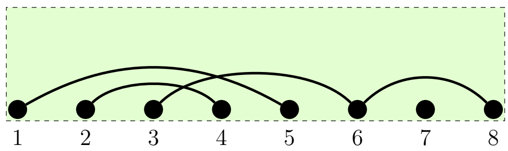
We create g_π as follows. Flatten the orbit basis diagram x_π that corresponds to π. Add a new top row of n vertices above the row of $l+k$ vertices. From the bottom row, take the lowest numbered vertex in block i of π, $1 \leq i \leq t$, where $t = n-1$ or n, and connect that vertex to vertex i in the top row. Call this new diagram b_π.

Example 4.11. Let π be the set partition $\{1, 5 \mid 2, 4 \mid 3, 6, 8 \mid 7\}$ having 4 blocks and suppose that $l = 3$, $k = 5$ and $n = 5$. The orbit basis diagram x_π corresponding to π is



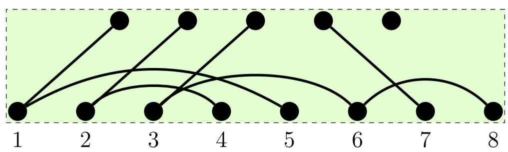
(4.6)

Flattening the orbit basis diagram gives



(4.7)

Consequently, after adding $n = 5$ new nodes to the top row and connecting the lowest numbered vertex in each block in the bottom row to its appropriate top row vertex, the new diagram b_π is



(4.8)

We claim the following:

Proposition 4.12. *The diagram b_π that we constructed from an orbit basis diagram x_π in $P_k^l(n)$ having either $n-1$ or n blocks corresponds bijectively with a basis element of* $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes l+k}, (\mathbb{R}^n)^{\otimes n})$*. Consequently, we define g_π to be this basis element.*

Proof. It is clear that the diagram b_π is an orbit basis diagram in $P_{l+k}^n(n)$ having exactly n blocks. Since each orbit basis diagram in $P_{l+k}^n(n)$ having at most n blocks corresponds bijectively with a basis element of $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes l+k}, (\mathbb{R}^n)^{\otimes n})$, by Theorem 3.18, we obtain the result. $\square$

The following result is immediate from the construction of the diagram b_π.

Proposition 4.13. *In the standard basis of matrix units of* $\mathrm{Hom}((\mathbb{R}^n)^{\otimes l+k}, (\mathbb{R}^n)^{\otimes n})$, g_π *is given by*

$$g_\pi := \sum_{(K,I,J)\in O_{S_n}((K_\pi,I_\pi,J_\pi))} E_{K,(I,J)} \tag{4.9}$$

where

$$K_\pi := (1, 2, \dots, n) \tag{4.10}$$

and $O_{S_n}((K_\pi, I_\pi, J_\pi))$ *is defined to be*

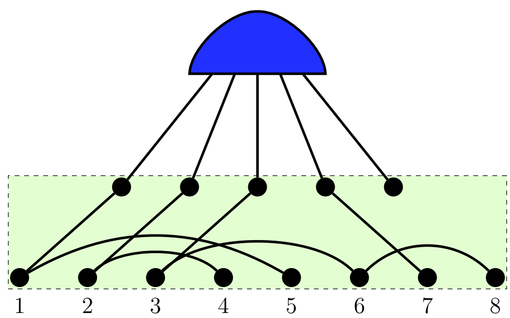
$$\left\{(K, I, J) \;\middle|\; (I, J) \in O_{S_n}((I_\pi, J_\pi)) \text{ and if } (I, J) = \sigma(I_\pi, J_\pi), \text{ then } K := \sigma(K_\pi) \right\} \tag{4.11}$$

In order to see that g_π has the properties that we need, we first define the following map.

Definition 4.14. Let x_π be an orbit basis diagram in $P_k^l(n)$ having either $n-1$ or n blocks. Then we define $f_\pi := \det \circ g_\pi : (\mathbb{R}^n)^{\otimes l+k} \to \mathbb{R}$. Clearly, f_π corresponds bijectively with the underlying set partition π of $[l+k]$.

We can associate to f_π a diagram that is built from the composition of b_π — which corresponds to the map g_π — and a jellyfish — which corresponds to the determinant map.

Example 4.15. If π is the set partition from Example 4.11, where we had that $l = 3$, $k = 5$ and $n = 5$, then the diagram that is associated to f_π is



(4.12)

Proposition 4.16. *The map* f_π *is an element of* $\mathrm{Hom}_{A_n}((\mathbb{R}^n)^{\otimes l+k}, \mathbb{R})$.

Proof. Let $e_{(I,J)}$ be any standard basis vector in $(\mathbb{R}^n)^{\otimes l+k}$, where $I \in [n]^l$ and $J \in [n]^k$.

Then, for all $\sigma \in A_n$, we have that

$$f_\pi(\rho_{l+k}(\sigma)[e_{(I,J)}]) = \det \circ g_\pi(\rho_{l+k}(\sigma)[e_{(I,J)}]) \tag{4.13}$$
$$= \det(\rho_n(\sigma)[g_\pi(e_{(I,J)})]) \tag{4.14}$$
$$= \det(g_\pi(e_{(I,J)})) \tag{4.15}$$
$$= f_\pi(e_{(I,J)}) \tag{4.16}$$

where, in (4.14), we have used Proposition 4.12, and in (4.15), we have used (4.4). □

We have arrived at the crucial result for our purposes.

Theorem 4.17. *Let $e_{(I,J)}$ be any standard basis vector in $(\mathbb{R}^n)^{\otimes l+k}$, where $I \in [n]^l$, $J \in [n]^k$.*

Then

$$f_\pi : e_{(I,J)} \rightarrow \begin{cases} \pm 1 & \textit{if } (I,J) \in O_{S_n}((I_\pi, J_\pi)) \\ 0 & \textit{otherwise} \end{cases} \tag{4.17}$$

Proof. Let $e_{(I,J)}$ be any standard basis vector in $(\mathbb{R}^n)^{\otimes l+k}$, and suppose that $E_{K,(L,M)}$ is a matrix unit of $\mathrm{Hom}((\mathbb{R}^n)^{\otimes l+k}, (\mathbb{R}^n)^{\otimes n})$. Then, we have that

$$E_{K,(L,M)} e_{(I,J)} = \delta_{((L,M),(I,J))} e_K \tag{4.18}$$

where e_K is a standard basis vector in $(\mathbb{R}^n)^{\otimes n}$.

Hence, by (4.9), we see that, for the otherwise case, $g_\pi(e_{(I,J)}) = 0$, and so $f_\pi(e_{(I,J)}) = 0$.

Now, if $(I,J) \in O_{S_n}((I_\pi, J_\pi))$, then $(I,J) = \sigma(I_\pi, J_\pi)$ for some $\sigma \in S_n$. Hence, by (4.9), (4.18) and the definition of $O_{S_n}((K_\pi, I_\pi, J_\pi))$ given in (4.11), we see that $g_\pi(e_{(I,J)}) = e_K$, where $K = \sigma(K_\pi)$. Since K is a permutation of $[n]$, we can apply Lemma 4.9 to get that $f_\pi(e_{(I,J)}) = \det(e_K) = \pm 1$. □

Consequently, we can define the following two sets:

$$O_\pi^+ \coloneqq \{(I,J) \in O_{S_n}((I_\pi, J_\pi)) \mid f_\pi(e_{(I,J)}) = +1\} \tag{4.19}$$

and

$$O_\pi^- \coloneqq \{(I,J) \in O_{S_n}((I_\pi, J_\pi)) \mid f_\pi(e_{(I,J)}) = -1\} \tag{4.20}$$

We claim the following result.

Theorem 4.18. *O_π^+ and O_π^- are the two A_n orbits that $O_{S_n}((I_\pi, J_\pi))$ splits into.*

Proof. It is enough to show that if $(I, J) \in O_\pi^+$, then $\sigma(I, J) \in O_\pi^+$ for all $\sigma \in A_n$, and if $(I, J) \in O_\pi^-$, then $\sigma(I, J) \in O_\pi^-$ for all $\sigma \in A_n$, where the action of σ on the pair (I, J) is given in (3.19). Indeed, by Proposition 4.16, we have that

$$f_\pi(e_{\sigma(I,J)}) = f_\pi(\rho_{l+k}(\sigma)[e_{(I,J)}]) = f_\pi(e_{(I,J)}) \tag{4.21}$$

Applying Theorem 4.17 gives the result. □

Relabelling O_π^+ as $O_{A_n}((I_\pi^+, J_\pi^+))$, and O_π^- as $O_{A_n}((I_\pi^-, J_\pi^-))$, we obtain the two basis elements of $\mathrm{Hom}_{A_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ from the one orbit basis diagram x_π in $P_k^l(n)$ that has either $n-1$ or n blocks, namely

$$X_\pi^+ := \sum_{(I,J)\in O_{A_n}((I_\pi^+, J_\pi^+))} E_{I,J} \tag{4.22}$$

and

$$X_\pi^- := \sum_{(I,J)\in O_{A_n}((I_\pi^-, J_\pi^-))} E_{I,J} \tag{4.23}$$

We summarise the above results into the following theorems.

Theorem 4.19. *For all non-negative integers l, k and positive integers n, the union of the two sets*

$$\{X_\pi \mid \pi \text{ is a set partition of } [l+k] \text{ having at most } n-2 \text{ blocks}\} \tag{4.24}$$

$$\{X_\pi^+, X_\pi^- \mid \pi \text{ is a set partition of } [l+k] \text{ having either } n-1 \text{ or } n \text{ blocks}\} \tag{4.25}$$

forms a basis of $\mathrm{Hom}_{A_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$. *Its dimension is given in Theorem 4.5.*

Theorem 4.20 (Alternating Group Equivariant Weight Matrices). *For all non-negative integers l, k and positive integers n, the weight matrix W that appears in an A_n-equivariant linear layer function from $(\mathbb{R}^n)^{\otimes k}$ to $(\mathbb{R}^n)^{\otimes l}$ must be of the form*

$$W = \sum_{\pi\in\Pi_{l+k,n-2}} \lambda_\pi X_\pi + \sum_{\pi\in\Pi_{l+k,n}\setminus\Pi_{l+k,n-2}} \left[\lambda_\pi^+ X_\pi^+ + \lambda_\pi^- X_\pi^-\right] \tag{4.26}$$

A General Procedure for Calculating the Weight Matrix of an A_n-Equivariant Linear Layer Function from $(\mathbb{R}^n)^{\otimes k}$ to $(\mathbb{R}^n)^{\otimes l}$.

Perform the following steps:

1. Calculate all of the set partitions π of $\{1, \dots, l+k\}$ that have at most n blocks.
2. Express each set partition π as an orbit basis diagram x_π in $P_k^l(n)$. Consider each x_π in turn.
3. If x_π has at most $n-2$ blocks, then calculate its associated basis matrix X_π using the procedure that was given for the symmetric group S_n (see the blue-coloured box in Chapter 3).
4. Otherwise, flatten x_π, insert a new top row consisting of n vertices, connect the lowest numbered vertex in each block i to vertex i in the top row, and attach a jellyfish with n legs to the top row. The function that is associated with this new diagram is f_π.
5. Calculate the orbit $O_{S_n}((I_\pi, J_\pi))$, and split it into two A_n orbits, $O_{A_n}((I_\pi^+, J_\pi^+))$ and $O_{A_n}((I_\pi^-, J_\pi^-))$. Let each (I,J)-tuple index a standard basis element $e_{(I,J)}$ of $(\mathbb{R}^n)^{\otimes l+k}$. The first orbit consists of all (I,J)-tuples such that $e_{(I,J)}$ is mapped to $+1$ under f_π, whereas the second orbit consists of all (I,J)-tuples such that $e_{(I,J)}$ is mapped to -1.
6. The basis matrix X_π^+ is the sum of all matrix units $E_{I,J}$ that are indexed by the (I,J)-tuples in $O_{A_n}((I_\pi^+, J_\pi^+))$, and a similar sum gives the basis matrix X_π^-.
7. Weight all basis matrices by a parameter, and add them all together to give the overall A_n-equivariant weight matrix.

4.1.2 General Procedure and Examples

In the pale-coloured box, we provide a general procedure for how to calculate the weight matrix that appears in an A_n-equivariant neural network from $(\mathbb{R}^n)^{\otimes k}$ to $(\mathbb{R}^n)^{\otimes l}$ in the standard basis of $\mathbb{R}^n$. The procedure is very similar to the one that appeared in the teal-coloured box for permutation equivariant neural networks, except we now have to calculate two basis matrices using the n-legged jellyfish for each set partition of $[l+k]$ that has either $n-1$ or n blocks.

We finish this chapter with some examples.

Example 4.21. Recall Example 3.20, where we found the weight matrix for an S_4-equivariant linear layer function from $\mathbb{R}^4$ to $\mathbb{R}^4$. The weight matrix was presented in (3.36).

Suppose that we change S_4 to A_4. Now, since the orbit basis diagrams that determined the weight matrix for S_4 had either 1 or 2 blocks, Proposition 4.2 implies that neither of their associated S_4 orbits in $[4]^{1+1}$ split.

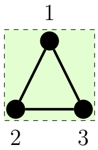
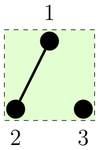
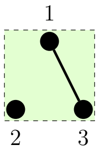
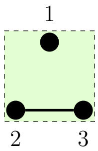
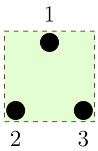
Orbit Basis Diagram x_π	Block Labelling $(I_\pi \mid J_\pi)$	Basis Matrix X_π
	$(1 \mid 1, 1)$	$\begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \left[\begin{array}{ccc\|ccc\|ccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right] \end{array}$
	$(1 \mid 1, 2)$	$\begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \left[\begin{array}{ccc\|ccc\|ccc} 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{array}\right] \end{array}$
	$(1 \mid 2, 1)$	$\begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \left[\begin{array}{ccc\|ccc\|ccc} 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \end{array}\right] \end{array}$
	$(1 \mid 2, 2)$	$\begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \left[\begin{array}{ccc\|ccc\|ccc} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{array}\right] \end{array}$
	$(1 \mid 2 \mid 3)$	$\begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \left[\begin{array}{ccc\|ccc\|ccc} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array}$

Figure 4.1: We use Theorem 3.18, or, equivalently, we apply the procedure that appears in the teal-coloured box in Chapter 3, to obtain a basis of $\mathrm{Hom}_{S_3}((\mathbb{R}^3)^{\otimes 2}, \mathbb{R}^3)$ from all of the orbit basis diagrams in $P_2^1(3)$ that have at most 3 blocks.

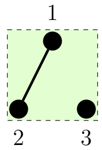
Hence, the weight matrix for an A_4-equivariant linear layer function from $\mathbb{R}^4$ to $\mathbb{R}^4$ is exactly the same matrix.

Example 4.22. Suppose that we would like to find the weight matrix for an A_3-equivariant linear layer function from $(\mathbb{R}^3)^{\otimes 2}$ to $\mathbb{R}^3$.

We first find a basis of $\mathrm{Hom}_{S_3}((\mathbb{R}^3)^{\otimes 2}, \mathbb{R}^3)$. Since $n = 3$, $k = 2$, and $l = 1$, we need to find all orbit basis diagrams in $P_2^1(3)$ that have at most 3 blocks. They are shown in Figure 4.1, together with the basis matrices for $\mathrm{Hom}_{S_3}((\mathbb{R}^3)^{\otimes 2}, \mathbb{R}^3)$ that correspond with them.

To find the basis for $\mathrm{Hom}_{A_3}((\mathbb{R}^3)^{\otimes 2}, \mathbb{R}^3)$, we need to show which orbit basis diagrams in Figure 4.1 split and which do not. As $n = 3$, only the first diagram corresponding to the set partition $\{1, 2, 3\}$ does not split, since it has only one block.

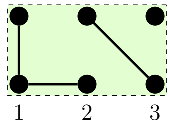
We show in full how to find the two basis elements X_π^+, X_π^- in $\mathrm{Hom}_{A_3}((\mathbb{R}^3)^{\otimes 2}, \mathbb{R}^3)$ that corresponds to the second orbit basis diagram x_π in Figure 4.1, namely



(4.27)

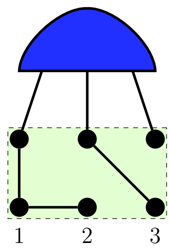
and state in Figure 4.2 what they are for the other orbit basis diagrams that split.

First, we flatten the orbit basis diagram x_π. Then we add a new top row consisting of $n = 3$ vertices, and connect the lowest numbered vertex in each block i of π to vertex i in the top row. Hence, we obtain the diagram b_π, which is



(4.28)

Next, we attach a three-legged jellyfish to the top row of vertices, giving



(4.29)

This is the diagram that is associated with the function f_π that is given in Proposition 4.16.

To obtain the two A_3 orbits $O_{A_3}((I_\pi^+, J_\pi^+))$ and $O_{A_3}((I_\pi^-, J_\pi^-))$ corresponding to π, we apply Theorem 4.17.

From Figure 4.1, we see that

$$O_{S_3}((I_\pi, J_\pi)) = \{(1,1,2),(1,1,3),(2,2,1),(2,2,3),(3,3,1),(3,3,2)\} \tag{4.30}$$

We now split this orbit up by applying f_π to every $e_{(I,J)}$ such that (I,J) is a tuple in $O_{S_3}((I_\pi, J_\pi))$.

For example, we see that

$$f_\pi(e_{(1,1,2)}) = \det \circ g_\pi(e_{(1,1,2)}) = \det(e_{(1,2,1)} + e_{(1,2,2)} + e_{(1,2,3)}) = +1 \tag{4.31}$$

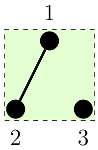
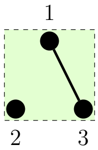
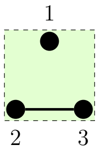
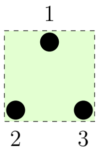
Orbit Basis Diagram x_π	Standard Basis Element X_π^+	Standard Basis Element X_π^-
	$\begin{array}{cc} & \begin{array}{ccc\|ccc\|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc\|ccc\|ccc} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right] \end{array}$	$\begin{array}{cc} & \begin{array}{ccc\|ccc\|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc\|ccc\|ccc} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{array}\right] \end{array}$
	$\begin{array}{cc} & \begin{array}{ccc\|ccc\|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc\|ccc\|ccc} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array}$	$\begin{array}{cc} & \begin{array}{ccc\|ccc\|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc\|ccc\|ccc} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{array}\right] \end{array}$
	$\begin{array}{cc} & \begin{array}{ccc\|ccc\|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc\|ccc\|ccc} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array}$	$\begin{array}{cc} & \begin{array}{ccc\|ccc\|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc\|ccc\|ccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{array}\right] \end{array}$
	$\begin{array}{cc} & \begin{array}{ccc\|ccc\|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc\|ccc\|ccc} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array}$	$\begin{array}{cc} & \begin{array}{ccc\|ccc\|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc\|ccc\|ccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array}$

Figure 4.2: By considering which set partitions of $[1+2]$ having at most 3 blocks split, together with those that do not split, we obtain a basis of $\mathrm{Hom}_{A_3}((\mathbb{R}^3)^{\otimes 2}, \mathbb{R}^3)$. Note that we have not shown the first orbit basis diagram that appeared in Figure 4.1 here since it does not split.

whereas

$$f_\pi(e_{(1,1,3)}) = \det \circ g_\pi(e_{(1,1,3)}) = \det(e_{(1,3,1)} + e_{(1,3,2)} + e_{(1,3,3)}) = -1 \tag{4.32}$$

recalling, in the process, that the determinant map is linear.

Calculating this for all six tuples in $O_{S_3}((I_\pi, J_\pi))$, by (4.19) and (4.20), we get that

$$O_{A_3}((I_\pi^+, J_\pi^+)) = \{(1,1,2), (2,2,3), (3,3,1)\} \tag{4.33}$$

and

$$O_{A_3}((I_\pi^-, J_\pi^-)) = \{(1,1,3), (2,2,1), (3,3,2)\} \tag{4.34}$$

and so, by (4.22) and (4.23),

$$X_\pi^+ = E_{(1|1,2)} + E_{(2|2,3)} + E_{(3|3,1)} \tag{4.35}$$

and

$$X_\pi^- = E_{(1|1,3)} + E_{(2|2,1)} + E_{(3|3,2)} \tag{4.36}$$

Consequently, using the matrix that corresponds to the first orbit basis diagram in Figure 4.1 and all of the matrices that are given in Figure 4.2, we see that the weight matrix for an A_3-equivariant linear layer function from $(\mathbb{R}^3)^{\otimes 2}$ to $\mathbb{R}^3$ is of the form

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array}\begin{array}{c} \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \left[\begin{array}{ccc|ccc|ccc} \lambda_1 & \lambda_2 & \lambda_3 & \lambda_4 & \lambda_6 & \lambda_8 & \lambda_5 & \lambda_9 & \lambda_7 \\ \lambda_7 & \lambda_5 & \lambda_9 & \lambda_3 & \lambda_1 & \lambda_2 & \lambda_8 & \lambda_4 & \lambda_6 \\ \lambda_6 & \lambda_8 & \lambda_4 & \lambda_9 & \lambda_7 & \lambda_5 & \lambda_2 & \lambda_3 & \lambda_1 \end{array}\right] \end{array} \tag{4.37}$$

for weights $\lambda_1, \ldots, \lambda_9 \in \mathbb{R}$.

Chapter 5

Brauer's Group Equivariant Neural Networks

Although much of the initial focus in equivariant deep learning was on constructing permutation equivariant neural networks, some researchers have looked at the problem of how to construct neural networks that are equivariant to other groups. In particular, Finzi et al. (2021) introduced a numerical algorithm for constructing neural networks that are equivariant to a number of matrix groups, including the orthogonal group $O(n)$, the special orthogonal group $SO(n)$, and the symplectic group $Sp(n)$, where the layers are tensor power spaces of $\mathbb{R}^n$. However, their algorithm is only viable for small values of n and for low tensor power orders of $\mathbb{R}^n$, since it runs out of memory on larger values of n and on higher tensor power orders. For the orthogonal and special orthogonal groups, Villar et al. (2021) characterised all invariant scalar functions $(\mathbb{R}^n)^{\otimes k} \rightarrow \mathbb{R}$ and all equivariant vector functions $(\mathbb{R}^n)^{\otimes k} \rightarrow \mathbb{R}^n$ as a sum involving $O(n)$ and $SO(n)$ invariant scalar functions. They then used standard multilayer perceptrons to learn the invariant scalar functions.

In this chapter, we consider group equivariant neural networks of the form given in Section 2.3 for three groups that were defined in Section 2.1: the orthogonal group $O(n)$, the special orthogonal group $SO(n)$, and the symplectic group $Sp(n)$. We improve upon Finzi et al. (2021) and Villar et al. (2021) by deriving a complete, analytic characterisation of all of the weight matrices that can appear in the linear layers of these neural networks. In particular, we construct a spanning set of matrices for the vector space $\mathrm{Hom}_{G(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ in the standard basis of $\mathbb{R}^n$ for each of the groups $G(n)$ in question.

Our approach is motivated once again by Schur–Weyl duality. In the Introduction, we said that Richard Brauer, a student of Issai Schur, found a number of Schur–Weyl dualities. Brauer established a version of Schur–Weyl duality for the orthogonal group $O(n)$, the special orthogonal group $SO(n)$, and the symplectic group $Sp(n)$ in his 1937 paper *On Algebras Which are Connected with the Semisimple*

Continuous Groups (Brauer, 1937). Similar to the other Schur–Weyl dualities that we have seen so far, Brauer's version describes a one-to-one correspondence between the irreducibles of a group and the irreducibles of an algebra that appear in the decomposition of the tensor power space $(\mathbb{R}^n)^{\otimes k}$. The most important of these algebras, now known as the Brauer algebra, has since been studied by a number of authors, including Weyl (1939), Brown (1955, 1956), Hanlon and Wales (1989), Grood (1999), and Lehrer and Zhang (2012).

The key result behind each of these Schur–Weyl dualities is a map that Brauer constructed from the algebra that corresponds to the group $G(n)$ onto the endomorphism algebra $\text{End}_{G(n)}((\mathbb{R}^n)^{\otimes k})$. Our approach is adapted from Brauer (1937), which focused, in part, on finding a spanning set of matrices for $\text{End}_{G(n)}((\mathbb{R}^n)^{\otimes k})$ for each of the groups $G(n)$ in question. Brauer achieved his main result by applying the First Fundamental Theorem of Invariant Theory for each of the groups to a spanning set of invariants, one for each group $G(n)$, that he showed are in bijective correspondence with the spanning set of matrices for $\text{End}_{G(n)}((\mathbb{R}^n)^{\otimes k})$. In particular, for each group $G(n)$, he associated a diagram with each element of the spanning set of invariants, which we describe in further detail below. In doing so, he constructed a bijective correspondence between a set of such diagrams and a spanning set of matrices for $\text{End}_{G(n)}((\mathbb{R}^n)^{\otimes k})$ in the standard/symplectic basis of $\mathbb{R}^n$. By adapting the approach taken in Brauer's paper, we provide a theoretical foundation for how to construct these neural networks.

The content of this chapter is modelled on the paper:

Edward Pearce-Crump. (2023a) Brauer's Group Equivariant Neural Networks. *Proceedings of the 40th International Conference on Machine Learning* (ICML 2023), PMLR 202:27461-27482. [Live Oral]

5.1 Preliminaries for Brauer's Invariant Argument

We begin by presenting some preliminary material that will be used in the adapted version of Brauer's Invariant Argument. We show that, for each k-order tensor power space of $\mathbb{R}^n$ that is a representation of a subgroup of $GL(n)$, we can construct a representation on the k-order tensor power of the dual space of $\mathbb{R}^n$, $((\mathbb{R}^n)^*)^{\otimes k}$. Then, for both of these representations, we can derive the form of their matrix representations when a basis of $\mathbb{R}^n$ is chosen.

Let $G(n)$ be any subgroup of $GL(n)$. Recall that the vector space $\mathbb{R}^n$ has, associated with it, its dual vector space, $(\mathbb{R}^n)^*$. Let $B \coloneqq \{b_i \mid i \in [n]\}$ be any basis of $\mathbb{R}^n$. Then there is a basis B^* of the dual space, $(\mathbb{R}^n)^*$, that is associated with the basis B of $\mathbb{R}^n$. B^* is called the dual basis, and it is defined to be $\{b_i^* \mid i \in [n]\}$ where $b_i^*(b_j) := \delta_{i,j}$.

In particular, coordinates on $\mathbb{R}^n$ with respect to the basis B are linear functions, that is, elements of $(\mathbb{R}^n)^*$. Indeed, if $v = \sum_j x_j b_j$, then the coordinate function x_j can be identified with the dual basis vector b_j^* since

$$b_j^*(v) = b_j^*(\sum_i x_i b_i) = \sum_i x_i b_j^*(b_i) = x_j \tag{5.1}$$

Since $G(n)$ is a subgroup of $GL(n)$, we have from (2.14) that $\mathbb{R}^n$ is a representation of $G(n)$, which is denoted by ρ_1. In fact, for all $f \in G(n)$, we have that $\rho_1(f) = f$.

Consequently, if v is any vector in $\mathbb{R}^n$, and f is any element of $G(n)$, then the linear transformation

$$\rho_1(f) = f : v \to f(v) \tag{5.2}$$

can be expressed in its matrix representation, choosing B as the basis for each copy of $\mathbb{R}^n$, as the matrix $a(f) = (a_{i,j})$ such that

$$y_i = \sum_j a_{i,j} x_j \tag{5.3}$$

where, in the basis B,

$$v = \sum_j x_j b_j \text{ and } f(v) = \sum_j y_j b_j \tag{5.4}$$

for some coefficients $x_j, y_j \in \mathbb{R}$.

We will sometimes express (5.3) in the form $y = a(f)x$, where x, y are column vectors such that the i^{th} component of each vector is x_i and y_i, respectively.

As $(\mathbb{R}^n, \rho_1)$ is a representation of $G(n)$, we can define another representation of $G(n)$, called the contragredient representation, on the dual space $(\mathbb{R}^n)^*$, as follows

$$(\rho_1^{-1})^\top : G(n) \to GL((\mathbb{R}^n)^*) \tag{5.5}$$

where, for all $f \in G(n)$,

$$(\rho_1^{-1})^\top(f) : (\mathbb{R}^n)^* \to (\mathbb{R}^n)^* \tag{5.6}$$

is defined by

$$(\rho_1^{-1})^\top(f)[u] : v \mapsto u(\rho_1^{-1}(f)(v)) \tag{5.7}$$

One way of understanding the contragredient representation $((\mathbb{R}^n)^*, (\rho_1^{-1})^\top)$ of $G(n)$ is as the action on $(\mathbb{R}^n)^*$ such that all pairings between $(\mathbb{R}^n)^*$ and $\mathbb{R}^n$ remain invariant under their respective actions. Indeed, if $u \in (\mathbb{R}^n)^*$, and $v \in \mathbb{R}^n$, then we see that, for all $f \in G(n)$

$$v \mapsto \rho_1(f)(v) \tag{5.8}$$

and

$$u \mapsto (\rho_1^{-1})^\top(f)[u] \tag{5.9}$$

and so

$$u(v) \mapsto (\rho_1^{-1})^\top(f)[u](\rho_1(f)(v)) = u(\rho_1^{-1}(f)(\rho_1(f)(v))) = u(v) \tag{5.10}$$

Hence, expressing $u \in (\mathbb{R}^n)^*$ in the dual basis B^* as

$$u = \sum_j p_j b_j^* \tag{5.11}$$

and $(\rho_1^{-1})^\top(f)[u] \in (\mathbb{R}^n)^*$ as

$$(\rho_1^{-1})^\top(f)[u] = \sum_j q_j b_j^* \tag{5.12}$$

we see that, for any $f \in G(n)$, the linear transformation

$$(\rho_1^{-1})^\top(f) : u \to (\rho_1^{-1})^\top(f)[u] \tag{5.13}$$

can be expressed in its matrix representation as

$$pa(f)^{-1} = q \tag{5.14}$$

as a result of (5.7), and so

$$p = qa(f) \tag{5.15}$$

that is,

$$p_i = \sum_j q_j a_{j,i} \tag{5.16}$$

In (5.15), p, q are row vectors such that the i^{th} component of each vector is p_i and q_i, respectively.

We also have from (2.15) that $(\mathbb{R}^n)^{\otimes k}$ is a representation of $G(n)$, which is denoted by ρ_k. In particular, for all $f \in G(n)$, we see that $\rho_k(f) = \rho_1(f)^{\otimes k} = f^{\otimes k}$.

Consequently, if v is any vector in $(\mathbb{R}^n)^{\otimes k}$, and f is any element of $G(n)$, then the linear transformation

$$\rho_k(f) = f^{\otimes k} : v \to f^{\otimes k}(v) \tag{5.17}$$

can be expressed in its matrix representation, choosing B as the basis for each copy of $\mathbb{R}^n$, as the matrix $A(f) = (A_{I,J})$, over all tuples $I := (i_1, i_2, \dots, i_k)$, $J := (j_1, j_2, \dots, j_k) \in [n]^k$, such that

$$y_I = \sum_J A_{I,J} x_J \tag{5.18}$$

where

$$A_{I,J} = \prod_{r=1}^{k} a_{i_r, j_r} \tag{5.19}$$

and

$$v = \sum_J x_J b_J \text{ and } f(v) = \sum_J y_J b_J \tag{5.20}$$

for some coefficients $x_J, y_J \in \mathbb{R}$ in the basis $\{b_J \mid J \in [n]^k\}$ of $(\mathbb{R}^n)^{\otimes k}$, where

$$b_J := b_{j_1} \otimes b_{j_2} \otimes \dots \otimes b_{j_k} \tag{5.21}$$

As before, since $((\mathbb{R}^n)^{\otimes k}, \rho_k)$ is a representation of $G(n)$, we obtain the contragredient representation, $(((\mathbb{R}^n)^*)^{\otimes k}, (\rho_k^{-1})^\top)$

$$(\rho_k^{-1})^\top : G(n) \to GL(((\mathbb{R}^n)^*)^{\otimes k}) \tag{5.22}$$

where

$$(\rho_k^{-1})^\top(f) : ((\mathbb{R}^n)^*)^{\otimes k} \to ((\mathbb{R}^n)^*)^{\otimes k} \tag{5.23}$$

is defined by

$$(\rho_k^{-1})^\top(f)[u] : v \mapsto u(\rho_k^{-1}(f)(v)) \tag{5.24}$$

In particular, we see that $(\rho_k^{-1})^\top = ((\rho_1^{-1})^\top)^{\otimes k}$.

Hence, expressing $u \in ((\mathbb{R}^n)^*)^{\otimes k}$ in the dual basis $\{b_J^* \mid J \in [n]^k\}$ of $(\mathbb{R}^n)^{\otimes k}$, where

$$b_J^* := b_{j_1}^* \otimes b_{j_2}^* \otimes \dots \otimes b_{j_k}^* \tag{5.25}$$

as

$$u = \sum_J p_J b_J^* \tag{5.26}$$

and $(\rho_k^{-1})^\top(f)[u] \in ((\mathbb{R}^n)^*)^{\otimes k}$ as

$$(\rho_k^{-1})^\top(f)[u] = \sum_J q_J b_J^* \tag{5.27}$$

we see that, for any $f \in G(n)$, the linear transformation

$$(\rho_k^{-1})^\top(f) : u \to (\rho_k^{-1})^\top(f)[u] \tag{5.28}$$

can be expressed in its matrix representation form as

$$p_I = \sum_J q_J A_{J,I} \tag{5.29}$$

where

$$A_{J,I} = \prod_{r=1}^{k} a_{j_r,i_r} \tag{5.30}$$

5.2 Brauer's Invariant Argument, Adapted

We use the preliminaries that we established in the previous section to construct an invariant of $G(n)$ that is in bijective correspondence with an element of $\mathrm{Hom}_{G(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$. The invariant that we will construct will be an element of $\mathrm{Hom}_{G(n)}(((\mathbb{R}^n)^*)^{\otimes l} \otimes (\mathbb{R}^n)^{\otimes k}, \mathbb{R})$. The following argument is an adaptation of the one that appears in Brauer (1937).

Recall that a linear map $\phi : (\mathbb{R}^n)^{\otimes k} \to (\mathbb{R}^n)^{\otimes l}$ is an element of $\mathrm{Hom}_{G(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ if and only if

$$\phi \circ \rho_k(f) = \rho_l(f) \circ \phi \tag{5.31}$$

for all $f \in G(n)$.

Choosing the basis B as the basis for each copy of $\mathbb{R}^n$, the matrix representation of (5.31) is

$$C(\phi)K(f) = L(f)C(\phi) \tag{5.32}$$

where $K(f) = (K_{I,J})$ and $L(f) = (L_{I,J})$ are as in (5.17) and $C(\phi) = (C_{I,J})$.

Hence, (5.32) gives

$$\sum_{J \in [n]^k} C_{I,J} K_{J,M} = \sum_{J \in [n]^l} L_{I,J} C_{J,M} \tag{5.33}$$

where $I \in [n]^l$ and $M \in [n]^k$.

Brauer's trick is as follows.

Let $v(1), v(2), \ldots, v(k)$ be elements of $\mathbb{R}^n$, and suppose that they are all mapped by the same transformation $\rho_1(f)$, for some $f \in G(n)$. Then, by (5.4), in the basis B of $\mathbb{R}^n$, we see that

$$v(r) = \sum_j x_j(r) b_j \tag{5.34}$$

for all $r \in [k]$, and so, by (5.3), we have that

$$y_i(r) = \sum_j a_{i,j} x_j(r) \tag{5.35}$$

for all $r \in [k]$.

Then, considering the tensor product $v(1) \otimes v(2) \otimes \cdots \otimes v(k)$, an element of $(\mathbb{R}^n)^{\otimes k}$, and considering its transformation under $\rho_k(f)$, for the same $f \in G(n)$, we see that the coefficient of the basis vector b_J for $v(1) \otimes v(2) \otimes \cdots \otimes v(k)$, as in (5.20), is

$$\prod_{r=1}^{k} x_{j_r}(r) \tag{5.36}$$

and the coefficient of the basis vector b_I for $\rho_k(f)[v(1) \otimes v(2) \otimes \cdots \otimes v(k)]$ is

$$\prod_{r=1}^{k} y_{i_r}(r) \tag{5.37}$$

and that (5.19) holds, namely that

$$K_{I,J} = \prod_{r=1}^{k} a_{i_r, j_r} \tag{5.38}$$

Hence, by (5.18), we have that

$$\prod_{r=1}^{k} y_{i_r}(r) = \sum_{J \in [n]^k} K_{I,J} \prod_{r=1}^{k} x_{j_r}(r) \tag{5.39}$$

Also, let $u(1), u(2), \ldots, u(l)$ be elements of $(\mathbb{R}^n)^*$, the dual space of $\mathbb{R}^n$. Then, by (5.11), in the dual basis B^*, we have that

$$u(t) = \sum_j p_j(t) b_j^* \tag{5.40}$$

for all $t \in [l]$, and so, by (5.16), we see that

$$p_i(t) = \sum_j q_j(t) a_{j,i} \tag{5.41}$$

for all $t \in [l]$.

Then, considering the tensor product $u(1) \otimes u(2) \otimes \cdots \otimes u(l)$, an element of $((\mathbb{R}^n)^*)^{\otimes l}$, and considering its transformation under $(\rho_l^{-1})^\top(f)$, for the same $f \in G(n)$, we see that the coefficient of the basis vector b_I^* for $u(1) \otimes u(2) \otimes \cdots \otimes u(l)$, as in (5.26), is

$$\prod_{t=1}^{l} p_{i_t}(t) \tag{5.42}$$

and the coefficient of the basis vector b_J^* for $(\rho_l^{-1})^\top(f)[u(1) \otimes u(2) \otimes \cdots \otimes u(l)]$ is

$$\prod_{t=1}^{l} q_{j_t}(t) \tag{5.43}$$

and that (5.30) holds, namely that

$$L_{J,I} = \prod_{t=1}^{l} a_{j_t,i_t} \tag{5.44}$$

Hence, by (5.29), we have that

$$\prod_{t=1}^{l} p_{i_t}(t) = \sum_{J \in [n]^l} \prod_{t=1}^{l} q_{j_t}(t) L_{J,I} \tag{5.45}$$

Multiplying both sides of (5.33) by

$$\prod_{t=1}^{l} q_{i_t}(t) \prod_{r=1}^{k} x_{m_r}(r) \tag{5.46}$$

adding over all tuples $I \in [n]^l, M \in [n]^k$, and applying (5.39) and (5.45) gives us, on the left hand side

$$\sum_{I \in [n]^l, J \in [n]^k} C_{I,J} \prod_{t=1}^{l} q_{i_t}(t) \prod_{r=1}^{k} y_{j_r}(r) \tag{5.47}$$

and on the right hand side

$$\sum_{J \in [n]^l, M \in [n]^k} C_{J,M} \prod_{t=1}^{l} p_{j_t}(t) \prod_{r=1}^{k} x_{m_r}(r) \tag{5.48}$$

This means that

$$\sum_{I \in [n]^l, J \in [n]^k} C_{I,J} \prod_{t=1}^{l} p_{i_t}(t) \prod_{r=1}^{k} x_{j_r}(r) \tag{5.49}$$

is an invariant for the group $G(n)$, that is, it is a linear transformation

$$((\mathbb{R}^n)^*)^{\otimes l} \otimes (\mathbb{R}^n)^{\otimes k} \to \mathbb{R} \tag{5.50}$$

which maps an element of the form

$$u(1) \otimes u(2) \otimes \cdots \otimes u(l) \otimes v(1) \otimes v(2) \otimes \cdots \otimes v(k) \tag{5.51}$$

to (5.49) that is invariant under the action of $G(n)$.

We see that each stage of the argument, from (5.31) to (5.51), is an if and only if statement, since any invariant of $G(n)$ which is linear in any subset $\{v(1), v(2), \ldots, v(k)\}$ of k vectors of $\mathbb{R}^n$ and any subset $\{u(1), u(2), \ldots, u(l)\}$ of l vectors in $(\mathbb{R}^n)^*$, and is a homogeneous function of their union, must be of the form (5.49), since any invariant of these $l + k$ elements must be a linear combination of the elements $\prod_{t=1}^{l} p_{i_t}(t) \prod_{r=1}^{k} x_{j_r}(r)$, where $x_{j_r}(r)$ is the j_r^{th} coefficient of $v(r)$ when expressed in some basis B of $\mathbb{R}^n$, and $p_{i_t}(t)$ is the i_t^{th} coefficient of $u(t)$ when expressed in its dual basis B^*. Hence, starting from (5.49) and running the argument in reverse gives (5.31), and therefore shows that it is an if and only if statement.

In particular, this means that each element of $\mathrm{Hom}_{G(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$, having chosen the basis B for each copy of $\mathbb{R}^n$, is in bijective correspondence with an invariant

$$((\mathbb{R}^n)^*)^{\otimes l} \otimes (\mathbb{R}^n)^{\otimes k} \to \mathbb{R} \tag{5.52}$$

of $G(n)$ of the form (5.49), as desired.

5.3 Spanning Set of Invariants

It can be shown that $\mathbb{R}^n$, as a representation of each of the groups $G(n) = O(n)$, $Sp(n)$ and $SO(n)$ (for $Sp(n)$, $n = 2m$), is isomorphic to its dual space $(\mathbb{R}^n)^*$, by using the appropriate bilinear form that is used to define each of the groups in question. See Goodman and Wallach (2009) for more details. Applying this to the outcome of Brauer's Invariant Argument, our goal now is to find a spanning set for $\mathrm{Hom}_{G(n)}((\mathbb{R}^n)^{\otimes l+k}, \mathbb{R})$, where the basis B that we have chosen is the standard/symplectic basis that was given in (2.1) and (2.4), respectively.

We use that each of the groups $G(n) = O(n)$, $Sp(n)$ and $SO(n)$ has a First Fundamental Theorem. These are stated below. For their proofs, see Goodman and Wallach (2009) for more details.

Theorem 5.1 (First Fundamental Theorem for $O(n)$)**.** *Let n be a non-negative integer, and suppose that the real vector space $\mathbb{R}^n$ has associated with it a non-degenerate, symmetric, bilinear form $(\cdot,\cdot)$, as in Section 2.1.*

Let us choose the standard basis for $\mathbb{R}^n$, so that $(\cdot,\cdot)$ becomes the Euclidean inner product on $\mathbb{R}^n$, as defined in (2.3).

If $f : (\mathbb{R}^n)^{\otimes l+k} \to \mathbb{R}$ is a polynomial function on elements in $(\mathbb{R}^n)^{\otimes l+k}$ of the form

$$u(1) \otimes u(2) \otimes \cdots \otimes u(l) \otimes v(1) \otimes v(2) \otimes \cdots \otimes v(k) \tag{5.53}$$

that is $O(n)$-invariant, then it must be a polynomial of the Euclidean inner products

$$(u(i), u(j)), (u(i), v(j)), (v(i), v(j)) \tag{5.54}$$

Theorem 5.2 (First Fundamental Theorem for $Sp(n)$, $n = 2m$)**.** *Let n be a non-negative even integer, and suppose that the real vector space $\mathbb{R}^n$ has associated with it a non-degenerate, skew-symmetric, bilinear form $\langle\cdot,\cdot\rangle$, as in Section 2.1.*

Let us choose the symplectic basis for $\mathbb{R}^n$, so that $\langle\cdot,\cdot\rangle$ becomes the form given in (2.7). Note that, in this basis, the non-degenerate, symmetric, bilinear form $(\cdot,\cdot)$, which we can also associate with $\mathbb{R}^n$, becomes the Euclidean inner product on $\mathbb{R}^n$, as defined in (2.3), since the symplectic basis is standard with respect to $(\cdot,\cdot)$.

If $f : (\mathbb{R}^n)^{\otimes l+k} \to \mathbb{R}$ is a polynomial function on elements in $(\mathbb{R}^n)^{\otimes l+k}$ of the form

$$u(1) \otimes u(2) \otimes \cdots \otimes u(l) \otimes v(1) \otimes v(2) \otimes \cdots \otimes v(k) \tag{5.55}$$

that is $Sp(n)$-invariant, then it must be a polynomial of the Euclidean inner products

$$(u(i), v(j)) \tag{5.56}$$

together with the skew products

$$\langle u(i), u(j)\rangle, \langle v(i), v(j)\rangle \tag{5.57}$$

such that $i < j$ in (5.57).

Theorem 5.3 (First Fundamental Theorem for $SO(n)$)**.** *Let n be a non-negative integer, and suppose that the real vector space $\mathbb{R}^n$ has associated with it a non-degenerate, symmetric, bilinear form $(\cdot,\cdot)$, as in Section 2.1.*

Let us choose the standard basis for $\mathbb{R}^n$, so that $(\cdot,\cdot)$ becomes the Euclidean inner product on $\mathbb{R}^n$, as defined in (2.3).

If $f : (\mathbb{R}^n)^{\otimes l+k} \to \mathbb{R}$ is a polynomial function on elements in $(\mathbb{R}^n)^{\otimes l+k}$ of the form

$$u(1) \otimes u(2) \otimes \cdots \otimes u(l) \otimes v(1) \otimes v(2) \otimes \cdots \otimes v(k) \tag{5.58}$$

that is $SO(n)$-invariant, then it must be a polynomial of the Euclidean inner products

$$(u(i), u(j)), (u(i), v(j)), (v(i), v(j)) \tag{5.59}$$

together with the $n \times n$ subdeterminants of the $n \times (l + k)$ matrix M, which is the matrix having as its columns

$$M := \begin{pmatrix} | & | & & | & | & | & & | \\ u(1) & u(2) & \dots & u(l) & v(1) & v(2) & \dots & v(k) \\ | & | & & | & | & | & & | \end{pmatrix} \tag{5.60}$$

Now, for each group $G(n)$, looking at the invariant (5.49) that was constructed in Brauer's Invariant Argument, considered as a function $(\mathbb{R}^n)^{\otimes l+k} \to \mathbb{R}$, we see that

- each term in (5.49) contains a single component from each of the vectors $u(1), u(2), \dots, u(l)$, $v(1), v(2), \dots, v(k)$, and
- the sum (5.49) contains every combination where a single component has been chosen from each of the vectors $u(1), u(2), \dots, u(l), v(1), v(2), \dots, v(k)$ exactly once.

Consequently, we obtain the following spanning set of invariants $(\mathbb{R}^n)^{\otimes l+k} \to \mathbb{R}$ for each of $O(n)$, $Sp(n)$ and $SO(n)$.

Theorem 5.4 (Spanning Set of Invariants $(\mathbb{R}^n)^{\otimes l+k} \to \mathbb{R}$ for $O(n)$). *The functions*

$$(z(1), z(2))(z(3), z(4)) \dots (z(l+k-1), z(l+k)) \tag{5.61}$$

where $z(1), \dots, z(l+k)$ is a permutation of $u(1), u(2), \dots, u(l), v(1), v(2), \dots, v(k)$ form a spanning set of invariants $(\mathbb{R}^n)^{\otimes l+k} \to \mathbb{R}$ for $O(n)$.

Clearly, functions of the form (5.61) can only be formed when $l+k$ is even, hence there are no invariants $(\mathbb{R}^n)^{\otimes l+k} \to \mathbb{R}$ for $O(n)$ when $l+k$ is odd.

Theorem 5.5 (Spanning Set of Invariants $(\mathbb{R}^n)^{\otimes l+k} \to \mathbb{R}$ for $Sp(n)$, $n = 2m$). *The functions*

$$[z(1), z(2)][z(3), z(4)] \dots [z(l+k-1), z(l+k)] \tag{5.62}$$

where $z(1), \dots, z(l+k)$ is a permutation of $u(1), u(2), \dots, u(l), v(1), v(2), \dots, v(k)$ and

$$[z(i), z(i+1)] := \begin{cases} (z(i), z(i+1)) & \text{if } z(i) = u(j) \text{ and } z(i+1) = v(m), \text{ or} \\ & z(i) = v(m) \text{ and } z(i+1) = u(j), \text{ for} \\ & \text{some } j \in [l],\ m \in [k] \\ \langle z(i), z(i+1) \rangle & \text{otherwise} \end{cases} \tag{5.63}$$

form a spanning set of invariants $(\mathbb{R}^n)^{\otimes l+k} \to \mathbb{R}$ for $Sp(n)$, with $n = 2m$.

Similar to the $O(n)$ case, functions of the form (5.62) can only be formed when $l+k$ is even, hence there are no invariants $(\mathbb{R}^n)^{\otimes l+k} \to \mathbb{R}$ for $Sp(n)$ when $l+k$ is odd.

Theorem 5.6 (Spanning Set of Invariants $(\mathbb{R}^n)^{\otimes l+k} \to \mathbb{R}$ for $SO(n)$). *Functions of the form (5.61) together with functions of the form*

$$\det(z(1), \dots, z(n))(z(n+1), z(n+2)) \dots (z(l+k-1), z(l+k)) \tag{5.64}$$

where $z(1), \dots, z(l+k)$ is a permutation of $u(1), u(2), \dots, u(l), v(1), v(2), \dots, v(k)$ form a spanning set of invariants $(\mathbb{R}^n)^{\otimes l+k} \to \mathbb{R}$ for $SO(n)$.

Note that functions of the form (5.64) can only be formed when $n \leq l+k$, and either when n is odd and $l+k$ is odd, or when n is even and $l+k$ is even.

5.4 The Brauer and Brauer–Grood Vector Spaces

In order to obtain the characterisation of the weight matrices for these three groups, we need to introduce two types of diagrams that come from certain set partitions, whose definitions we adapt from Brauer (1937) for our purposes.

The first is defined as follows:

Definition 5.7. For all non-negative integers l and k, a (k, l)–Brauer partition β is a partitioning of the set $[l + k]$ into a disjoint union of pairs. We call each pair a block. Clearly, if $l + k$ is odd, then no such partitions exist. Therefore, in the following, we assume that $l + k$ is even. By convention, if $k = l = 0$, then there is just one $(0, 0)$–Brauer partition.

We can represent each (k, l)–Brauer partition β by a diagram d_β, called a $\boldsymbol{(k, l)}$**–Brauer diagram**, consisting of two rows of vertices and edges between vertices such that there are

1. l white vertices in the top row, labelled left to right by $1, \dots l$;
2. k white vertices in the bottom row, labelled left to right by $l + 1, \dots, l + k$; and
3. the edges between vertices correspond to the blocks of β.

In particular, this means that each vertex is incident to exactly one edge; hence there are precisely $\frac{l+k}{2}$ edges in total in d_β.

It is clear that the number of (k, l)–Brauer diagrams, if $l + k$ is even, is

$$(l + k - 1)!! := (l + k - 1)(l + k - 3) \cdots 5 \cdot 3 \cdot 1 \tag{5.65}$$

and is 0 otherwise.

Example 5.8. We see that

$$\beta := \{1, 5 \mid 2, 8 \mid 3, 4 \mid 6, 7 \mid 9, 10\} \tag{5.66}$$

is a valid $(6, 4)$–Brauer partition. Figure 5.1a) shows the $(6, 4)$–Brauer diagram d_β corresponding to β.

The second type of diagram is what we have decided to call an $(l + k)\backslash n$–diagram (pronounced "*l plus k without n*"). These diagrams were originally hinted at by Brauer (1937) in the case where $l = k$ and were looked at in greater detail by Grood (1999), but again only in the case where $l = k$. In their paper,

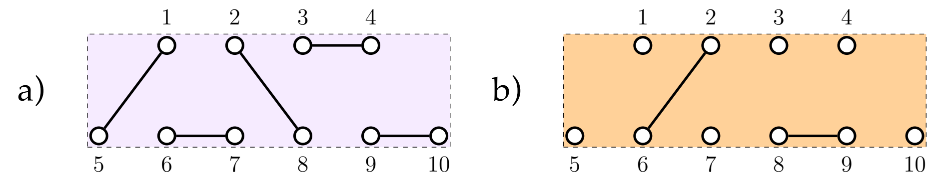


Figure 5.1: a) The diagram d_β corresponds to the $(6,4)$–Brauer partition β given in (5.66). b) The diagram d_α corresponds to the $(4+6)\backslash 6$–partition α given in (5.67).

they called these diagrams $k\backslash m$–diagrams, since they only considered the situation where $l = k$ and $n = 2m$. We will see that the definition below is equivalent to their definition in this case. Our naming convention for the diagrams makes more sense since they are a generalisation of those considered by Brauer and Grood.

Definition 5.9. For all non-negative integers l, k and positive integers n, an $(l+k)\backslash n$–partition α is a partitioning of the set $[l+k]$ with some n elements removed into a disjoint union of pairs. Once again, each pair is called a block.

An $\boldsymbol{(l+k)\backslash n}$**–diagram** d_α is the representation of an $(l+k)\backslash n$–partition α in diagram form, constructed in a similar way to the (k,l)–Brauer diagrams above, where there is still a top row consisting of l vertices and a bottom row consisting of k vertices, but now there are only $\frac{l+k-n}{2}$ edges between pairs of vertices. In this case, the n vertices removed from the set $[l+k]$ will not be incident to any edge, and an edge exists between any two vertices that are in the same pair. We call the n vertices whose labels have been removed from $[l+k]$ **free** vertices.

It is clear that if $n > l+k$, then no $(l+k)\backslash n$–diagrams exist. Also, no such diagrams exist if n is odd and $l+k$ is even, or if n is even and $l+k$ is odd.

Otherwise, that is, if $n \leq l+k$ and either n is even and $l+k$ is even, or n is odd and $l+k$ is odd, the number of $(l+k)\backslash n$–diagrams is $\binom{l+k}{n}(l+k-n-1)!!$, since there are $\binom{l+k}{n}$ ways to pick n free vertices, and $(l+k-n-1)!!$ ways to pair up the remaining $l+k-n$ vertices.

Example 5.10. We see that, if $k = 6$ and $l = 4$, then

$$\alpha := \{2, 6 \mid 8, 9\} \tag{5.67}$$

is a $(4+6)\backslash 6$–partition. Figure 5.1b) shows the $(4+6)\backslash 6$–diagram d_α corresponding to α.

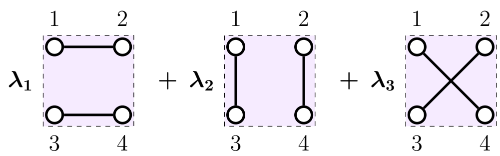
From the two types of diagrams defined above, we form the following two vector spaces.

Definition 5.11. We define the **Brauer vector space**, $B_k^l(n)$, which exists for all non-negative integers l, k and positive integers n, as follows. Let $B_0^0(n) \coloneqq \mathbb{R}$. Otherwise, define $B_k^l(n)$ to be the $\mathbb{R}$–linear span of the set of all (k, l)–Brauer diagrams.

Definition 5.12. We define the **Brauer–Grood vector space**, $D_k^l(n)$, which exists for all non-negative integers l, k and positive integers n, as follows. Let $D_0^0(n) \coloneqq \mathbb{R}$. Otherwise, define $D_k^l(n)$ to be the $\mathbb{R}$–linear span of the set of all (k, l)–Brauer diagrams together with the set of all $(l+k)\backslash n$–diagrams.

Remark 5.13. Clearly, $D_k^l(n) = B_k^l(n)$ if $n > l + k$, or if n is odd and $l + k$ is even, or if n is even and $l + k$ is odd.

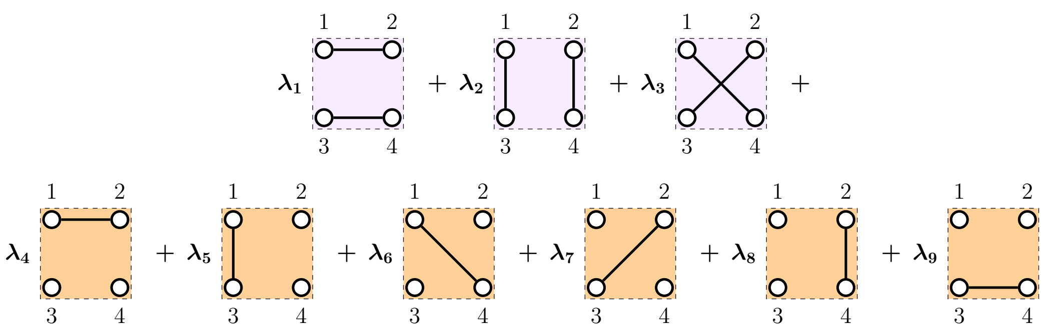
Example 5.14. For any positive integer n, all elements of $B_2^2(n)$ can be expressed as



(5.68)

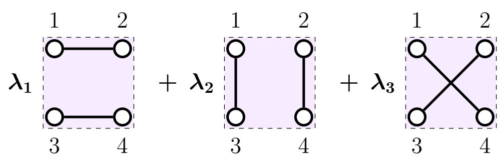
since the elements that appear in the Brauer vector spaces do not depend on n.

However, the elements in the Brauer–Grood vector spaces do depend on n. For example, the elements of $D_2^2(2)$ must be the form



(5.69)

since there are three $(2, 2)$–Brauer diagrams and six $(2+2)\backslash 2$–diagrams, whereas the elements of $D_2^2(3)$ are of the form



(5.70)

since no $(2+2)\backslash 3$–diagrams exist, hence $D_2^2(3) = B_2^2(3)$.

5.5 Main Results

We can now construct a bijective correspondence between each of the functions (5.61), (5.62), and (5.64), and either a (k,l)–Brauer diagram or an $(l+k)\backslash n$–diagram, as follows.

Indeed, consider the spanning set of invariants of the form (5.61). Then we can associate a (k,l)–Brauer diagram with each element of the set by labelling the top l vertices of the diagram from left-to-right by $u(1), u(2), \dots, u(l)$ and the bottom k vertices from left-to-right by $v(1), v(2), \dots, v(k)$, and drawing an edge between two vertices if and only if they appear in the same inner product in (5.61).

We do a similar thing for the spanning set of invariants of the form (5.62), associating a (k,l)–Brauer diagram with each element of the set, labelling the vertices in the same mamner, and drawing an edge between two vertices if and only if they appear in the same inner or skew product in (5.62).

Finally, consider the functions of the form (5.64). Then we can associate an $(l+k)\backslash n$–diagram with each element of the set by labelling the top l vertices of the diagram by $u(1), u(2), \dots, u(l)$ and the bottom k vertices by $v(1), v(2), \dots, v(k)$, leaving the vertices $z(1), \dots, z(n)$ free and drawing an edge between the other vertices if and only if they appear in the same inner product in (5.64).

As a result, we have constructed a bijective correspondence between a spanning set of invariants $(\mathbb{R}^n)^{\otimes l+k} \to \mathbb{R}$ for $O(n)$ with the set of (k,l)–Brauer diagrams whose span is the Brauer vector space $B_k^l(n)$, for $SO(n)$ with the set of (k,l)–Brauer diagrams together with the set of $(l+k)\backslash n$–diagrams whose span is the Brauer–Grood vector space $D_k^l(n)$, and for $Sp(n)$ with the set of (k,l)–Brauer diagrams whose span is the Brauer vector space $B_k^l(n)$.

Consequently, for each group $G(n)$, we combine the bijective correspondence (5.49) that exists between the spannning set of invariants $(\mathbb{R}^n)^{\otimes l+k} \to \mathbb{R}$ (given in Theorems 5.4, 5.5, and 5.6) and a spanning set of matrices for $\mathrm{Hom}_{G(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ in the standard/symplectic basis of $\mathbb{R}^n$ with the bijective correspondence that exists between the spannning set of invariants $(\mathbb{R}^n)^{\otimes l+k} \to \mathbb{R}$ and the set of diagrams that span either the Brauer vector space $B_k^l(n)$, for $O(n)$ and $Sp(n)$, or the Brauer–Grood vector space $D_k^l(n)$, for $SO(n)$, to obtain the following key results, which are stated in Theorems 5.15, 5.17, and 5.19 below.

Theorem 5.15 (Spanning set when $G(n) = O(n)$)**.** *For all non-negative integers l, k and positive integers n, there is a surjection of vector spaces*

$$\Phi_{k,n}^l : B_k^l(n) \to \mathrm{Hom}_{O(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l}) \tag{5.71}$$

which is defined as follows.

If $l + k$ is odd, then we map the empty set onto the empty set.

Otherwise, for each (k, l)–Brauer diagram d_β, associate the indices $i_1, i_2, \ldots, i_l$ with the vertices in the top row of d_β, and $j_1, j_2, \ldots, j_k$ with the vertices in the bottom row of d_β. Then, define

$$E_\beta := \sum_{I \in [n]^l, J \in [n]^k} \delta_{r_1,u_1} \delta_{r_2,u_2} \ldots \delta_{r_{\frac{l+k}{2}}, u_{\frac{l+k}{2}}} E_{I,J} \tag{5.72}$$

where $r_1, u_1, \ldots, r_{\frac{l+k}{2}}, u_{\frac{l+k}{2}}$ is any permutation of the indices $i_1, i_2, \ldots, i_l, j_1, j_2, \ldots, j_k$ such that the vertices corresponding to r_p, u_p are in the same block of β.

We see from the adapted version of Brauer's Invariant Argument that (5.72) defines a bijective correspondence between the set of all (k, l)–Brauer diagrams and a spanning set for $\mathrm{Hom}_{O(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$.

Consequently, when $l + k$ is even, the surjection of vector spaces given in (5.71) is defined by

$$d_\beta \mapsto E_\beta \tag{5.73}$$

for all (k, l)–Brauer diagrams d_β, and is extended linearly on the basis of such diagrams for $B_k^l(n)$.

Hence, the set

$$\{E_\beta \mid d_\beta \text{ is a } (k, l)\text{–Brauer diagram}\} \tag{5.74}$$

is a spanning set for $\mathrm{Hom}_{O(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ in the standard basis of $\mathbb{R}^n$, of size 0 when $l + k$ is odd, and of size $(l + k - 1)!!$ when $l + k$ is even.

Remark 5.16. Lehrer and Zhang (2012) showed that when $2n \geq l + k$, $\Phi_{k,n}^l$ is an isomorphism of vector spaces, and so the set (5.74) forms a basis of $\mathrm{Hom}_{O(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ in this case.

Theorem 5.17 (Spanning set when $G(n) = Sp(n), n = 2m$)**.** *For all non-negative integers l, k and positive even integers n, there is a surjection of vector spaces*

$$X_{k,n}^l : B_k^l(n) \to \mathrm{Hom}_{Sp(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l}) \tag{5.75}$$

which is defined as follows.

If $l + k$ is odd, then we map the empty set onto the empty set.

A General Procedure for Calculating the Weight Matrix of an $O(n)$-Equivariant Linear Layer Function from $(\mathbb{R}^n)^{\otimes k}$ to $(\mathbb{R}^n)^{\otimes l}$.

If $l+k$ is odd, return the zero matrix. Otherwise, perform the following steps:

1. Calculate all of the (k,l)–Brauer diagrams d_β.
2. Obtain the spanning set matrix E_β that is associated with each d_β by calculating each (I,J)-entry of E_β as follows: place the entries of I on the top row and the entries of J on the bottom row of d_β and see whether the entries within each block are the same, for each block in the diagram. If they are, then the (I,J)-entry of E_β is 1, otherwise 0.
3. Attach a weight $\lambda_\beta \in \mathbb{R}$ to each matrix E_β.
4. Finally, calculate $\sum \lambda_\beta E_\beta$ to give the overall weight matrix.

Otherwise, for each (k,l)–Brauer diagram d_β, associate the indices $i_1, i_2, \dots, i_l$ with the vertices in the top row of d_β, and $j_1, j_2, \dots, j_k$ with the vertices in the bottom row of d_β. Then, define

$$F_\beta \coloneqq \sum_{I,J} \gamma_{r_1,u_1}\gamma_{r_2,u_2}\dots\gamma_{r_{\frac{l+k}{2}},u_{\frac{l+k}{2}}} E_{I,J} \tag{5.76}$$

where the indices i_p, j_p range over $1, 1', \dots, m, m'$, where $r_1, u_1, \dots, r_{\frac{l+k}{2}}, u_{\frac{l+k}{2}}$ is any permutation of the indices $i_1, i_2, \dots, i_l, j_1, j_2, \dots, j_k$ such that the vertices corresponding to r_p, u_p are in the same block of β, and

$$\gamma_{r_p,u_p} \coloneqq \begin{cases} \delta_{r_p,u_p} & \text{if the vertices corresponding to } r_p, u_p \text{ are in different rows of } d_\beta \\ \epsilon_{r_p,u_p} & \text{if the vertices corresponding to } r_p, u_p \text{ are in the same row of } d_\beta \end{cases} \tag{5.77}$$

where ϵ_{r_p,u_p} was defined in (2.8).

We see from the adapted version of Brauer's Invariant Argument that (5.76) defines a bijective correspondence between the set of all (k,l)–Brauer diagrams and a spanning set for $\mathrm{Hom}_{Sp(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$.

Consequently, when $l+k$ is even, the surjection of vector spaces given in (5.75) is defined by

$$d_\beta \mapsto F_\beta \tag{5.78}$$

for all (k,l)–Brauer diagrams d_β, and is extended linearly on the basis of such diagrams for $B_k^l(n)$.

Hence, the set

$$\{F_\beta \mid d_\beta \text{ is a } (k,l)\text{–Brauer diagram}\} \tag{5.79}$$

A General Procedure for Calculating the Weight Matrix of an $Sp(n)$-Equivariant Linear Layer Function from $(\mathbb{R}^n)^{\otimes k}$ to $(\mathbb{R}^n)^{\otimes l}$.

If $l + k$ is odd, return the zero matrix. Otherwise, perform the following steps:

1. Calculate all of the (k, l)–Brauer diagrams d_β.
2. Obtain the spanning set matrix F_β that is associated with each d_β by calculating each (I, J)-entry of F_β as follows: place the entries of I on the top row and the entries of J on the bottom row of d_β, and consider each block in the diagram. We assign a value to each block according to the following rules.
   - If the block consists of two vertices that are in different rows, we see whether the entries that have been placed on the vertices are the same. If they are, then we assign the value 1 to that block, otherwise 0.
   - Otherwise, the block consists of two vertices that are in the same row. If x is the entry that has been placed on the left most vertex, and y is the other entry, then we calculate $\epsilon_{x,y}$, which was defined in (2.8).

   The (I, J)-entry of F_β is the value that is obtained by multiplying together all of the values that have been assigned to the blocks.
3. Attach a weight $\lambda_\beta \in \mathbb{R}$ to each matrix F_β.
4. Finally, calculate $\sum \lambda_\beta F_\beta$ to give the overall weight matrix.

is a spanning set for $\mathrm{Hom}_{Sp(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$*, for* $n = 2m$*, in the symplectic basis of* $\mathbb{R}^n$*, of size* 0 *when* $l + k$ *is odd, and of size* $(l + k - 1)!!$ *when* $l + k$ *is even.*

Remark 5.18. Lehrer and Zhang (2012) showed that when $n \geq l + k$, $X^l_{k,n}$ is an isomorphism of vector spaces, and so the set (5.79) forms a basis of $\mathrm{Hom}_{Sp(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$, for $n = 2m$, in this case.

Theorem 5.19 (Spanning set when $G(n) = SO(n)$)**.** *For all non-negative integers* l, k *and positive integers* n*, we construct a surjection of vector spaces*

$$\Psi^l_{k,n} : D^l_k(n) \to \mathrm{Hom}_{SO(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l}) \tag{5.80}$$

as follows.

If $n > l + k$*, or if* n *is odd and* $l + k$ *is even, or if* n *is even and* $l + k$ *is odd, then we saw that* $D^l_k(n) = B^l_k(n)$*. Hence, in these cases,* $\Psi^l_{k,n} = \Phi^l_{k,n}$*, and so* $\mathrm{Hom}_{SO(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l}) = \mathrm{Hom}_{O(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$*.*

Otherwise, that is, if $n \leq l + k$*, and either* n *is even and* $l + k$ *is even, or* n *is odd and* $l + k$ *is odd, there exist* $(l + k)\backslash n$*–diagrams.*

A General Procedure for Calculating the Weight Matrix of an $SO(n)$-Equivariant Linear Layer Function from $(\mathbb{R}^n)^{\otimes k}$ to $(\mathbb{R}^n)^{\otimes l}$.

Perform the following steps:

1. Calculate all of the (k, l)–Brauer diagrams d_β and all of the $(l + k)\backslash n$–diagrams d_α, if they exist.

2. Obtain the spanning set matrix E_β that is associated with each d_β using the procedure that is given for $O(n)$.

3. Obtain the spanning set matrix H_α that is associated with each d_α by calculating each (I, J)-entry of H_α as follows: place the entries of I on the top row and the entries of J on the bottom row of d_α.

   - We assign a value to each block that contains two vertices: the value is 1 if the entries that have been placed on the vertices of the block are the same, otherwise 0.
   - We then assign the value $\det(e_{t_1} \otimes e_{t_2} \otimes \cdots \otimes e_{t_s} \otimes e_{b_1} \otimes \cdots \otimes e_{b_{n-s}})$ to the set of free vertices, where the values $t_1, \ldots, t_s$ label the free vertices in the top row and $b_1, \ldots, b_{n-s}$ label the free vertices in the bottom row of the diagram.

   The (I, J)-entry of H_α is the value that is obtained from multiplying together the values that are assigned to the blocks together with the value that is assigned to the set of free vertices.

4. Attach a weight $\lambda_\beta \in \mathbb{R}$ to each matrix E_β and a weight $\lambda_\alpha \in \mathbb{R}$ to each matrix H_α.

5. Finally, calculate $\sum \lambda_\beta E_\beta + \sum \lambda_\alpha H_\alpha$ to give the overall weight matrix.

For each such diagram d_α, again associate the indices $i_1, i_2, \ldots, i_l$ with the vertices in the top row of d_α, and $j_1, j_2, \ldots, j_k$ with the vertices in the bottom row of d_α. Suppose that there are s free vertices in the top row. Then there are $n - s$ free vertices in the bottom row. Relabel the s free indices in the top row from left-to-right by $t_1, \ldots, t_s$, and the $n - s$ free indices in the bottom row from left-to-right by $b_1, \ldots, b_{n-s}$.

As a result, define H_α to be

$$\sum_{I \in [n]^l, J \in [n]^k} \det(e_{T,B}) \delta(R, U) E_{I,J} \tag{5.81}$$

where $\det$ *is the determinant map given in Definition 4.3,*

$$e_{T,B} \coloneqq e_{t_1} \otimes e_{t_2} \otimes \cdots \otimes e_{t_s} \otimes e_{b_1} \otimes \cdots \otimes e_{b_{n-s}} \tag{5.82}$$

and

$$\delta(R, U) \coloneqq \delta_{r_1, u_1} \delta_{r_2, u_2} \ldots \delta_{r_{\frac{l+k-n}{2}}, u_{\frac{l+k-n}{2}}} \tag{5.83}$$

Here, $r_1, u_1, \dots, r_{\frac{l+k-n}{2}}, u_{\frac{l+k-n}{2}}$ is any permutation of the indices

$$\{i_1, \dots, i_l, j_1, \dots, j_k\} \backslash \{t_1, \dots, t_s, b_1, \dots, b_{n-s}\} \tag{5.84}$$

such that the vertices corresponding to r_p, u_p are in the same block of α.

We see from the adapted version of Brauer's Invariant Argument that (5.72) and (5.81) defines a bijective correspondence between the set of all (k,l)–Brauer diagrams together with the set of all $(l+k)\backslash n$–diagrams, and a spanning set for $\mathrm{Hom}_{SO(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$.

Consequently, when $n \leq l+k$ and n is even and $l+k$ is even, the surjection of vector spaces (5.80) is given by

$$d_\beta \mapsto E_\beta \tag{5.85}$$

if d_β is a (k,l)–Brauer diagram, where E_β was defined in Theorem 5.15, and by

$$d_\alpha \mapsto H_\alpha \tag{5.86}$$

if d_α is an $(l+k)\backslash n$–diagram, and is extended linearly on the basis of such diagrams for $D_k^l(n)$.

When $n \leq l+k$ and n is odd and $l+k$ is odd, the surjection of vector spaces (5.80) is given solely by (5.86), since no (k,l)–Brauer diagrams exist in this case.

Hence, in all cases, the set

$$\{E_\beta\}_\beta \cup \{H_\alpha\}_\alpha \tag{5.87}$$

is a spanning set for $\mathrm{Hom}_{SO(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ in the standard basis of $\mathbb{R}^n$, where d_β is a (k,l)–Brauer diagram and d_α is an $(l+k)\backslash n$–diagram.

We obtain the weight matrix for a $G(n)$-equivariant linear layer function from $(\mathbb{R}^n)^{\otimes k}$ to $(\mathbb{R}^n)^{\otimes l}$ in the standard or symplectic basis of $\mathbb{R}^n$ for each of $G(n) = O(n), Sp(n)$, and $SO(n)$ from the spanning sets that are given in Theorems 5.15, 5.17, and 5.19. They are presented in the following three theorems.

Theorem 5.20 (Orthogonal Group Equivariant Weight Matrices). *For all non-negative integers l, k and positive integers n, the weight matrix W that appears in an $O(n)$-equivariant linear layer function from $(\mathbb{R}^n)^{\otimes k}$ to $(\mathbb{R}^n)^{\otimes l}$ must be of the form*

$$W = \sum_{d_\beta \in B_k^l(n)} \lambda_\beta E_\beta \tag{5.88}$$

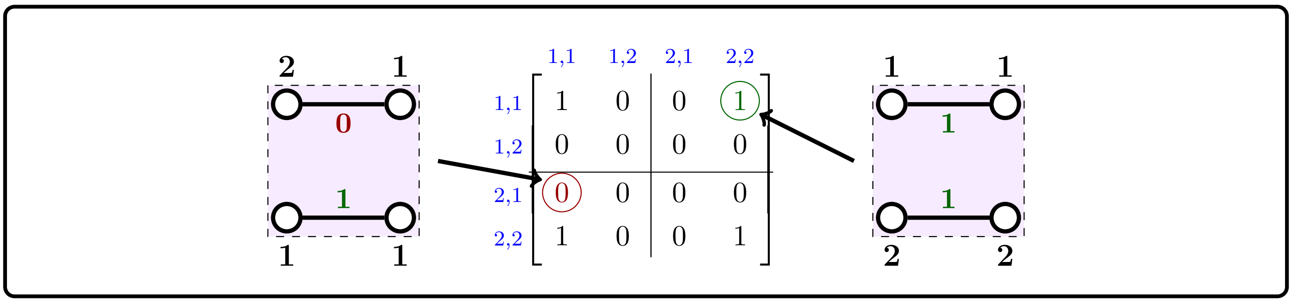


Figure 5.2: We show one of the three basis matrices whose weighted linear combination gives all of the possible weight matrices that can appear in an $O(2)$-equivariant neural network from $(\mathbb{R}^2)^{\otimes 2}$ to $(\mathbb{R}^2)^{\otimes 2}$. We obtain these matrices from the $(2, 2)$–Brauer diagrams in $B_2^2(2)$. For each diagram, to calculate the (I, J)-entry of its associated basis matrix, we place the I-tuple on the top row of the diagram and the J-tuple on the bottom row of the diagram, and see whether the entries within each block are the same for each block in the diagram. If they are, then we put a 1 in the (I, J)-entry of the matrix, otherwise 0.

Theorem 5.21 (Symplectic Group Equivariant Weight Matrices)**.** *For all non-negative integers l, k and positive even integers n, the weight matrix W that appears in an $Sp(n)$-equivariant linear layer function from $(\mathbb{R}^n)^{\otimes k}$ to $(\mathbb{R}^n)^{\otimes l}$ must be of the form*

$$W = \sum_{d_\beta \in B_k^l(n)} \lambda_\beta F_\beta \tag{5.89}$$

Theorem 5.22 (Special Orthogonal Group Equivariant Weight Matrices)**.** *For all non-negative integers l, k and positive integers n, the weight matrix W that appears in an $SO(n)$-equivariant linear layer function from $(\mathbb{R}^n)^{\otimes k}$ to $(\mathbb{R}^n)^{\otimes l}$ must be of the form*

$$W = \sum_{d_\beta \in D_k^l(n)} \lambda_\beta E_\beta + \sum_{d_\alpha \in D_k^l(n)} \lambda_\alpha H_\alpha \tag{5.90}$$

5.6 Examples

In the following, in order to save space, we use some temporary notation to denote a sum of a number of weight parameters. We represent a sum of weight parameters, where the sum is over some index set A, by a single element that is indexed by the set of indices itself, that is

$$\lambda_A := \sum_{i \in A} \lambda_i \tag{5.91}$$

For example, $\lambda_{8,11,12}$ represents the sum $\lambda_8 + \lambda_{11} + \lambda_{12}$, and $-\lambda_{4,5,7}$ represents the sum $-\lambda_4 - \lambda_5 - \lambda_7$.

Example 5.23. Suppose that we would like to find the weight matrix for an $O(2)$-equivariant linear

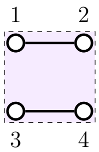
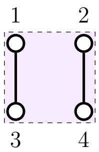
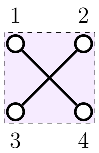
Basis Diagram d_β	Matrix Entries	Basis Matrix E_β
	$(\delta_{i_1,i_2}\delta_{j_1,j_2})$	$\begin{array}{c c} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc\|cc} 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{array}\right] \end{array}$
	$(\delta_{i_1,j_1}\delta_{i_2,j_2})$	$\begin{array}{c c} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc\|cc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right] \end{array}$
	$(\delta_{i_1,j_2}\delta_{i_2,j_1})$	$\begin{array}{c c} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc\|cc} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right] \end{array}$

Figure 5.3: The images under $\Phi_{2,2}^2$ of the basis diagrams of $B_2^2(2)$ make up a basis of $\mathrm{End}_{O(2)}((\mathbb{R}^2)^{\otimes 2})$.

layer function from $(\mathbb{R}^2)^{\otimes 2}$ to $(\mathbb{R}^2)^{\otimes 2}$.

Since $l = k = n = 2$, we apply Theorem 5.15 by considering the surjective map

$$\Phi_{2,2}^2 : B_2^2(2) \to \mathrm{End}_{O(2)}((\mathbb{R}^2)^{\otimes 2}) \tag{5.92}$$

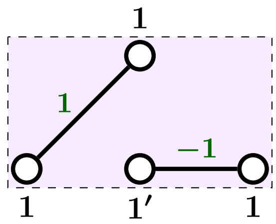
noting that $l + k$ is even. Also, by Remark 5.16, as $2n \geq l + k$, this map is an isomorphism of vector spaces. Hence, by taking the image of all of the $(2, 2)$–Brauer diagrams that appear in $B_2^2(2)$, we actually obtain a basis of $\mathrm{End}_{O(2)}((\mathbb{R}^2)^{\otimes 2})$.

$B_2^2(2)$ has three basis diagrams, which are given in the left hand column of Figure 5.3. To find the basis matrix E_β that corresponds to its Brauer diagram d_β, we perform the procedure that is given in the purple box. Fix a Brauer diagram d_β. For each (I, J)-entry of the matrix E_β, we place the entries of I on the top row and the entries of J on the bottom row of the diagram and see whether the entries within each block are the same for each block in the diagram. If they are, then the (I, J)-entry of the matrix is 1, otherwise 0.

For example, take the first diagram in Figure 5.3 which corresponds to the set partition

$$\beta = \{1, 2 \mid 3, 4\} \tag{5.93}$$

Consider the (I, J)-entry where $I = (1, 1)$ and $J = (2, 2)$. We place $(1, 1)$ on the top row and $(2, 2)$ on



$$
\begin{array}{c|cc|cc|cc|cc|}
 & 1,1,1 & 1,1,1' & 1,1',1 & 1,1',1' & 1',1,1 & 1',1,1' & 1',1',1 & 1',1',1' \\
\hline
1 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\
1' & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0
\end{array}
$$

Figure 5.4: We show one of the three spanning set matrices whose weighted linear combination gives all of the possible weight matrices that can appear in an $Sp(2)$-equivariant neural network from $(\mathbb{R}^2)^{\otimes 3}$ to $\mathbb{R}^2$. We obtain these matrices from the $(3,1)$–Brauer diagrams in $B_3^1(2)$. For each diagram, to calculate the (I,J)-entry of its associated spanning set matrix, we place the I-tuple on the top row of the diagram and the J-tuple on the bottom row of the diagram, and label the blocks according to the procedure that is given in the light blue box. The (I,J)-entry of the matrix is the product of the values that are given to the individual blocks.

the bottom row of the diagram d_β, and see that the entries within each block are the same, for each block in the diagram. Hence, the $(1,1 \mid 2,2)$-entry for the matrix E_β corresponding to this diagram is 1. However, if we consider the (I,J)-entry where $I = (2,1)$ and $J = (1,1)$, then this entry is 0 since the labels for the block on the bottom row are not the same. Figure 5.2 shows how the (I,J)-entries are obtained from the Brauer diagram d_β for these examples.

Figure 5.3 shows in full the basis of $\mathrm{End}_{O(2)}((\mathbb{R}^2)^{\otimes 2})$ that is obtained from the basis of $B_2^2(2)$.

Hence, the weight matrix for an $O(2)$-equivariant linear layer function from $(\mathbb{R}^2)^{\otimes 2}$ to $(\mathbb{R}^2)^{\otimes 2}$, in the standard basis of $\mathbb{R}^2$, is of the form

$$
\begin{array}{c|cc|cc|}
 & 1,1 & 1,2 & 2,1 & 2,2 \\
\hline
1,1 & \lambda_{1,2,3} & 0 & 0 & \lambda_1 \\
1,2 & 0 & \lambda_2 & \lambda_3 & 0 \\
\hline
2,1 & 0 & \lambda_3 & \lambda_2 & 0 \\
2,2 & \lambda_1 & 0 & 0 & \lambda_{1,2,3}
\end{array}
\tag{5.94}
$$

for weights $\lambda_1, \lambda_2, \lambda_3 \in \mathbb{R}$.

Example 5.24. Suppose that we would like to find the weight matrix for an $Sp(2)$-equivariant linear layer function from $(\mathbb{R}^2)^{\otimes 3}$ to $\mathbb{R}^2$.

We saw that calculating the weight matrix is similar to the $O(n)$ case, except we replace δ by ϵ if there is an edge between two vertices that are in the same row.

We apply Theorem 5.17 by considering the surjective map

$$X_{3,2}^1 : B_3^1(2) \rightarrow \mathrm{Hom}_{Sp(2)}((\mathbb{R}^2)^{\otimes 3}, \mathbb{R}^2) \tag{5.95}$$

noting that $l + k$ is even.

$B_3^1(2)$ has three basis diagrams, which are given in the left hand column of Figure 5.5. To find the spanning set matrices F_β corresponding to each diagram d_β, we perform the procedure that is given in the light blue box. For each (I, J)-entry of the matrix, we place the entries of I on the top row and the entries of J on the bottom row of the diagram, and consider each block in the diagram. We assign a value to each block as follows. If the block consists of two vertices that are in different rows, we see whether the entries that have been placed on the vertices are the same. If they are, then we assign the value 1 to that block, otherwise 0. Otherwise, the block consists of two vertices that are in the same row. If x is the entry that has been placed on the left most vertex, and y is the other entry, then we calculate $\epsilon_{x,y}$, which was defined in (2.8). Consequently, the (I, J)-entry of the matrix is the value that is obtained by multiplying together all of the values that have been assigned to the blocks.

For example, take the first diagram in Figure 5.5 corresponding to the set partition

$$\beta = \{1, 2 \mid 3, 4\} \tag{5.96}$$

Consider the (I, J)-entry of F_β where $I = (1)$ and $J = (1, 1', 1)$. We place (1) on the top row and $(1, 1', 1)$ on the bottom row of the diagram. We see that the entries that have been placed on the block that connects vertices 1 and 2, which are in different rows, are the same. Consequently, we assign the value 1 to this block. For the other block, $1'$ has been placed on vertex 3, and 1 has been placed on vertex 4. Hence, we assign the value of $\epsilon_{1',1}$ to this block, which, by (2.8), is -1. Therefore, the $(1 \mid 1, 1', 1)$-entry of F_β is $1 \times -1 = -1$. Figure 5.4 shows how the $(1 \mid 1, 1', 1)$-entry is obtained from the Brauer diagram d_β itself.

Figure 5.5 shows in full the spanning set for $\mathrm{Hom}_{Sp(2)}((\mathbb{R}^2)^{\otimes 3}, \mathbb{R}^2)$ that is obtained from the basis of $B_3^1(2)$.

Hence, the weight matrix for an $Sp(2)$-equivariant linear layer function from $(\mathbb{R}^2)^{\otimes 3}$ to $\mathbb{R}^2$, in the

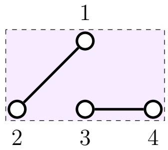
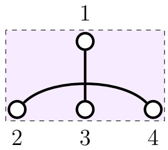
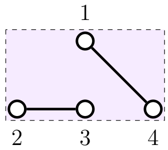
<table>
<tr><th>Basis Diagram d_β</th><th>Matrix Entries</th><th>Spanning Set Matrix F_β</th></tr>
<tr><td></td><td>$(\delta_{i_1,j_1}\epsilon_{j_2,j_3})$</td><td>$\begin{array}{cc} & \begin{array}{cccccccc} 1,1,1 & 1,1,1' & 1,1',1 & 1,1',1' & 1',1,1 & 1',1,1' & 1',1',1 & 1',1',1' \end{array} \\ \begin{array}{c} 1 \\ 1' \end{array} & \left[\begin{array}{cc|cc|cc|cc} 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 \end{array}\right] \end{array}$</td></tr>
<tr><td></td><td>$(\delta_{i_1,j_2}\epsilon_{j_1,j_3})$</td><td>$\begin{array}{cc} & \begin{array}{cccccccc} 1,1,1 & 1,1,1' & 1,1',1 & 1,1',1' & 1',1,1 & 1',1,1' & 1',1',1 & 1',1',1' \end{array} \\ \begin{array}{c} 1 \\ 1' \end{array} & \left[\begin{array}{cc|cc|cc|cc} 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 \end{array}\right] \end{array}$</td></tr>
<tr><td></td><td>$(\delta_{i_1,j_3}\epsilon_{j_1,j_2})$</td><td>$\begin{array}{cc} & \begin{array}{cccccccc} 1,1,1 & 1,1,1' & 1,1',1 & 1,1',1' & 1',1,1 & 1',1,1' & 1',1',1 & 1',1',1' \end{array} \\ \begin{array}{c} 1 \\ 1' \end{array} & \left[\begin{array}{cc|cc|cc|cc} 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 & 0 & 0 \end{array}\right] \end{array}$</td></tr>
</table>

Figure 5.5: The images under $X_{3,2}^1$ of the basis diagrams of $B_3^1(2)$ make up a spanning set for $\mathrm{Hom}_{Sp(2)}((\mathbb{R}^2)^{\otimes 3}, \mathbb{R}^2)$.

symplectic basis of $\mathbb{R}^2$, is of the form

$$\begin{array}{cc} & \begin{array}{cccccccc} 1,1,1 & 1,1,1' & 1,1',1 & 1,1',1' & 1',1,1 & 1',1,1' & 1',1',1 & 1',1',1' \end{array} \\ \begin{array}{c} 1 \\ 1' \end{array} & \left[\begin{array}{cc|cc|cc|cc} 0 & \lambda_1+\lambda_2 & -\lambda_1+\lambda_3 & 0 & -\lambda_2-\lambda_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & \lambda_2+\lambda_3 & 0 & \lambda_1-\lambda_3 & -\lambda_1-\lambda_2 & 0 \end{array}\right] \end{array} \tag{5.97}$$

for weights $\lambda_1, \lambda_2, \lambda_3 \in \mathbb{R}$.

Example 5.25. Finally, suppose that we would like to find the weight matrix for an $SO(2)$-equivariant linear layer function from $(\mathbb{R}^2)^{\otimes 3}$ to $\mathbb{R}^2$. Again we apply Theorem 5.19.

As $n \leq l+k$, and n is even and $l+k$ is even, we see that three $(3,1)$–Brauer diagrams and six $(1+3)\backslash 2$–diagrams make up a basis of $D_3^1(2)$. Their images under

$$\Psi_{3,2}^1 : D_3^1(2) \to \mathrm{Hom}_{SO(2)}((\mathbb{R}^2)^{\otimes 3}, \mathbb{R}^2) \tag{5.98}$$

form a spanning set of $\mathrm{Hom}_{SO(2)}((\mathbb{R}^2)^{\otimes 3}, \mathbb{R}^2)$.

Calculating the images of the three $(3,1)$–Brauer diagrams is the same as for the $O(n)$ case.

To find the spanning set matrices H_α corresponding to each $(1+3)\backslash 2$–diagram d_α, we perform the procedure that is given in the orange box. For each (I,J)-entry of the matrix, we place the entries

$$
\begin{array}{c c}
 & \begin{array}{cccccccc} 1,1,1 & 1,1,2 & 1,2,1 & 1,2,2 & 2,1,1 & 2,1,2 & 2,2,1 & 2,2,2 \end{array} \\
\begin{array}{c} 1 \\ 2 \end{array} & \left[\begin{array}{cc|cc|cc|cc} 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 \end{array}\right]
\end{array}
$$

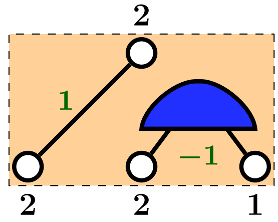


Figure 5.6: We show one of the nine spanning set matrices whose weighted linear combination gives all of the possible weight matrices that can appear in an $SO(2)$-equivariant neural network from $(\mathbb{R}^2)^{\otimes 3}$ to $\mathbb{R}^2$. We obtain these matrices from the three $(3,1)$–Brauer diagrams and six $(1+3)\backslash 2$–diagrams in $D_3^1(2)$. For each $(1+3)\backslash 2$–diagram, to calculate the (I,J)-entry of its associated spanning set matrix, we place the I-tuple on the top row of the diagram and the J-tuple on the bottom row of the diagram, and label the blocks according to the procedure that is given in the orange box. The (I,J)-entry of the matrix is the product of the values that are given to the individual blocks.

of I on the top row and the entries of J on the bottom row of the diagram. To the block containing two vertices, we assign the value 1 to that block if the entries that have been placed on the vertices are the same, otherwise 0. We then assign a value to the set of free vertices as follows. We attach a two-legged jellyfish to the set of free vertices, attaching the legs to the free vertices in the top row first, from left-to-right, before attaching the legs to the free vertices in the bottom row, also from left-to-right. We calculate the value of the determinant map that corresponds to this jellyfish. In this case, the value will be $\det(e_x \otimes e_y)$, where x is the entry that is associated with the first free vertex and y is the entry that is associated with the other free vertex. Consequently, the (I,J)-entry of the matrix is the value that is obtained from multiplying the value that is assigned to the block with the value that is assigned to the set of free vertices.

For example, take the first diagram in Figure 5.7 corresponding to the set partition

$$\alpha = \{1, 2 \mid 3 \mid 4\} \tag{5.99}$$

Consider the (I,J)-entry of H_α where $I = (2)$ and $J = (2,2,1)$. We place (2) on the top row and $(2,2,1)$ on the bottom row of the diagram. We see that the entries that have been placed on the block that connects vertices 1 and 2 are the same. Consequently, we assign 1 to this block. For the set of free vertices, we attach a two-legged jellyfish to these vertices and see that the value of the determinant map associated with the jellyfish is $\det(e_2 \otimes e_1)$, which is -1. Therefore, the $(2 \mid 2,2,1)$-entry of H_α is $1 \times -1 = -1$. Figure 5.6 shows how the $(2 \mid 2,2,1)$-entry is obtained from the $(1+3)\backslash 2$-diagram d_α itself.

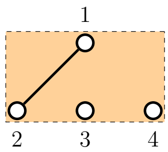
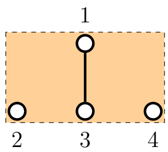
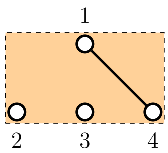
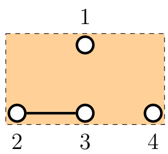
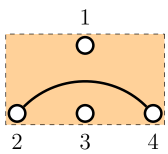
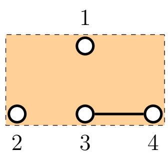
Basis Diagram d_α	Matrix Entries	Spanning Set Matrix H_α
	$(\det(e_{j_2} \otimes e_{j_3})\delta_{i_1,j_1})$	$\begin{array}{cc} & \begin{array}{cccccccc} \text{1,1,1} & \text{1,1,2} & \text{1,2,1} & \text{1,2,2} & \text{2,1,1} & \text{2,1,2} & \text{2,2,1} & \text{2,2,2} \end{array} \\ \begin{array}{c} 1 \\ 2 \end{array} & \left[\begin{array}{cc\|cc\|cc\|cc} 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 \end{array}\right] \end{array}$
	$(\det(e_{j_1} \otimes e_{j_3})\delta_{i_1,j_2})$	$\begin{array}{cc} & \begin{array}{cccccccc} \text{1,1,1} & \text{1,1,2} & \text{1,2,1} & \text{1,2,2} & \text{2,1,1} & \text{2,1,2} & \text{2,2,1} & \text{2,2,2} \end{array} \\ \begin{array}{c} 1 \\ 2 \end{array} & \left[\begin{array}{cc\|cc\|cc\|cc} 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 \end{array}\right] \end{array}$
	$(\det(e_{j_1} \otimes e_{j_2})\delta_{i_1,j_3})$	$\begin{array}{cc} & \begin{array}{cccccccc} \text{1,1,1} & \text{1,1,2} & \text{1,2,1} & \text{1,2,2} & \text{2,1,1} & \text{2,1,2} & \text{2,2,1} & \text{2,2,2} \end{array} \\ \begin{array}{c} 1 \\ 2 \end{array} & \left[\begin{array}{cc\|cc\|cc\|cc} 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 & 0 & 0 \end{array}\right] \end{array}$
	$(\det(e_{i_1} \otimes e_{j_3})\delta_{j_1,j_2})$	$\begin{array}{cc} & \begin{array}{cccccccc} \text{1,1,1} & \text{1,1,2} & \text{1,2,1} & \text{1,2,2} & \text{2,1,1} & \text{2,1,2} & \text{2,2,1} & \text{2,2,2} \end{array} \\ \begin{array}{c} 1 \\ 2 \end{array} & \left[\begin{array}{cc\|cc\|cc\|cc} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \end{array}\right] \end{array}$
	$(\det(e_{i_1} \otimes e_{j_2})\delta_{j_1,j_3})$	$\begin{array}{cc} & \begin{array}{cccccccc} \text{1,1,1} & \text{1,1,2} & \text{1,2,1} & \text{1,2,2} & \text{2,1,1} & \text{2,1,2} & \text{2,2,1} & \text{2,2,2} \end{array} \\ \begin{array}{c} 1 \\ 2 \end{array} & \left[\begin{array}{cc\|cc\|cc\|cc} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \end{array}\right] \end{array}$
	$(\det(e_{i_1} \otimes e_{j_1})\delta_{j_2,j_3})$	$\begin{array}{cc} & \begin{array}{cccccccc} \text{1,1,1} & \text{1,1,2} & \text{1,2,1} & \text{1,2,2} & \text{2,1,1} & \text{2,1,2} & \text{2,2,1} & \text{2,2,2} \end{array} \\ \begin{array}{c} 1 \\ 2 \end{array} & \left[\begin{array}{cc\|cc\|cc\|cc} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \end{array}\right] \end{array}$

Figure 5.7: The images under $\Psi^1_{3,2}$ of the six $(1+3)\backslash 2$–diagrams in $D^1_3(2)$, together with the images under $\Psi^1_{3,2}$ of the three $(3,1)$–Brauer diagrams in $D^1_3(2)$ (not shown), make up a spanning set for $\mathrm{Hom}_{SO(2)}((\mathbb{R}^2)^{\otimes 3}, \mathbb{R}^2)$.

Figure 5.7 shows in full the matrices that are obtained from the six $(1+3)\backslash 2$–diagrams.

Hence, the weight matrix for an $SO(2)$-equivariant linear layer function from $(\mathbb{R}^2)^{\otimes 3}$ to $\mathbb{R}^2$, in the standard basis of $\mathbb{R}^2$, is of the form

$$\begin{array}{cc} & \begin{array}{cccccccc} \text{1,1,1} & \text{1,1,2} & \text{1,2,1} & \text{1,2,2} & \text{2,1,1} & \text{2,1,2} & \text{2,2,1} & \text{2,2,2} \end{array} \\ \begin{array}{c} 1 \\ 2 \end{array} & \left[\begin{array}{cc|cc|cc|cc} \lambda_{1,2,3} & \lambda_{4,5,7} & -\lambda_4 + \lambda_{6,8} & \lambda_1 & -\lambda_{5,6} + \lambda_9 & \lambda_2 & \lambda_3 & \lambda_{7,8,9} \\ -\lambda_{7,8,9} & \lambda_3 & \lambda_2 & \lambda_{5,6} - \lambda_9 & \lambda_1 & \lambda_4 - \lambda_{6,8} & -\lambda_{4,5,7} & \lambda_{1,2,3} \end{array}\right] \end{array} \tag{5.100}$$

for weights $\lambda_1, \ldots, \lambda_9 \in \mathbb{R}$.

Chapter 6

Partition Categories

In the preceding chapters, we found linear maps between vector spaces containing certain types of set partition diagrams and vector spaces of equivariant linear maps between tensor power representations for some subgroups $G(n)$ of $GL(n)$. We used these linear maps to obtain the weight matrices that appear in neural networks that are equivariant to these groups $G(n)$, where the layers are tensor power spaces of $\mathbb{R}^n$. However, for a given group $G(n)$, each of the weight matrices in the neural network is in some sense similar, since they are determined in part by the layers of the network, which are the tensor power representations that differ only in their order. Likewise, the vector spaces containing certain types of set partition diagrams are also, in some sense, similar, in that, for a fixed value of n, the vector spaces mostly differ by the values chosen for l and k, which represent the number of vertices in each row of a set partition diagram. We would like to formalise this intuition of similarity across different values of l and k for both the vector spaces of equivariant linear maps and the vector spaces containing certain types of set partition diagrams. For this, the concepts that appear in category theory are ideal, in that they are useful for abstracting away the specifics of structures in order to study their general properties. By replacing set-theoretic notions with category-theoretic ones, we create a deeper structure for understanding the group equivariant neural networks themselves. From the vector spaces that have similar properties, we create so-called monoidal categories, which are categories that have an additional operation for composing objects and morphisms, known as the tensor product, and then use them to create so-called monoidal functors, which are functors that preserve the tensor product between monoidal categories. We are particularly interested in the tensor product because the layers of the neural networks are tensor power spaces of $\mathbb{R}^n$, and they differ only in the number of tensor products that are involved.

We note that we are not solely rewriting existing results in a different language, but that, as an outcome of this process, we obtain new insights into these neural networks. In particular, we show that set partition diagrams that appear in monoidal categories have a string-like quality to them. By pulling on the strings or dragging their ends to different locations, we can use the monoidal functors to obtain new results for the weight matrices that appear in these group equivariant neural networks.

Before we define monoidal categories and monoidal functors, we first introduce a new basis of matrices for the linear maps that appear in permutation equivariant neural networks that was originally found by Godfrey et al. (2023). We show that this basis comes with additional benefits relating to the tensor product compared with the one that we obtained from the orbit basis of the partition vector space $P_k^l(n)$ in Chapter 3.

The content of this chapter is modelled on the paper:

Edward Pearce-Crump and William J. Knottenbelt. (2024a) A Diagrammatic Approach to Improve Computational Efficiency in Group Equivariant Neural Networks. `arXiv:2412.10837`.

6.1 Diagram Basis for Permutation Equivariant Linear Layers

After we had established the relationship between the weight matrices that appear in permutation equivariant neural networks and the partition vector spaces in the preprint of our paper Pearce-Crump (2024), Godfrey et al. (2023) constructed another basis for $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$, which they called the **diagram basis**, using an algebra formed from the direct sum of all vector spaces of permutation equivariant linear layers. We show instead that we can use the diagram basis of the partition vector space $P_k^l(n)$ that we introduced in Chapter 3 to obtain the diagram basis of $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$.

Recall from Section 3.2.2 that the equation

$$d_\pi = \sum_{\pi \preceq \theta} x_\theta \tag{6.1}$$

given previously in (3.9) defines a unitriangular change of basis matrix between the diagram basis $\{d_\pi\}$ and the orbit basis $\{x_\theta\}$ of $P_k^l(n)$. In particular, this equation also defines a change of basis matrix on the subspace of $P_k^l(n)$ spanned by all diagram basis elements d_π that have at most n blocks. Indeed, for each diagram d_π in this subspace, since it has at most n blocks, the number of blocks in each x_θ that appears in this equation can be at most n, by definition of the partial order $\preceq$ on set partitions of $[l+k]$.

Recall also from Definition 3.17 that for all non-negative integers l, k and positive integers n, the surjective map

$$\Gamma^l_{k,n} : P^l_k(n) \to \mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l}) \tag{6.2}$$

was defined on the orbit basis of $P^l_k(n)$ as follows, and extended linearly:

$$\Gamma^l_{k,n}(x_\pi) := \begin{cases} X_\pi & \text{if } \pi \text{ has } n \text{ or fewer blocks} \\ 0 & \text{if } \pi \text{ has more than } n \text{ blocks} \end{cases} \tag{6.3}$$

We can combine these results to define another map from $P^l_k(n)$ to $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$, this time on the diagram basis of $P^l_k(n)$. We will use the notation $\Theta^l_{k,n}$ to refer to this map in what follows.

Let π be a set partition of $[l+k]$, and let d_π be its representation in the diagram basis of $P^l_k(n)$. We define $\Theta^l_{k,n}(d_\pi)$ in the following way. First change the basis in $P^l_k(n)$ from the diagram basis to the orbit basis using (6.1), and then apply $\Gamma^l_{k,n}$ to each of the orbit basis diagrams that appear in (6.1) for d_π. We get that

$$\Theta^l_{k,n}(d_\pi) := \Gamma^l_{k,n}\Big(\sum_{\pi \preceq \theta} x_\theta\Big) = \sum_{\pi \preceq \theta} \Gamma^l_{k,n}(x_\theta) = \sum_{\pi \preceq \theta,\ b(\theta) \leq n} X_\theta \tag{6.4}$$

where $b(\theta)$ is the number of blocks in θ. Now define

$$D_\pi := \sum_{\pi \preceq \theta,\ b(\theta) \leq n} X_\theta \tag{6.5}$$

Clearly, D_π is an element of $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$. Hence we have defined $\Theta^l_{k,n}(d_\pi) := D_\pi$. $\Theta^l_{k,n}$ becomes a map on $P^l_k(n)$ by extending the definition by linearity to the entire diagram basis of $P^l_k(n)$. We claim the following result.

Proposition 6.1. *For all non-negative integers l, k and positive integers n, the set*

$$\{D_\pi \mid d_\pi \text{ has at most } n \text{ blocks}\} \tag{6.6}$$

forms a basis of $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$. *In particular, the map $\Theta^l_{k,n}$ is surjective.*

Note that whilst $\Theta^l_{k,n}$ is defined on all diagram basis elements in $P^l_k(n)$, it is only those that have at most n blocks whose image under $\Theta^l_{k,n}$ creates a basis of $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$.

Proof. Consider the two sets $B_D := \{D_\pi \mid \pi \in \Pi_{l+k,n}\}$ and $B_O := \{X_\theta \mid \theta \in \Pi_{l+k,n}\}$. We see that, by definition of D_π in (6.5), the transition matrix from B_D to B_O is unitriangular. As B_O forms a basis of $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$, this implies that B_D is also a basis of $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$. Since B_D is a basis, where each element in B_D is the image under $\Theta^l_{k,n}$ of a diagram basis element in $P^l_k(n)$, and $\Theta^l_{k,n}$ is linear, the surjectivity of $\Theta^l_{k,n}$ is immediate. $\square$

Given that Godfrey et al. (2023) referred to the basis $\{D_\pi \mid \pi \in \Pi_{l+k,n}\}$ of $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ as the diagram basis, we will do the same.

From now on, we refer to the diagram basis elements in $P^l_k(n)$ as **(k,l)–partition diagrams**.

We formulate the construction given above as the following theorem, with an immediate corollary relating to permutation equivariant weight matrices.

Theorem 6.2 (Diagram Basis when $G(n) = S_n$). *For all non-negative integers l, k and positive integers n, there is a surjection of vector spaces*

$$\Theta^l_{k,n} : P^l_k(n) \to \mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l}) \tag{6.7}$$

that is given by

$$\Theta^l_{k,n}(d_\pi) := D_\pi \tag{6.8}$$

for all (k,l)–partition diagrams d_π, where D_π is defined in (6.5).

In particular, the set

$$\{D_\pi \mid d_\pi \text{ is a } (k,l)\text{–partition diagram having at most } n \text{ blocks}\} \tag{6.9}$$

is a basis for $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ in the standard basis of $\mathbb{R}^n$, of size $\mathrm{B}(l+k,n)$.

Corollary 6.3 (Permutation Equivariant Weight Matrices). *For all non-negative integers l, k and positive integers n, the weight matrix W that appears in an S_n-equivariant linear layer function from $(\mathbb{R}^n)^{\otimes k}$ to $(\mathbb{R}^n)^{\otimes l}$ must be of the form*

$$W = \sum_{\pi \in \Pi_{l+k,n}} \lambda_\pi D_\pi \tag{6.10}$$

for $\mathrm{B}(l+k,n)$ many weights $\lambda_\pi \in \mathbb{R}$.

We will see in what follows that there is an advantage in using weight matrices that come from the diagram basis of $P^l_k(n)$ as opposed to those that come from the orbit basis when we look to construct permutation equivariant neural networks.

A Procedure for Calculating the (I, J)-entry of each Permutation Equivariant Basis Matrix D_π from $(\mathbb{R}^n)^{\otimes k}$ to $(\mathbb{R}^n)^{\otimes l}$ using the Diagram Basis of $P_k^l(n)$.

Assume that d_π is a diagram basis element in $P_k^l(n)$ having at most n blocks.

Perform the following steps:

1. Place the indices I on the top row of d_π and the indices J on the bottom row of d_π.
2. If all of the vertices in each block in d_π have been overlaid with the same number, then the (I, J)-entry of D_π is 1, otherwise it is 0.

However, as it stands, to calculate the diagram basis of $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$, we would first need to calculate the orbit basis of $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ and then use (6.5). To calculate the diagram basis of $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ directly from the diagram basis for $P_k^l(n)$, we use the following theorem.

Theorem 6.4. *The (I, J)-entry of the matrix D_π given in Theorem 6.2 can be calculated directly from the diagram basis element $d_\pi \in P_k^l(n)$ as follows.*

1. *Place the indices I on the top row of d_π and the indices J on the bottom row of d_π.*
2. *If all of the vertices in each block in d_π have been overlaid with the same number, then the (I, J)-entry of D_π is 1, otherwise it is 0.*

Proof. Suppose that d_π is a diagram basis element in $P_k^l(n)$. Assume that we have placed the indices I on the top row of d_π and the indices J on the bottom row of d_π.

Consider the following cases.

Case 1: (I, J) does not label the blocks of d_π consistently.

Then, by (6.1) and the definition of the partial order $\preceq$ on set partitions of $[l + k]$, (I, J) does not label the blocks of any of the x_θ consistently. Hence, the (I, J)-entry of each X_θ that is the image of x_θ under $\Gamma_{k,n}^l$ is 0, and so the (I, J)-entry of D_π is also 0, by (6.5).

Case 2: (I, J) labels the blocks of d_π consistently.

Then, by (6.1), we see that (I, J) labels the blocks of exactly one x_θ in (6.1) consistently *and distinctly*. Indeed, this set partition θ is obtained from π by taking the union of the blocks in π where the vertices in those blocks have been labelled by the same values after the indices I have been placed on the top row of d_π and the indices J have been placed on the bottom row of d_π. Hence, the (I, J)-entry for

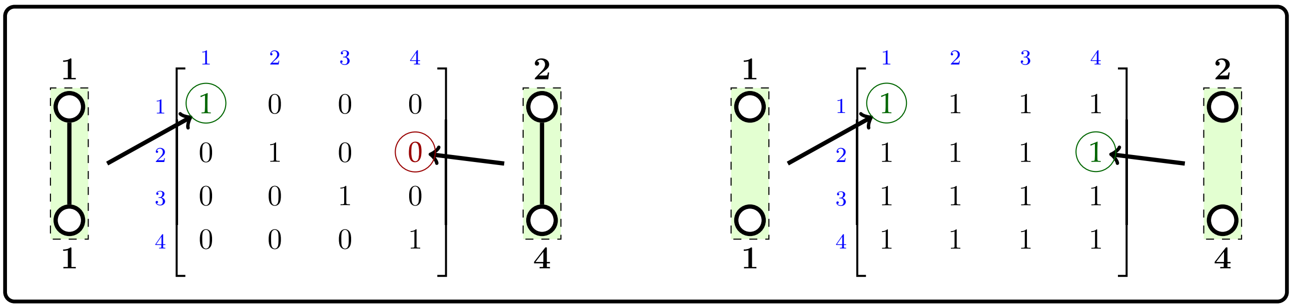


Figure 6.1: We obtain the two diagram basis matrices whose weighted linear combination gives all of the possible weight matrices that can appear in an S_4-equivariant neural network from $\mathbb{R}^4$ to $\mathbb{R}^4$. We obtain these matrices from the diagram basis elements in $P_1^1(4)$ that have at most 4 blocks. For each diagram, to calculate the (I, J)-entry of its associated basis matrix, we place the I-tuple on the top row of the diagram and the J-tuple on the bottom row of the diagram, and see if they label the diagram's blocks consistently. If the labelling is consistent, then we put a 1 in the (I, J)-entry of the matrix, otherwise 0. Compare these matrices with those that appeared in Figure 3.1.

X_θ that is the image of this x_θ under $\Gamma_{k,n}^l$ is 1, and is 0 for the images of all of the other orbit basis diagrams in (6.1) that are not x_θ. Hence, the (I, J)-entry of D_π is 1, by (6.5). □

In the clay box, we have turned Theorem 6.4 into a simple procedure for calculating the diagram basis of $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ that is similar to the procedure that appeared in Chapter 3 for calculating the orbit basis of $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$.

Consequently, from Theorem 6.4, we obtain the following corollary that describes another way to express each matrix D_π in $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ in terms of matrix units.

Corollary 6.5. *Let D_π be the image of a diagram basis element d_π in $P_k^l(n)$.*

Then, if $S_\pi((I, J))$ is defined to be the set

$$\left\{(I, J) \in [n]^{l+k} \mid \text{if } x, y \text{ are in the same block of } \pi, \text{then } i_x = i_y\right\} \tag{6.11}$$

(where we have momentarily replaced the elements of J by $i_{l+m} := j_m$ for all $m \in [k]$), we have that

$$D_\pi := \sum_{I \in [n]^l, J \in [n]^k} \delta_{\pi,(I,J)} E_{I,J} \tag{6.12}$$

where

$$\delta_{\pi,(I,J)} := \begin{cases} 1 & \text{if } (I, J) \in S_\pi((I, J)) \\ 0 & \text{otherwise} \end{cases} \tag{6.13}$$

One important consequence in what follows is that the tensor product of images of diagram basis elements in partition vector spaces is preserved under the map $\Theta^l_{k,n}$, in the following sense.

Corollary 6.6. *If D_π is the image of a diagram basis element d_π in $P^l_k(n)$, and D_τ is the image of a diagram basis element d_τ in $P^m_q(n)$, then their tensor product, $D_\pi \otimes D_\tau$, is D_ω, the image of a diagram basis element d_ω in $P^{l+m}_{k+q}(n)$.*

Proof. Let ω be the set partition $\pi \cup \tau$ of $[l+m+k+q]$. Then, by Corollary 6.5, we see that

$$D_\pi \otimes D_\tau = \left(\sum_{I \in [n]^l, J \in [n]^k} \delta_{\pi,(I,J)} E_{I,J} \right) \otimes \left(\sum_{X \in [n]^m, Y \in [n]^q} \delta_{\tau,(X,Y)} E_{X,Y} \right) \tag{6.14}$$

$$= \sum_{(I,X) \in [n]^{l+m}, (J,Y) \in [n]^{k+q}} \delta_{\omega,(I,X),(J,Y))} E_{(I,X),(J,Y)} \tag{6.15}$$

$$= D_\omega \tag{6.16}$$

where (6.15) holds because $S_\pi((I,J)) \cup S_\tau((X,Y)) = S_\omega((I,X),(J,Y))$. □

Remark 6.7. Note that in constructing the diagram basis for $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ from the diagram basis of $P^l_k(n)$, we do not require different blocks in the set partition diagram to be overlaid with different numbers, unlike with the construction given in Chapter 3 that uses the orbit basis of $P^l_k(n)$, where the blocks need to be labelled consistently *and distinctly*. This also explains why it is important that there are two different types of set partition diagrams for $P^l_k(n)$ — each type gives a different way of constructing permutation equivariant weight matrices.

Example 6.8. Compare the following with Example 3.20. Suppose that we would like to find the weight matrix for an S_4-equivariant linear layer function from $\mathbb{R}^4$ to $\mathbb{R}^4$, except we now use the diagram basis of $\mathrm{Hom}_{S_4}(\mathbb{R}^4, \mathbb{R}^4)$. Note that $l = k = 1$ and $n = 4$.

As before, to calculate this weight matrix, we first need to calculate all of the set partitions of $[1+1]$ having at most 4 blocks. These are $\pi_1 = \{1, 2\}$ and $\pi_2 = \{1 \mid 2\}$. But now, we express each of these set partitions as a diagram basis element in $P^1_1(4)$, and apply the map $\Theta^1_{1,4}$ to each of these diagrams to obtain the basis matrices D_{π_1} and D_{π_2}. Figure 6.1 gives examples of how to calculate the (I,J)-entries for both of these matrices using the procedure that is given in the clay-coloured box. In particular, we see that the $(1,1)$-entry of D_{π_1} is 1, since the only block in D_{π_1} is consistently labelled (by 1), whereas the $(2,4)$-entry of D_{π_1} is 0, since the only block in d_{π_1} is inconsistently labelled ($2 \neq 4$). However, compared with the matrix X_{π_2} that comes from the orbit basis diagram x_{π_2} in $P^1_1(4)$ (see

Example 3.20), we see that all of the entries of D_{π_2} are 1 since the two blocks in d_{π_2} are consistently labelled automatically!

Finally, we multiply each matrix by a weight, namely λ_1 and λ_2, respectively, and then add the two matrices together to obtain the overall weight matrix. Hence, the weight matrix for an S_4-equivariant linear layer function from $\mathbb{R}^4$ to $\mathbb{R}^4$, using the diagram basis of $\mathrm{Hom}_{S_4}(\mathbb{R}^4, \mathbb{R}^4)$, is of the form

$$\begin{array}{c c} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{bmatrix} \lambda_{1,2} & \lambda_2 & \lambda_2 & \lambda_2 \\ \lambda_2 & \lambda_{1,2} & \lambda_2 & \lambda_2 \\ \lambda_2 & \lambda_2 & \lambda_{1,2} & \lambda_2 \\ \lambda_2 & \lambda_2 & \lambda_2 & \lambda_{1,2} \end{bmatrix} \end{array} \tag{6.17}$$

for weights $\lambda_1, \lambda_2 \in \mathbb{R}$, where $\lambda_{1,2} := \lambda_1 + \lambda_2$.

6.2 Strict $\mathbb{R}$–Linear Monoidal Categories and String Diagrams

The result given in Corollary 6.6 hints at something more general, in that the tensor product of diagram basis matrices, which come from partition vector spaces, holds no matter how we choose the values of l, k, m and q.

To achieve this generality, we need to study these concepts at the level of categories. Recall from Section 2.5 the definition of a category and the definition of a functor between two categories.

We are specifically interested in categories that have a specific property called **monoidal**, and the monoidal functors between these categories. The monoidal property gives additional structure to the way in which objects and morphisms can be related. In particular, monoidal categories have an additional operation, known as a **tensor product**, that enables objects and morphisms to be composed in another way. We call this additional composition operation **horizontal**. Crucially, monoidal functors preserve the tensor product across monoidal categories.

We assume throughout that all categories are **locally small**, which, recalling Section 2.5, means that the collection of morphisms between any two objects is a set. In fact, all of the categories that we consider going forward have morphism sets that are vector spaces. Hence, the morphisms between objects become linear maps. We follow the presentation that is given in Hu (2019) and Savage (2021) below.

Definition 6.9. A category $\mathcal{C}$ is said to be **strict monoidal** if it comes with a functor $\otimes : \mathcal{C} \times \mathcal{C} \to \mathcal{C}$, known as a **bifunctor** or a **tensor product**, and a unit object $\mathbb{1}$, such that, for all objects X, Y, Z in $\mathcal{C}$, we have that

$$(X \otimes Y) \otimes Z = X \otimes (Y \otimes Z) \tag{6.18}$$

$$(\mathbb{1} \otimes X) = X = (X \otimes \mathbb{1}) \tag{6.19}$$

and, for all morphisms f, g, h in $\mathcal{C}$, we have that

$$(f \otimes g) \otimes h = f \otimes (g \otimes h) \tag{6.20}$$

$$(1_{\mathbb{1}} \otimes f) = f = (f \otimes 1_{\mathbb{1}}) \tag{6.21}$$

where $1_{\mathbb{1}}$ is the identity morphism $\mathbb{1} \to \mathbb{1}$. We often use the tuple $(\mathcal{C}, \otimes_{\mathcal{C}}, \mathbb{1}_{\mathcal{C}})$ to refer to the strict monoidal category $\mathcal{C}$.

We have started with the definition of a *strict* monoidal category as opposed to that of a monoidal category since we can assume that all monoidal categories are strict, owing to a technical result known as Mac Lane's Coherence Theorem. See Mac Lane (1998) for more details.

Some categories come with additional structure:

Definition 6.10. A category $\mathcal{C}$ is said to be **$\mathbb{R}$–linear** if, for any two objects X, Y in $\mathcal{C}$, the morphism set $\mathrm{Hom}_{\mathcal{C}}(X, Y)$ is, in fact, a vector space over $\mathbb{R}$, and the composition of morphisms is $\mathbb{R}$–bilinear.

By combining Definitions 6.9 and 6.10, we obtain the following definition:

Definition 6.11. A category $\mathcal{C}$ is said to be **strict $\mathbb{R}$–linear monoidal** if it is a category that is both strict monoidal and $\mathbb{R}$–linear, such that the bifunctor $\otimes$ is $\mathbb{R}$–bilinear.

In Section 2.5, we saw that there exists a notion of "maps" between categories, which are called functors. We are interested in a particular type of functor, which is defined as follows.

Definition 6.12. Suppose that $(\mathcal{C}, \otimes_{\mathcal{C}}, \mathbb{1}_{\mathcal{C}})$ and $(\mathcal{D}, \otimes_{\mathcal{D}}, \mathbb{1}_{\mathcal{D}})$ are two strict $\mathbb{R}$–linear monoidal categories.

A **strict $\mathbb{R}$–linear monoidal functor** from $\mathcal{C}$ to $\mathcal{D}$ is a functor $\mathcal{F} : \mathcal{C} \to \mathcal{D}$ such that

1. for all objects X, Y in $\mathcal{C}$, $\mathcal{F}(X \otimes_{\mathcal{C}} Y) = \mathcal{F}(X) \otimes_{\mathcal{D}} \mathcal{F}(Y)$

2. for all morphisms f, g in $\mathcal{C}$, $\mathcal{F}(f \otimes_{\mathcal{C}} g) = \mathcal{F}(f) \otimes_{\mathcal{D}} \mathcal{F}(g)$

3. $\mathcal{F}(\mathbb{1}_{\mathcal{C}}) = \mathbb{1}_{\mathcal{D}}$, and

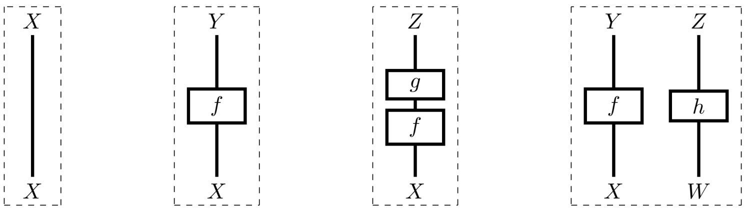
4. for all objects X, Y in $\mathcal{C}$, the map

$$\mathrm{Hom}_{\mathcal{C}}(X, Y) \to \mathrm{Hom}_{\mathcal{D}}(\mathcal{F}(X), \mathcal{F}(Y)) \tag{6.22}$$

   given by $f \mapsto \mathcal{F}(f)$ is $\mathbb{R}$–linear.

Strict monoidal categories are particularly interesting because they can be represented by a very useful diagrammatic language known as **string diagrams**. We will see that, as this language is, in some sense, geometric in nature, it is much easier to work with these diagrams compared with their equivalent algebraic form.

Definition 6.13 (String Diagrams)**.** Suppose that $\mathcal{C}$ is a strict monoidal category. Let W, X, Y and Z be objects in $\mathcal{C}$, and let $f : X \to Y$, $g : Y \to Z$, and $h : W \to Z$ be morphisms in $\mathcal{C}$. Then we can represent the morphisms $1_X : X \to X$, $f : X \to Y$, $g \circ f : X \to Z$ and $f \otimes h : X \otimes W \to Y \otimes Z$ as string diagrams in the following way:



(6.23)

In particular, the vertical composition of morphisms $g \circ f$ is obtained by placing g above f, and the horizontal composition of morphisms $f \otimes h$ is obtained by horizontally placing f to the left of h.

We will typically omit the labelling of the objects when they are clear or when they are not important.

As an example of how useful string diagrams are when working with strict monoidal categories, the associativity of the bifunctor given in (6.20) becomes immediately apparent. Another, more involved, example is given by the interchange law that exists for any strict monoidal category. It can be expressed algebraically as

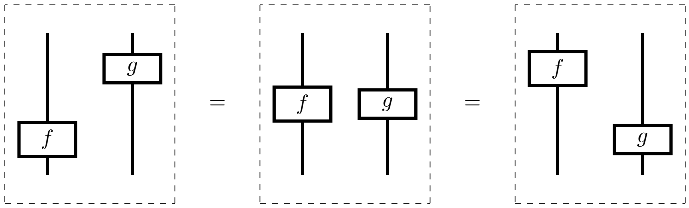
$$(\mathbb{1} \otimes g) \circ (f \otimes \mathbb{1}) = f \otimes g = (f \otimes \mathbb{1}) \circ (\mathbb{1} \otimes g) \tag{6.24}$$

Without string diagrams, it is somewhat tedious to prove this result. Indeed, for the left hand equality of (6.24), we have that

$$(\mathbb{1} \otimes g) \circ (f \otimes \mathbb{1}) = (\otimes(\mathbb{1}, g)) \circ (\otimes(f, \mathbb{1})) = \otimes((\mathbb{1}, g) \circ (f, \mathbb{1})) = \otimes((f, g)) = f \otimes g \tag{6.25}$$

where we have used the definition of composition in $\mathcal{C} \times \mathcal{C}$ and the fact that $\otimes : \mathcal{C} \times \mathcal{C} \to \mathcal{C}$ is a functor. A similar calculation also holds for the right hand equality of (6.24).

However, with string diagrams, the result is intuitively obvious, if we allow ourselves to deform the diagrams by pulling on the strings:



(6.26)

6.3 Categorification

In previous chapters, we defined a vector space for each non-negative integer l and k that is the $\mathbb{R}$–linear span of a certain subset of (k, l)–partition diagrams. However, it is clear that, for all values of l and k, these vector spaces are all similar in nature, in that the set partition diagrams only differ by the number of vertices that appear in each row and by the connections that are made between vertices. Moreover, set partition diagrams look like string diagrams. Given that string diagrams represent strict monoidal categories, and that we have a collection of vector spaces for certain subsets of set partition diagrams, this implies that we should have a number of strict $\mathbb{R}$–linear monoidal categories! Indeed we do — we formalise this intuition below.

6.3.1 Partition Categories

We assume throughout that l and k are non-negative integers and that n is a positive integer. In Section 3.2.1, we defined the partition vector space $P_k^l(n)$, which has a basis consisting of all possible (k, l)–partition diagrams. In order to obtain a category from these vector spaces, we need to define the following two $\mathbb{R}$–bilinear operations on (k,l)–partition diagrams:

$$\text{composition: } \bullet : P_l^m(n) \times P_k^l(n) \to P_k^m(n) \tag{6.27}$$

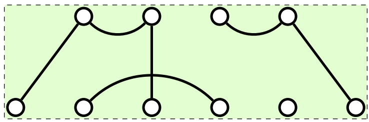
$$\text{tensor product: } \otimes : P_k^l(n) \times P_q^m(n) \to P_{k+q}^{l+m}(n) \tag{6.28}$$

They are given as follows:

Definition 6.14 (Composition)**.** Let $d_{\pi_1} \in P_k^l(n)$ and $d_{\pi_2} \in P_l^m(n)$. First, we concatenate the diagrams, written $d_{\pi_2} \circ d_{\pi_1}$, by putting d_{π_1} below d_{π_2}, concatenating the edges in the middle row of vertices, and then removing all connected components that lie entirely in the middle row of the concatenated diagrams. Let $c(d_{\pi_2}, d_{\pi_1})$ be the number of connected components that are removed from the middle row in $d_{\pi_2} \circ d_{\pi_1}$. Then the composition is defined, using infix notation, as

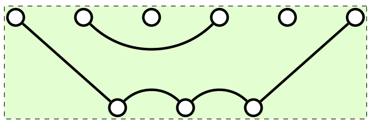
$$d_{\pi_2} \bullet d_{\pi_1} := n^{c(d_{\pi_2}, d_{\pi_1})}(d_{\pi_2} \circ d_{\pi_1}) \tag{6.29}$$

Example 6.15. If d_{π_2} is the $(6,4)$–partition diagram



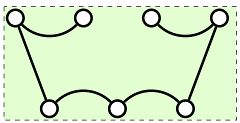
(6.30)

and d_{π_1} is the $(3,6)$–partition diagram



(6.31)

then we have that $d_{\pi_2} \circ d_{\pi_1}$ is the $(3,4)$–partition diagram

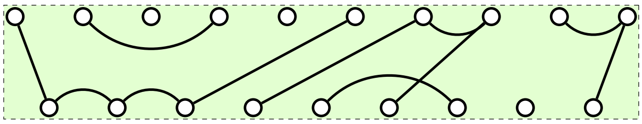


(6.32)

and so $d_{\pi_2} \bullet d_{\pi_1}$ is the diagram (6.32) multiplied by n^2, since two connected components were removed from the middle row of $d_{\pi_2} \circ d_{\pi_1}$.

Definition 6.16 (Tensor Product)**.** Let $d_{\pi_1} \in P_k^l(n)$ and $d_{\pi_2} \in P_q^m(n)$. Then $d_{\pi_1} \otimes d_{\pi_2}$ is defined to be the $(k+q, l+m)$–partition diagram that is obtained by horizontally placing d_{π_1} to the left of d_{π_2} without any overlapping of vertices.

Example 6.17. The tensor product $d_{\pi_1} \otimes d_{\pi_2}$, for d_{π_1} and d_{π_2} given in Example 6.15, is the $(9, 10)$–partition diagram



(6.33)

It is clear that both the composition and tensor product operations are associative.

Consequently, we have the following category:

Definition 6.18. We define the **partition category** $\mathcal{P}(n)$ to be the category whose objects are the non-negative integers, and, for any pair of objects k and l, the morphism space $\text{Hom}_{\mathcal{P}(n)}(k, l)$ is $P_k^l(n)$.

The vertical composition of morphisms is the composition of partition diagrams given in Definition 6.14; the bifunctor (the horizontal composition of morphisms) is the tensor product of partition diagrams given in Definition 6.16; and the unit object is 0.

Given that we can inherit the composition and tensor product operations for $B_k^l(n)$ from the composition and tensor product operations for $P_k^l(n)$, we have the following definition.

Definition 6.19. We define the **Brauer category** $\mathcal{B}(n)$ to be the category whose objects are the same as those of $\mathcal{P}(n)$ and, for any pair of objects k and l, the morphism space $\text{Hom}_{\mathcal{B}(n)}(k, l)$ is $B_k^l(n)$.

The vertical and horizontal composition of morphisms and the unit object are the same as those defined for $\mathcal{P}(n)$.

Proposition 6.20. *The partition category $\mathcal{P}(n)$ and the Brauer category $\mathcal{B}(n)$ are strict $\mathbb{R}$–linear monoidal categories.*

Proof. We prove this result for the partition category since the proof for the Brauer category is effectively the same.

$\mathcal{P}(n)$ is a strict monoidal category because the bifunctor on objects reduces to the addition of natural numbers, which is associative, and the bifunctor on morphisms is the tensor product of linear combinations of partition diagrams that is given in Definition 6.14, which is associative by definition.

$\mathcal{P}(n)$ is $\mathbb{R}$–linear because the morphism space between any two objects is by definition a vector space, and the composition of morphisms is $\mathbb{R}$–bilinear by definition. For the same reason, the bifunctor is also $\mathbb{R}$–bilinear. $\square$

We can also obtain a strict $\mathbb{R}$–linear monoidal category $\mathcal{BG}(n)$ from the Brauer–Grood vector spaces $D_k^l(n)$. We call $\mathcal{BG}(n)$ the **Brauer–Grood category**, although it is also known in the literature as the Jellyfish Brauer category (Comes, 2020), for reasons which will become apparent. Showing that $\mathcal{BG}(n)$ is strict $\mathbb{R}$–linear monoidal is a long and arduous computation that is the result of combining the proof of Lehrer and Zhang (2012, Theorem 2.6) together with the proof of Lehrer and Zhang (2018, Theorem 6.1), hence we omit it here.

Instead, we show how the composition and tensor product operations are defined for $D_k^l(n)$, which are needed to be able to define the Brauer–Grood category. Recall that $D_k^l(n)$ is the $\mathbb{R}$–linear span of the set of all (k, l)–Brauer diagrams together with the set of all $(l + k)\backslash n$–diagrams.

We want to define two $\mathbb{R}$–bilinear operations

$$\text{composition: } \bullet : D_l^m(n) \times D_k^l(n) \to D_k^m(n) \tag{6.34}$$

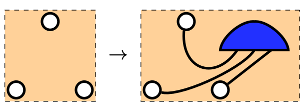
$$\text{tensor product: } \otimes : D_k^l(n) \times D_q^m(n) \to D_{k+q}^{l+m}(n) \tag{6.35}$$

It is clear that we need to define these operations on ordered pairs of diagrams in each of the following four cases:

1. Brauer and Brauer
2. Brauer and $(l + k)\backslash n$
3. $(l + k)\backslash n$ and Brauer, and
4. $(l + k)\backslash n$ and $(l + k)\backslash n$

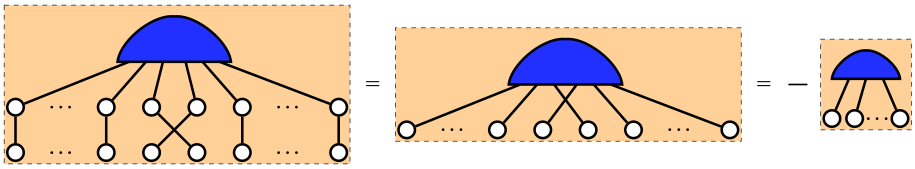
In particular, to define the composition and tensor product operations in Cases 2, 3, and 4 where an $(l + k)\backslash n$–diagram appears, we reintroduce the n-legged jellyfish by attaching the free vertices in the diagram to the head of the jellyfish, in top-to-bottom, left-to-right order.

Example 6.21. If d_α is a $(1 + 2)\backslash 3$–diagram, then by attaching the jellyfish we obtain the following diagram:

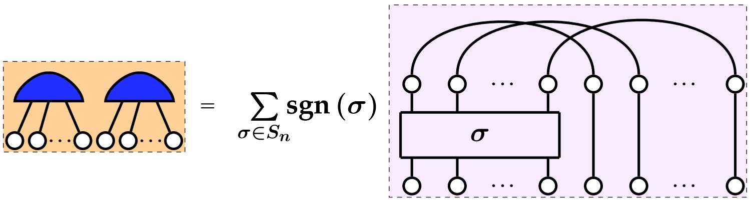

(6.36)

We allow the jellyfish to satisfy the following two rules.

Rule 1: composing an adjacent permutation with a jellyfish, which is equivalent to having two adjacent legs of the jellyfish crossed, is equal to the negative of the uncrossed jellyfish:

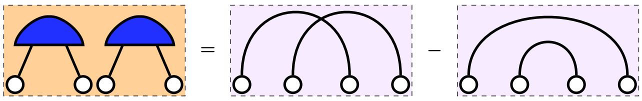

(6.37)

Rule 2: the juxtaposition of two jellyfish, each with n legs, is equal to the following linear combination of Brauer diagrams:

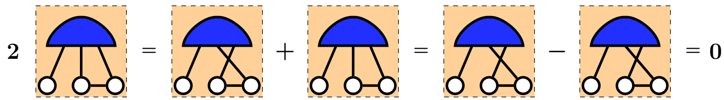


(6.38)

Example 6.22. If $n = 2$, then Rule 2 becomes


(6.39)

One consequence of Rule 1 is that if two legs of the same jellyfish are connected, then it is equal to 0. For example,


(6.40)

With this, we can now define the composition and tensor product operations for each of the four cases.

Definition 6.23 (Composition)**.** We define composition for each of the four cases given above.

Case 1: The composition of a (k, l)–Brauer diagram with an (l, m)–Brauer diagram is the same as the composition that was given in Definition 6.14.

Cases 2, 3: We explain this in the situation where we have a (k, l)–Brauer diagram d_β and an $(m + l)\backslash n$–diagram d_α, but the same steps apply in the other case. We perform the following procedure:

- Attach a jellyfish to d_α, as in (6.36).
- Place d_β below d_α, and concatenate the diagrams to obtain a diagram with three rows of vertices.

- If any two legs of the same jellyfish are connected in this diagram, the result is 0, and we can stop.

- Otherwise, fully concatenate the diagrams to obtain a diagram with two rows of vertices, removing all connected components that lie entirely in the middle row of the concatenated diagrams. Let $c(d_\alpha, d_\beta)$ be the number of connected components that are removed from the middle row.

- Now repeatedly apply Rule 1 to fully uncross the legs of the jellyfish. This is equivalent to a permutation $\sigma \in S_n$ acting on the legs of the jellyfish, and so the sign that we put in front of the final diagram is sgn(σ), the sign of the permutation.

- Finally, detach/remove the jellyfish to obtain an $(m+k)\backslash n$–diagram d_γ. Consequently, the result that we obtain is sgn(σ)$n^{c(d_\alpha, d_\beta)} d_\gamma$.

Case 4: Suppose that we have an $(l+k)\backslash n$–diagram d_{α_1} and an $(m+l)\backslash n$–diagram d_{α_2}. We perform the following procedure:

- Attach separate jellyfish to d_{α_1} and d_{α_2}, as in (6.36).

- Place d_{α_1} below d_{α_2}, and concatenate the diagrams to obtain a diagram with three rows of vertices.

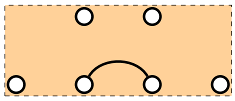
- If any two legs of the same jellyfish are connected in this diagram, the result is 0, and we can stop.

- Otherwise, fully concatenate the diagrams to obtain a diagram with two rows of vertices, removing all connected components that lie entirely in the middle row of the concatenated diagrams. Let $c(d_{\alpha_2}, d_{\alpha_1})$ be the number of connected components that are removed from the middle row.

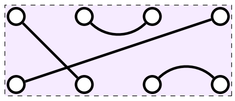
- Now repeatedly apply Rule 1 to fully uncross the legs of each of the jellyfish. This is equivalent to a product of permutations $\sigma\tau \in S_n$. Hence, the sign that we put in front of the final diagram is sgn($\sigma\tau$) $=$ sgn(σ) sgn(τ).

- Now apply Rule 2 to obtain a linear combination of (k, m)–Brauer diagrams. Call this linear combination d. Consequently, the result that we obtain is sgn(σ) sgn(τ)$n^{c(d_{\alpha_2}, d_{\alpha_1})} d$.

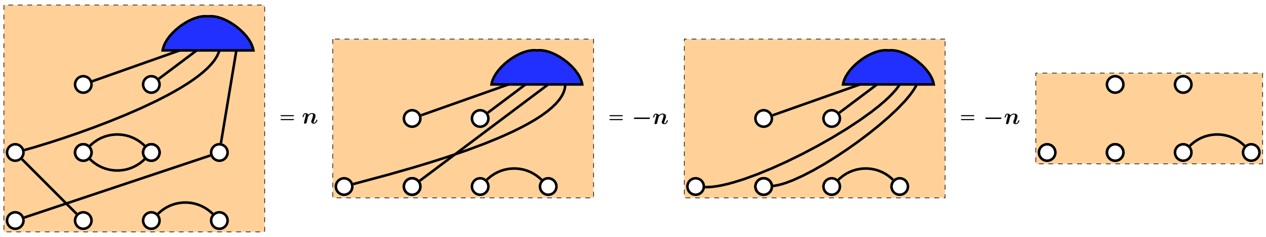
Example 6.24 (Composition, Case 3). Let d_α be the diagram



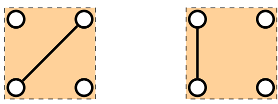
and let d_β be the diagram



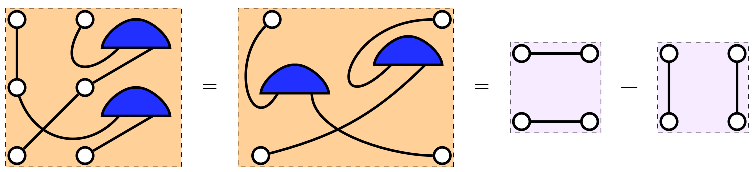
Then $d_\alpha \bullet d_\beta$ is equal to



Example 6.25 (Composition, Case 4). Let d_{α_1} and d_{α_2} be the diagrams



respectively. Then $d_{\alpha_2} \bullet d_{\alpha_1}$ is equal to



Definition 6.26 (Tensor Product)**.** Only the definition for Case 4 is non-trivial.

Case 1: The tensor product of a (k, l)–Brauer diagram with a (q, m)–Brauer diagram is the same as the tensor product that was given in Definition 6.16.

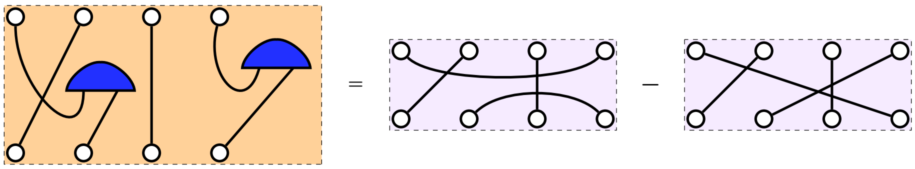
Case 2: The tensor product of a (k, l)–Brauer diagram d_β with an $(m + q)\backslash n$–diagram d_α is defined to be the $(l + m + k + q)\backslash n$–diagram that is obtained by horizontally placing d_β to the left of d_α without any overlapping of vertices.

Case 3: The tensor product of an $(l + k)\backslash n$–diagram d_α with a (q, m)–Brauer diagram d_β is defined to be the $(l + m + k + q)\backslash n$–diagram that is obtained by horizontally placing d_α to the left of d_β without any overlapping of vertices.

Case 4: Suppose that we have an $(l+k)\backslash n$–diagram d_{α_1} and an $(m+q)\backslash n$–diagram d_{α_2}. We perform the following procedure:

- Attach separate jellyfish to d_{α_1} and d_{α_2}, as in (6.36).
- Horizontally place d_{α_1} to the left of d_{α_2} without any overlapping of vertices.
- Apply Rule 2 to obtain a linear combination of $(k+q, l+m)$–Brauer diagrams. This is the end result.

Example 6.27 (Tensor Product, Case 4). Let d_{α_1} and d_{α_2} be as above. Then $d_{\alpha_1} \otimes d_{\alpha_2}$ is equal to



It is a consequence of Lehrer and Zhang (2018, Theorem 6.1) that these operations are associative, and so we obtain the following strict $\mathbb{R}$–linear monoidal category.

Definition 6.28. The **Brauer–Grood category** $\mathcal{BG}(n)$ is the category whose objects are the same as those of $\mathcal{P}(n)$ and, for any pair of objects k and l, the morphism space $\mathrm{Hom}_{\mathcal{BG}(n)}(k,l)$ is $D_k^l(n)$.

The vertical composition of morphisms and the horizontal composition of morphisms are the same as those defined for $D_k^l(n)$. The unit object is 0.

Remark 6.29. We will refer to these categories in general as **partition categories**. When we wish to specifically reference the partition category $\mathcal{P}(n)$, we will explicitly write $\mathcal{P}(n)$ to clarify that we mean this particular category.

6.3.2 Group Representation Categories

It is not just the partition vector spaces that are similar for different values of l and k. The tensor power spaces of $\mathbb{R}^n$ that make up the layers of the group equivariant neural networks defined in Section 2.3 and, in particular, the linear layer functions between them, are also similar for different values of l and k. To form a category from these representations and the linear maps between them, we first need to define a category for all representations of a group.

Definition 6.30. Let G be a group. Then $\mathrm{Rep}(G)$ is the category whose objects are pairs (V, ρ_V), where $\rho_V : G \to GL(V)$ is a representation of G, and, for any pair of objects (V, ρ_V) and (W, ρ_W), the morphism space, $\mathrm{Hom}_{\mathrm{Rep}(G)}((V, \rho_V), (W, \rho_W))$, is precisely the vector space of G-equivariant linear maps from V to W, $\mathrm{Hom}_G(V, W)$.

The vertical composition of morphisms is given by the composition of linear maps, the horizontal composition of morphisms is given by the tensor product of linear maps, both of which are associative operations, and the unit object is given by $(\mathbb{R}, 1_{\mathbb{R}})$, where $1_{\mathbb{R}}$ is the one-dimensional trivial representation of G.

If $G = G(n)$ is a subgroup of $GL(n)$, then we have the following category.

Definition 6.31. If $G(n)$ is a subgroup of $GL(n)$, then we define $\mathcal{C}(G(n))$ to be the category whose objects are pairs $\{((\mathbb{R}^n)^{\otimes k}, \rho_k)\}_{k \in \mathbb{Z}_{\geq 0}}$, where $\rho_k : G(n) \to GL((\mathbb{R}^n)^{\otimes k})$ is the representation of $G(n)$ given in Section 2.2, and, for any pair of objects $((\mathbb{R}^n)^{\otimes k}, \rho_k)$ and $((\mathbb{R}^n)^{\otimes l}, \rho_l)$, the morphism space $\mathrm{Hom}_{\mathcal{C}(G(n))}(((\mathbb{R}^n)^{\otimes k}, \rho_k), ((\mathbb{R}^n)^{\otimes l}, \rho_l))$ is precisely $\mathrm{Hom}_{G(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$.

The vertical composition of morphisms is given by the composition of linear maps, the horizontal composition of morphisms is given by the tensor product of linear maps, both of which are associative operations, and the unit object is given by $(\mathbb{R}, 1_{\mathbb{R}})$, where $1_{\mathbb{R}}$ is the one-dimensional trivial representation of $G(n)$.

Proposition 6.32. *For any subgroup $G(n)$ of $GL(n)$, $\mathcal{C}(G(n))$ is a full subcategory of the category of representations of $G(n)$, $\mathrm{Rep}(G(n))$. In particular, $\mathcal{C}(G(n))$ is a strict $\mathbb{R}$–linear monoidal category.*

Proof. It is clear from the definitions that $\mathcal{C}(G(n))$ is a subcategory of $\mathrm{Rep}(G(n))$. $\mathcal{C}(G(n))$ is a full subcategory of $\mathrm{Rep}(G(n))$, as, for any pair of objects $((\mathbb{R}^n)^{\otimes k}, \rho_k)$ and $((\mathbb{R}^n)^{\otimes l}, \rho_l)$ in $C(G(n))$, the morphism space $\mathrm{Hom}_{\mathcal{C}(G(n))}(((\mathbb{R}^n)^{\otimes k}, \rho_k), ((\mathbb{R}^n)^{\otimes l}, \rho_l))$ is $\mathrm{Hom}_{\mathrm{Rep}(G(n))}(((\mathbb{R}^n)^{\otimes k}, \rho_k), ((\mathbb{R}^n)^{\otimes l}, \rho_l))$.

$\mathcal{C}(G(n))$ can immediately be seen to be a strict monoidal category because the bifunctor on objects is the tensor product of vector spaces, which is associative, and the bifunctor on morphisms is the tensor product on linear maps between vector spaces, which is also associative.

$\mathcal{C}(G(n))$ is $\mathbb{R}$–linear because the morphism space for any two objects is a vector space over $\mathbb{R}$, and the composition of morphisms is $\mathbb{R}$–bilinear because composition is $\mathbb{R}$–bilinear for linear maps on vector spaces. It is also clear that the bifunctor $\otimes$ is $\mathbb{R}$–bilinear because it is the standard tensor product for vector spaces. $\square$

6.4 Monoidal Functors from Partition Categories to Group Representation Categories

In previous chapters, we established a number of linear maps between certain partition vector spaces and the vector space of equivariant linear maps between tensor power representations of certain subgroups $G(n)$ of $GL(n)$. Given that we have formed a number of strict $\mathbb{R}$–linear monoidal partition categories from these partition vector spaces, and a strict $\mathbb{R}$–linear monoidal category $\mathcal{C}(G(n))$ from the equivariant linear maps between tensor power representations of these subgroups $G(n)$, there should now exist strict $\mathbb{R}$–linear monoidal functors between the categories. Indeed, we have that

Theorem 6.33. *There exists a full, strict $\mathbb{R}$–linear monoidal functor*

$$\Theta : \mathcal{P}(n) \to \mathcal{C}(S_n) \tag{6.41}$$

that is defined on the objects of $\mathcal{P}(n)$ by $\Theta(k) := ((\mathbb{R}^n)^{\otimes k}, \rho_k)$ and, for any objects k, l of $\mathcal{P}(n)$, the map

$$\mathrm{Hom}_{\mathcal{P}(n)}(k, l) \to \mathrm{Hom}_{\mathcal{C}(S_n)}(\Theta(k), \Theta(l)) \tag{6.42}$$

is precisely the map

$$\Theta^l_{k,n} : P^l_k(n) \to \mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l}) \tag{6.43}$$

given in Theorem 6.2.

Theorem 6.34. *There exists a full, strict $\mathbb{R}$–linear monoidal functor*

$$\Phi : \mathcal{B}(n) \to \mathcal{C}(O(n)) \tag{6.44}$$

that is defined on the objects of $\mathcal{B}(n)$ by $\Phi(k) := ((\mathbb{R}^n)^{\otimes k}, \rho_k)$ and, for any objects k, l of $\mathcal{B}(n)$, the map

$$\mathrm{Hom}_{\mathcal{B}(n)}(k, l) \to \mathrm{Hom}_{\mathcal{C}(O(n))}(\Phi(k), \Phi(l)) \tag{6.45}$$

is the map

$$\Phi^l_{k,n} : B^l_k(n) \to \mathrm{Hom}_{O(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l}) \tag{6.46}$$

given in Theorem 5.15.

Theorem 6.35. *There exists a full, strict $\mathbb{R}$–linear monoidal functor*

$$X : \mathcal{B}(n) \to \mathcal{C}(Sp(n)) \tag{6.47}$$

that is defined on the objects of $\mathcal{B}(n)$ by $X(k) := ((\mathbb{R}^n)^{\otimes k}, \rho_k)$ and, for any objects k, l of $\mathcal{B}(n)$, the map

$$\mathrm{Hom}_{\mathcal{B}(n)}(k, l) \to \mathrm{Hom}_{\mathcal{C}(Sp(n))}(\Phi(k), \Phi(l)) \tag{6.48}$$

is the map

$$X_{k,n}^l : B_k^l(n) \to \mathrm{Hom}_{Sp(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l}) \tag{6.49}$$

given in Theorem 5.17.

Theorem 6.36. *There exists a full, strict $\mathbb{R}$–linear monoidal functor*

$$\Psi : \mathcal{BG}(n) \to \mathcal{C}(SO(n)) \tag{6.50}$$

that is defined on the objects of $\mathcal{BG}(n)$ by $\Psi(k) := ((\mathbb{R}^n)^{\otimes k}, \rho_k)$ and, for any objects k, l of $\mathcal{B}(n)$, the map

$$\mathrm{Hom}_{\mathcal{BG}(n)}(k, l) \to \mathrm{Hom}_{\mathcal{C}(SO(n))}(\Phi(k), \Phi(l)) \tag{6.51}$$

is the map

$$\Psi_{k,n}^l : D_k^l(n) \to \mathrm{Hom}_{SO(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l}) \tag{6.52}$$

given in Theorem 5.19.

Proof of Theorem 6.33. We prove this theorem in a series of steps.

Step 1: We need to show that $\Theta : \mathcal{P}(n) \to \mathcal{C}(S_n)$ is a functor.

We show that Θ satisfies the definition of a functor which was given in Section 2.5.

- Θ maps the objects of $\mathcal{P}(n)$ to the objects of $\mathcal{C}(S_n)$ by definition.
- It is enough to show that $\Theta(g)\Theta(f) = \Theta(g \bullet f)$ on arbitrary basis elements of arbitrary morphism spaces where the codomain of f is the domain of g because the morphism spaces are vector spaces.

Let $f = d_{\pi_1}$ be a (k,l)–partition diagram, and let $g = d_{\pi_2}$ be a (l,m)–partition diagram. Then, by Definition 6.14, we have that

$$d_{\pi_2} \bullet d_{\pi_1} = n^{c(d_{\pi_2}, d_{\pi_1})} d_{\pi_3} \tag{6.53}$$

where d_{π_3} be the composition $d_{\pi_2} \circ d_{\pi_1}$ expressed as a (k,m)–partition diagram. Then

$$\Theta(g \bullet f) = \Theta(d_{\pi_2} \bullet d_{\pi_1}) = n^{c(d_{\pi_2}, d_{\pi_1})} D_{\pi_3} \tag{6.54}$$

We also have that

$$\Theta(g)\Theta(f) = D_{\pi_2} D_{\pi_1} = \left(\sum_{I \in [n]^m, K \in [n]^l} \delta_{\pi_2,(I,K)} E_{I,K} \right) \left(\sum_{L \in [n]^l, J \in [n]^k} \delta_{\pi_1,(L,J)} E_{L,J} \right) \tag{6.55}$$

$$= \sum_{I \in [n]^m, K \in [n]^l, J \in [n]^k} \delta_{\pi_2,(I,K)} \delta_{\pi_1,(K,J)} E_{I,K} E_{K,J} \tag{6.56}$$

For fixed I, J, consider

$$\sum_{K \in [n]^l} \delta_{\pi_2,(I,K)} \delta_{\pi_1,(K,J)} \tag{6.57}$$

This is equal to

$$n^{c(d_{\pi_2}, d_{\pi_1})} \delta_{\pi_3,(I,J)} \tag{6.58}$$

Indeed, for fixed I, J, $\delta_{\pi_3,(I,J)}$ is 1 if and only if both $\delta_{\pi_2,(I,K)}$ and $\delta_{\pi_1,(K,J)}$ are 1 for some $K \in [n]^l$ since d_{π_3} is the composition $d_{\pi_2} \circ d_{\pi_1}$. The number of such K is determined only by the vertices that appear in connected components that are removed from the middle row of $d_{\pi_2} \circ d_{\pi_1}$, since, for fixed I, J, only these vertices can be freely chosen if we want both $\delta_{\pi_2,(I,K)}$ and $\delta_{\pi_1,(K,J)}$ to be 1. However, since the entries in each connected component must be the same for both $\delta_{\pi_2,(I,K)}$ and $\delta_{\pi_1,(K,J)}$ to be 1, this implies that the number of $K \in [n]^l$ such that both $\delta_{\pi_2,(I,K)}$ and $\delta_{\pi_1,(K,J)}$ are 1 is $n^{c(d_{\pi_2}, d_{\pi_1})}$.

Hence (6.56) becomes

$$\sum_{I \in [n]^m, J \in [n]^k} n^{c(d_{\pi_2}, d_{\pi_1})} \delta_{\pi_3,(I,J)} E_{I,J} = n^{c(d_{\pi_2}, d_{\pi_1})} E_{\pi_3} \tag{6.59}$$

and so, by (6.54), we have that $\Theta(g)\Theta(f) = \Theta(g \bullet f)$, as required.

- For each object k in $\mathcal{P}(n)$, the identity morphism 1_k is the (k,k)–partition diagram d_π where π is the set partition

$$\{1, k+1 \mid 2, k+2 \mid \cdots \mid k, 2k\} \tag{6.60}$$

of $[2k]$, and its image under $d_\pi \mapsto D_\pi$ is the $n^k \times n^k$ identity matrix, as required.

Step 2: We need to show that $\Theta : \mathcal{P}(n) \to \mathcal{C}(S_n)$ is full.

The functor Θ is full because, for all objects k, l in $\mathcal{P}(n)$, the map (6.42) is (6.43), by definition, and this map is surjective by Theorem 6.2.

Step 3: We need to show that $\Theta : \mathcal{P}(n) \to \mathcal{C}(S_n)$ is strict $\mathbb{R}$–linear monoidal.

To show that Θ is strict $\mathbb{R}$–linear monoidal, we need to show that Θ satisfies the conditions given in Definition 6.12. The picture to have in mind for point 2 below is that the tensor product on set partition diagrams places the diagrams side-by-side, without any overlapping of vertices.

1. Let k, l be any two objects in $\mathcal{P}(n)$. Then

$$\Theta(k \otimes l) = \Theta(k+l) \tag{6.61}$$

$$= ((\mathbb{R}^n)^{\otimes k+l}, \rho_{k+l}) \tag{6.62}$$

$$= ((\mathbb{R}^n)^{\otimes k} \otimes (\mathbb{R}^n)^{\otimes l}, \rho_k \otimes \rho_l) \tag{6.63}$$

$$= \Theta(k) \otimes \Theta(l) \tag{6.64}$$

2. It is enough to prove this condition on arbitrary basis elements of arbitrary morphism spaces as the morphism spaces are vector spaces. Suppose that $f : k \to l$ and $g : q \to m$ are two basis elements in $\mathrm{Hom}_{\mathcal{P}(n)}(k, l)$ and $\mathrm{Hom}_{\mathcal{P}(n)}(q, m)$ respectively. Then $f = d_\pi$ for some set partition π of $[l+k]$, and $g = d_\tau$ for some set partition τ of $[m+q]$.

   As $\mathrm{Hom}_{\mathcal{P}(n)}(k, l) = P_k^l(n)$ and $\mathrm{Hom}_{\mathcal{P}(n)}(q, m) = P_q^m(n)$, we have, by Definition 6.16, that $f \otimes g$ is an element of $P_{k+q}^{l+m}(n) = \mathrm{Hom}_{\mathcal{P}(n)}(k+q, l+m)$.

   In particular, $f \otimes g = d_\omega$ for the set partition $\omega := \pi \cup \tau$ of $[l+m+k+q]$.

   By Theorem 6.2, we have that $\Theta(f) = D_\pi$, $\Theta(g) = D_\tau$, and $\Theta(f \otimes g) = D_\omega$.

   By Corollary 6.6, we see that $\Theta(f) \otimes \Theta(g) = \Theta(f \otimes g)$, as required.

3. It is clear from the statement of the theorem that Θ sends the unit object 0 in $\mathcal{P}(n)$ to $(\mathbb{R}, 1_\mathbb{R})$, which is the unit object in $\mathcal{C}(S_n)$.

4. This is immediate because the map (6.43) is $\mathbb{R}$–linear by Theorem 6.2. □

Proof of Theorem 6.34. The proof is practically identical to that of Theorem 6.33, except we replace Θ by Φ. □

Proof of Theorem 6.35. The proof is very similar to that of Theorem 6.33. For step 3, point 2, we need to consider the form of the matrix (5.76), but it is not hard to see how the proof of step 3, point 2 for Theorem 6.33 can be adapted for this situation, since, if d_{β_1} is a (k, l)–Brauer diagram, and if d_{β_2} is a (q, m)–Brauer diagram, then, defining $\omega := \beta_1 \cup \beta_2$, that is, d_ω is a $(k + q, l + m)$–Brauer diagram, we have that two vertices in d_{β_i} are in different rows (the same row) if and only if they are in different rows (the same row) of d_ω. □

Proof of Theorem 6.36. This is Lehrer and Zhang (2018, Theorem 6.1). □

6.5 Implications for Group Equivariant Neural Networks

These results imply a number of statements that are important in the context of the group equivariant neural networks for $G(n)$ that we have studied, where $G(n)$ is a subgroup of $GL(n)$. They can be stated as follows.

1. To understand and work with any matrix in $\mathrm{Hom}_{G(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$, it is enough to work with the subset of (k, l)–partition diagrams that corresponds to $G(n)$. This is because we can express any matrix in terms of the set of spanning set elements for $\mathrm{Hom}_{G(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$, and this set corresponds bijectively with the subset of (k, l)–partition diagrams that corresponds to $G(n)$. We can recover the matrix itself by applying the appropriate monoidal functor to the set partition diagrams because the monoidal functors are full.

2. We can manipulate the connected components and vertices of (k, l)–partition diagrams to obtain new set partition diagrams. Indeed, as the partition categories are strict monoidal, this means that the (k, l)–partition diagrams in a partition category are string diagrams. Statement 1 immediately implies that we will obtain new matrices between tensor power spaces of $\mathbb{R}^n$ that are equivariant to $G(n)$ from the resulting set partition diagrams.

3. If a (k,l)–partition diagram can be decomposed as a tensor product of smaller set partition diagrams, then the corresponding matrix can also be decomposed as a tensor product of smaller sized matrices, each of which is equivariant to $G(n)$. This is because the functors are monoidal.

In particular, for the symmetric group S_n, point 3 immediately implies the advantage that we claimed in using weight matrices that come from the diagram basis of $P_k^l(n)$, as opposed to those that come from the orbit basis of $P_k^l(n)$. These weight matrices can be decomposed as a tensor product of smaller sized matrices, each of which can be determined by a smaller sized partition diagram, since the map $\Theta_{k,n}^l$ corresponding to the diagram basis of $P_k^l(n)$ that is used to define the functor in Theorem 6.33 makes it monoidal, meaning that it preserves the tensor product. If we had used the map $\Gamma_{k,n}^l$ corresponding to the orbit basis of $P_k^l(n)$ instead to define the functor, it would not be monoidal, and so tensor products would not be preserved under it.

Chapter 7

Algorithm for Performing Linear Equivariant Layer Functions

We now use the monoidal functors that we established in the previous chapter to construct a fast multiplication algorithm for passing a vector through a linear layer function that appears in a group equivariant neural network of the form given in Section 2.3 for any of the groups $G(n)$ that are associated with the functors.

In particular, suppose that we would like to pass a vector $v \in (\mathbb{R}^n)^{\otimes k}$ through a linear layer function that lives in $\mathrm{Hom}_{G(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$. A naïve implementation of the matrix multiplication between the weight matrix and the vector has a time complexity equal to $O(n^{l+k})$. We show that, by using the monoidal functors, and, in particular, the implications for group equivariant neural networks that we listed at the end of the previous chapter, we can construct a faster multiplication algorithm that reduces the time complexity exponentially for each of the groups $G(n)$ in question.

It is important to highlight that, since we have at least a spanning set of $\mathrm{Hom}_{G(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ for each group $G(n)$ that appeared in the previous chapter, it is enough to describe an algorithm for how to multiply v by a spanning set element, since we can extend the result by linearity. The linearity provides an extra advantage in that it enables the fast matrix multiplication of an input vector $v \in (\mathbb{R}^n)^{\otimes k}$ with a weight matrix in $\mathrm{Hom}_{G(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ to be executed in parallel, namely by performing the fast matrix multiplication on each of the spanning set elements that make up the weight matrix.

The content of this chapter is modelled on the paper:

Edward Pearce-Crump and William J. Knottenbelt. (2024a) A Diagrammatic Approach to Improve Computational Efficiency in Group Equivariant Neural Networks. `arXiv:2412.10837`.

7.1 Algorithmically Planar Set Partition Diagrams

Recall that each spanning set element in $\mathrm{Hom}_{G(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ corresponds to a set partition diagram under a monoidal functor for $G(n)$, and that a set partition diagram is a string diagram since it is a morphism in a monoidal category. This leads to the following question that we use as motivation: can we use the string-like property of each set partition diagram to gain control over how the matrix multiplication operation is performed between the spanning set element and the vector?

There are a few guiding principles that we use to construct the algorithm.

Ideally, we would like to break the matrix multiplication operation down into a series of computations, each of which is indecomposable. Given the monoidal functor relationships between set partition diagrams and matrix operations, we know that the smallest possible computations that are available to us correspond precisely with the smallest indecomposable set partition diagrams. In general, set partition diagrams that consist only of one block are the smallest indecomposable diagrams. The exception is for $(l+k)\backslash n$–diagrams, where the smallest indecomposable diagrams are either those consisting of a single block of two vertices or a diagram consisting of all of the free vertices taken together. Moreover, a single block can live either entirely in the top row, entirely in the bottom row, or in both rows. Hence, we see that there are only a few classes of indecomposable diagrams that are smallest. As a result, there are only a few classes of indecomposable matrix operations.

We can say more. Since the matrices that we obtain must come from set partition diagrams, the most efficient matrices that we can construct are a Kronecker product of indecomposable matrices. But also, since indecomposable matrices come from indecomposable set partition diagrams, this implies that the most efficient matrices that we can construct come from set partition diagrams that decompose into a tensor product of the smallest indecomposable set partition diagrams. Moreover, we would like to choose the order in which the indecomposable matrices appear in the Kronecker product because each indecomposable matrix performs a certain operation and we would like them to be executed in an order that is as efficient as possible. For example, it is best to zero out as many terms as possible in the input vector first, if such an operation can be performed, since this reduces the number of elements that need to be operated on going forward. After this, it is better to perform tensor contraction operations before performing copying operations, because this reduces the number of elements that need to be copied. It is clear that choosing any other order for these operations would make the overall algorithm less efficient. We will see that each of these operations can be understood easily in their equivalent diagram form.

Consequently, from the set partition diagram that corresponds to a spanning set element, we would like to create another set partition diagram that decomposes into a tensor product of the smallest possible indecomposable set partition diagrams and are ordered such that the indecomposable matrix operations that appear in the corresponding Kronecker product will be performed most efficiently.

We call these special set partition diagrams **algorithmically planar**, and by construction they are the most efficient diagrams for performing matrix multiplication. They are defined as follows.

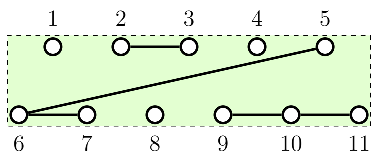
Definition 7.1. A (k, l)–partition diagram is said to be **algorithmically planar** if it satisfies the following conditions:

- The connected components that are solely in the bottom row are lined up sequentially, starting from the far right hand side, such that the vertices in each connected component are in consecutive order. We also require that these connected components are ordered by their size, starting on the far right hand side with the largest and decreasing in size as we move leftward, for reasons that will become clear when we conduct a time complexity analysis of the multiplication algorithm for the symmetric group.

- The connected components that are solely in the top row are lined up sequentially, starting from the far left hand side, such that the vertices in each connected component are in consecutive order. (Note that these connected components can be in any order.)

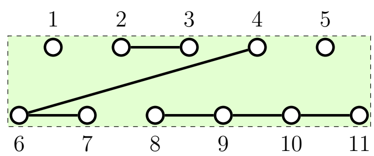
- The connected components between different rows are such that no two subsets cross.

Example 7.2. The $(6, 5)$–partition diagram



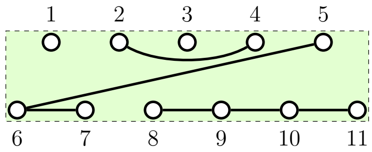
(7.1)

is algorithmicallly planar. However, the $(6, 5)$–partition diagram



(7.2)

is not algorithmically planar because the connected component consisting solely of vertex 5 is not adjacent to a connected component that lives solely in the top row, and the $(6, 5)$–partition diagram

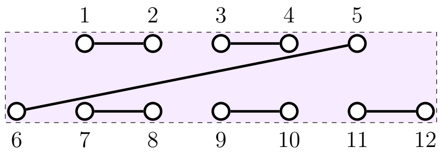


(7.3)

is also not algorithmically planar because the vertices in the connected component $\{2, 4\}$ are not in consecutive order.

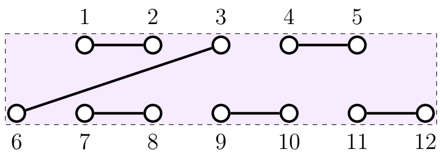
Definition 7.3. A (k, l)–Brauer diagram is said to be **algorithmically planar** if it is an algorithmically planar (k, l)–partition diagram that is also a Brauer diagram.

Example 7.4. The $(7, 5)$–Brauer diagram



(7.4)

is algorithmicallly planar, whereas the $(7, 5)$–Brauer diagram



(7.5)

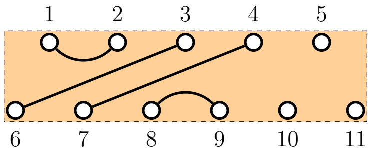
is not algorithmicallly planar.

Definition 7.5. An $(l + k)\backslash n$–diagram is said to be **algorithmically planar** if it satisfies the following conditions:

- The free vertices in the bottom row start from the far right hand side and are lined up sequentially.
- The free vertices in the top row also start from the far right hand side and are lined up sequentially.
- The connected components that are solely in the bottom row are lined up sequentially, starting from the right hand side to the left of the free vertices, such that the vertices in each connected component are in consecutive order.
- The connected components that are solely in the top row are lined up sequentially, starting from the far left hand side, such that the vertices in each connected component are in consecutive order.

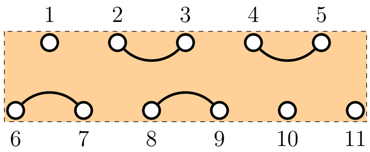
- The connected components between different rows are such that no two subsets cross.

Example 7.6. The $(5+6)\backslash 3$–diagram



(7.6)

is algorithmicallly planar. However, the $(5+6)\backslash 3$–diagram



(7.7)

is not algorithmically planar because the free vertex labelled 1 is not at the far right hand side of the top row.

Remark 7.7. It is clear that an algorithmically planar set partition diagram is **planar**, that is, none of the connected components in the diagram cross.

7.2 Multiplication Algorithm

In Algorithm 1, we outline a procedure called **MatrixMult** that performs the matrix multiplication of v by a spanning set element in $\mathrm{Hom}_{G(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ using the guiding principles that were given in the previous section. We assume that we have the set partition diagram that is associated with the spanning set element. Note that in the description of the algorithm, we have used d_π to represent a generic (k,l)–partition diagram; however, the type of set partition diagrams that we can use as input depends entirely upon the group $G(n)$. For example, if $G(n) = O(n)$, then only (k,l)–Brauer diagrams are valid as inputs to the procedure.

We now describe each of the subprocedures that appear in Algorithm 1 in more detail.

Factor takes as input a (k,l)–partition diagram that is appropriate for the group $G(n)$. The procedure uses the string-like property of these diagrams to output three diagrams whose composition is equivalent to the original input diagram. The first is a (k,k)–partition diagram that corresponds to a permutation $\sigma_k \in S_k$. Specifically, if i is a vertex in the top row, then the vertex in the bottom row that is connected to it is $k + \sigma_k(i)$. The second is a (k,l)–partition diagram that is algorithmically planar.

Algorithm 1: MatrixMult$(G(n), d_\pi, v)$

Inputs:

- $G(n)$ is a group.
- d_π is an appropriate (k, l)–partition diagram for $G(n)$.
- $v \in (\mathbb{R}^n)^{\otimes k}$.

Perform the following steps:

1. $\sigma_k, d_\pi, \sigma_l \leftarrow$ **Factor**$(G(n), d_\pi)$
2. $v \leftarrow$ **Permute**(v, σ_k)
3. $w \leftarrow$ **PlanarMult**$(G(n), d_\pi, v)$
4. $w \leftarrow$ **Permute**(w, σ_l)

Output: $w \in (\mathbb{R}^n)^{\otimes l}$.

The third is an (l, l)–partition diagram which can be interpreted as a permutation $\sigma_l \in S_l$ in the same way as the first diagram, replacing k with l.

Permute takes as input a vector $w \in (\mathbb{R}^n)^{\otimes m}$, for some n, m, that is expressed in the standard basis of $\mathbb{R}^n$, and a permutation σ in S_m, and outputs another vector in $(\mathbb{R}^n)^{\otimes m}$ where only the indices of the basis vectors — and not the indices of the coefficients of w — have been permuted according to σ. Said differently, **Permute** performs the following operation, which is extended linearly:

$$\sigma \cdot w_I e_I := w_I(e_{i_{\sigma(1)}} \otimes e_{i_{\sigma(2)}} \otimes \cdots \otimes e_{i_{\sigma(m)}}) \tag{7.8}$$

Note that in Algorithm 1, m will be equal to either k or l.

PlanarMult takes as input an algorithmically planar set partition diagram of the form returned by **Factor** together with a vector, and performs a fast matrix multiplication on this vector. Since the set partition diagram is algorithmically planar, we first use the monoidal property of the category in which it is a morphism to decompose it as a tensor product of smaller set partition diagrams. Next, we apply the monoidal functor that is appropriate for $G(n)$ to express this tensor product of diagrams as a Kronecker product of smaller matrices. Finally, we perform matrix multiplication by applying these smaller matrices to the input vector from "right-to-left, diagram-by-diagram" — to be described in more detail for each group below — returning another vector as output.

Remark 7.8. In effect, the **Factor** procedure takes as input a (k, l)–partition diagram that is appropriate for the group $G(n)$ and swaps as many pairs of vertices as necessary in each row in order to obtain an algorithmically planar (k, l)–partition diagram. The composition of the swaps in each row is represented by a permutation diagram.

Remark 7.9. We note that the implementation of the **Factor** and **PlanarMult** procedures vary according to the group $G(n)$ and the type of set partition diagrams that correspond to $G(n)$, although they share many commonalities. We describe the implementation of these procedures for each of the groups below.

Remark 7.10. We also analyse the time complexity of the implementation of **MatrixMult** for each group $G(n)$ and compare it with the time complexity of the naïve implementation of matrix multiplication. Note that in our analysis, we are viewing memory operations, such as permuting basis vectors and making copies of coefficients, as having no cost. Hence, we view the procedures **Factor** and **Permute** as having no cost. As a result, it is enough to consider the computational cost of **PlanarMult** only.

7.2.1 Symmetric Group S_n

The implementation of Algorithm 1 that we give here for the symmetric group effectively recovers the algorithm that was presented in Godfrey et al. (2023, Appendix C); however, we take an entirely different approach that uses monoidal categories instead. The major difference in implementation between the two versions relates to how, in our terminology, the connected components between vertices in different rows of d_π are pulled into the middle diagram in **Factor**. We choose to make sure that the connected components do not cross, making the resulting diagram algorithmically planar, whereas in Godfrey et al. (2023) they choose to connect the vertices in different rows such that the left-most vertices in the top row of the new diagram connect to the right-most vertices in the bottom row, that is, they make the connected components cross in "opposites". While this does not make any significant difference in terms of performing the matrix multiplication for the symmetric group — indeed, there is only a difference in how the tensor indices are ordered — our decision to make the middle diagram in the composition algorithmically planar leads to significant performance improvements when we extend our approach to the other groups below, since for these groups, we will show that these operations reduce to the identity transformation.

Implementation

We provide specific implementations of **Factor** and **PlanarMult** for the symmetric group S_n.

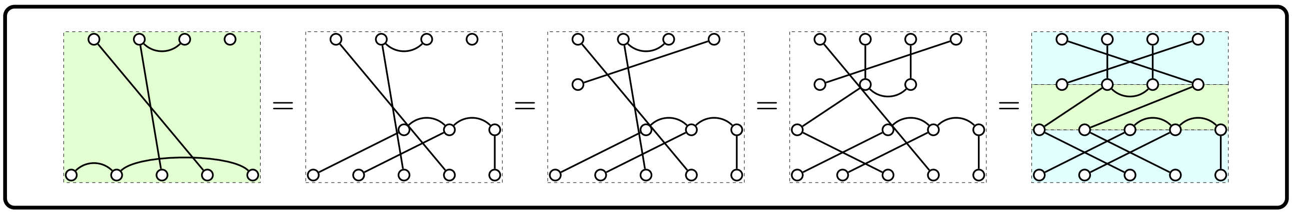


Figure 7.1: We use the string-like property of (k,l)–partition diagrams to **Factor** them as a composition of a permutation in S_k, an algorithmically planar (k,l)–partition diagram, and a permutation in S_l. Here $k = 5$ and $l = 4$.

Factor: The input is a (k,l)–partition diagram d_π. We drag and bend the strings representing the connected components of d_π to obtain three diagrams whose composition is equivalent to d_π: a (k,k)–partition diagram that represents a permutation σ_k in the symmetric group S_k; an algorithmically planar (k,l)–partition diagram; and a (l,l)–partition diagram that represents a permutation σ_l in the symmetric group S_l.

To obtain the algorithmically planar (k,l)–partition diagram, we drag and bend the strings in any way such that

- the connected components that are solely in the bottom row of d_π are pulled up to be next to each other in the far right hand side of the bottom row of the algorithmically planar (k,l)–partition diagram, such that the connected components are ordered by their size, in decreasing order from right to left,
- the connected components that are solely in the top row of d_π are pulled down to be next to each other in the far left hand side of the top row of the algorithmically planar (k,l)–partition diagram, and
- the connected components between vertices in different rows of d_π are bent to be in between the other vertices of the algorithmically planar (k,l)–partition diagram such that no two subsets of connected components cross.

We give an example of this procedure in Figure 7.1.

PlanarMult: We take as input the algorithmically planar (k,l)–partition diagram d_π having at most n blocks that is the output of **Factor**, and a vector $v \in (\mathbb{R}^n)^{\otimes k}$ that is the output of **Permute**, as per Algorithm 1.

Let t be the number of blocks that are solely in the top row of d_π, let d be the number of blocks that connect vertices in different rows of d_π, and let b be the number of blocks that are solely in the bottom row of d_π.

Given how **Factor** constructs the algorithmically planar (k, l)–partition diagram d_π, the set partition π corresponding to d_π will be of the form

$$\left(\bigcup_{i=1}^{t} T_i\right) \bigcup \left(\bigcup_{i=1}^{d} D_i\right) \bigcup \left(\bigcup_{i=1}^{b} B_i\right) \tag{7.9}$$

such that

$$|B_1| \leq |B_2| \leq \cdots \leq |B_b| \tag{7.10}$$

where we have used T_i to refer to a top row block, D_i to refer to a different row block, and B_i to refer to a bottom row block. If we let

$$D_i \coloneqq D_i^U \cup D_i^L \tag{7.11}$$

where D_i^U is the subset of D_i whose vertices are in the top row of d_π, and D_i^L is the subset of D_i whose vertices are in the bottom row of d_π, then we have that the vertices in each subset are given by

- $T_i = \left\{\sum_{j=1}^{i-1} |T_j| + 1, \dots, \sum_{j=1}^{i} |T_j|\right\}$ for all $i = 1 \to t$
- $D_i^U = \left\{\sum_{j=1}^{t} |T_j| + \sum_{j=1}^{i-1} |D_j^U| + 1, \dots, \sum_{j=1}^{t} |T_j| + \sum_{j=1}^{i} |D_j^U|\right\}$ for all $i = 1 \to d$,
- $D_i^L = \left\{l + \sum_{j=1}^{i-1} |D_j^L| + 1, \dots, l + \sum_{j=1}^{i} |D_j^L|\right\}$ for all $i = 1 \to d$, and
- $B_i = \left\{l + \sum_{j=1}^{d} |D_j^L| + \sum_{j=1}^{i-1} |B_j| + 1, \dots, l + \sum_{j=1}^{d} |D_j^L| + \sum_{j=1}^{i} |B_j|\right\}$ for all $i = 1 \to b$.

Note, in particular, that

$$l = \sum_{j=1}^{t} |T_j| + \sum_{j=1}^{d} |D_j^U| \tag{7.12}$$

and

$$k = \sum_{j=1}^{d} |D_j^L| + \sum_{j=1}^{b} |B_j| \tag{7.13}$$

Next, we take d_π and express it as a tensor product of three types of set partition diagrams. The right-most type is itself a tensor product of diagrams corresponding to the B_i; the middle type is a single diagram corresponding to $\bigcup_{i=1}^{d} D_i$; and the left-most type is itself a tensor product of diagrams corresponding to the T_i.

We apply the monoidal functor Θ to this tensor product decomposition of diagrams, which returns a Kronecker product of matrices. We perform the matrix multiplication by applying the matrices "right-to-left, diagram-by-diagram", as follows.

Step 1: Apply each matrix corresponding to a bottom row block diagram, one-by-one, starting from the one that corresponds to B_b and ending with the one that corresponds to B_1.

Suppose that we are performing the part of the matrix multiplication that corresponds to B_i, for some $i = 1 \rightarrow b$. The input will be a vector $w \in (\mathbb{R}^n)^{\otimes k - \sum_{j=i+1}^{b} |B_j|}$.

We can express w in the standard basis of $\mathbb{R}^n$ as

$$w = \sum_{L \in [n]^{k - \sum_{j=i+1}^{b} |B_j|}} w_L e_L \tag{7.14}$$

This will be mapped to the vector $r \in (\mathbb{R}^n)^{\otimes k - \sum_{j=i}^{b} |B_j|}$, where r is of the form

$$r = \sum_{M \in [n]^{k - \sum_{j=i}^{b} |B_j|}} r_M e_M \tag{7.15}$$

and

$$r_M = \sum_{j \in [n]} w_{M,j,\dots,j} \tag{7.16}$$

where the number of indices j in (7.16) is $|B_i|$.

At the end of this process, we obtain a vector in $(\mathbb{R}^n)^{\otimes k - \sum_{j=1}^{b} |B_j|}$.

Note that the matrices corresponding to bottom row connected components are merely performing indexing and summation operations, that is, ultimately, tensor contractions.

Step 2: Now apply the matrix corresponding to the middle diagram, that is, to the set $\bigcup_{i=1}^{d} D_i$.

The input will be a vector $w \in (\mathbb{R}^n)^{\otimes k - \sum_{j=1}^{b} |B_j|}$. Using (7.13), we can express w in the standard basis of $\mathbb{R}^n$ as

$$w = \sum_{j_1, \dots, j_d \in [n]} w_{j_1,\dots,j_1,j_2,\dots,j_2,\dots,j_d,\dots,j_d} \bigotimes_{k=1}^{d} \left(\bigotimes_{p=1}^{|D_k^L|} e_{j_k} \right) \tag{7.17}$$

where each index j_k in the coefficient is repeated $|D_k^L|$ times.

This will be mapped to the vector $r \in (\mathbb{R}^n)^{\otimes \sum_{j=i}^{d} |D_j^U|}$, where r is of the form

$$r = \sum_{j_1,\dots,j_d \in [n]} r_{j_1,\dots,j_1,j_2,\dots,j_2,\dots,j_d,\dots,j_d} \bigotimes_{k=1}^{d} \left(\bigotimes_{p=1}^{|D_k^U|} e_{j_k} \right) \tag{7.18}$$

where each index j_k in the coefficient is repeated $|D_k^U|$ times, and

$$r_{j_1,\dots,j_1,j_2,\dots,j_2,\dots,j_d,\dots,j_d} = w_{j_1,\dots,j_1,j_2,\dots,j_2,\dots,j_d,\dots,j_d} \tag{7.19}$$

These operations are called transfer operations, a term that first appeared in Pan and Kondor (2022).

Step 3: Finally, apply each matrix corresponding to a top row block diagram, one-by-one, starting from the one that corresponds to T_t and ending with the one that corresponds to T_1.

Suppose that we are performing the part of the matrix multiplication that corresponds to T_i, for some $i = 1 \to t$.

Then we begin with a vector $x \in (\mathbb{R}^n)^{\otimes l - \sum_{j=1}^{i} |T_j|}$ that is of the form

$$x = \sum_{l_{i+1},\dots,l_t \in [n]} \sum_{j_1,\dots,j_d \in [n]} x_{j_1,j_2,\dots,j_d} \bigotimes_{q=i+1}^{t} \left(\bigotimes_{m=1}^{|T_q|} e_{l_q} \right) \bigotimes_{k=1}^{d} \left(\bigotimes_{p=1}^{|D_k^U|} e_{j_k} \right) \tag{7.20}$$

where $x_{j_1,j_2,\dots,j_d}$ is the coefficient $r_{j_1,\dots,j_1,j_2,\dots,j_2,\dots,j_d,\dots,j_d}$ appearing in (7.19).

This will be mapped to the vector $y \in (\mathbb{R}^n)^{\otimes l - \sum_{j=1}^{i-1} |T_j|}$, where y is of the form

$$y = \sum_{l_i,l_{i+1},\dots,l_t \in [n]} \sum_{j_1,\dots,j_d \in [n]} x_{j_1,j_2,\dots,j_d} \bigotimes_{q=i}^{t} \left(\bigotimes_{m=1}^{|T_q|} e_{l_q} \right) \bigotimes_{k=1}^{d} \left(\bigotimes_{p=1}^{|D_k^U|} e_{j_k} \right) \tag{7.21}$$

At the end of this process, we obtain a vector in $(\mathbb{R}^n)^{\otimes l}$, by (7.12), which is of the form

$$\sum_{l_1,\dots,l_t \in [n]} \sum_{j_1,\dots,j_d \in [n]} x_{j_1,j_2,\dots,j_d} \bigotimes_{q=1}^{t} \left(\bigotimes_{m=1}^{|T_q|} e_{l_q} \right) \bigotimes_{k=1}^{d} \left(\bigotimes_{p=1}^{|D_k^U|} e_{j_k} \right) \tag{7.22}$$

This is the vector that is returned by **PlanarMult** for the symmetric group S_n.

To summarise, the implementation of **PlanarMult** for the symmetric group S_n takes as input the algorithmically planar (k,l)–partition diagram that comes from **Factor** and expresses it as a tensor product of three types of set partition diagrams. The right-most type is itself a tensor product of set

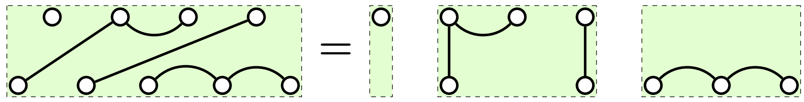


Figure 7.2: The decomposition of the algorithmically planar $(5, 4)$–partition diagram that appears in Figure 7.1 into a tensor product of smaller partition diagrams. These diagrams correspond, from right-to-left, to tensor contraction, transfer, and copying operations under the functor Θ. This tensor product decomposition is used in **PlanarMult** for the symmetric group S_n.

partition diagrams having only vertices in the bottom row, where each diagram represents a single block. These diagrams correspond to tensor contraction operations under the functor Θ. The middle type is a set partition diagram that consists of all of the connected components in the algorithmically planar (k, l)–partition diagram between vertices in different rows. These diagrams correspond to transfer operations under the functor Θ. The left-most type is itself a tensor product of set partition diagrams having only vertices in the top row, where each diagram represents a single block. These diagrams correspond to indexing operations that perform copies under the functor Θ.

Figure 7.2 gives the tensor product decomposition of the algorithmically planar $(5, 4)$–partition diagram that is shown in Figure 7.1.

To show diagrammatically how the matrix multiplication is performed, we see that the tensor product of the three types of set partition diagrams corresponds to a Kronecker product of smaller matrices under the functor Θ, defined in Theorem 6.33, by the monoidal property of Θ. We would like to apply each smaller matrix to the input vector from right-to-left, diagram-by-diagram. To do this, we first deform the entire tensor product decomposition of set partition diagrams by pulling each individual diagram up one level higher than the previous one, going from right-to-left, and then apply the functor Θ at each level. The newly inserted strings correspond to an identity matrix, hence only the matrices corresponding to the original tensor product decomposition act on the input vector at each stage.

Figure 7.3 gives an example of how the computation takes place at each stage for the tensor product decomposition given in Figure 7.2, using its equivalent diagram form.

Example 7.11. Suppose that we wish to perform the multiplication of D_π by $v \in (\mathbb{R}^n)^{\otimes 5}$, where D_π corresponds to the $(5, 4)$–partition diagram d_π given in Figure 7.1 under Θ, and v is given by

$$\sum_{L \in [n]^5} v_L e_L \tag{7.23}$$

We know that D_π is a matrix in $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes 5}, (\mathbb{R}^n)^{\otimes 4})$.

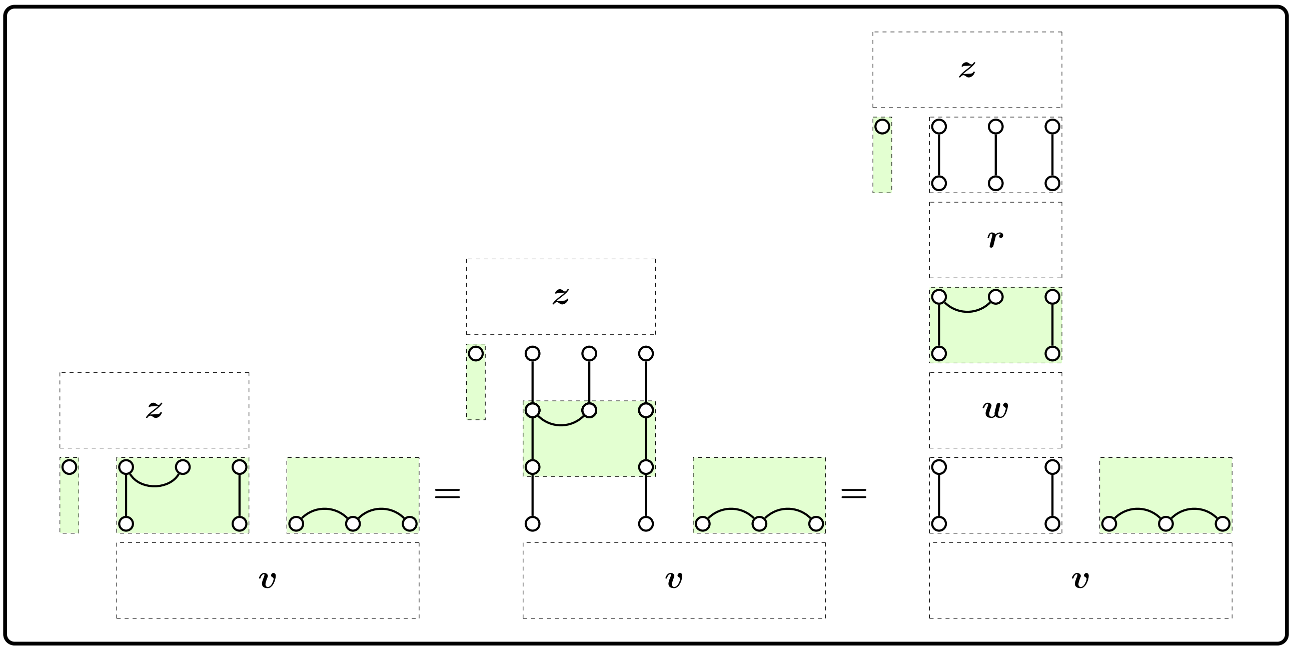


Figure 7.3: We show how matrix multiplication is implemented in **PlanarMult** for S_n using the tensor product decomposition of the algorithmically planar $(5, 4)$–partition diagram given in Figure 7.2 as an example. We perform the matrix multiplication as follows: first, we deform the entire tensor product decomposition diagram by pulling each individual diagram up one level higher than the previous one, going from right-to-left, and then we apply the functor Θ at each level. Finally, we perform matrix multiplication at each level to obtain the final output vector.

First, we apply the procedure **Factor**, which returns the three diagrams given in Figure 7.1. The first diagram corresponds to the permutation $(13)(24)$ in S_5; hence, the result of **Permute**$(v, (13)(24))$ is the vector

$$\sum_{L\in[n]^5} v_{l_1,l_2,l_3,l_4,l_5} \left(e_{l_3} \otimes e_{l_4} \otimes e_{l_1} \otimes e_{l_2} \otimes e_{l_5}\right) \tag{7.24}$$

Next we apply **PlanarMult** with the decomposition given in Figure 7.2.

Step 1: Apply the matrices that correspond to the bottom row blocks.

We obtain the vector

$$w = \sum_{l_3,l_4\in[n]} w_{l_3,l_4} \left(e_{l_3} \otimes e_{l_4}\right) \tag{7.25}$$

where

$$w_{l_3,l_4} = \sum_{j\in[n]} v_{j,j,l_3,l_4,j} \tag{7.26}$$

Step 2: Apply the matrices that correspond to the middle diagram.

We obtain the vector

$$r = \sum_{l_3 \in [n]} \sum_{l_4 \in [n]} r_{l_3,l_3,l_4} \left(e_{l_3} \otimes e_{l_3} \otimes e_{l_4}\right) \tag{7.27}$$

where

$$r_{l_3,l_3,l_4} = w_{l_3,l_4} \tag{7.28}$$

Step 3: Apply the matrices that correspond to the top row blocks.

We obtain the vector

$$z = \sum_{m \in [n]} \sum_{l_3 \in [n]} \sum_{l_4 \in [n]} r_{l_3,l_3,l_4} \left(e_m \otimes e_{l_3} \otimes e_{l_3} \otimes e_{l_4}\right) \tag{7.29}$$

Substituting in, we get that

$$z = \sum_{m \in [n]} \sum_{l_3 \in [n]} \sum_{l_4 \in [n]} w_{l_3,l_4} \left(e_m \otimes e_{l_3} \otimes e_{l_3} \otimes e_{l_4}\right) \tag{7.30}$$

and hence

$$z = \sum_{m \in [n]} \sum_{l_3 \in [n]} \sum_{l_4 \in [n]} \sum_{j \in [n]} v_{j,j,l_3,l_4,j} \left(e_m \otimes e_{l_3} \otimes e_{l_3} \otimes e_{l_4}\right) \tag{7.31}$$

Finally, as the third diagram returned from **Factor** corresponds to the permutation (14) in S_4, we perform **Permute**$(z, (14))$, which returns the vector

$$z = \sum_{m \in [n]} \sum_{l_3 \in [n]} \sum_{l_4 \in [n]} \sum_{j \in [n]} v_{j,j,l_3,l_4,j} \left(e_{l_4} \otimes e_{l_3} \otimes e_{l_3} \otimes e_m\right) \tag{7.32}$$

This is the vector that is returned by **MatrixMult**.

Time Complexity

We look at each step of the implementation that was given above.

Step 1: For the part of the matrix multiplication that corresponds to B_i, for some $i = 1 \to b$, we map a vector in $(\mathbb{R}^n)^{\otimes k - \sum_{j=i+1}^{b} |B_j|}$ to a vector in $(\mathbb{R}^n)^{\otimes k - \sum_{j=i}^{b} |B_j|}$.

Since the matrix corresponds to a bottom row block, for each tuple M of indices in the output coefficient r_M, as in (7.16), there are only n terms to multiply (an improvement over $n^{|B_i|}$), and consequently only $n - 1$ additions.

Hence, in total, there are

$$\sum_{i=1}^{b} n^{k-\sum_{j=b+1-i}^{b}|B_j|} \cdot n \tag{7.33}$$

multiplications and

$$\sum_{i=1}^{b} n^{k-\sum_{j=b+1-i}^{b}|B_j|} \cdot (n-1) \tag{7.34}$$

additions. Note that, in calculating the overall time complexity for this step, the highest order term is given by the size of B_b. Hence, the worst case occurs when $|B_b| = 1$, giving an overall time complexity of $O(n^k)$. However, assuming that Step 1 must take place, the best case occurs when $|B_b| = k$, that is, there is only one bottom row block of size k, giving an overall time complexity of $O(n)$.

Step 2: As these are transfer operations, there is no cost since we are simply copying elements from an array.

Step 3: Here we are copying arrays, hence there is no cost.

Consequently, in the worst case, we have reduced the overall time complexity from $O(n^{l+k})$ to $O(n^k)$. If Step 1 must occur, then in the best case, we have reduced the overall time complexity from $O(n^{l+k})$ to $O(n)$. However, the true best case occurs when Step 1 does not take place at all, that is, when the number of bottom row blocks b is zero, because, in this case, the computation is effectively free!

This analysis also shows why, in Definition 7.1 of an algorithmically planar (k, l)–partition diagram, we have chosen to order the connected components that are solely in the bottom row by their size, in decreasing order from right to left. We want to obtain the best overall time complexity for the multiplication algorithm, which, for the symmetric group, is determined by the number of additions and multiplications that occur in Step 1. It is clear from (7.33) and (7.34) that the time complexity is best when the bottom row blocks are ordered in decreasing size order from right to left.

7.2.2 Orthogonal Group $O(n)$

The implementation of Algorithm 1 for the orthogonal group $O(n)$ is similar to the implementation for the symmetric group S_n as a result of the similarity between the monoidal functors that are associated with each group. In fact, the implementation for the orthogonal group will be simpler because the only valid set partition diagrams that can be input into the **MatrixMult** procedure for this group are Brauer diagrams.

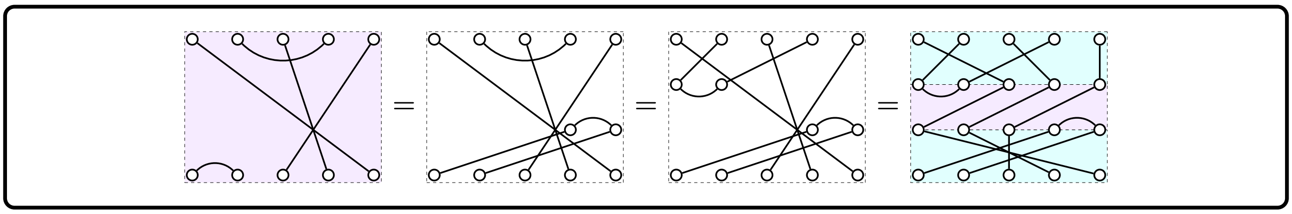


Figure 7.4: We use the string-like aspect of (k,l)–Brauer diagrams to **Factor** them as a composition of a permutation in S_k, an algorithmically planar (k,l)–Brauer diagram, and a permutation in S_l. Here $k = l = 5$.

Implementation

We provide specific implementations of **Factor** and **PlanarMult** for the orthogonal group $O(n)$.

Factor: The input is a (k,l)–Brauer diagram d_β. We drag and bend the strings representing the connected components of d_β to obtain a factoring of d_β into three diagrams whose composition is equivalent to d_β: a (k,k)–Brauer diagram that represents a permutation σ_k in the symmetric group S_k; another (k,l)–Brauer diagram that is algorithmically planar; and a (l,l)–Brauer diagram that represents a permutation σ_l in the symmetric group S_l.

To obtain the algorithmically planar (k,l)–Brauer diagram, we drag and bend the strings in any way such that

- the pairs that are solely in the bottom row of d_β are pulled up to be next to each other in the far right hand side of the bottom row of the algorithmically planar (k,l)–Brauer diagram,
- the pairs that are solely in the top row of d_β are pulled down to be next to each other in the far left hand side of the top row of the algorithmically planar (k,l)–Brauer diagram, and
- the pairs between vertices in different rows of d_β are bent to be in between the other vertices of the algorithmically planar (k,l)–Brauer diagram such that no two pairings in the algorithmically planar diagram intersect each other.

We give an example of this procedure in Figure 7.4.

PlanarMult: We take as input the planar (k,l)–Brauer diagram d_β that is the output of **Factor**, and a vector $v \in (\mathbb{R}^n)^{\otimes k}$ that is the output of **Permute**, as per Algorithm 1.

Let t be the number of pairs that are solely in the top row of d_β, let d be the number of pairs that are in different rows of d_β, and let b be the number of pairs that are solely in the bottom row of d_β.

Then it is clear that

$$2t + d = l \quad \text{and} \quad 2b + d = k \tag{7.35}$$

and so

$$2t + 2d + 2b = l + k \tag{7.36}$$

Given how **Factor** constructs the planar (k, l)–Brauer diagram d_β, the Brauer partition β corresponding to d_β will be of the form

$$\left(\bigcup_{i=1}^{t} T_i\right) \bigcup \left(\bigcup_{i=1}^{d} D_i\right) \bigcup \left(\bigcup_{i=1}^{b} B_i\right) \tag{7.37}$$

where we have used T_i to refer to a top row pair, D_i to refer to a different row pair, and B_i to refer to a bottom row pair. In particular, we have that

- $T_i = \{2i - 1, 2i\}$ for all $i = 1 \to t$
- $D_i = \{l + i - d, l + i\}$ for all $i = 1 \to d$, and
- $B_i = \{l + k - 2b + 2i - 1, l + k - 2b + 2i\}$ for all $i = 1 \to b$.

Next, we take d_β and express it as a tensor product of three types of Brauer diagrams. The right-most type is itself a tensor product of diagrams corresponding to the B_i; the middle type is a single diagram corresponding to $\bigcup_{i=1}^{d} D_i$; and the left-most type is itself a tensor product of diagrams corresponding to the T_i.

We apply the monoidal functor Φ to this tensor product decomposition of diagrams, which returns a Kronecker product of matrices. We perform the matrix multiplication by applying the matrices "right-to-left, diagram-by-diagram", as follows.

Step 1: Apply each matrix corresponding to a bottom row pair diagram, one-by-one, starting from the one that corresponds to B_b and ending with the one that corresponds to B_1.

Suppose that we are performing the part of the matrix multiplication that corresponds to B_i, for some $i = 1 \to b$. The input will be a vector $w \in (\mathbb{R}^n)^{\otimes k-2(b-i)}$.

We can express w in the standard basis of $\mathbb{R}^n$ as

$$w = \sum_{L \in [n]^{k-2(b-i)}} w_L e_L \tag{7.38}$$

This will be mapped to the vector $r \in (\mathbb{R}^n)^{\otimes k-2(b-i)-2}$, where r is of the form

$$r = \sum_{M \in [n]^{k-2(b-i)-2}} r_M e_M \tag{7.39}$$

and

$$r_M = \sum_{j \in [n]} w_{M,j,j} \tag{7.40}$$

At the end of this process, we obtain a vector in $(\mathbb{R}^n)^{\otimes k-2b}$.

As before, the matrices corresponding to bottom row pairs are merely performing tensor contractions.

Step 2: Now apply the matrix corresponding to the middle diagram, that is, to the set $\bigcup_{i=1}^{d} D_i$.

The input will be a vector $w \in (\mathbb{R}^n)^{\otimes k-2b}$.

Expressing w in the standard basis of $\mathbb{R}^n$ as

$$w = \sum_{L \in [n]^{k-2b}} w_L e_L \tag{7.41}$$

the multiplication of the matrix merely returns w itself!

Hence, the transfer operations for the orthogonal group are simply the identity map.

Step 3: Finally, apply each matrix corresponding to a top row pair diagram, one-by-one, starting from the one that corresponds to T_t and ending with the one that corresponds to T_1.

Suppose that we are performing the part of the matrix multiplication that corresponds to T_i, for some $i = 1 \rightarrow t$.

Then we begin with a vector $w \in (\mathbb{R}^n)^{\otimes k-2b+2(t-i)}$ that is of the form

$$w = \sum_{J \in [n]^{t-i}} \sum_{L \in [n]^{k-2b}} v_L \left(e_{j_1} \otimes e_{j_1} \otimes \cdots \otimes e_{j_{t-i}} \otimes e_{j_{t-i}} \otimes e_L\right) \tag{7.42}$$

where v_L is the coefficient of e_L appearing in the vector at the end of Step 2.

This will be mapped to the vector $r \in (\mathbb{R}^n)^{\otimes k-2b+2(t-i)+2}$, where r is of the form

$$r = \sum_{m \in [n]} \sum_{J \in [n]^{t-i}} \sum_{L \in [n]^{k-2b+2(t-i)}} v_L \left(e_m \otimes e_m \otimes e_{j_1} \otimes e_{j_1} \otimes \cdots \otimes e_{j_{t-i}} \otimes e_{j_{t-i}} \otimes e_L\right) \tag{7.43}$$

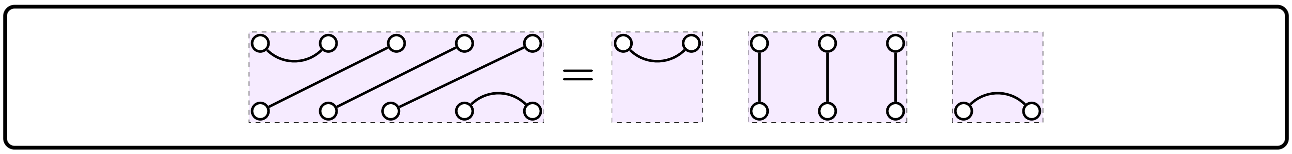


Figure 7.5: The tensor product decomposition of the planar $(5,5)$–Brauer diagram that appears in Figure 7.4.

At the end of this process, we obtain a vector in $(\mathbb{R}^n)^{\otimes l}$, since $k - 2b + 2t = l$, which is of the form

$$\sum_{J \in [n]^t} \sum_{L \in [n]^{k-2b}} v_L \left(e_{j_1} \otimes e_{j_1} \otimes \cdots \otimes e_{j_t} \otimes e_{j_t} \otimes e_L\right) \tag{7.44}$$

This is the vector that is returned by **PlanarMult** for the orthogonal group $O(n)$.

To summarise, the implementation of **PlanarMult** for the orthogonal group $O(n)$ takes as input the algorithmically planar (k,l)–Brauer diagram that comes from **Factor** and expresses it as a tensor product of three types of Brauer diagrams. The right-most type is itself a tensor product of Brauer diagrams, where each diagram has only two connected vertices in the bottom row. These diagrams correspond to tensor contraction operations under the functor Φ given in Theorem 6.34. The middle type is a Brauer diagram that consists of all of the pairs in the planar (k,l)–Brauer diagram between vertices in different rows. This diagram corresponds to the identity under the functor Φ. The left-most type is a tensor product of Brauer diagrams having only two connected vertices in the top row. These diagrams correspond to indexing operations that perform copies under the functor Φ.

Figure 7.5 gives the tensor product decomposition of the algorithmically planar $(5,5)$–Brauer diagram that is shown in Figure 7.4.

The diagrammatic representation of the matrix multiplication step is very similar to the one for the symmetric group, in that to obtain the matrices we perform the same deformation of the tensor product decomposition of diagrams before applying the functor Φ at each level. We give an example in Figure 7.6 of how the computation takes place at each stage for the tensor product decomposition given in Figure 7.5, using its equivalent diagram form.

Example 7.12. Suppose that we wish to perform the multiplication of E_β by $v \in (\mathbb{R}^n)^{\otimes 5}$, where E_β corresponds to the $(5,5)$–Brauer diagram d_β given in Figure 7.4 under Φ, and v is given by

$$\sum_{L \in [n]^5} v_L e_L \tag{7.45}$$

We know that E_β is a matrix in $\mathrm{Hom}_{O(n)}((\mathbb{R}^n)^{\otimes 5}, (\mathbb{R}^n)^{\otimes 5})$.

First, we apply the procedure **Factor**, which returns the three diagrams given in Figure 7.4. The first diagram corresponds to the permutation (1524) in S_5, hence, the result of **Permute**$(v, (1524))$ is the vector

$$\sum_{L \in [n]^5} v_{l_1,l_2,l_3,l_4,l_5} \left(e_{l_5} \otimes e_{l_4} \otimes e_{l_3} \otimes e_{l_1} \otimes e_{l_2}\right) \tag{7.46}$$

Now we apply **PlanarMult** with the decomposition given in Figure 7.5.

Step 1: Apply the matrices that correspond to the bottom row pairs.

We obtain the vector

$$w = \sum_{l_5,l_4,l_3 \in [n]} w_{l_5,l_4,l_3} \left(e_{l_5} \otimes e_{l_4} \otimes e_{l_3}\right) \tag{7.47}$$

where

$$w_{l_5,l_4,l_3} = \sum_{j \in [n]} v_{j,j,l_3,l_4,l_5} \tag{7.48}$$

Step 2: Apply the matrices that correspond to the middle diagram.

As the transfer operations are the identity map, we get w.

Step 3: Apply the matrices that correspond to the top row.

We obtain the vector

$$z = \sum_{m \in [n]} \sum_{l_5,l_4,l_3 \in [n]} w_{l_5,l_4,l_3} \left(e_m \otimes e_m \otimes e_{l_5} \otimes e_{l_4} \otimes e_{l_3}\right) \tag{7.49}$$

Substituting in, we get that

$$z = \sum_{m \in [n]} \sum_{l_5,l_4,l_3 \in [n]} \sum_{j \in [n]} v_{j,j,l_3,l_4,l_5} \left(e_m \otimes e_m \otimes e_{l_5} \otimes e_{l_4} \otimes e_{l_3}\right) \tag{7.50}$$

Finally, as the third diagram returned from **Factor** corresponds to the permutation (1342) in S_5, we perform **Permute**$(z, (1342))$, which returns the vector

$$\sum_{m \in [n]} \sum_{l_5,l_4,l_3 \in [n]} \sum_{j \in [n]} v_{j,j,l_3,l_4,l_5} \left(e_{l_5} \otimes e_m \otimes e_{l_4} \otimes e_m \otimes e_{l_3}\right) \tag{7.51}$$

This is the vector that is returned by **MatrixMult**.

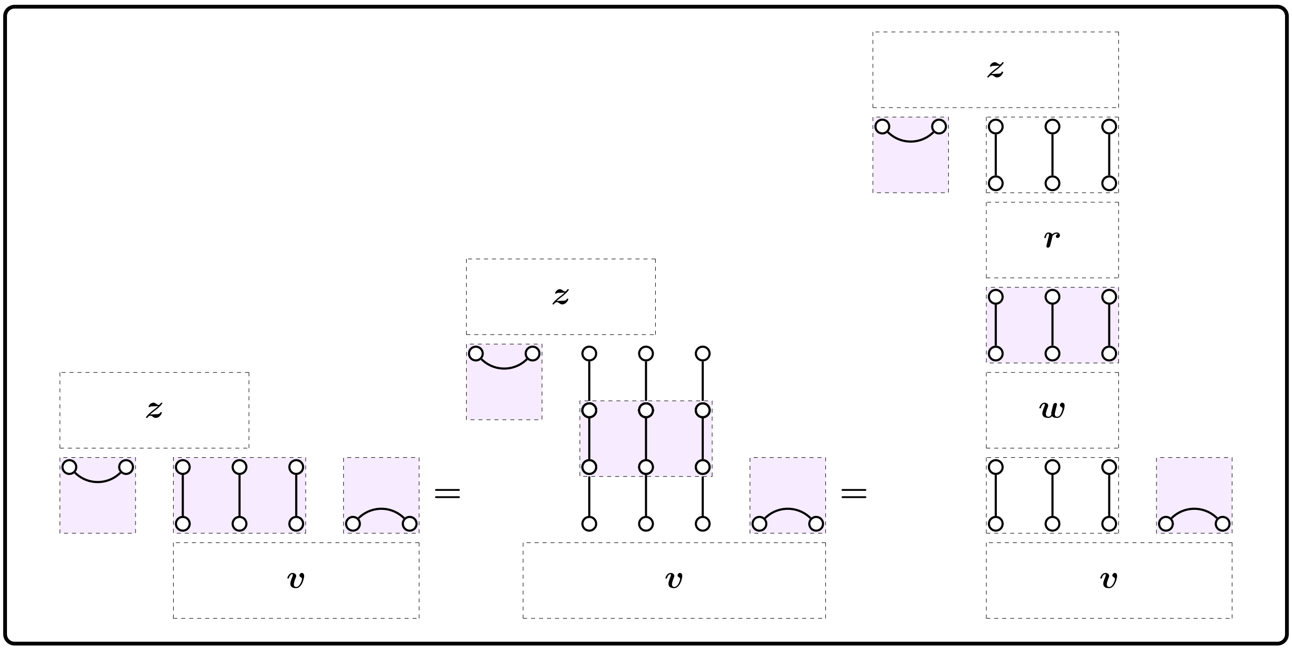


Figure 7.6: We show how matrix multiplication is implemented in **PlanarMult** for $O(n), Sp(n)$ and $SO(n)$ using the tensor product decomposition of the planar $(5,5)$–Brauer diagram given in Figure 7.5 as an example. Effectively, we perform the matrix multiplication by applying the matrices "right-to-left, diagram-by-diagram". In reality, we perform the matrix multiplication as follows: first, we deform the entire tensor product decomposition diagram by pulling each individual diagram up one level higher than the previous one, going from right–to-left, and then we apply the functor that corresponds to the group at each level. Finally, we perform matrix multiplication at each level to obtain the final output vector.

Time Complexity

We look at each step of the implementation that was given above.

Step 1: For the part of the matrix multiplication that corresponds to B_i, for some $i = 1 \to b$, we map a vector in $(\mathbb{R}^n)^{\otimes k-2(b-i)}$ to a vector in $(\mathbb{R}^n)^{\otimes k-2(b-i)-2}$.

Since the matrix corresponds to a bottom row pair that is connected, for each tuple M of indices in the output coefficient r_M, as in (7.40), there are only n terms to multiply (an improvement over n^2), and consequently only $n-1$ additions.

Hence, in total, there are

$$\sum_{i=1}^{b} n^{k-2(b-i)-2} \cdot n \tag{7.52}$$

multiplications and

$$\sum_{i=1}^{b} n^{k-2(b-i)-2} \cdot (n-1) \tag{7.53}$$

additions, for an overall time complexity of $O(n^{k-1})$.

Step 2: This corresponds to the identity transformation, hence there is no cost, as we do not need to perform this operation.

Step 3: Here we are copying arrays, hence there is no cost.

Consequently, we have reduced the overall time complexity from $O(n^{l+k})$ to $O(n^{k-1})$.

7.2.3 Symplectic Group $Sp(n)$

The implementation of Algorithm 1 for the symplectic group $Sp(n)$ is related to the implementation for the orthogonal group $O(n)$ because the only valid set partition diagrams that can be input into the procedure in each case are Brauer diagrams. The difference in the implementations comes from the difference in the monoidal functor that is associated with each group.

Implementation

The implementation of the **Factor** procedure is the same as for the orthogonal group.

PlanarMult: Again, we take as input the planar (k, l)–Brauer diagram d_β that is the output of **Factor**, and a vector $v \in (\mathbb{R}^n)^{\otimes k}$ that is the output of **Permute**, as per Algorithm 1.

The tensor product decomposition of d_β is the same as for the orthogonal group $O(n)$; in particular, the Brauer partition β corresponding to d_β is of the form (7.37). The important difference here is that we apply the monoidal functor X, instead of Φ, to this tensor product decomposition of diagrams, which returns a different Kronecker product of matrices. Again, we perform the same steps for the matrix multiplication, but this time the vectors that are returned at each stage are different.

Step 1: Apply each matrix corresponding to a bottom row pair diagram, one-by-one, starting from the one that corresponds to B_b and ending with the one that corresponds to B_1.

Suppose that we are performing the part of the matrix multiplication that corresponds to B_i, for some $i = 1 \rightarrow b$. The input will be a vector $w \in (\mathbb{R}^n)^{\otimes k-2(b-i)}$.

We can express w in the standard basis of $\mathbb{R}^n$ as

$$w = \sum_{L \in [n]^{k-2(b-i)}} w_L e_L \tag{7.54}$$

This will be mapped to the vector $r \in (\mathbb{R}^n)^{\otimes k-2(b-i)-2}$ where r is of the form

$$r = \sum_{M \in [n]^{k-2(b-i)-2}} r_M e_M \tag{7.55}$$

where

$$r_M = \sum_{j_{k-2(b-i)-1}, j_{k-2(b-i)} \in [n]} \epsilon_{j_{k-2(b-i)-1}, j_{k-2(b-i)}} w_{M, j_{k-2(b-i)-1}, j_{k-2(b-i)}} \tag{7.56}$$

Recall that $\epsilon_{j_{k-2(b-i)-1}, j_{k-2(b-i)}}$ was defined in (2.8).

At the end of this process, we obtain a vector in $(\mathbb{R}^n)^{\otimes k-2b}$.

As before, the matrices corresponding to bottom row pairs are performing tensor contractions.

Step 2: Now apply the matrix corresponding to the middle diagram, that is, to the set $\bigcup_{i=1}^{d} D_i$.

This will be exactly the same as for the orthogonal group, by the definition of γ_{r_p, u_p} given in (5.77).

Step 3: Finally, apply each matrix corresponding to a top row pair diagram, one-by-one, starting from the one that corresponds to T_t and ending with the one that corresponds to T_1.

Suppose that we are performing the part of the matrix multiplication that corresponds to T_i, for some $i = 1 \to t$.

Then we begin with a vector $w \in (\mathbb{R}^n)^{\otimes k-2b+2(t-i)}$ that is of the form

$$w = \sum_{J \in [n]^{2(t-i)}} \sum_{L \in [n]^{k-2b}} \epsilon_J v_L \left(e_J \otimes e_L\right) \tag{7.57}$$

where

$$\epsilon_J \coloneqq \epsilon_{j_1, j_2} \dots \epsilon_{j_{2(t-i)-1}, j_{2(t-i)}} \tag{7.58}$$

and where v_L is the coefficient of e_L appearing in the vector at the end of Step 2.

This will be mapped to the vector $r \in (\mathbb{R}^n)^{\otimes k-2b+2(t-i)+2}$, where r is of the form

$$r = \sum_{M \in [n]^2} \sum_{J \in [n]^{2(t-i)}} \sum_{L \in [n]^{k-2b}} \epsilon_{M,J} v_L \left(e_M \otimes e_J \otimes e_L\right) \tag{7.59}$$

where

$$\epsilon_{M,J} \coloneqq \epsilon_{m_1,m_2}\epsilon_{j_1,j_2} \dots \epsilon_{j_{2(t-i)-1},j_{2(t-i)}} \tag{7.60}$$

At the end of this process, we obtain a vector in $(\mathbb{R}^n)^{\otimes l}$, since $k - 2b + 2t = l$, which is of the form

$$\sum_{J \in [n]^{2t}} \sum_{L \in [n]^{k-2b}} \epsilon_J v_L \left(e_J \otimes e_L\right) \tag{7.61}$$

where ϵ_J is redefined to be

$$\epsilon_{j_1,j_2} \dots \epsilon_{j_{2t-1},j_{2t}} \tag{7.62}$$

This is the vector that is returned by **PlanarMult** for the symplectic group $Sp(n)$.

Example 7.13. Suppose that we wish to perform the multiplication of F_β by $v \in (\mathbb{R}^n)^{\otimes 5}$, for the same $(5,5)$–Brauer diagram d_β given in Figure 7.4, and v is given by

$$\sum_{L \in [n]^5} v_L e_L \tag{7.63}$$

We know that F_β is a matrix in $\mathrm{Hom}_{Sp(n)}((\mathbb{R}^n)^{\otimes 5}, (\mathbb{R}^n)^{\otimes 5})$.

The **Factor** and **Permute** steps are the same as for Example 7.12, hence, prior to the **PlanarMult** step, we have the vector

$$\sum_{L \in [n]^5} v_{l_1,l_2,l_3,l_4,l_5} \left(e_{l_5} \otimes e_{l_4} \otimes e_{l_3} \otimes e_{l_1} \otimes e_{l_2}\right) \tag{7.64}$$

We now apply **PlanarMult** with the decomposition given in Figure 7.5.

Step 1: Apply the matrices that correspond to the bottom row pairs.

We obtain the vector

$$w = \sum_{l_5,l_4,l_3 \in [n]} w_{l_5,l_4,l_3} \left(e_{l_5} \otimes e_{l_4} \otimes e_{l_3}\right) \tag{7.65}$$

where

$$w_{l_5,l_4,l_3} = \sum_{j_1,j_2 \in [n]} \epsilon_{j_1,j_2} v_{j_1,j_2,l_3,l_4,l_5} \tag{7.66}$$

Step 2: Apply the matrices that correspond to the middle diagram.

As the transfer operations are the identity map, we get w.

Step 3: Apply the matrices that correspond to the top row.

We obtain the vector

$$z = \sum_{m_1,m_2 \in [n]} \sum_{l_5,l_4,l_3 \in [n]} \epsilon_{m_1,m_2} w_{l_5,l_4,l_3} \left(e_{m_1} \otimes e_{m_2} \otimes e_{l_5} \otimes e_{l_4} \otimes e_{l_3}\right) \tag{7.67}$$

Substituting in, we get that

$$z = \sum_{m_1,m_2 \in [n]} \sum_{l_5,l_4,l_3 \in [n]} \sum_{j_1,j_2 \in [n]} \epsilon_{m_1,m_2} \epsilon_{j_1,j_2} v_{j_1,j_2,l_3,l_4,l_5} \left(e_{m_1} \otimes e_{m_2} \otimes e_{l_5} \otimes e_{l_4} \otimes e_{l_3}\right) \tag{7.68}$$

Finally, as the third diagram returned from **Factor** corresponds to the permutation (1342) in S_5, we perform **Permute**$(z, (1342))$, which returns the vector

$$\sum_{m_1,m_2 \in [n]} \sum_{l_5,l_4,l_3 \in [n]} \sum_{j_1,j_2 \in [n]} \epsilon_{m_1,m_2} \epsilon_{j_1,j_2} v_{j_1,j_2,l_3,l_4,l_5} \left(e_{l_5} \otimes e_{m_1} \otimes e_{l_4} \otimes e_{m_2} \otimes e_{l_3}\right) \tag{7.69}$$

This is the vector that is returned by **MatrixMult**.

Time Complexity

The time complexity is exactly the same as for the orthogonal group.

7.2.4 Special Orthogonal Group $SO(n)$

We can perform matrix multiplication between either E_β or $H_\alpha \in \mathrm{Hom}_{SO(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ and $v \in (\mathbb{R}^n)^{\otimes k}$, where d_β is a (k, l)–Brauer diagram, and d_α is an $(l + k)\backslash n$–diagram. Given that the implementation for the E_β case is the same as for the orthogonal group, by Theorem 5.19, we only consider the H_α case below.

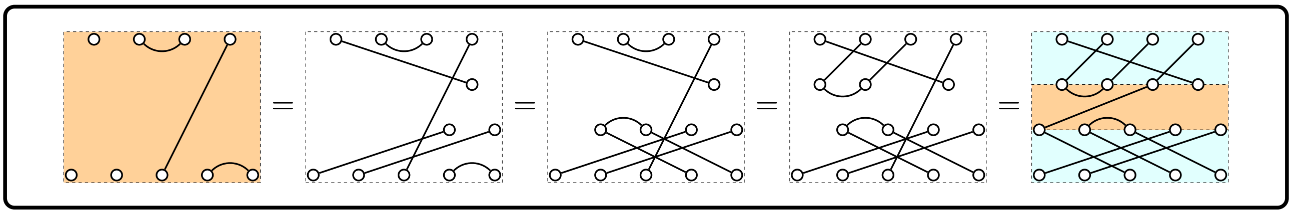


Figure 7.7: We use the string-like aspect of $(l+k)\backslash n$–diagrams to **Factor** them as a composition of a permutation in S_k, an algorithmically planar $(l+k)\backslash n$–diagram, and a permutation in S_l. Here $k=5$ and $l=4$.

Implementation

Factor: The input is an $(l+k)\backslash n$–diagram d_α. As before, we drag and bend the strings representing the connected components of d_α to obtain a factoring of d_α into the three diagrams, except this time the middle diagram will be an algorithmically planar $(l+k)\backslash n$–diagram.

To obtain the algorithmically planar $(l+k)\backslash n$–diagram we want to drag and bend the strings in any way such that

- the free vertices in the top row of d_α are pulled down to the far right of the top row of the algorithmically planar $(l+k)\backslash n$–diagram, maintaining their order,
- the free vertices in the bottom row of d_α are pulled up to the far right of the bottom row of the algorithmically planar $(l+k)\backslash n$–diagram, maintaining their order,
- the pairs in the bottom row of d_α are pulled up to be next to each other in the right hand side of the bottom row of the algorithmically planar $(l+k)\backslash n$–diagram, but next to and to the left of the free vertices in the bottom row of the algorithmically planar $(l+k)\backslash n$–diagram,
- the pairs in the top row of d_α are pulled down to be next to each other in the far left hand side of the top row of the algorithmically planar $(l+k)\backslash n$–diagram,
- the pairs connecting vertices in different rows of d_α are ordered in the algorithmically planar $(l+k)\backslash n$–diagram in between the other vertices such that no two pairings in the algorithmically planar diagram intersect each other.

We give an example of this procedure in Figure 7.7.

PlanarMult: We take as input the algorithmically planar $(l+k)\backslash n$–diagram d_α that is the output of **Factor**, and a vector $v \in (\mathbb{R}^n)^{\otimes k}$ that is the output of **Permute**, as per Algorithm 1.

We need new subsets and notation to consider the impact of the free vertices in the $(l+k)\backslash n$–diagram d_α on the implementation for **PlanarMult**.

As before, let t be the number of pairs that are solely in the top row of d_α, let d be the number of pairs that are in different rows of d_α, and let b be the number of pairs that are solely in the bottom row of d_α. Now, let s be the number of free vertices in the top row of d_α. Hence there are $n-s$ free vertices in the bottom row of d_α.

Then it is clear that

$$2t+d+s=l \quad \text{and} \quad 2b+d+n-s=k \tag{7.70}$$

and so

$$2t+2d+2b+n=l+k \tag{7.71}$$

Given how **Factor** constructs the algorithmically planar $(l+k)\backslash n$–diagram d_α, the set partition α corresponding to d_α will be of the form

$$\left(\bigcup_{i=1}^{t} T_i\right) \bigcup \left(\bigcup_{i=1}^{d} D_i\right) \bigcup \left(\bigcup_{i=1}^{s} TF_i\right) \bigcup \left(\bigcup_{i=1}^{b} B_i\right) \bigcup \left(\bigcup_{i=1}^{n-s} BF_i\right) \tag{7.72}$$

where we have used T_i to refer to a top row pair, D_i to refer to a different row pair, TF_i to refer to a top row free vertex, B_i to refer to a bottom row pair, and BF_i to refer to a bottom row free vertex. In particular, we have that

- $T_i = \{2i-1, 2i\}$ for all $i = 1 \to t$,
- $D_i = \{l+i-d-s, l+i\}$ for all $i = 1 \to d$,
- $TF_i = \{l-s+i\}$ for all $i = 1 \to s$,
- $B_i = \{l+d+2i-1, l+d+2i\}$ for all $i = 1 \to b$, and
- $BF_i = \{l+d+2b+i\}$ for all $i = 1 \to n-s$.

We take d_α and express it as a tensor product of four types of set partition diagrams.

The right-most type is a diagram consisting of all the free vertices. Consequently, it corresponds to $(\bigcup_{i=1}^{s} TF_i) \bigcup (\bigcup_{i=1}^{n-s} BF_i)$. The type to its left is itself a tensor product of diagrams corresponding to the B_i; the next type is a single diagram corresponding to $\left(\bigcup_{i=1}^{d} D_i\right)$; and, finally, the left-most type is itself a tensor product of diagrams corresponding to the T_i.

We now apply the monoidal functor Ψ to this tensor product decomposition of diagrams, which returns a Kronecker product of matrices. We perform the matrix multiplication by applying the matrices "right-to-left, diagram-by-diagram", as follows.

Step 1: Apply the matrix that corresponds to the free vertices, that is, to $(\bigcup_{i=1}^{s} TF_i) \bigcup (\bigcup_{i=1}^{n-s} BF_i)$.

The input will be a vector $v \in (\mathbb{R}^n)^{\otimes k}$.

As $k = 2b + d + (n - s)$, we can express v in the standard basis of $\mathbb{R}^n$ as

$$v = \sum_{J \in [n]^{2b+d}} \sum_{B \in [n]^{n-s}} v_{J,B} e_{J,B} \tag{7.73}$$

This will be mapped to the vector $w \in (\mathbb{R}^n)^{\otimes 2b+d+s}$, where w is of the form

$$w = \sum_{J \in [n]^{2b+d}} \sum_{T \in [n]^{s}} w_{J,T} e_{J,T} \tag{7.74}$$

where

$$w_{J,T} = \sum_{B \in [n]^{n-s}} v_{J,B} \det(e_{T,B}) \tag{7.75}$$

Step 2: Apply each matrix corresponding to a bottom row pair diagram, one-by-one, starting from the one that corresponds to B_b and ending with the one that corresponds to B_1.

This is exactly the same as Step 1 for the orthogonal group. We obtain a vector $y \in (\mathbb{R}^n)^{\otimes d+s}$

Step 3: Now apply the matrix corresponding to the middle diagram, that is, to the set $\left(\bigcup_{i=1}^{d} D_i\right)$.

This is exactly the same as Step 2 for the orthogonal group. We obtain a vector $r \coloneqq y \in (\mathbb{R}^n)^{\otimes d+s}$

Step 4: Finally, apply each matrix corresponding to a top row pair diagram, one-by-one, starting from the one that corresponds to T_t and ending with the one that corresponds to T_1.

This is exactly the same as Step 3 for the orthogonal group. We obtain a vector $z \in (\mathbb{R}^n)^{\otimes 2t+d+s}$. As $2t + d + s = l$ by (7.70), we have that $z \in (\mathbb{R}^n)^{\otimes l}$, as required.

To summarise, the implementation of **PlanarMult** for $SO(n)$ differs slightly from the implementation of **PlanarMult** for the groups that have come before. In the case where the spanning set element

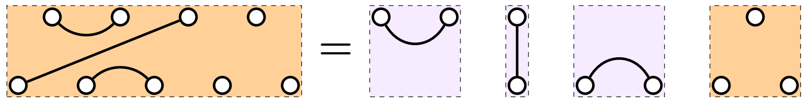


Figure 7.8: The tensor product decomposition of the algorithmically planar $(4+5)\backslash 3$–diagram that appears in Figure 7.7.

corresponds to a Brauer diagram, the implementation is the same as for the orthogonal group $O(n)$. Otherwise, the implementation takes as input the algorithmically planar $(l+k)\backslash n$–diagram that comes from **Factor**, but expresses it as a tensor product of *four* types of set partition diagrams. The right-most is a diagram consisting of all of the free vertices. This corresponds to the determinant map that is given in Definition 4.3. The three other types of diagrams that appear in the decomposition and their corresponding operations are exactly the same as for the orthogonal group $O(n)$.

Figure 7.8 gives an example of the tensor product decomposition for the algorithmically planar $(4+5)\backslash 3$–diagram given in Figure 7.7.

The diagrammatic representation of the matrix multiplication step is also very similar to the orthogonal group, in that to obtain the matrices we perform the same deformation of the tensor product decomposition of diagrams before applying the functor Ψ, defined in Theorem 6.36, at each level. Note, in particular, that we need to attach identity strings to the free vertices that appear in the top row.

We give an example in Figure 7.9 of how the computation takes place at each stage for the tensor product decomposition given in Figure 7.8, using its equivalent diagram form.

Example 7.14. Suppose that we wish to perform the multiplication of H_α by $v \in (\mathbb{R}^3)^{\otimes 5}$, where H_α corresponds to the $(4+5)\backslash 3$–diagram d_α given in Figure 7.7 under Ψ, and v is given by

$$\sum_{L \in [3]^5} v_L e_L \tag{7.76}$$

when expressed in the standard basis of $\mathbb{R}^3$.

We know that H_α is a matrix in $\mathrm{Hom}_{SO(3)}((\mathbb{R}^3)^{\otimes 5}, (\mathbb{R}^3)^{\otimes 4})$.

First, we apply the procedure **Factor**, which returns the three diagrams given in Figure 7.7. The first diagram corresponds to the permutation (13524) in S_5, hence, the result of **Permute**$(v, (13524))$ is the vector

$$\sum_{L \in [3]^5} v_{l_1,l_2,l_3,l_4,l_5} \left(e_{l_3} \otimes e_{l_4} \otimes e_{l_5} \otimes e_{l_1} \otimes e_{l_2} \right) \tag{7.77}$$

We now apply **PlanarMult** with the decomposition given in Figure 7.8.

Step 1: Apply the matrices that correspond to the free vertices.

We obtain the vector

$$w = \sum_{l_3,l_4,l_5,t_1 \in [3]} w_{l_3,l_4,l_5,t_1} \left(e_{l_3} \otimes e_{l_4} \otimes e_{l_5} \otimes e_{t_1} \right) \tag{7.78}$$

where

$$w_{l_3,l_4,l_5,t_1} = \sum_{l_1,l_2 \in [3]} v_{l_1,l_2,l_3,l_4,l_5} \det(e_{t_1} \otimes e_{l_1} \otimes e_{l_2}) \tag{7.79}$$

Step 2: Apply the matrices that correspond to the bottom row pairs.

We obtain the vector

$$y = \sum_{l_3,t_1 \in [3]} y_{l_3,t_1} \left(e_{l_3} \otimes e_{t_1} \right) \tag{7.80}$$

where

$$y_{l_3,t_1} = \sum_{j \in [3]} w_{l_3,j,j,t_1} \tag{7.81}$$

Step 3: Apply the matrices that correspond to the different row pairs.

Here, we get that $r = y$, as the transfer operations correspond to the identity map.

Step 4: Apply the matrices that correspond to the top row pairs.

We obtain the vector

$$z = \sum_{m \in [3]} \sum_{l_3,t_1 \in [3]} y_{l_3,t_1} \left(e_m \otimes e_m \otimes e_{l_3} \otimes e_{t_1} \right) \tag{7.82}$$

Substituting in, we get that

$$z = \sum_{m \in [3]} \sum_{l_3,t_1 \in [3]} \sum_{j \in [3]} w_{l_3,j,j,t_1} \left(e_m \otimes e_m \otimes e_{l_3} \otimes e_{t_1} \right) \tag{7.83}$$

and hence

$$z = \sum_{m \in [3]} \sum_{l_3,t_1 \in [3]} \sum_{j \in [3]} \sum_{l_1,l_2 \in [3]} v_{l_1,l_2,l_3,j,j} \det(e_{t_1} \otimes e_{l_1} \otimes e_{l_2}) \left(e_m \otimes e_m \otimes e_{l_3} \otimes e_{t_1} \right) \tag{7.84}$$

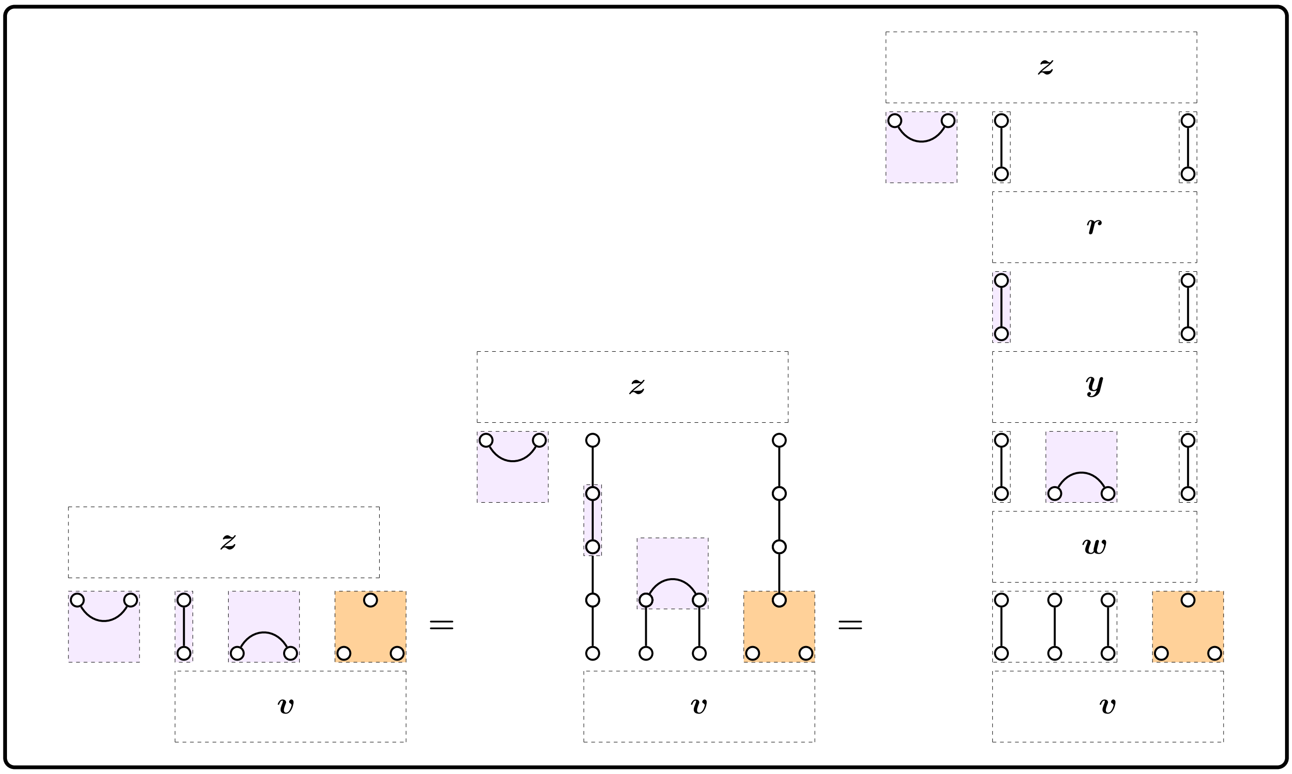


Figure 7.9: We show how matrix multiplication is implemented in **PlanarMult** for $SO(n)$ (here $n = 3$) using the tensor product decomposition of the algorithmically planar $(4 + 5)\backslash 3$–diagram given in Figure 7.8 as an example. We perform the matrix multiplication as follows: first, we deform the entire tensor product decomposition diagram by pulling each individual diagram up one level higher than the previous one, going from right-to-left, and then we apply the functor Ψ at each level. Note, in particular, that we need to attach identity strings to the free vertices appearing in the top row. Finally, we perform matrix multiplication at each level to obtain the final output vector.

Finally, as the third diagram returned from **Factor** corresponds to the permutation (1432) in S_4, we perform **Permute**$(z, (1432))$ which returns the vector

$$\sum_{m\in[3]} \sum_{l_3,t_1\in[3]} \sum_{j\in[3]} \sum_{l_1,l_2\in[3]} v_{l_1,l_2,l_3,j,j} \det(e_{t_1} \otimes e_{l_1} \otimes e_{l_2}) \, (e_{t_1} \otimes e_m \otimes e_m \otimes e_{l_3}) \tag{7.85}$$

This is the vector that is returned by **MatrixMult**.

Time Complexity

For a matrix corresponding to a (k, l)–Brauer diagram, the analysis is the same as for the orthogonal group. For a matrix corresponding to an $(l+k)\backslash n$–diagram, we look at each step of the implementation that was given above.

Step 1: Recall that we map a vector in $(\mathbb{R}^n)^{\otimes k}$ to a vector in $(\mathbb{R}^n)^{\otimes 2b+d+s}$ in this step.

To obtain the best time complexity, we need to look at the tuples $J \in [n]^{2b+d}, T \in [n]^s$ and $B \in [n]^{n-s}$ that appear in (7.73), (7.74), and (7.75).

Indeed, for each tuple J, we first need to consider how many tuples T come with pairwise different entries in $[n]$. The number of such tuples is $\frac{n!}{(n-s)!}$. Consequently, we only need to perform multiplications and additions for entries with such indices (J, T). Let us call any such pair of indices (J, T) **valid**.

Hence, for each valid tuple (J, T), there are $(n-s)!$ multiplications and $(n-s)! - 1$ additions. Since the number of valid tuples of the form (J, T) is

$$n^{2b+d}\frac{n!}{(n-s)!} \tag{7.86}$$

we have that the overall time complexity is $O(n^{k-(n-s)}n!)$ for this step, since $2b + d = k - (n-s)$ by (7.70).

Step 2: By the operations performed in this step, we can immediately apply the analysis of Step 1 for the orthogonal group.

Since the part of the matrix multiplication that corresponds to B_i, for some $i = 1 \to b$, maps a vector in $(\mathbb{R}^n)^{\otimes d+s+2i}$ to a vector in $(\mathbb{R}^n)^{\otimes d+s+2i-2}$, we have that the overall time complexity is $O(n^{k+s-(n-s)-1})$, since $2b + d + s = k + s - (n-s)$ by (7.70).

Steps 3, 4: These steps correspond to Steps 2, 3 for the orthogonal group, hence they have no cost.

Hence, we have reduced the overall time complexity from $O(n^{l+k})$ to

$$O(n^{k-(n-s)}(n! + n^{s-1})) \tag{7.87}$$

Chapter 8

Graph Automorphism Group Equivariant Neural Networks

In many cases, the relationships that exist between certain entities can be structured in the form of a graph. These include the interactions between people in a social network (Leskovec et al., 2010), the bonds that connect atoms in a molecule (Gilmer et al., 2017), and the relationships between users and items in a recommendation system (Konstas et al., 2009). As a result, there is a high level of motivation to design neural network architectures that can offer new insights into data with this structure. A number of methods for learning from graphs exist in the machine learning literature. These include Graph Attention Networks (GATs) (Velicković et al., 2018), Graph Convolutional Networks (GCNs) (Kipf and Welling, 2017), and Graph Neural Networks (GNNs) (Gori et al., 2005; Scarselli et al., 2009), among others. In particular, many of these approaches use some form of equivariance to the symmetric group, which we characterised fully in Chapter 3. However, using neural networks that are equivariant only to the symmetric group to learn from graphs does not take into account the relations that exist between the vertices in a graph. Given that the actual group of symmetries of a graph is its automorphism group, we would like to construct neural networks that satisfy the stronger condition of being equivariant to this group instead.

In this chapter, writing the automorphism group of a graph G having some n vertices as $\mathrm{Aut}(G)$, we give a full characterisation of all of the possible $\mathrm{Aut}(G)$-equivariant neural networks whose layers are some tensor power of $\mathbb{R}^n$ by finding a spanning set of matrices for the learnable, linear, $\mathrm{Aut}(G)$-equivariant layer functions between such tensor power spaces in the standard basis of $\mathbb{R}^n$. Our approach is similar to the one that we have seen in the preceding chapters for other groups. However, instead of calculating the spanning set by studying set partitions, we obtain it by applying a monoidal functor to isomorphism classes of so-called **bilabelled graphs**. In short, a (k, l)–bilabelled graph is a graph that comes with two tuples, one of length k and the other of length l, whose entries are taken from the vertex set of the graph (with repetitions amongst entries allowed). Consequently, by looking at the combinatorics of bilabelled graphs, we can determine the learnable, linear, $\text{Aut}(G)$-equivariant layer functions between any two tensor power spaces of $\mathbb{R}^n$.

We leverage the work of Mančinska and Roberson (2020), who studied the problem of determining under what circumstances two graphs are quantum isomorphic. They built upon the work of Chassaniol (2019), who showed that a vertex-transitive graph has no quantum symmetries. We show how their results and methods can be applied instead for the purpose of learning from data that has an underlying symmetry to the automorphism group of a graph.

There are important consequences for performing such a characterisation. A famous theorem in algebraic graph theory known as Frucht's Theorem (Frucht, 1938) states that every finite group is isomorphic to the automorphism group of a finite undirected graph. As a result, for any finite group of our choosing, if we know the graph (having some n vertices) whose automorphism group is isomorphic to the group in question, then we will be able to characterise all of the learnable, linear, group equivariant layer functions between any two tensor power spaces of $\mathbb{R}^n$ for that group. In particular, we show that we can recover the diagram basis that appears in Godfrey et al. (2023) — and which we studied in Section 6.1 — for the learnable, linear, S_n-equivariant layer functions between tensor power spaces of $\mathbb{R}^n$ in the standard basis of $\mathbb{R}^n$. We also show that we can determine characterisations for other groups, such as for D_4, considered as a subgroup of S_4, which have been missing from the literature.

Furthermore, even if, for a given finite undirected graph, we are unable to determine the finite group that is isomorphic to the automorphism group of the graph, we know that we can calculate the learnable, linear, automorphism group equivariant linear layers using this method, which will be sufficient for performing learning in situations where we do not need to know explicitly what the actual automorphism group is.

The content of this chapter is modelled on the paper:

Edward Pearce-Crump and William J. Knottenbelt. (2024b) Graph Automorphism Group Equivariant Neural Networks. *Proceedings of the 41st International Conference on Machine Learning* (ICML 2024), PMLR 235:40051-40077. [Poster]

8.1 Characterisation Result using Bilabelled Graphs

In this section, we construct a spanning set of matrices for $\mathrm{Hom}_{\mathrm{Aut}(G)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ by relating each spanning set matrix with the isomorphism class of a so-called (k,l)–bilabelled graph. We leverage the work of Mančinska and Roberson (2020) throughout, but begin by stating a result found by Chassaniol (2019) which describes, in terms of a generating set of matrices, the category whose morphisms are the linear layer functions that we wish to characterise.

8.1.1 Chassaniol's Result

For any subgroup $G(n)$ of $GL(n)$, recall from Proposition 6.31 the category $\mathcal{C}(G(n))$, which consists of objects that are the k-order tensor power spaces of $\mathbb{R}^n$, viewed as representations of $G(n)$, and morphism spaces between any two objects that are the vector spaces $\mathrm{Hom}_{G(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$.

In this chapter, we will sometimes write $\mathcal{C}(G(n))(k,l)$ for $\mathrm{Hom}_{G(n)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$, and reuse the notation $\mathcal{C}(G(n))$ for the set $\bigcup_{k,l=0}^{\infty} \mathcal{C}(G(n))(k,l)$.

It will be useful to define a number of spider maps $M^{k,l}$ together with the swap map S.

Definition 8.1. For all non-negative integers k, l, the **spider map** $M^{k,l} \in \mathrm{Hom}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ is defined as follows:

- if $k, l > 0$, then $M^{k,l}$ maps $e_i^{\otimes k}$ to $e_i^{\otimes l}$ for all $i \in [n]$ and maps all other vectors to the zero vector.
- if $k = 0$ and $l > 0$, then $M^{0,l}$ maps 1 to $\sum_i e_i^{\otimes l}$.
- if $k > 0$ and $l = 0$, then $M^{k,0}$ maps $e_i^{\otimes k}$ to 1 for all $i \in [n]$ and maps all other vectors to 0.
- if $k = 0$ and $l = 0$, then $M^{0,0} := (n)$.

The **swap map** $S \in \mathrm{Hom}((\mathbb{R}^n)^{\otimes 2}, (\mathbb{R}^n)^{\otimes 2})$ maps $e_i \otimes e_j$ to $e_j \otimes e_i$ for all $i, j \in [n]$.

Then, for any graph G having n vertices, Chassaniol (2019) found the following generating set for the category $\mathcal{C}(\mathrm{Aut}(G))$:

Theorem 8.2 (Chassaniol (2019), Proposition 3.5)**.** *Let A_G denote the adjacency matrix of the graph G. Then*

$$\mathcal{C}(\mathrm{Aut}(G)) = \langle M^{0,1}, M^{2,1}, A_G, S \rangle_{+,\circ,\otimes,*} \tag{8.1}$$

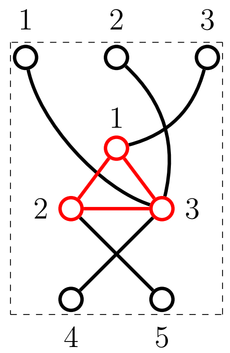


Figure 8.1: The $(2,3)$–bilabelled graph diagram that is associated with the isomorphism class of the $(2,3)$–bilabelled graph $\boldsymbol{K} := (K_3, (3,2), (3,3,1))$.

where the right hand side denotes all matrices that can be generated from the four matrices using the operations of $\mathbb{R}$–linear combinations, matrix product, Kronecker product, and transposition.

8.1.2 The Category of All Bilabelled Graphs

Mančinska and Roberson (2020) showed how Chassaniol's generating set for $\mathcal{C}(\text{Aut}(G))$ can be improved by relating it to the combinatorics of bilabelled graphs.

Definition 8.3 (Bilabelled Graph)**.** A **$(\boldsymbol{k}, \boldsymbol{l})$–bilabelled graph $\boldsymbol{H}$** is a triple $(H, \boldsymbol{k}, \boldsymbol{l})$, where H is a graph having some m vertices with labels in $[m]$, $\boldsymbol{k} := (k_1, \dots, k_k)$ is a tuple in $[m]^k$, and $\boldsymbol{l} := (l_1, \dots, l_l)$ is a tuple in $[m]^l$.

We call $\boldsymbol{k}$ and $\boldsymbol{l}$ the **input** and **output tuples** to $\boldsymbol{H}$ respectively, and we call H the **underlying graph** of $\boldsymbol{H}$. A vertex in the underlying graph H of $\boldsymbol{H}$ is said to be **free** if it does not appear in either of the input or output tuples.

In order to provide a diagrammatic representation of bilabelled graphs, we need the following definition.

Definition 8.4. Let $\boldsymbol{H_1} = (H_1, \boldsymbol{k}, \boldsymbol{l})$ be a (k,l)–bilabelled graph and let $\boldsymbol{H_2} = (H_2, \boldsymbol{k'}, \boldsymbol{l'})$ be another (k,l)–bilabelled graph. Then $\boldsymbol{H_1}$ and $\boldsymbol{H_2}$ are **isomorphic** as (k,l)–bilabelled graphs, written $\boldsymbol{H_1} \cong \boldsymbol{H_2}$, if and only if there is a graph isomorphism from H_1 to H_2 such that $k_i \mapsto k'_i$ for all $i \in [k]$ and $l_j \mapsto l'_j$ for all $j \in [l]$. We denote $\boldsymbol{H_1}$'s **isomorphism class** by $[\boldsymbol{H_1}]$.

Remark 8.5. An isomorphism between two (k,l)–bilabelled graphs can effectively be thought of as a relabelling of the vertices of the *same* underlying graph, resulting in an appropriate relabelling of the elements of the tuples themselves. Consequently, the isomorphism can be thought of as a relabelling of the same bilabelled graph.

With this in mind, we can represent the isomorphism class $[\boldsymbol{H}]$ for a (k,l)–bilabelled graph $\boldsymbol{H} = (H, \boldsymbol{k}, \boldsymbol{l})$ in diagrammatic form. We choose $\boldsymbol{H}$ as a class representative of $[\boldsymbol{H}]$, and proceed as follows: we draw

- l black vertices on the top row, labelled left to right by $1, \ldots, l$,
- k black vertices on the bottom row, labelled left to right by $l+1, \ldots, l+k$,
- the underlying graph H in red, with labelled red vertices and red edges, in between the two rows of black vertices, and
- a black line connecting vertex i in the top row to l_i, and a black line connecting vertex $l+j$ in the bottom row to k_j.

Note that if we drew a diagram for another representative of the same isomorphism class $[\boldsymbol{H}]$, by the definition of an isomorphism for (k,l)–bilabelled graphs, we would obtain exactly the same diagram as for the first class representative, except the labels of the underlying graph's vertices (and consequently the elements of the tuples) would be different. As a result, the diagram for the isomorphism class $[\boldsymbol{H}]$ is independent of the choice of class representative, and so we choose the class' label, $\boldsymbol{H}$, to draw the diagram throughout, unless otherwise stated. With the technicalities being understood, we often refer to the construction defined above as a (k,l)–bilabelled graph diagram *for $\boldsymbol{H}$ itself*, and use the same notation $\boldsymbol{H}$. We give an example of a $(2,3)$–bilabelled graph diagram in Figure 8.1.

We can define a number of **operations** on isomorphism classes of bilabelled graphs, namely, composition, tensor product, and involution, as follows.

Definition 8.6 (Composition)**.** Let $[\boldsymbol{H_1}]$ be the isomorphism class of the (k,l)–bilabelled graph $\boldsymbol{H_1} = (H_1, \boldsymbol{k}, \boldsymbol{l})$, and let $[\boldsymbol{H_2}]$ be the isomorphism class of the (l,m)–bilabelled graph $\boldsymbol{H_2} = (H_2, \boldsymbol{l'}, \boldsymbol{m})$.

We define the **composition** $[\boldsymbol{H_2}] \circ [\boldsymbol{H_1}]$ to be the isomorphism class $[\boldsymbol{H}]$ of the (k,m)–bilabelled graph $\boldsymbol{H} = (H, \boldsymbol{k}, \boldsymbol{m})$ that is obtained as follows: drawing each isomorphism class as a diagram, we first relabel the red vertices in H_2 (and consequently the elements of $\boldsymbol{l'}, \boldsymbol{m}$) under the map $i \mapsto i'$, for all $i \in [V(H_2)]$. Then, we connect the top row of black vertices of $\boldsymbol{H_1}$ with the bottom row of black vertices of $\boldsymbol{H_2}$, and delete the vertices themselves. We are now left with l black lines that are edges between red vertices of the underlying graphs H_1 and H_2. Next, we contract these, forming set unions of the vertex labels where appropriate, and remove any red multiedges between red vertices that appear in the contraction to obtain the new underlying graph H. Note that we keep any loops that appear in the new underlying graph H. Finally, we relabel the vertex set of the new underlying graph H so that each vertex is labelled by an integer only, and consequently relabel the entries of the tuples $\boldsymbol{k}$ and $\boldsymbol{m}$ accordingly.

Since this operation has been defined on diagrams, it means that the operation is well-defined on the isomorphism classes themselves. We give an example of this composition in Figure 8.2.

Remark 8.7. In order to compose two isomorphism classes of bilabelled graphs, note that only the number of bottom row vertices in the diagram for $\boldsymbol{H_2}$ needs to be equal to the number of top row vertices in the diagram for $\boldsymbol{H_1}$. In particular, the number of vertices in the underlying graphs of $\boldsymbol{H_1}$ and $\boldsymbol{H_2}$ do *not* need to be the same.

Definition 8.8 (Tensor Product)**.** Let $[\boldsymbol{H_1}]$ be the isomorphism class of the (k, l)–bilabelled graph $\boldsymbol{H_1} = (H_1, \boldsymbol{k}, \boldsymbol{l})$ and let $[\boldsymbol{H_2}]$ be the isomorphism class of the (q, m)–bilabelled graph $\boldsymbol{H_2} = (H_2, \boldsymbol{q}, \boldsymbol{m})$.

We define the **tensor product** $[\boldsymbol{H_1}] \otimes [\boldsymbol{H_2}]$ to be the isomorphism class of the $(k+q, l+m)$–bilabelled graph $(H_1 \cup H_2, \boldsymbol{kq}, \boldsymbol{lm})$ where $\boldsymbol{kq}$ is the $(k+q)$-length tuple obtained by concatenating $\boldsymbol{k}$ and $\boldsymbol{q}$, and likewise for $\boldsymbol{lm}$.

Definition 8.9 (Involution)**.** Let $[\boldsymbol{H}]$ be the isomorphism class of the (k, l)–bilabelled graph $\boldsymbol{H} = (H, \boldsymbol{k}, \boldsymbol{l})$. We define the **involution** $[\boldsymbol{H}^*]$ to be the isomorphism class of the (l, k)–bilabelled graph $\boldsymbol{H}^* = (H, \boldsymbol{l}, \boldsymbol{k})$.

We can form a category for the bilabelled graphs, as follows:

Definition 8.10 (Category of Bilabelled Graphs)**.** The **category of all bilabelled graphs** $\mathcal{G}$ is the category whose objects are the non-negative integers, and, for any pair of objects k and l, the morphism space $\mathcal{G}(k, l) := \mathrm{Hom}_{\mathcal{G}}(k, l)$ is defined to be the $\mathbb{R}$–linear span of the set of all isomorphism classes of (k, l)–bilabelled graphs.

The vertical composition of morphisms is the composition of isomorphism classes of bilabelled graphs given in Definition 8.6 that is extended to be $\mathbb{R}$–bilinear, the tensor product of morphisms is the tensor product of isomorphism classes of bilabelled graphs given in Definition 8.8 that is also extended to be $\mathbb{R}$–bilinear, and the unit object is 0.

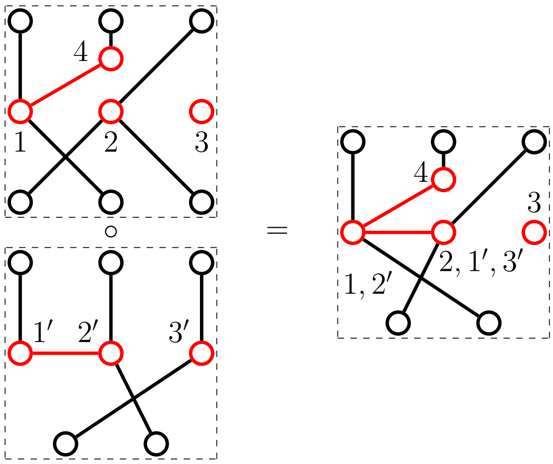


Figure 8.2: The composition $[\boldsymbol{H_2}] \circ [\boldsymbol{H_1}]$ of the $(2,3)$–bilabelled graph diagram for $\boldsymbol{H_1} = (H_1, (3', 2'), (1', 2', 3'))$ with the $(3,3)$–bilabelled graph diagram for $\boldsymbol{H_2} = (H_2, (2,1,2), (1,4,2))$, where H_1 is the graph having (relabelled) vertex set $\{1', 2', 3'\}$ and edge set $\{(1', 2')\}$ and H_2 is the graph having vertex set $\{1,2,3,4\}$ and edge set $\{(1,4)\}$. The vertices of the resulting bilabelled graph diagram on the right hand side would be relabelled. For example, $\{1, 2'\}$ could be relabelled as 1 and $\{2, 1', 3'\}$ could be relabelled as 2 to give the $(2,3)$–bilabelled graph diagram for $\boldsymbol{H} = (H, (2,1), (1,4,2))$, where H is the graph having vertex set $\{1,2,3,4\}$ and edge set $\{(1,4),(1,2)\}$.

Proposition 8.11. *The category of all bilabelled graphs, $\mathcal{G}$, is a strict $\mathbb{R}$–linear monoidal category.*

Proof. $\mathcal{G}$ is a strict monoidal category because the bifunctor on objects reduces to the addition operation on natural numbers, which is associative, and the bifunctor on morphisms is the tensor product of isomorphism classes of bilabelled graphs given in Definition 8.8, which is associative because both the concatenation of tuples and the (set) unions of graphs are associative operations.

$\mathcal{G}$ is $\mathbb{R}$–linear because the morphism space between any two objects is by definition a vector space, and the composition of morphisms is $\mathbb{R}$–bilinear by definition. For the same reason, the bifunctor is also $\mathbb{R}$–bilinear. $\square$

8.1.3 G-Homomorphism Matrices

Mančinska and Roberson (2020) established a relationship between the abstract and the concrete: namely, between isomorphism classes of (k,l)–bilabelled graphs and, for a fixed graph G having n vertices, matrices that are linear maps $(\mathbb{R}^n)^{\otimes k} \to (\mathbb{R}^n)^{\otimes l}$, which they termed G-homomorphism matrices. We express this relationship more formally in terms of functors and categories at the end of this section.

Definition 8.12 (G-Homomorphism Matrix)**.** Suppose that G is a graph having n vertices, and let $[\boldsymbol{H}]$ be the isomorphism class of the (k, l)–bilabelled graph $\boldsymbol{H} := (H, \boldsymbol{k}, \boldsymbol{l})$.

The G**-homomorphism matrix** of $[\boldsymbol{H}]$ is the $n^l \times n^k$ matrix where each (I, J)-entry is given by the number of graph homomorphisms from H to G such that $\boldsymbol{l}$ is mapped to I and $\boldsymbol{k}$ is mapped to J. We denote this matrix by $X_{\boldsymbol{H}}^G$.

Remark 8.13. Note that a G-homomorphism matrix $X_{\boldsymbol{H}}^G$ is independent of the choice of class representative for $[\boldsymbol{H}]$, and so we can refer to a G-homomorphism matrix *of* $\boldsymbol{H}$ *itself*, with the technicalities being understood. Also, any such matrix must have only real entries, by definition.

In Figure 8.3, we present an example that shows how to calculate the G-homomorphism matrix of $[\boldsymbol{H}]$ for the graph G having vertex set $V(G) = \{1, 2, 3\}$ and edge set $E(G) = \{(1, 2)\}$ and for the $(1, 1)$–bilabelled graph $\boldsymbol{H} = (H, (3), (1))$, where H is the graph having vertex set $V(H) = \{1, 2, 3, 4\}$ and edge set $E(H) = \{(1, 2), (3, 4)\}$. To determine the (I, J)-entry of $X_{\boldsymbol{H}}^G$, we superimpose the values of I onto the top row of black vertices in $\boldsymbol{H}$, and the values of J onto the bottom row of black vertices in $\boldsymbol{H}$. For each red vertex that is connected with a set of black vertices, we look to update its label. If all of the black vertices in the set have the same label, then we update the red vertex with that label, otherwise we stop and immediately determine that the (I, J)-entry of $X_{\boldsymbol{H}}^G$ is 0. Assuming that these red vertices can and have been updated, we determine the number of possible mappings to G for the red vertices that have not been updated in $\boldsymbol{H}$ such that the overall mapping is a graph homomorphism. The total number of possibilities is the (I, J)-entry of $X_{\boldsymbol{H}}^G$.

We now describe some important examples of G-homomorphism matrices where G is a graph having n vertices throughout.

Example 8.14. If $\boldsymbol{A}$ is the $(1, 1)$–bilabelled graph $(K_2, (1), (2))$, where K_2 is the complete graph on two vertices, then $X_{\boldsymbol{A}}^G$ is the adjacency matrix A_G of G.

Example 8.15. If $\boldsymbol{M}^{k,l}$ is the (k, l)–bilabelled graph $(K_1, \boldsymbol{k} = (1, \dots, 1), \boldsymbol{l} = (1, \dots, 1))$, where K_1 is the complete graph on one vertex, then $X_{\boldsymbol{M}^{k,l}}^G$ is the spider matrix $M^{k,l}$ that is given in Definition 8.1.

Example 8.16. If $\boldsymbol{S}$ is the $(2, 2)$–bilabelled graph $(\overline{K_2}, (2, 1), (1, 2))$, where $\overline{K_2}$ is the edgeless graph on two vertices, then $X_{\boldsymbol{S}}^G$ is the swap map S that is also given in Definition 8.1.

For a fixed graph G having n vertices, we can form a category for the G-homomorphism matrices, as follows:

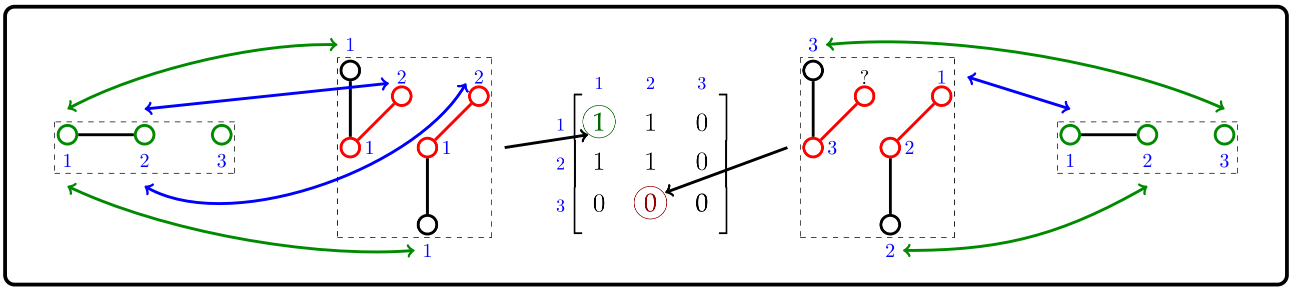


Figure 8.3: For the graph G having vertex set $V(G) = \{1, 2, 3\}$ and edge set $E(G) = \{(1, 2)\}$, we show how to calculate the (I, J)-entries of the G-homomorphism matrix $X^G_{\boldsymbol{H}}$ corresponding to the isomorphism class $[\boldsymbol{H}]$ of the $(1, 1)$–bilabelled graph $\boldsymbol{H} = (H, (3), (1))$ where H is the graph having vertex set $V(H) = \{1, 2, 3, 4\}$ and edge set $E(H) = \{(1, 2), (3, 4)\}$. On the left hand side, we calculate the $(1, 1)$-entry of $X^G_{\boldsymbol{H}}$. Since both of the black labelled vertices are being mapped to vertex 1 in G, we can superimpose 1 onto these vertices, and hence also onto the red labelled vertices that are connected with them, to determine where the other red vertices can be mapped to under a graph homomorphism. In this case, the only possible vertex in G that the other red vertices can be mapped to is 2. Hence, there is only one possible graph homomorphism from H to G such that $1 \mapsto 1$ and $3 \mapsto 1$, and so the $(1, 1)$-entry of $X^G_{\boldsymbol{H}}$ is 1. On the right hand side, we calculate the $(3, 2)$-entry of $X^G_{\boldsymbol{H}}$. We follow the same approach by superimposing 3 in G onto the black vertex labelled 1 in $\boldsymbol{H}$, and 2 in G onto the black vertex labelled 3 in $\boldsymbol{H}$. We relabel the red vertices that the black vertices are connected with and see where the other red vertices can be mapped to under a graph homomorphism. Whilst one of these red vertices can only be mapped to 1 in G, the other red vertex cannot be mapped to any vertex in G, since the vertex 3 in G is not connected with any other vertex in G. Hence, the number of graph homomorphisms from H to G such that $1 \mapsto 3$ and $3 \mapsto 2$ is 0, and so the $(3, 2)$-entry of $X^G_{\boldsymbol{H}}$ is 0.

Definition 8.17 (Category of G-Homomorphism Matrices)**.** For a given graph G having n vertices, **the category of all G-homomorphism matrices**, $\mathcal{C}^G$, is the category whose objects are the vector spaces $(\mathbb{R}^n)^{\otimes k}$, and, for any pair of objects $(\mathbb{R}^n)^{\otimes k}$ and $(\mathbb{R}^n)^{\otimes l}$, the morphism space $\mathrm{Hom}_{\mathcal{C}^G}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ is defined to be the $\mathbb{R}$–linear span of the set of all G-homomorphism matrices obtained from all isomorphism classes of (k, l)–bilabelled graphs.

The vertical composition of morphisms is given by the usual multiplication of matrices, the tensor product of morphisms is given by the usual Kronecker product of matrices, and the unit object is $\mathbb{R}$.

Proposition 8.18. *The category of all G-homomorphism matrices for a given graph G having n vertices, $\mathcal{C}^G$, is a strict $\mathbb{R}$–linear monoidal category.*

Proof. This is effectively the same proof as the one that is given for Proposition 6.32, except we replace linear maps by matrices. $\square$

Mančinska and Roberson (2020) showed that the operations given on isomorphism classes of bilabelled graphs $\boldsymbol{H}$ correspond bijectively with the matrix operations on G-homomorphism matrices $X^G_{\boldsymbol{H}}$. This is stated more formally in the following three lemmas:

Lemma 8.19 (Mančinska and Roberson (2020), Lemma 3.21)**.** *Suppose that G is a graph having n vertices. Let $[\boldsymbol{H_1}]$ be the isomorphism class of the (k,l)–bilabelled graph $\boldsymbol{H_1} = (H_1, \boldsymbol{k}, \boldsymbol{l})$, and let $[\boldsymbol{H_2}]$ be the isomorphism class of the (l,m)–bilabelled graph $\boldsymbol{H_2} = (H_2, \boldsymbol{l'}, \boldsymbol{m})$. Then*

$$X^G_{\boldsymbol{H_2}} X^G_{\boldsymbol{H_1}} = X^G_{\boldsymbol{H_2} \circ \boldsymbol{H_1}} \tag{8.2}$$

Lemma 8.20 (Mančinska and Roberson (2020), Lemma 3.23)**.** *Suppose that G is a graph having n vertices. Let $[\boldsymbol{H_1}]$ be the isomorphism class of the (k,l)–bilabelled graph $\boldsymbol{H_1} = (H_1, \boldsymbol{k}, \boldsymbol{l})$, and let $[\boldsymbol{H_2}]$ be the isomorphism class of the (q,m)–bilabelled graph $\boldsymbol{H_2} = (H_2, \boldsymbol{q}, \boldsymbol{m})$. Then*

$$X^G_{\boldsymbol{H_1}} \otimes X^G_{\boldsymbol{H_2}} = X^G_{\boldsymbol{H_1} \otimes \boldsymbol{H_2}} \tag{8.3}$$

Lemma 8.21 (Mančinska and Roberson (2020), Lemma 3.24)**.** *Suppose that G is a graph having n vertices, and let $[\boldsymbol{H}]$ be the isomorphism class of the (k,l)–bilabelled graph $\boldsymbol{H} = (H, \boldsymbol{k}, \boldsymbol{l})$. Then*

$$(X^G_{\boldsymbol{H}})^* = X^G_{\boldsymbol{H^*}} \tag{8.4}$$

where $(X^G_{\boldsymbol{H}})^*$ is the transpose of the matrix $X^G_{\boldsymbol{H}}$.

Consequently, we obtain the following theorem, expressed in terms of functors and categories.

Theorem 8.22. *Suppose that G is a graph having n vertices. Then there exists a full, strict $\mathbb{R}$–linear monoidal functor*

$$\mathcal{F}^G : \mathcal{G} \to \mathcal{C}^G \tag{8.5}$$

that is defined on the objects of $\mathcal{G}$ by $\mathcal{F}^G(k) := (\mathbb{R}^n)^{\otimes k}$ and, for any objects k, l of $\mathcal{G}$, the map

$$\mathrm{Hom}_{\mathcal{G}}(k,l) \to \mathrm{Hom}_{\mathcal{C}^G}(\mathcal{F}^G(k), \mathcal{F}^G(l)) \tag{8.6}$$

is given by

$$[\boldsymbol{H}] \mapsto X^G_{\boldsymbol{H}} \tag{8.7}$$

for all isomorphism classes of (k,l)–bilabelled graphs.

Proof. Let G be a graph having n vertices. Lemma 8.19 immediately implies that $\mathcal{F}^G$ is a functor. Hence, we show each of the four conditions of Definition 6.12 in turn.

1. Let k, l be any two objects in $\mathcal{G}$. Then

$$\mathcal{F}^G(k \otimes l) = \mathcal{F}^G(k + l) = (\mathbb{R}^n)^{\otimes k+l} = (\mathbb{R}^n)^{\otimes k} \otimes (\mathbb{R}^n)^{\otimes l} = \mathcal{F}^G(k) \otimes \mathcal{F}^G(l) \tag{8.8}$$

2. This is Lemma 8.20.

3. It is clear from the statement of the theorem that $\mathcal{F}^G$ sends the unit object 0 in $\mathcal{G}$ to $\mathbb{R}$, which is the unit object in $\mathcal{C}^G$.

4. This is immediate from the definition of a G-homomorphism matrix.

It is clear that the functor $\mathcal{F}^G$ is full, once again by the definition of a G-homomorphism matrix. $\square$

8.1.4 A Spanning Set of Matrices for the Learnable, Linear, $\mathrm{Aut}(G)$-Equivariant Layer Functions

Compare the following proposition with Chassaniol's result, given in Theorem 8.2.

Proposition 8.23 (Mančinska and Roberson (2020), Theorem 8.4)**.** *We have that*

$$\mathcal{G} = \langle [\boldsymbol{M}^{0,1}], [\boldsymbol{M}^{2,1}], [\boldsymbol{A}], [\boldsymbol{S}] \rangle_{\circ,\otimes,*} \tag{8.9}$$

where $\boldsymbol{M}^{0,1}, \boldsymbol{M}^{2,1}, \boldsymbol{A}, \boldsymbol{S}$ are the bilabelled graphs defined in Examples 8.14, 8.15, and 8.16, and the operations $\circ, \otimes, *$ on bilabelled graphs are those that are given in Definitions 8.6, 8.8 and 8.9, respectively.

We have come to the main results of this chapter, namely the following two theorems.

Theorem 8.24. *Suppose that G is a graph having n vertices. The vector space of all $\mathrm{Aut}(G)$-equivariant, linear maps between tensor power spaces of $\mathbb{R}^n$, $\mathrm{Hom}_{\mathrm{Aut}(G)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$, when the standard basis of $\mathbb{R}^n$ is chosen, is spanned by all of the G-homomorphism matrices $X^G_{\boldsymbol{H}}$ that are obtained from all of the isomorphism classes of (k, l)–bilabelled graphs.*

Proof. The proof of this theorem is inspired by Mančinska and Roberson (2020), Theorem 8.5.

We know, by Chassaniol, that

$$\mathcal{C}(\mathrm{Aut}(G)) = \langle M^{0,1}, M^{2,1}, A_G, S \rangle_{+,\circ,\otimes,*} \tag{8.10}$$

By Examples 8.14, 8.15, and 8.16, we get that

$$\mathcal{C}(\mathrm{Aut}(G)) = \{X_{\boldsymbol{H}}^G \mid [\boldsymbol{H}] \in \langle [\boldsymbol{M}^{0,1}], [\boldsymbol{M}^{2,1}], [\boldsymbol{A}], [\boldsymbol{S}] \rangle_{+,\circ,\otimes,*}\} \tag{8.11}$$

and so, by Proposition 8.23, we have that

$$\mathcal{C}(\mathrm{Aut}(G)) = \mathbb{R}\text{–span}\{X_{\boldsymbol{H}}^G \mid [\boldsymbol{H}] \in \mathcal{G}\} \tag{8.12}$$

Consequently, for any k, l, we get that

$$\mathcal{C}(\mathrm{Aut}(G))(k,l) = \mathbb{R}\text{–span}\{X_{\boldsymbol{H}}^G \mid [\boldsymbol{H}] \in \mathcal{G}(k,l)\} \tag{8.13}$$

As the left hand side of (8.13) is equivalent notation for $\mathrm{Hom}_{\mathrm{Aut}(G)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$, we obtain our result. □

Theorem 8.25 (Graph Automorphism Group Equivariant Weight Matrices)**.** *For all non-negative integers l and k, if G is a graph having n vertices, then the weight matrix W that appears in an* Aut(G)*-equivariant linear layer function from $(\mathbb{R}^n)^{\otimes k}$ to $(\mathbb{R}^n)^{\otimes l}$ must be of the form*

$$W = \sum_{[\boldsymbol{H}] \in \mathcal{G}(k,l)} \lambda_{\boldsymbol{H}} X_{\boldsymbol{H}}^G \tag{8.14}$$

Remark 8.26. In particular, Theorem 8.24 shows that $\mathcal{C}^G$ is isomorphic to $\mathcal{C}(\mathrm{Aut}(G))$ as categories, and that the objects of the category $\mathcal{C}^G$ come from representations of $\mathrm{Aut}(G)$.

Theorem 8.24 is especially powerful when it is combined with Frucht's Theorem (Frucht, 1938). Frucht's Theorem states that every finite group is isomorphic to the automorphism group of a finite undirected graph. Hence, for any finite group, by determining a graph (having n vertices) whose automorphism group is isomorphic to the group in question, we can determine the weight matrix that appears between any two layers that are some tensor power of $\mathbb{R}^n$ in a neural network that is equivariant to the group.

Remark 8.27. It is very important to note the following. If H is a group that is isomorphic to the automorphism group of a graph G having n vertices, then the spanning set that we obtain for $\mathrm{Hom}_H((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ using Theorem 8.24 depends on how the automorphism group is embedded in the symmetric group S_n, itself thought of as a subgroup of matrices in $GL(n)$ having chosen the standard basis of $\mathbb{R}^n$. The spanning set is determined not only by how the vertices of the underlying graph G are labelled (up to all automorphisms), but also by what the edges are in G.

For example, in Example 8.33, we show not only that D_4 has three embeddings in S_4, but also that each embedding can be obtained separately from two graphs that are the complement of each other, where both graphs have the same labelling of the vertices. All six instances (an embedding of D_4 coming from a graph with a certain labelling of its vertices) give rise to different, albeit related, spanning sets (in fact, bases) of $\mathrm{Hom}_{D_4}(\mathbb{R}^4, \mathbb{R}^4)$. Hence, we obtain a basis for each specific embedding and labelled graph that is chosen.

We have seen that we can find a spanning set for $\mathrm{Hom}_{\mathrm{Aut}(G)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ by considering all isomorphism classes of (k, l)–bilabelled graphs. However, with the following result, we can reduce the number of elements that appear in the spanning set by reducing the number of isomorphism classes that we need to consider.

Proposition 8.28. *Let $\boldsymbol{H}$ be a (k, l)–bilabelled graph whose underlying graph H contains a subset of free vertices that, whilst they may have edges amongst themselves, are entirely disconnected from any vertices that are in a connected component containing a non-free vertex.*

Then $X_{\boldsymbol{H}}^G$ is a scalar multiple of $X_{\boldsymbol{H'}}^G$, where $\boldsymbol{H'}$ is the (k, l)–bilabelled graph that is obtained from $\boldsymbol{H}$ by removing the subset.

Proof. Suppose that $\boldsymbol{H} = (H, \boldsymbol{k}, \boldsymbol{l})$. Let H_2 be the subgraph consisting of the subset of free vertices (including the edges that are solely between these vertices) that are entirely disconnected from any vertices that are in a connected component containing a non-free vertex. Define H_1 to be the graph that is H without H_2.

Then, by construction, H is a disjoint union of H_1 and H_2, as graphs, and, if we define $\boldsymbol{H_1} := (H_1, \boldsymbol{k}, \boldsymbol{l})$, then it is clear that

$$X_{\boldsymbol{H}}^G = c_{\boldsymbol{H_2}}^G X_{\boldsymbol{H_1}}^G \tag{8.15}$$

where $c_{\boldsymbol{H_2}}^G$ is a constant denoting the number of graph homomorphisms from H_2 to G, as required. □

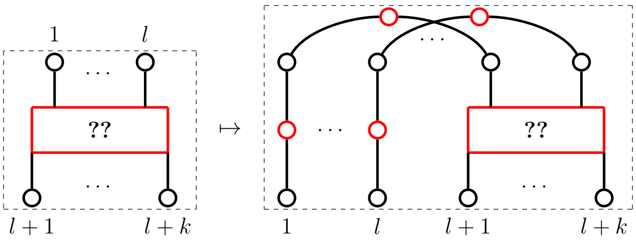
We also have the following result that is useful for generating the isomorphism classes of (k, l)–bilabelled graphs.

Proposition 8.29 (Frobenius Duality). *Suppose that we use Theorem 8.24 to obtain a spanning set for* $\mathrm{Hom}_{\mathrm{Aut}(G)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$. *Then, for any $q, m \geq 0$ such that $q + m = k + l$, we can immediately obtain a spanning set for* $\mathrm{Hom}_{\mathrm{Aut}(G)}((\mathbb{R}^n)^{\otimes q}, (\mathbb{R}^n)^{\otimes m})$.

Proof. Firstly, there is an $\mathbb{R}$–linear isomorphism

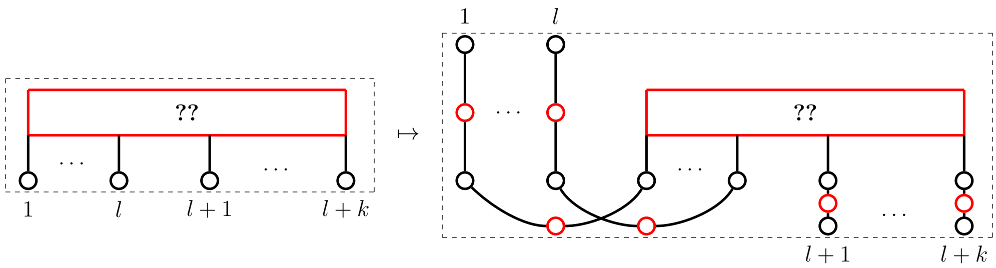
$$\mathrm{Hom}_{\mathcal{G}}(k, l) \to \mathrm{Hom}_{\mathcal{G}}(l + k, 0) \tag{8.16}$$

that is given on bilabelled graph diagrams by



(8.17)

with inverse given by



(8.18)

Similarly, for any graph G having n vertices, there is an $\mathbb{R}$–linear isomorphism

$$\mathrm{Hom}_{\mathrm{Aut}(G)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l}) \to \mathrm{Hom}_{\mathrm{Aut}(G)}((\mathbb{R}^n)^{\otimes l+k}, \mathbb{R}) \tag{8.19}$$

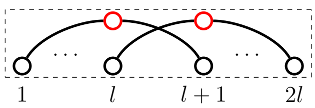
that is given by

$$X_{\boldsymbol{H}}^{G} \mapsto \mathcal{F}^{G}([\boldsymbol{MM}_l]) \circ (Id^{\otimes l} \otimes X_{\boldsymbol{H}}^{G}) \tag{8.20}$$

where Id is the $n \times n$ identity matrix, with inverse given by

$$X_{\boldsymbol{H}}^{G} \mapsto (Id^{\otimes l} \otimes X_{\boldsymbol{H}}^{G}) \circ (\mathcal{F}^{G}([\boldsymbol{UU}_l]) \otimes Id^{\otimes k}) \tag{8.21}$$

Here, $\boldsymbol{MM}_l$ is the $(2l, 0)$–bilabelled graph diagram



(8.22)

and $\boldsymbol{UU}_l$ is its involution, as defined in Definition 8.9.

Since, for any graph G having n vertices, the functor $\mathcal{F}^G$ is strict monoidal, by Theorem 8.22, we get that the following diagram commutes:

$$
\begin{tikzcd}
\mathrm{Hom}_{\mathcal{G}}(k, l) \arrow[r] \arrow[d] & \mathrm{Hom}_{\mathcal{G}}(l+k, 0) \arrow[d] \\
\mathrm{Hom}_{\mathrm{Aut}(G)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l}) \arrow[r] & \mathrm{Hom}_{\mathrm{Aut}(G)}((\mathbb{R}^n)^{\otimes l+k}, \mathbb{R})
\end{tikzcd}
\tag{8.23}
$$

Since $\mathcal{F}^G$ gives a bijective correspondence between all isomorphism classes of bilabelled graphs and the spanning set elements for $\mathrm{Hom}_{\mathrm{Aut}(G)}$, as shown in Theorem 8.24, we can use the commuting square to obtain a spanning set for $\mathrm{Hom}_{\mathrm{Aut}(G)}((\mathbb{R}^n)^{\otimes k+l}, \mathbb{R})$ from the spanning set for $\mathrm{Hom}_{\mathrm{Aut}(G)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$.

Now, fixing $q, m \geq 0$ such that $q + m = k + l$, we can run the arrows of the commuting square in reverse to obtain a spanning set for $\mathrm{Hom}_{\mathrm{Aut}(G)}((\mathbb{R}^n)^{\otimes q}, (\mathbb{R}^n)^{\otimes m})$, as required. □

We also have the following result.

Proposition 8.30. *We can recover the diagram basis of* $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ *that first appeared in Godfrey et al. (2023) (see also Section 6.1) from the homomorphism matrices for the complement of the complete graph on* n *vertices,* $\overline{K_n}$*.*

Proof. We saw in Section 2.4 that the automorphism group of the complement of the complete graph, $\mathrm{Aut}(\overline{K_n})$, is the symmetric group S_n. To recover the diagram basis, we reduce the set of isomorphism classes of (k, l)–bilabelled graphs $\boldsymbol{H}$ that we need to consider in (8.13), as follows.

Firstly, we only need to consider isomorphism classes of (k, l)–bilabelled graphs $\boldsymbol{H}$ whose underlying graphs H are edgeless, since if H has either some edge between two distinct vertices or a loop, then for there to be a graph homomorphism from H to $\overline{K_n}$, the graph $\overline{K_n}$ would need to have an edge between two distinct vertices or a loop. Secondly, we only need to consider isomorphism classes of (k, l)–bilabelled graphs $\boldsymbol{H}$ whose underlying graph H is edgeless, having at most n vertices, since the image under $\mathcal{F}^G$ of any (k, l)–bilabelled graph whose underlying graph is edgeless having more than n vertices would correspond to a scalar multiple of the image of a (k, l)–bilabelled graph whose underlying graph is edgeless having at most n vertices. Thirdly, we only need to consider isomorphism classes of (k, l)–bilabelled graphs $\boldsymbol{H}$ whose underlying graph H is edgeless having at most n vertices where no vertex is left *free* in $\boldsymbol{H}$, that is, there is no red vertex in H that is not attached by a black edge to some black vertex, by Proposition 8.28.

But this subset of isomorphism classes of (k, l)–bilabelled graphs is precisely all set partitions of $\{1, \dots, l+k\}$ having at most n blocks! By construction, the image of this subset under $\mathcal{F}^G$ is a spanning set of $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ and, by a dimension count, the image forms a basis, which is precisely the diagram basis of $\mathrm{Hom}_{S_n}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ that appeared in Section 6.1. □

Remark 8.31. Proposition 8.30 implies that, for a graph G having n vertices, the spanning set of $\mathrm{Hom}_{\mathrm{Aut}(G)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$ automatically includes the image, under the functor $\mathcal{F}^G$, of all set partitions of $[l+k]$ expressed as (k, l)–bilabelled graph diagrams, since $\mathrm{Aut}(G)$ is a subgroup of S_n.

We use all of the results given above to obtain a procedure for constructing the weight matrix that appears in an $\mathrm{Aut}(G)$-equivariant linear layer function from $(\mathbb{R}^n)^{\otimes k}$ to $(\mathbb{R}^n)^{\otimes l}$ from isomorphism classes of (k, l)–bilabelled graphs. In the procedure, we create all $(q, 0)$–bilabelled graph diagrams that are appropriate for the graph G, where $q = k + l$, and then use Frobenius duality and the functor $\mathcal{F}^G$ to obtain a spanning set of matrices for $\mathrm{Hom}_{\mathrm{Aut}(G)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$. To capture all of the $(q, 0)$–bilabelled graph diagrams that are appropriate for G, we first need to calculate the length of the longest path m between any two (not necessarily distinct) vertices in G where each edge in G is traversed at most once. This is the longest path that needs to be mapped onto by a graph homomorphism from the underlying graph of a $(q, 0)$–bilabelled graph diagram to G. We use m to generate the all of the $(q, 0)$–bilabelled graph diagrams as follows. We focus on the **non-free red vertices** since they play a key role in how the entries of a spanning set matrix $X_{\boldsymbol{H}}^G$ are determined. We use the following ideas to generate all of the $(q, 0)$–bilabelled graph diagrams from the set partitions of $[l+k]$ expressed as $(q, 0)$–bilabelled graph diagrams.

- Between any pair of non-free red vertices, we must consider all paths of lengths 0 to $2m$ between them consisting solely of free red vertices, since such a path of length $2m$ is a double cover of the longest path in G.
- Starting with a single non-free red vertex that is not connected with any other red vertex, we must consider all paths of lengths 0 to m consisting solely of free red vertices from this non-free red vertex, since such a path of length m is a single cover of the longest path in G.

We also need to consider the impact of loops on $(q, 0)$–bilabelled graph diagrams if G has loops (see Step 5 of the procedure). These points show that the set of $(q, 0)$–bilabelled graph diagrams that we need to consider depends on the graph G.

A General Procedure for Calculating the Weight Matrix of an $\mathrm{Aut}(G)$-Equivariant Linear Layer Function from $(\mathbb{R}^n)^{\otimes k}$ to $(\mathbb{R}^n)^{\otimes l}$.

First, calculate all $(q, 0)$–bilabelled graph diagrams that are appropriate for G where $q = l + k$, as follows. Let m be the maximum length of a path between any two vertices in G such that each edge in G is traversed at most once. Then

1. Calculate all set partitions of $\{1, \dots, q = l + k\}$, and express them as $(q, 0)$–bilabelled graph diagrams. For uniformity, we choose to order the blocks in a set partition by their size, in descending order. Each block B_i of size b_i corresponds to a spider diagram having a single red vertex and b_i black vertices connected to it with black edges. The vertices in the spider diagram for B_i are labelled with the elements of B_i in ascending order, from left to right. If $m = 0$, go to Step 6. Otherwise:

2. From all of the diagrams found in Step 1, create all possible $(q, 0)$–bilabelled graph diagrams that have only internal red edges between red vertices, as follows. For each diagram $\boldsymbol{H}$ found in Step 1:

   - Let c be the number of red vertices in $\boldsymbol{H}$. Let $e := \frac{c(c-1)}{2}$: this is the number of edges in the complete graph K_c having c vertices. Create all possible e length strings having values in $0 \to 2m$. Each position in the string represents a different pair of distinct red vertices in $\boldsymbol{H}$. Remove the all 0 string.

   - For each string, create a new $(q, 0)$–bilabelled graph diagram as follows: for each position in the string, if t is its value, then, assuming that the position represents the pair of red vertices v and w, insert t new edges between v and w into $\boldsymbol{H}$, adding in $t - 1$ new red vertices to make these t new edges possible.

3. Also, from all of the diagrams found in Step 1, create all possible $(q, 0)$–bilabelled graph diagrams that have only external red edges, as follows. For each diagram $\boldsymbol{H}$ found in Step 1:

   - Let c be the number of red vertices in $\boldsymbol{H}$. Create all possible c length strings having values in $0 \to m$, where each position in the string represents a red vertex in $\boldsymbol{H}$. Remove the all 0 string.

   - For each string, create a new $(q, 0)$–bilabelled graph diagram as follows: for each position in the string, if t is its value, then, assuming that the position represents the red vertex v, add t new edges outwards from v, adding in $t-1$ new red vertices to make these t new edges possible.

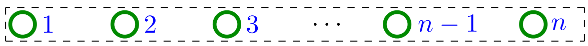
4. From all of the diagrams found in Step 2, create all possible $(q, 0)$–bilabelled graph diagrams having both internal and external red edges, as follows. For each diagram $\boldsymbol{H_I}$ found in Step 2:
   - Let $\boldsymbol{H}$ be the $(q, 0)$–bilabelled graph diagram from Step 1 that $\boldsymbol{H_I}$ came from. Let c be the number of red vertices in $\boldsymbol{H}$, and label these vertices in $\boldsymbol{H_I}$ as $1, \dots, c$. Create an empty set named *originals*, and add only the vertices $1, \dots, c$ in $\boldsymbol{H_I}$ that are not connected to any other red vertex in $\boldsymbol{H_I}$.
   - If *originals* is empty, then no new diagrams come from $\boldsymbol{H_I}$. Otherwise, create all possible c length strings, indexed by the vertices $1, \dots, c$, allowing only the values in the positions in *originals* to range from $0 \to m$, with the rest being 0. Remove the all 0 string. Now follow the instructions given in Step 3.2.
5. Finally, if G has loops, then, for each $(q, 0)$–bilabelled graph diagram found in Steps 1–4:
   - If c is the number of red vertices in $\boldsymbol{H}$, label these vertices as $1, \dots, c$, and create all binary strings of length c. For each string, create a new $(q, 0)$–bilabelled graph diagram as follows: for each position i in the string, if the value is 1, then attach a loop to the red vertex labelled as i.

The weight matrix is obtained from the set of all $(q, 0)$–bilabelled graph diagrams found in Steps 1–5, as follows:

6. Apply Frobenius duality to each $(q, 0)$–bilabelled graph diagram $\boldsymbol{H}$ to obtain its form as a (k, l)–bilabelled graph diagram. This is the same as dragging the labelled black vertices in $\boldsymbol{H}$ into two rows of vertices such that the vertices in the top row are ordered $1, \dots, l$ and the vertices in the bottom row are ordered $l+1, \dots, l+k$.
7. Apply the strict $\mathbb{R}$–linear monoidal functor $\mathcal{F}^G$ to each (k, l)–bilabelled graph diagram to obtain the spanning set matrices in $\mathrm{Hom}_{\mathrm{Aut}(G)}((\mathbb{R}^n)^{\otimes k}, (\mathbb{R}^n)^{\otimes l})$. Remove all duplicate matrices as well as the all zero matrices from this set. Weight each matrix that remains in the set by a parameter, and then add them all together to give the overall $\mathrm{Aut}(G)$-equivariant weight matrix.

8.2 Examples

Example 8.32 (Diagram Basis for $\mathrm{Hom}_{S_n}(\mathbb{R}^n, \mathbb{R}^n)$). Suppose we take the complement of the complete graph having n vertices, $\overline{K_n}$. There is only one possible way to label the vertices of this graph, up to all automorphisms:



In Section 2.4, we saw that its automorphism group, Aut($\overline{K_n}$), is the symmetric group S_n.

Since the order of each tensor power space is 1, by the arguments in the proof of Proposition 8.30, it is enough to consider the isomorphism classes of $(1, 1)$–bilabelled graphs $\boldsymbol{H}$ whose underlying graph H is edgeless having at most n vertices where no vertex is left free in $\boldsymbol{H}$. Because only at most two vertices cannot be left free for $(1, 1)$–bilabelled graphs $\boldsymbol{H}$, we only need to consider isomorphism classes of $(1, 1)$–bilabelled graphs $\boldsymbol{H}$ whose underlying graph H is edgeless having at most two vertices. This leads to two possibilities, namely $[\boldsymbol{H_1}]$ and $[\boldsymbol{H_2}]$, where $\boldsymbol{H_1} = (K_1, (1), (1))$ and $\boldsymbol{H_2} = (\overline{K_2}, (1), (2))$. They are given in the left hand column of Figure 8.4. (Note, for example, that the bilabelled graph $(\overline{K_2}, (2), (1))$ is in the same isomorphism class as $\boldsymbol{H_2}$.)

In the right hand column of Figure 8.4, we show the corresponding G-homomorphism matrices for the case where $G = \overline{K_4}$, that is, $n = 4$, with the case for general n being very similar. Here we have used Definition 8.12 to find the entries of each of the matrices. To be clear, for general n, the (i, j)-entry of $X_{\boldsymbol{H_1}}^{\overline{K_n}}$, by definition, is equal to the number of graph homomorphisms from K_1 to $\overline{K_n}$ such that 1 is mapped to i and 1 is mapped to j, which is $\delta_{i,j}$. By contrast, the (i, j)-entry of $X_{\boldsymbol{H_2}}^{\overline{K_n}}$ is equal to the number of graph homomorphisms from $\overline{K_2}$ to $\overline{K_n}$ such that 2 is mapped to i and 1 is mapped to j, which is 1 for all $i, j \in [n]$.

In the middle column, we have also shown the equivalent set partition diagrams that first appeared in Figure 6.1 to highlight how the $(1, 1)$–bilabelled graph diagrams are related to set partition diagrams in this case. This is also consistent with the procedure that is given in the orange box since the longest path in $\overline{K_n}$ is 0. Hence, only the set partitions of $\{1, 2\}$, expressed as $(2, 0)$–bilabelled graph diagrams that are then mapped under Frobenius duality to $(1, 1)$–bilabelled graph diagrams, are needed to obtain a spanning set — which is actually a basis — of $\mathrm{Hom}_{S_n}(\mathbb{R}^n, \mathbb{R}^n)$.

Example 8.33 (Basis for $\mathrm{Hom}_{D_4}(\mathbb{R}^4, \mathbb{R}^4)$). In Section 2.4, we said that the automorphism group of two copies of the complete graph on two vertices, Aut($2K_2$), is isomorphic to the dihedral group D_4.

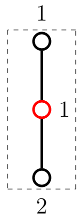
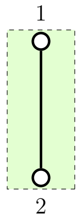
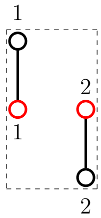
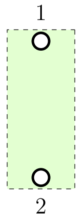
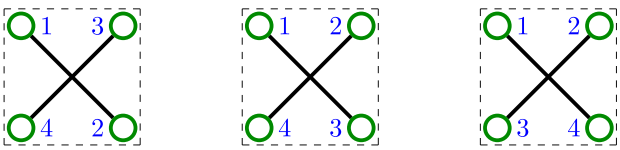
$(1,1)$–Bilabelled Graph Diagram $\boldsymbol{H}$	Set Partition Diagram	Standard Basis Element $X_{\boldsymbol{H}}^{\overline{K_4}}$
		$\begin{array}{cc} & \begin{matrix}1&2&3&4\end{matrix} \\ \begin{matrix}1\\2\\3\\4\end{matrix} & \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}\end{array}$
		$\begin{array}{cc} & \begin{matrix}1&2&3&4\end{matrix} \\ \begin{matrix}1\\2\\3\\4\end{matrix} & \begin{bmatrix}1&1&1&1\\1&1&1&1\\1&1&1&1\\1&1&1&1\end{bmatrix}\end{array}$

Figure 8.4: We use Theorem 8.24 to obtain the diagram basis of $\mathrm{Hom}_{S_4}(\mathbb{R}^4, \mathbb{R}^4)$ from all of the $(1,1)$–bilabelled graph diagrams.

There are three different ways to label the vertices of the graph $2K_2$, up to all automorphisms, namely:



We will refer to these graphs as A, B and C, respectively, throughout the rest of this example.

The automorphism group of each graph is isomorphic to D_4, but each automorphism group is a different embedding of D_4 in S_4. Said differently, while each automorphism group is isomorphic to D_4, the elements in each automorphism group are not the same when thought of as elements of S_4. Specifically, the group corresponding to the first diagram is $\langle(1423), (13)(24)\rangle$, whereas the second is $\langle(1432), (12)(34)\rangle$, and the third is $\langle(1342), (12)(34)\rangle$.

Consequently, when we would like to find a spanning set for $\mathrm{Hom}_{D_4}(\mathbb{R}^4, \mathbb{R}^4)$, we need to consider which embedding of D_4 in S_4 we are referring to, or, more simply, which labels we have used for the graph $2K_2$. As a result, we will obtain a spanning set (which we will show is actually a basis) for $\mathrm{Hom}_{D_4}(\mathbb{R}^4, \mathbb{R}^4)$, one for each embedding of D_4 in S_4. Note that these spanning sets, whilst different, will all be isomorphic under a change of basis that reorders the standard basis vectors in $\mathbb{R}^4$. This is a consequence of all of the automorphism groups being isomorphic to each other.

We present the bases for each instance of $2K_2$ in Figure 8.5.

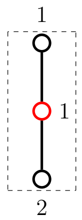
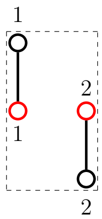
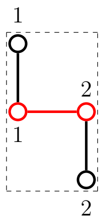
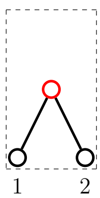
$(1,1)$–Bilabelled Graph Diagram $\boldsymbol{H}$	Standard Basis Element $X_{\boldsymbol{H}}^{A}$	Standard Basis Element $X_{\boldsymbol{H}}^{B}$	Standard Basis Element $X_{\boldsymbol{H}}^{C}$
	$\begin{array}{cc} & \begin{matrix}1&2&3&4\end{matrix} \\ \begin{matrix}1\\2\\3\\4\end{matrix} & \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}\end{array}$	$\begin{array}{cc} & \begin{matrix}1&2&3&4\end{matrix} \\ \begin{matrix}1\\2\\3\\4\end{matrix} & \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}\end{array}$	$\begin{array}{cc} & \begin{matrix}1&2&3&4\end{matrix} \\ \begin{matrix}1\\2\\3\\4\end{matrix} & \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}\end{array}$
	$\begin{array}{cc} & \begin{matrix}1&2&3&4\end{matrix} \\ \begin{matrix}1\\2\\3\\4\end{matrix} & \begin{bmatrix}1&1&1&1\\1&1&1&1\\1&1&1&1\\1&1&1&1\end{bmatrix}\end{array}$	$\begin{array}{cc} & \begin{matrix}1&2&3&4\end{matrix} \\ \begin{matrix}1\\2\\3\\4\end{matrix} & \begin{bmatrix}1&1&1&1\\1&1&1&1\\1&1&1&1\\1&1&1&1\end{bmatrix}\end{array}$	$\begin{array}{cc} & \begin{matrix}1&2&3&4\end{matrix} \\ \begin{matrix}1\\2\\3\\4\end{matrix} & \begin{bmatrix}1&1&1&1\\1&1&1&1\\1&1&1&1\\1&1&1&1\end{bmatrix}\end{array}$
	$\begin{array}{cc} & \begin{matrix}1&2&3&4\end{matrix} \\ \begin{matrix}1\\2\\3\\4\end{matrix} & \begin{bmatrix}0&1&0&0\\1&0&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}\end{array}$	$\begin{array}{cc} & \begin{matrix}1&2&3&4\end{matrix} \\ \begin{matrix}1\\2\\3\\4\end{matrix} & \begin{bmatrix}0&0&1&0\\0&0&0&1\\1&0&0&0\\0&1&0&0\end{bmatrix}\end{array}$	$\begin{array}{cc} & \begin{matrix}1&2&3&4\end{matrix} \\ \begin{matrix}1\\2\\3\\4\end{matrix} & \begin{bmatrix}0&0&0&1\\0&0&1&0\\0&1&0&0\\1&0&0&0\end{bmatrix}\end{array}$

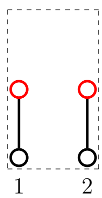
Figure 8.5: Depending on how D_4 is embedded as a subgroup of S_4, we obtain a basis of $\mathrm{Hom}_{D_4}(\mathbb{R}^4, \mathbb{R}^4)$, where here D_4 refers to the specific embedding in S_4 that is obtained from the different labellings of the vertices of $2K_2$, considered up to all automorphisms.

We focus on finding the basis for the automorphism group of the graph A, noting that the approach for the other graphs is similar. We follow the procedure that is given in the orange box. We see that the longest path m between any two vertices in A is 1.

Step 1: We calculate all set partitions of $[l+k] = \{1, 2\}$ and express them as $(2,0)$–bilabelled graph diagrams, labelling only the black vertices.

They are given by

$\boldsymbol{A_0} =$ 

and $\boldsymbol{B_0} =$ 

(8.24)

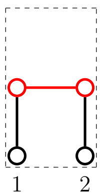
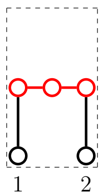
Step 2: From $\boldsymbol{A_0}$ and $\boldsymbol{B_0}$, we calculate all possible $(2,0)$–bilabelled graph diagrams that have only internal red edges between red vertices.

Since the number of red vertices c in $\boldsymbol{A_0}$ is 1, this implies that the number of edges in the complete graph K_1 is 0, and so we do not obtain any new $(2,0)$–bilabelled graph diagrams from $\boldsymbol{A_0}$.

However, for $\boldsymbol{B_0}$:

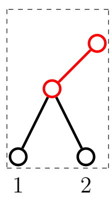
- The number of red vertices, c, in $\boldsymbol{B_0}$ is 2. Hence the number of edges in the complete graph K_2, e, is 1.
- We now create all possible length $e = 1$ strings having values in $0 \to 2m = 2$. Remove the all 0 string.

Hence we obtain the length one strings 1 and 2. We use each of these strings to create a new $(2,0)$–bilabelled graph diagram from $\boldsymbol{B_0}$, which we shall call $\boldsymbol{B_{1,1}}$ and $\boldsymbol{B_{1,2}}$, by inserting t new edges between the two red vertices, where t equals 1 and 2 respectively. Hence, $\boldsymbol{B_{1,1}}$ and $\boldsymbol{B_{1,2}}$ are given as follows:

$\boldsymbol{B_{1,1}} =$  ; $\boldsymbol{B_{1,2}} =$  (8.25)

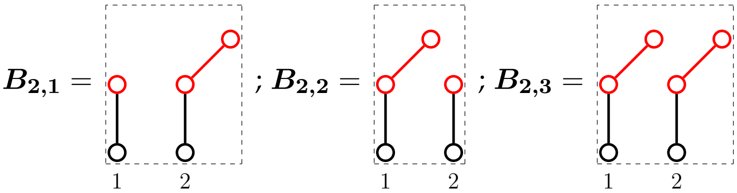
Step 3: From $\boldsymbol{A_0}$ and $\boldsymbol{B_0}$, we now calculate all possible $(2,0)$–bilabelled graph diagrams that have only external red edges between red vertices.

For $\boldsymbol{A_0}$, as the number of red vertices is 1, we create all length one strings having entries in 0 to $m = 1$, removing the all 0 string. We create a new $(2,0)$–bilabelled graph diagram from the string 1 by adding 1 new red edge outwards from the red vertex in $\boldsymbol{A_0}$, adding in a new red vertex to make this new red edge possible. Hence we obtain

$\boldsymbol{A_2} =$  (8.26)

For $\boldsymbol{B_0}$, as the number of red vertices is 2, we create all length two strings having entries in 0 to $m = 1$, removing the all 0 string. Hence we create three new $(2,0)$–bilabelled graph diagrams from the strings

01, 10 and 11, by adding 1 new red edge outwards from the red vertices in $\boldsymbol{A_0}$ corresponding to the ones in the string, adding in a new red vertex for each new red edge. Hence we obtain



(8.27)

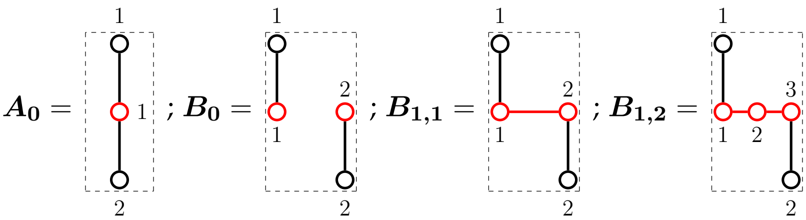
Step 4: We calculate all possible $(2, 0)$–bilabelled graph diagrams that have external red edges from $\boldsymbol{B_{1,1}}$ and $\boldsymbol{B_{2,2}}$.

We do not create any new $(2, 0)$–bilabelled graph diagrams in this step as the set *originals* is empty for $\boldsymbol{B_{1,1}}$ and $\boldsymbol{B_{2,2}}$.

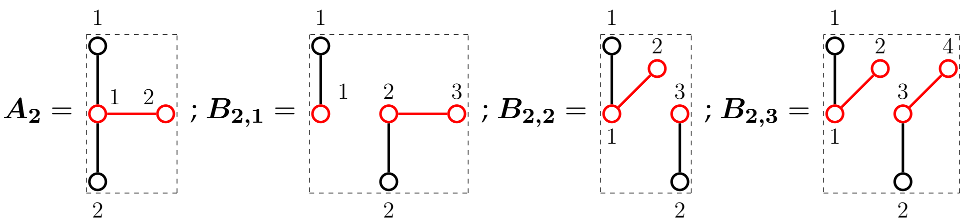
Step 5: As A does not have any loops, we immediately move onto Step 6.

Step 6: We now apply Frobenius duality to each $(2, 0)$–bilabelled graph diagram found in Steps 1 to 5 inclusive to obtain their form as $(1, 1)$–bilabelled graph diagrams.

This is equivalent to dragging the black vertex labelled 1 up to the top row. At this stage, we choose to arbitrarily label the red vertices as well. Using the same names for the bilabelled graph diagrams, we obtain:



(8.28)



(8.29)

Step 7: We calculate the A-homomorphism matrices that correspond to the $(1, 1)$–bilabelled graph diagrams given in Step 6. We remove all duplicate matrices as well as any all zero matrices, weight those that remain, and then add them together to obtain the weight matrix for an $\text{Aut}(A) \cong D_4$-equivariant linear layer function from $\mathbb{R}^4$ to $\mathbb{R}^4$.

The A-homomorphism matrices that correspond to the $(1,1)$–bilabelled graph diagrams given in (8.28) are

$$\begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 1 & 0 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 \\ 3 & 0 & 0 & 1 & 0 \\ 4 & 0 & 0 & 0 & 1 \end{array} ; \quad \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 1 & 1 & 1 & 1 \\ 2 & 1 & 1 & 1 & 1 \\ 3 & 1 & 1 & 1 & 1 \\ 4 & 1 & 1 & 1 & 1 \end{array} ; \quad \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 1 & 0 & 0 \\ 2 & 1 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 1 \\ 4 & 0 & 0 & 1 & 0 \end{array} \quad \text{and} \quad \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 1 & 0 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 \\ 3 & 0 & 0 & 1 & 0 \\ 4 & 0 & 0 & 0 & 1 \end{array} \tag{8.30}$$

whereas the A-homomorphism matrices that correspond to the $(1,1)$–bilabelled graph diagrams given in (8.29) are

$$\begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 1 & 0 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 \\ 3 & 0 & 0 & 1 & 0 \\ 4 & 0 & 0 & 0 & 1 \end{array} ; \quad \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 1 & 1 & 1 & 1 \\ 2 & 1 & 1 & 1 & 1 \\ 3 & 1 & 1 & 1 & 1 \\ 4 & 1 & 1 & 1 & 1 \end{array} ; \quad \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 1 & 1 & 1 & 1 \\ 2 & 1 & 1 & 1 & 1 \\ 3 & 1 & 1 & 1 & 1 \\ 4 & 1 & 1 & 1 & 1 \end{array} \quad \text{and} \quad \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 1 & 1 & 1 & 1 \\ 2 & 1 & 1 & 1 & 1 \\ 3 & 1 & 1 & 1 & 1 \\ 4 & 1 & 1 & 1 & 1 \end{array} \tag{8.31}$$

respectively.

Clearly, only the A-homomorphism matrices corresponding to $\boldsymbol{A_0}$, $\boldsymbol{B_0}$ and $\boldsymbol{B_{1,1}}$ are unique, and so they form a spanning set for $\mathrm{Hom}_{D_4}(\mathbb{R}^4, \mathbb{R}^4)$, for $D_4 = \mathrm{Aut}(A) = \langle (1423), (12)(34) \rangle$, considered as a subgroup of S_4.

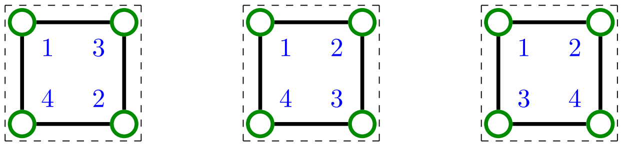
Hence, the weight matrix that we obtain for an $\mathrm{Aut}(A) \cong D_4$-equivariant linear layer function from $\mathbb{R}^4$ to $\mathbb{R}^4$ is

$$\begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & \lambda_{1,2} & \lambda_{2,3} & \lambda_2 & \lambda_2 \\ 2 & \lambda_{2,3} & \lambda_{1,2} & \lambda_2 & \lambda_2 \\ 3 & \lambda_2 & \lambda_2 & \lambda_{1,2} & \lambda_{2,3} \\ 4 & \lambda_2 & \lambda_2 & \lambda_{2,3} & \lambda_{1,2} \end{array} \tag{8.32}$$

for weights $\lambda_1, \lambda_2, \lambda_3 \in \mathbb{R}$, where $\lambda_{i,j}$ means $\lambda_i + \lambda_j$.

Note that, for this example, each spanning set corresponding to an instance of $2K_2$ happens to be a basis of $\mathrm{Hom}_{D_4}(\mathbb{R}^4, \mathbb{R}^4)$. Indeed, for each instance of $2K_2$, the subset consisting of the first two matrices in each column in Figure 8.5 is a basis for $\mathrm{Hom}_{S_4}(\mathbb{R}^4, \mathbb{R}^4)$, by Example 8.32, and since D_4 is a proper subgroup of S_4 and the spanning set for $\mathrm{Hom}_{D_4}(\mathbb{R}^4, \mathbb{R}^4)$ includes one further matrix that is not a linear combination of these two matrices, this implies that it must be a basis.

Note further that we could have used the cycle graph C_4 to obtain automorphism groups that are isomorphic to D_4. In fact, if we label the cycle graphs as follows, we obtain



which are the complements of the three instances A, B, C of $2K_2$. This is clear since $C_4 = \overline{2K_2}$. In fact, in this case, we actually have that $\text{Aut}(C_4) = \text{Aut}(2K_2)$, considered as subgroups of S_4, for each of the three possible labellings of the vertices.

However, the basis for $\text{Hom}_{D_4}(\mathbb{R}^4, \mathbb{R}^4)$ for each version of D_4 coming from the three instances $\overline{A}, \overline{B}, \overline{C}$ of C_4 is different, but related, to each of those for the three instances A, B, C of $2K_2$. We present these bases in Figure 8.6. Note that only the matrix corresponding to the third $(1, 1)$–bilabelled graph diagram $\boldsymbol{B_{1,1}}$ has changed, so even though the embeddings of D_4 in S_4 are the same for a given labelling of the vertices, how those vertices are connected (resulting in either the square C_4 or its complement $2K_2$) affects the basis that we obtain for $\text{Hom}_{D_4}(\mathbb{R}^4, \mathbb{R}^4)$. In fact, it is clear that, for each of $G = A, B, C$, we have that

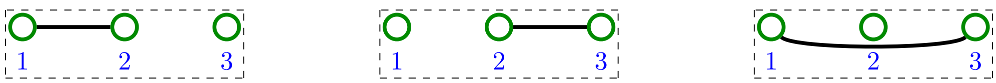
$$X^{\overline{G}}_{\boldsymbol{B_{1,1}}} = J - I - X^{G}_{\boldsymbol{B_{1,1}}} \tag{8.33}$$

where J is the all ones matrix and I is the identity matrix. By Example 8.14, we see that this is, in fact, the equation

$$A_{\overline{G}} = J - I - A_G \tag{8.34}$$

which describes the commonly known result of how to obtain the adjacency matrix of the complement of a graph G from the adjacency matrix of the original graph G!

Example 8.34 (Spanning Set for $\text{Hom}_{S_2}(\mathbb{R}^3, \mathbb{R}^3)$). It is clear that, for a graph G having $n = 3$ vertices such that its automorphism group is S_2, there are three different ways to label the vertices of the graph, up to all automorphisms, namely:



We will refer to these graphs as A, B and C, respectively, throughout the rest of this example.

We focus on graph A, with the approach for the other graphs being similar. In this case, $S_2 = \text{Aut}(A) = \{id, (12)\}$, as a subgroup of S_3.

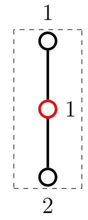
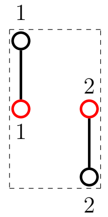
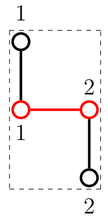
$(1,1)$–Bilabelled Graph Diagram $\boldsymbol{H}$	Standard Basis Element $X_{\boldsymbol{H}}^{\overline{A}}$	Standard Basis Element $X_{\boldsymbol{H}}^{\overline{B}}$	Standard Basis Element $X_{\boldsymbol{H}}^{\overline{C}}$
	$\begin{array}{cc} & \begin{array}{cccc}1&2&3&4\end{array} \\ \begin{array}{c}1\\2\\3\\4\end{array} & \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}\end{array}$	$\begin{array}{cc} & \begin{array}{cccc}1&2&3&4\end{array} \\ \begin{array}{c}1\\2\\3\\4\end{array} & \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}\end{array}$	$\begin{array}{cc} & \begin{array}{cccc}1&2&3&4\end{array} \\ \begin{array}{c}1\\2\\3\\4\end{array} & \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}\end{array}$
	$\begin{array}{cc} & \begin{array}{cccc}1&2&3&4\end{array} \\ \begin{array}{c}1\\2\\3\\4\end{array} & \begin{bmatrix}1&1&1&1\\1&1&1&1\\1&1&1&1\\1&1&1&1\end{bmatrix}\end{array}$	$\begin{array}{cc} & \begin{array}{cccc}1&2&3&4\end{array} \\ \begin{array}{c}1\\2\\3\\4\end{array} & \begin{bmatrix}1&1&1&1\\1&1&1&1\\1&1&1&1\\1&1&1&1\end{bmatrix}\end{array}$	$\begin{array}{cc} & \begin{array}{cccc}1&2&3&4\end{array} \\ \begin{array}{c}1\\2\\3\\4\end{array} & \begin{bmatrix}1&1&1&1\\1&1&1&1\\1&1&1&1\\1&1&1&1\end{bmatrix}\end{array}$
	$\begin{array}{cc} & \begin{array}{cccc}1&2&3&4\end{array} \\ \begin{array}{c}1\\2\\3\\4\end{array} & \begin{bmatrix}0&0&1&1\\0&0&1&1\\1&1&0&0\\1&1&0&0\end{bmatrix}\end{array}$	$\begin{array}{cc} & \begin{array}{cccc}1&2&3&4\end{array} \\ \begin{array}{c}1\\2\\3\\4\end{array} & \begin{bmatrix}0&1&0&1\\1&0&1&0\\0&1&0&1\\1&0&1&0\end{bmatrix}\end{array}$	$\begin{array}{cc} & \begin{array}{cccc}1&2&3&4\end{array} \\ \begin{array}{c}1\\2\\3\\4\end{array} & \begin{bmatrix}0&1&1&0\\1&0&0&1\\1&0&0&1\\0&1&1&0\end{bmatrix}\end{array}$

Figure 8.6: Choosing the cycle graph C_4 instead to obtain an embedding of D_4 inside S_4, we obtain a basis of $\mathrm{Hom}_{D_4}(\mathbb{R}^4, \mathbb{R}^4)$, one for each possible labelling of the vertices of C_4, that is related to the basis obtained from the complement graph, $2K_2$.

Given that the longest path m between any two vertices in A is 1, and that $l = k = 1$, we see that, by following the procedure that is given in the orange box, Steps 1 to 6 inclusive are exactly the same as in Example 8.33. Hence, the $(1,1)$–bilabelled graph diagrams that we need to consider are those that are given in (8.28) and (8.29).

However, given that the graph A is different from the one that appears in Example 8.33, the A-homomorphism matrices that correspond to the $(1,1)$–bilabelled graph diagrams given in (8.28) are now

$$\begin{array}{cc} & \begin{array}{ccc}1&2&3\end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix}\end{array} ; \begin{array}{cc} & \begin{array}{ccc}1&2&3\end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}\end{array} ; \begin{array}{cc} & \begin{array}{ccc}1&2&3\end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \begin{bmatrix}0&1&0\\1&0&0\\0&0&0\end{bmatrix}\end{array} \text{ and } \begin{array}{cc} & \begin{array}{ccc}1&2&3\end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \begin{bmatrix}1&0&0\\0&1&0\\0&0&0\end{bmatrix}\end{array} \tag{8.35}$$

and the A-homomorphism matrices that correspond to the $(1,1)$–bilabelled graph diagrams given in

(8.29) are now

$$\begin{array}{c|ccc} & 1 & 2 & 3 \\ \hline 1 & 1 & 0 & 0 \\ 2 & 0 & 1 & 0 \\ 3 & 0 & 0 & 0 \end{array} \; ; \; \begin{array}{c|ccc} & 1 & 2 & 3 \\ \hline 1 & 1 & 1 & 0 \\ 2 & 1 & 1 & 0 \\ 3 & 1 & 1 & 0 \end{array} \; ; \; \begin{array}{c|ccc} & 1 & 2 & 3 \\ \hline 1 & 1 & 1 & 1 \\ 2 & 1 & 1 & 1 \\ 3 & 0 & 0 & 0 \end{array} \text{ and } \begin{array}{c|ccc} & 1 & 2 & 3 \\ \hline 1 & 1 & 1 & 0 \\ 2 & 1 & 1 & 0 \\ 3 & 0 & 0 & 0 \end{array} \tag{8.36}$$

All of the A-homomorphism matrices are unique except for those coming from $\boldsymbol{B_{1,2}}$ and $\boldsymbol{A_2}$, and so by removing the matrix corresponding to $\boldsymbol{A_2}$, we obtain a spanning set $\mathrm{Hom}_{S_2}(\mathbb{R}^3, \mathbb{R}^3)$, for $S_2 = \mathrm{Aut}(A) = \langle (id, (12) \rangle$, as a subgroup of S_3. We label the remaining matrices in the order in which they are presented using the index set $\{1, \dots, 7\}$.

Hence, the weight matrix that we obtain for an $\mathrm{Aut}(A) \cong S_2$-equivariant linear layer function from $\mathbb{R}^3$ to $\mathbb{R}^3$ is

$$\begin{array}{c|ccc} & 1 & 2 & 3 \\ \hline 1 & \lambda_{1,2,4,5,6,7} & \lambda_{2,3,5,6,7} & \lambda_{2,6} \\ 2 & \lambda_{2,3,5,6,7} & \lambda_{1,2,4,5,6,7} & \lambda_{2,6} \\ 3 & \lambda_{2,5} & \lambda_{2,5} & \lambda_{1,2} \end{array} \tag{8.37}$$

for weights $\lambda_1, \dots, \lambda_7 \in \mathbb{R}$.

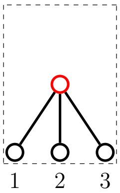
We can find the elements of the spanning set for $\mathrm{Hom}_{S_2}(\mathbb{R}^3, \mathbb{R}^3)$ for the other two embeddings of S_2 in S_3, given by $\mathrm{Aut}(B)$ and $\mathrm{Aut}(C)$. For $\mathrm{Aut}(B)$, we perform the permutation (13) on each index of the rows and columns of the spanning set matrices found for $\mathrm{Aut}(A)$, and for $\mathrm{Aut}(C)$, we perform the permutation (23) instead.

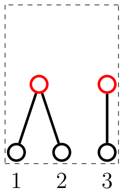
Example 8.35 (Spanning Set for $\mathrm{Hom}_{S_2}((\mathbb{R}^3)^{\otimes 2}, \mathbb{R}^3)$). We refer to the same three graphs, A, B, C, that were given in Example 8.34, and, once again, focus on graph A, with the approach for the other graphs being similar.

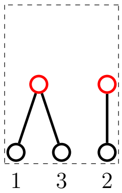
Recall that, in this case, $S_2 = \mathrm{Aut}(A) = \{id, (12)\}$, as a subgroup of S_3. We still have that the longest path in A is 1. We follow the procedure that is given in the orange box. However, in order to keep the A-homomorphism matrices in close proximity with the bilabelled graph diagrams that they correspond to, we have chosen to apply Step 6 and the first part of Step 7 during Steps 1 through 5 inclusive.

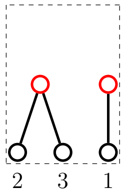
Step 1: We calculate all set partitions of $[l + k] = \{1, 2, 3\}$ and express them as $(3, 0)$–bilabelled graph diagrams, labelling only the black vertices.

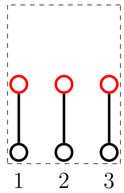
They are given by

$A_0 =$ 

; $B_0 =$ 

; $C_0 =$ 

; $D_0 =$ 

and $E_0 =$ 

(8.38)

and correspond, after Frobenius duality, to the A-homomorphism matrices

$$\begin{array}{cc} & \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc|ccc|ccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right] \end{array} ; \begin{array}{cc} & \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc|ccc|ccc} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{array}\right] \end{array} ; \begin{array}{cc} & \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc|ccc|ccc} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{array}\right] \end{array} \tag{8.39}$$

$$\begin{array}{cc} & \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc|ccc|ccc} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{array}\right] \end{array} \text{ and } \begin{array}{cc} & \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc|ccc|ccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}\right] \end{array} \tag{8.40}$$

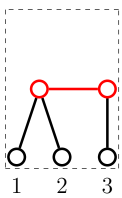
Step 2: We calculate all possible $(3, 0)$–bilabelled graph diagrams that have only internal red edges between red vertices from the five bilabelled graph diagrams given in Step 1.

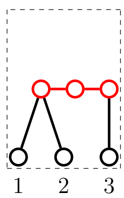
Clearly, there are no new bilabelled graph diagrams coming from $\boldsymbol{A_0}$ in this step.

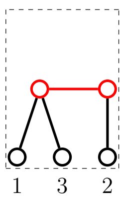
However, for $\boldsymbol{B_0}$, $\boldsymbol{C_0}$ and $\boldsymbol{D_0}$, calling these generically by $\boldsymbol{H}$, we have that

- The number of red vertices, c, in $\boldsymbol{H}$ is 2. Hence the number of edges in the complete graph K_2, e, is 1.
- We now create all possible length $e = 1$ strings having values in $0 \to 2m = 2$. Remove the all 0 string.

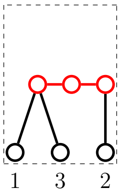
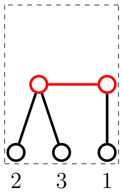
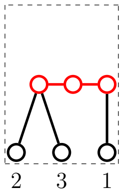
Hence we obtain the length one strings 1 and 2. We use these strings to create two new $(3, 0)$–bilabelled graph diagrams from each of $\boldsymbol{B_0}$, $\boldsymbol{C_0}$ and $\boldsymbol{D_0}$, by inserting t new edges between the two red vertices, where t equals 1 and 2, respectively. The new $(3, 0)$–bilabelled graph diagrams are given as follows:

$\boldsymbol{B_{1,1}} =$ 

; $\boldsymbol{B_{1,2}} =$ 

; $\boldsymbol{C_{1,1}} =$ 

(8.41)

$$C_{1,2} =$$  $$;\ D_{1,1} =$$  $$;\ D_{1,2} =$$  (8.42)

They correspond, after Frobenius duality, to the A-homomorphism matrices

$$\begin{array}{cc} & \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc|ccc|ccc} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} ; \begin{array}{cc} & \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc|ccc|ccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} ; \begin{array}{cc} & \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc|ccc|ccc} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} \tag{8.43}$$

$$\begin{array}{cc} & \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc|ccc|ccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} ; \begin{array}{cc} & \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc|ccc|ccc} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} ; \begin{array}{cc} & \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc|ccc|ccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} \tag{8.44}$$

For $\boldsymbol{E}_0$, we have that

- The number of red vertices, c, in $\boldsymbol{E}_0$ is 3. Hence the number of edges in the complete graph K_3, e, is 3.
- We now create all possible length $e = 3$ strings having values in $0 \to 2m = 2$. Remove the all 0 string.

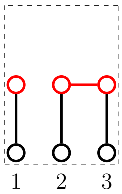
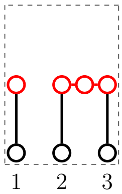
There are 26 such strings. If we label the red vertices in $\boldsymbol{E}_0$, without loss of generality, as 1, 2 and 3, from left to right, then we can let the indices in each string refer to possible connections between vertices 1 and 2, vertices 1 and 3, and vertices 2 and 3, written (12), (13) and (23), also from left to right and without loss of generality.

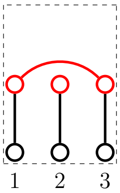
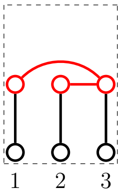
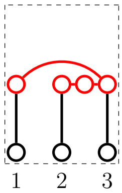
Table 8.1: All Possible Length 3 Strings With Entries in $0 \to 2m = 2$.

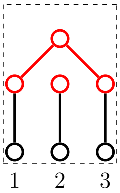
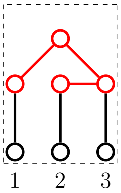
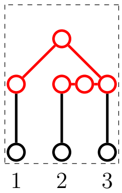
(12)(13)(23)	(12)(13)(23)	(12)(13)(23)
000	001	002
010	011	012
020	021	022
100	101	102
110	111	112
120	121	122
200	201	202
210	211	212
220	221	222

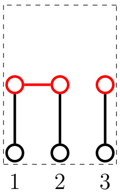
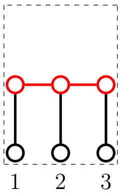
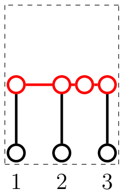
To create the 26 new $(3, 0)$–bilabelled graph diagrams, we now add in the edges (and new red vertices) between the labelled red vertices in $\boldsymbol{E}_0$, according to the entries in the string.

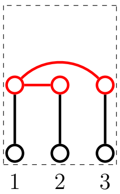
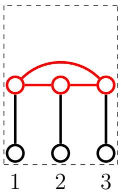
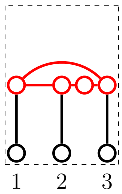
They are

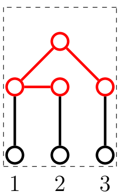
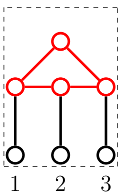
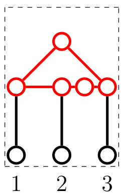
$\boldsymbol{E}_{001} =$  ; $\boldsymbol{E}_{002} =$  (8.45)

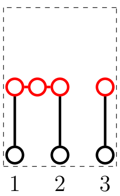
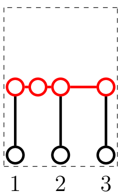
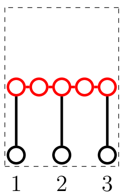
$\boldsymbol{E}_{010} =$  ; $\boldsymbol{E}_{011} =$  ; $\boldsymbol{E}_{012} =$  (8.46)

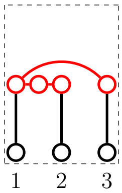
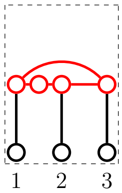
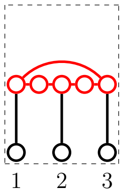
$\boldsymbol{E}_{020} =$  ; $\boldsymbol{E}_{021} =$  ; $\boldsymbol{E}_{022} =$  (8.47)

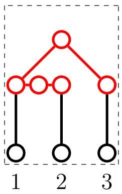
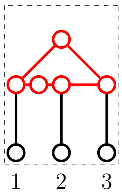
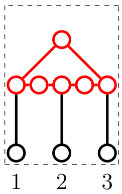
$\boldsymbol{E}_{100} =$  ; $\boldsymbol{E}_{101} =$  ; $\boldsymbol{E}_{102} =$  (8.48)

$\boldsymbol{E}_{110} =$  ; $\boldsymbol{E}_{111} =$  ; $\boldsymbol{E}_{112} =$  (8.49)

$\boldsymbol{E}_{120} =$  ; $\boldsymbol{E}_{121} =$  ; $\boldsymbol{E}_{122} =$  (8.50)

$\boldsymbol{E}_{200} =$  ; $\boldsymbol{E}_{201} =$  ; $\boldsymbol{E}_{202} =$  (8.51)

$\boldsymbol{E}_{210} =$  $;\ \boldsymbol{E}_{211} =$  $;\ \boldsymbol{E}_{212} =$  (8.52)

$\boldsymbol{E}_{220} =$  $;\ \boldsymbol{E}_{221} =$  $;\ \boldsymbol{E}_{222} =$  (8.53)

and correspond, after Frobenius duality, to the A-homomorphism matrices

$$\begin{array}{rl} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \left[\begin{array}{ccc|ccc|ccc} 0&1&0&1&0&0&0&0&0\\ 0&1&0&1&0&0&0&0&0\\ 0&1&0&1&0&0&0&0&0 \end{array}\right] \end{array} ; \begin{array}{rl} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \left[\begin{array}{ccc|ccc|ccc} 1&0&0&0&1&0&0&0&0\\ 1&0&0&0&1&0&0&0&0\\ 1&0&0&0&1&0&0&0&0 \end{array}\right] \end{array} \tag{8.54}$$

$$\begin{array}{rl} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \left[\begin{array}{ccc|ccc|ccc} 0&1&0&0&1&0&0&1&0\\ 1&0&0&1&0&0&1&0&0\\ 0&0&0&0&0&0&0&0&0 \end{array}\right] \end{array} ; \begin{array}{rl} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \left[\begin{array}{ccc|ccc|ccc} 0&1&0&0&1&0&0&0&0\\ 1&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0 \end{array}\right] \end{array} ; \begin{array}{rl} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \left[\begin{array}{ccc|ccc|ccc} 0&0&0&0&1&0&0&0&0\\ 1&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0 \end{array}\right] \end{array} \tag{8.55}$$

$$\begin{array}{rl} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \left[\begin{array}{ccc|ccc|ccc} 1&0&0&1&0&0&1&0&0\\ 0&1&0&0&1&0&0&1&0\\ 0&0&0&0&0&0&0&0&0 \end{array}\right] \end{array} ; \begin{array}{rl} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \left[\begin{array}{ccc|ccc|ccc} 0&0&0&1&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0 \end{array}\right] \end{array} ; \begin{array}{rl} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \left[\begin{array}{ccc|ccc|ccc} 1&0&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&0&0 \end{array}\right] \end{array} \tag{8.56}$$

$$\begin{array}{rl} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \left[\begin{array}{ccc|ccc|ccc} 0&0&0&1&1&1&0&0&0\\ 1&1&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0 \end{array}\right] \end{array} ; \begin{array}{rl} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \left[\begin{array}{ccc|ccc|ccc} 0&0&0&1&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0 \end{array}\right] \end{array} ; \begin{array}{rl} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \left[\begin{array}{ccc|ccc|ccc} 0&0&0&0&1&0&0&0&0\\ 1&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0 \end{array}\right] \end{array} \tag{8.57}$$

$$\begin{array}{rl} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \left[\begin{array}{ccc|ccc|ccc} 0&0&0&0&1&0&0&0&0\\ 1&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0 \end{array}\right] \end{array} ; \begin{array}{rl} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \left[\begin{array}{ccc|ccc|ccc} 0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0 \end{array}\right] \end{array} ; \begin{array}{rl} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \left[\begin{array}{ccc|ccc|ccc} 0&0&0&0&1&0&0&0&0\\ 1&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0 \end{array}\right] \end{array} \tag{8.58}$$

$$\begin{array}{rl} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \left[\begin{array}{ccc|ccc|ccc} 0&0&0&1&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0 \end{array}\right] \end{array} ; \begin{array}{rl} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \left[\begin{array}{ccc|ccc|ccc} 0&0&0&1&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0 \end{array}\right] \end{array} ; \begin{array}{rl} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \left[\begin{array}{ccc|ccc|ccc} 0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0 \end{array}\right] \end{array} \tag{8.59}$$

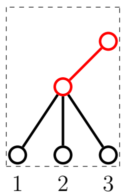
$$\begin{array}{rl} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \left[\begin{array}{ccc|ccc|ccc} 1&1&1&0&0&0&0&0&0\\ 0&0&0&1&1&1&0&0&0\\ 0&0&0&0&0&0&0&0&0 \end{array}\right] \end{array} ; \begin{array}{rl} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \left[\begin{array}{ccc|ccc|ccc} 0&1&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0 \end{array}\right] \end{array} ; \begin{array}{rl} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \left[\begin{array}{ccc|ccc|ccc} 1&0&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&0&0 \end{array}\right] \end{array} \tag{8.60}$$

$$\begin{array}{rl} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \left[\begin{array}{ccc|ccc|ccc} 0&1&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0 \end{array}\right] \end{array} ; \begin{array}{rl} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \left[\begin{array}{ccc|ccc|ccc} 0&1&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0 \end{array}\right] \end{array} ; \begin{array}{rl} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c}1\\2\\3\end{array} & \left[\begin{array}{ccc|ccc|ccc} 1&0&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&0&0 \end{array}\right] \end{array} \tag{8.61}$$

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array}\begin{array}{c} \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \left[\begin{array}{ccc|ccc|ccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} ; \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array}\begin{array}{c} \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \left[\begin{array}{ccc|ccc|ccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} ; \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array}\begin{array}{c} \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \left[\begin{array}{ccc|ccc|ccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} \tag{8.62}$$

Step 3: We calculate all possible $(3, 0)$–bilabelled graph diagrams that have only external red edges between red vertices from the five bilabelled graph diagrams given in Step 1.

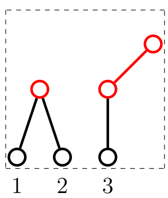
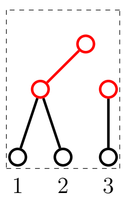
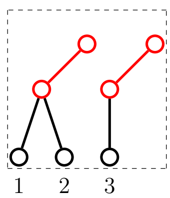
For $\boldsymbol{A_0}$, as the number of red vertices is 1, we create all length one strings having entries in 0 to $m = 1$, removing the all 0 string. As a result, we create a new $(3, 0)$–bilabelled graph diagram from the string 1 by adding 1 new red edge outwards from the red vertex in $\boldsymbol{A_0}$, adding in a new red vertex to make this new red edge possible. Hence we obtain

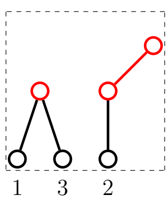
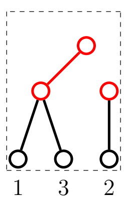
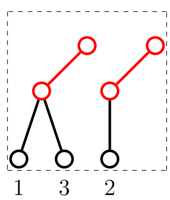
$\boldsymbol{A_2} =$  (8.63)

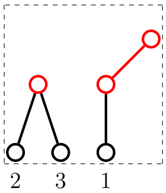
which, after Frobenius duality, corresponds to the A-homomorphism matrix

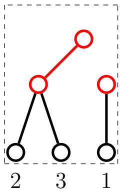
$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array}\begin{array}{c} \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \left[\begin{array}{ccc|ccc|ccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} \tag{8.64}$$

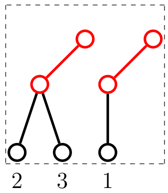
However, for $\boldsymbol{B_0}$, $\boldsymbol{C_0}$ and $\boldsymbol{D_0}$, calling these generically by $\boldsymbol{H}$, we know that the number of red vertices, c, in $\boldsymbol{H}$ is 2. As a result, we create all length two strings having entries in 0 to $m = 1$, removing the all 0 string. There are three such strings, 01, 10 and 11. Hence we create three new $(3, 0)$–bilabelled graph diagrams using these strings from each of $\boldsymbol{B_0}$, $\boldsymbol{C_0}$ and $\boldsymbol{D_0}$, by adding 1 new red edge outwards from a red vertex that corresponds to a 1 in the string, adding in a new red vertex to make this new red edge possible. They are given by

$\boldsymbol{B_{2,1}} =$  ; $\boldsymbol{B_{2,2}} =$  ; $\boldsymbol{B_{2,3}} =$  (8.65)

$\boldsymbol{C_{2,1}} =$  ; $\boldsymbol{C_{2,2}} =$  ; $\boldsymbol{C_{2,3}} =$  (8.66)

$$\boldsymbol{D_{2,1}} = $$ 

$$; \boldsymbol{D_{2,2}} = $$ 

$$; \boldsymbol{D_{2,3}} = $$ 

(8.67)

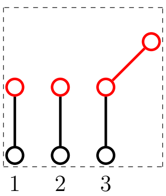
which, after Frobenius duality, correspond to the A-homomorphism matrices

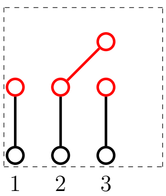
$$\begin{array}{cc} & \begin{matrix} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \end{matrix} & \left[\begin{array}{ccc|ccc|ccc} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{array}\right] \end{array} ; \begin{array}{cc} & \begin{matrix} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \end{matrix} & \left[\begin{array}{ccc|ccc|ccc} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} ; \begin{array}{cc} & \begin{matrix} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \end{matrix} & \left[\begin{array}{ccc|ccc|ccc} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} \tag{8.68}$$

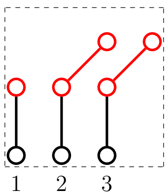
$$\begin{array}{cc} & \begin{matrix} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \end{matrix} & \left[\begin{array}{ccc|ccc|ccc} 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \end{array}\right] \end{array} ; \begin{array}{cc} & \begin{matrix} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \end{matrix} & \left[\begin{array}{ccc|ccc|ccc} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} ; \begin{array}{cc} & \begin{matrix} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \end{matrix} & \left[\begin{array}{ccc|ccc|ccc} 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} \tag{8.69}$$

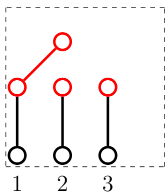
$$\begin{array}{cc} & \begin{matrix} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \end{matrix} & \left[\begin{array}{ccc|ccc|ccc} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} ; \begin{array}{cc} & \begin{matrix} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \end{matrix} & \left[\begin{array}{ccc|ccc|ccc} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} ; \begin{array}{cc} & \begin{matrix} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \end{matrix} & \left[\begin{array}{ccc|ccc|ccc} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} \tag{8.70}$$

For $\boldsymbol{E_0}$, we know that the number of red vertices, c, in $\boldsymbol{E_0}$ is 3. As a result, we create all length three strings having entries in 0 to $m = 1$, removing the all 0 string. There are seven such strings, 001, 010, 011, 100, 101, 110 and 111. Hence we create seven new $(3, 0)$–bilabelled graph diagrams using these strings from $\boldsymbol{E_0}$, by adding 1 new red edge outwards from a red vertex that corresponds to a 1 in the string, adding in a new red vertex to make this new red edge possible. They are given by

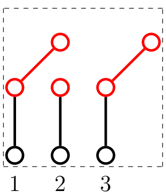
$$\boldsymbol{E_{2,1}} = $$ 

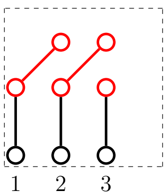
$$; \boldsymbol{E_{2,2}} = $$ 

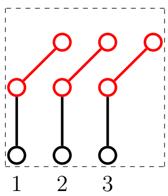
$$; \boldsymbol{E_{2,3}} = $$ 

$$; \boldsymbol{E_{2,4}} = $$ 

(8.71)

$$\boldsymbol{E_{2,5}} = $$ 

$$; \boldsymbol{E_{2,6}} = $$ 

$$; \boldsymbol{E_{2,7}} = $$ 

(8.72)

which, after Frobenius duality, correspond to the A-homomorphism matrices

$$\begin{array}{cc} & \begin{matrix} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \end{matrix} & \left[\begin{array}{ccc|ccc|ccc} 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \end{array}\right] \end{array} ; \begin{array}{cc} & \begin{matrix} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \end{matrix} & \left[\begin{array}{ccc|ccc|ccc} 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \end{array}\right] \end{array} \tag{8.73}$$

$$\begin{array}{cc} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc|ccc|ccc} 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} ; \begin{array}{cc} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc|ccc|ccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} \tag{8.74}$$

$$\begin{array}{cc} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc|ccc|ccc} 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} ; \begin{array}{cc} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc|ccc|ccc} 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} ; \begin{array}{cc} & \begin{array}{ccc|ccc|ccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc|ccc|ccc} 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} \tag{8.75}$$

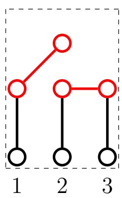
Step 4: We calculate all possible $(3, 0)$–bilabelled graph diagrams that have external red edges from the 32 bilabelled graph diagrams that were given in Step 2.

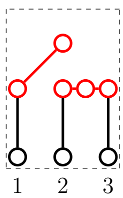
None of the bilabelled graph diagrams given in (8.41) and (8.42) will lead to new bilabelled graph diagrams, since each red vertex in each bilabelled graph diagram is connected with another red vertex in the diagram.

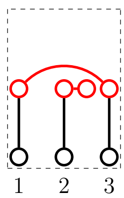
Only six bilabelled graph diagrams in (8.45) – (8.53) give rise to new bilabelled graph diagrams: they are $\boldsymbol{E}_{001}$, $\boldsymbol{E}_{002}$, $\boldsymbol{E}_{010}$, $\boldsymbol{E}_{020}$, $\boldsymbol{E}_{100}$, and $\boldsymbol{E}_{200}$, as they are the only bilabelled graph diagrams that have at least one (in this example, exactly one) red vertex that is not connected with any other vertex.

As each bilabelled graph diagram has three red vertices having only a single red vertex that is not connected with any other vertex, and since $m = 1$, this step produces exactly one string of length three, where each position in the string corresponds to a red vertex in the bilabelled graph diagram.

Hence, from $\boldsymbol{E}_{001}$, $\boldsymbol{E}_{002}$, and $\boldsymbol{E}_{010}$, we obtain

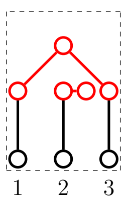
$\boldsymbol{E}_{001,\boldsymbol{E}} =$ 

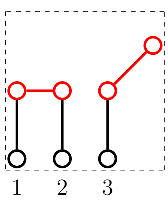
; $\boldsymbol{E}_{002,\boldsymbol{E}} =$ 

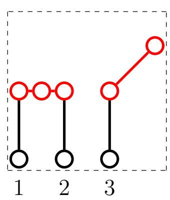
; $\boldsymbol{E}_{010,\boldsymbol{E}} =$ 

(8.76)

and from $\boldsymbol{E}_{020}$, $\boldsymbol{E}_{100}$, and $\boldsymbol{E}_{200}$, we obtain

$\boldsymbol{E}_{020,\boldsymbol{E}} =$ 

; $\boldsymbol{E}_{100,\boldsymbol{E}} =$ 

; $\boldsymbol{E}_{200,\boldsymbol{E}} =$ 

(8.77)

These bilabelled graph diagrams correspond, after Frobenius duality, to the A-homomorphism matrices

$$
\begin{array}{cc} & \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc|ccc|ccc} 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} ; \begin{array}{cc} & \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc|ccc|ccc} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} ; \begin{array}{cc} & \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc|ccc|ccc} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} \tag{8.78}
$$

$$
\begin{array}{cc} & \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc|ccc|ccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} ; \begin{array}{cc} & \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc|ccc|ccc} 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} ; \begin{array}{cc} & \begin{array}{ccccccccc} 1,1 & 1,2 & 1,3 & 2,1 & 2,2 & 2,3 & 3,1 & 3,2 & 3,3 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \end{array} & \left[\begin{array}{ccc|ccc|ccc} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array} \tag{8.79}
$$

Step 5: There are no new bilabelled graph diagrams to consider in this step as A does not have any loops.

In total, this gives us 60 $(3,0)$–bilabelled graph diagrams that are appropriate for A.

We have implicitly applied **Step 6** in calculating the A-homomorphism matrices during Steps 1 through 5 inclusive since we had to convert each $(3,0)$–bilabelled graph diagram into a $(2,1)$–bilabelled graph diagram first before applying the functor $\mathcal{F}^G$ to obtain each A-homomorphism matrix. Hence we move onto the second half of **Step 7**: we remove all duplicate matrices and the all zero matrices. We see that we are left with 31 matrices in the spanning set. After this, to obtain the weight matrix for an $\mathrm{Aut}(A) \cong S_2$-equivariant linear layer function from $(\mathbb{R}^3)^{\otimes 2}$ to $\mathbb{R}^3$, we would weight each of the 31 matrices in the spanning set and then add them together.

It is important to highlight that the vector space $\mathrm{Hom}((\mathbb{R}^3)^{\otimes 2}, \mathbb{R}^3)$, in which $\mathrm{Hom}_{S_2}((\mathbb{R}^3)^{\otimes 2}, \mathbb{R}^3)$ lives, is of dimension 27; hence, there must be linear dependencies amongst the 31 spanning set elements that we have found. However, determining these linear dependencies in general at the graph level *a priori* is left for future work.

We can also find the elements of the spanning set for $\mathrm{Hom}_{S_2}((\mathbb{R}^3)^{\otimes 2}, \mathbb{R}^3)$ for the other two embeddings of S_2 in S_3, given by $\mathrm{Aut}(B)$ and $\mathrm{Aut}(C)$. For $\mathrm{Aut}(B)$, we perform the permutation (13) on each index of the rows and columns of the spanning set matrices found for $\mathrm{Aut}(A)$, and for $\mathrm{Aut}(C)$, we perform the permutation (23) instead.

Chapter 9

Compact Matrix Quantum Group Equivariant Neural Networks

As a general theme, we have looked at the problem of encoding symmetries in neural networks as an inductive bias. Our approach has been founded upon the idea that spaces in classical (commutative) geometry are given by sets of points with additional structure, and that the data that lives in these spaces have symmetries that are formally described by groups. At each stage, we have chosen a particular group to study, and we have been able to characterise the weight matrices that appear in neural networks that are equivariant to this group.

However, when we consider tasks that involve learning from data that lives in a *non-commutative geometry*, such as learning properties of random variables in free probability theory (Voiculescu, 1985; Voiculescu et al., 1992), physical phenomena in particle physics (Bhowmick et al., 2014; van Suijlekom, 2014), and properties of exactly solvable models in statistical mechanics (Podlés, 1995), a group equivariant neural network is no longer a valid choice of model. This is because non-commutative geometry, as developed by Connes (1995), is formally described using non-commutative C^*-algebras, which generalize commutative algebras $C(X)$ of continuous functions on compact Hausdorff spaces X, and one can show that the C^*-algebra $C(G)$ of continuous functions on a compact matrix group G is indeed commutative (see Example 9.4). In particular, in non-commutative geometry, the classical notion of a space is replaced by an algebraic structure, thus making traditional group-based symmetries inadequate (Budzyński and Kondracki, 1995; Majid, 2000; Sitarz, 2013). Instead, a more general notion of symmetry is required. Quantum groups, which have been developed by a number of authors (Jimbo, 1985; Drinfeld, 1986; Woronowicz, 1987, 1991, 1998), are an important class of mathematical objects for describing these symmetries.

Given that there exist symmetries that are not captured by groups, this implies that there is a need for a new type of equivariant neural network. In this chapter, we focus on quantum groups that were constructed using a topological approach by Woronowicz (1987). Woronowicz generalised compact matrix groups to create a class of quantum groups called *compact matrix quantum groups*. We describe and motivate this construction in Section 9.2, and provide a comprehensive background on the prerequisite mathematical material, namely, C^*-algebras and operator theory, in Section 9.1.

The main contributions of this chapter are as follows. We broaden the scope of equivariant neural networks beyond classical symmetries by defining a new class of neural networks that encode, as an inductive bias, symmetries that are described by compact matrix quantum groups. We call these neural networks *compact matrix quantum group equivariant neural networks*. In fact, we mathematically derive their existence using Woronowicz's version of Tannaka–Krein duality for compact matrix quantum groups (Woronowicz, 1988), which is introduced in Section 9.4. After proving that these neural networks exist, we characterise their weight matrices for a large class of compact matrix quantum groups known as *easy*. We focus on easy compact matrix quantum groups not only because they have been studied extensively in the literature (Banica and Speicher, 2009; Banica et al., 2010; Weber, 2013; Freslon and Weber, 2016; Raum and Weber, 2016; Tarrago and Weber, 2016; Weber, 2017; Tarrago and Weber, 2018; Gromada, 2020; Freslon, 2023), but also because the way in which they are defined, namely, by set partitions, makes it possible to obtain a full characterisation of the equivariant weight matrices for these compact matrix quantum groups. We note that our contributions are primarily theoretical in nature: to demonstrate the practical potential of these neural networks, further work is needed to extend the characterisation to the equivariant non-linear layers so that they can be implemented.

We note as a corollory of our work that the new class of neural networks generalises and encompasses all neural networks that are equivariant to compact matrix groups, since we show, in Section 9.2, that compact matrix groups are, and can be expressed as, compact matrix quantum groups. As a result, we recover characterisations of the equivariant weight matrices that appeared in Chapters 5 and 6 (and originally in Ravanbakhsh et al. (2017); Zaheer et al. (2017); Maron et al. (2019a); Godfrey et al. (2023), and Pearce-Crump (2023a, 2024)) for the symmetric and orthogonal groups, and obtain characterisations of the equivariant weight matrices for five compact matrix groups that had not appeared in the machine learning literature before. They include classifications for the hyperoctahedral group H_n, the bistochastic group B_n, and the unitary group $U(n)$.

We note that in this chapter we change our field of scalars from $\mathbb{R}$ to $\mathbb{C}$ since most of the theory behind C^*-algebras is expressed using complex numbers.

The content of this chapter is modelled on the paper:

Edward Pearce-Crump. (2023c) Compact Matrix Quantum Group Equivariant Neural Networks. `arXiv:2311.06358`.

9.1 An Overview of C^*-Algebras and Operator Theory

We begin with an overview of the key results that appear in the theory of C^*-algebras, which is an area of mathematics that lies at the intersection of algebra and topology. The theory has its roots in the development of quantum mechanics in the 1920s, where the non-commutativity of bounded, linear operators on a Hilbert space is used to model physical observables. In 1943, Gelfand and Naimark (1943) developed an abstract definition of C^*-algebras that did not require the use of bounded, linear operators on a Hilbert space. Forty-five years later, Woronowicz (1987, 1988) used C^*-algebras to develop compact matrix quantum groups. Since a proper treatment of the theory of C^*-algebras would amount to several hundreds of pages, we focus solely on the key concepts that also appear in the theory of compact matrix quantum groups; notably, tensor products of C^*-algebras and the construction of the universal C^*-algebra from a set of generators and relations on those generators. When we quote a result without proof, we provide a reference to the literature where its proof can be found. We also rely on the background material on general topology that appeared in Chapter 2 throughout this chapter. For a more comprehensive treatment on C^*-algebras, we recommend any of the following references: Rørdam et al. (2000); Lin (2001); Blackadar (2006); Gromada (2020); Speicher and Weber (2020); Maaßen (2021), and Courtney and Gillaspy (2023).

Definition 9.1. We build up to the definition of a C^*-algebra with the following definitions.

- An **algebra** A over $\mathbb{C}$ is a complex vector space together with a bilinear, associative multiplication $A \times A \to A$ satisfying $\lambda(xy) = (\lambda x)y = x(\lambda y)$ for $x, y \in A$ and $\lambda \in \mathbb{C}$.
- An algebra A is **unital** if it contains an element $1 \in A$, called the **unit**, such that $1x = x1 = x$ for all $x \in A$.
- A subset $B \subseteq A$ is called a **subalgebra** if B is an algebra with respect to the same operations.
- A subset $I \subseteq A$ is called an **ideal** if I is a subalgebra of A that is invariant with respect to both left and right multiplication by elements of A.

- A **normed algebra** is an algebra A that has a norm $\|\cdot\| : A \to \mathbb{C}$ such that it is submultiplicative: $\|xy\| \leq \|x\| \|y\|$ for all $x, y \in A$.
- A **Banach algebra** is a normed algebra A which is complete with respect to the metric induced by the norm $\|\cdot\|$.
- An **involution** on an algebra A is an antilinear map $^* : A \to A$ such that $(x^*)^* = x$ and $(xy)^* = y^* x^*$ for all $x, y \in A$.
- A ***-algebra** is an algebra A that has an involution. Similarly, we define ***-subalgebra** and ***-ideal**.
- A **Banach *-algebra** is a Banach algebra A that has an involution.

Consequently, we have that

Definition 9.2. A ***C**-algebra** is a Banach *-algebra A that satisfies the ***C**-identity**: $\|x^* x\| = \|x\|^2$ for all $x \in A$.

Note that a closed *-subalgebra of a C^*-algebra is also a C^*-algebra.

Definition 9.3. We say that an element a of a C^*-algebra A is **self-adjoint** if $a^* = a$, and, if A is unital, we say that a is **unitary** if $a^* a = aa^* = 1$.

Example 9.4. We give some examples of C^*-algebras.

1. The complex numbers $\mathbb{C}$ form a unital C^*-algebra.
2. Let $\mathcal{H}$ be a complex Hilbert space with inner product denoted by $\langle \cdot, \cdot \rangle$. The collection of bounded linear operators $\mathcal{B}(\mathcal{H})$ is a unital C^*-algebra, where the norm is the operator norm

$$\|T\| := \sup\{\|Tv\|_{\mathcal{H}} \mid v \in \mathcal{H}, \|v\|_{\mathcal{H}} \leq 1\} \tag{9.1}$$

   for all $T \in \mathcal{B}(\mathcal{H})$. Note that multiplication is given by the composition of operators, and that the involution is the adjoint, defined with respect to the inner product by $\langle T^* v, w \rangle = \langle v, Tw \rangle$ for all $T \in \mathcal{B}(\mathcal{H})$ and $v, w \in \mathcal{H}$. Also, we see that $\mathcal{B}(\mathcal{H})$ is non-commutative.
3. For any positive integer n, let $M_n(\mathbb{C})$ denote the set of $n \times n$ matrices with elements in $\mathbb{C}$. Then $M_n(\mathbb{C})$ is a unital C^*-algebra, where the norm is the operator norm

$$\|M\| := \sup\{\|Mv\|_2 \mid v \in \mathbb{C}^n, \|v\|_2 \leq 1\} \tag{9.2}$$

where $\|\cdot\|_2$ is the Euclidean norm on $\mathbb{C}^n$, and the involution is given by the conjugate transpose. This is a special case of the previous example, where we have chosen $\mathcal{H} = \mathbb{C}^n$ and picked a basis to represent linear transformations as matrices.

4. Let X be a compact Hausdorff space. Then the space of continuous functions on X,

$$C(X) := \{f : X \to \mathbb{C} \mid f \text{ is continuous}\} \tag{9.3}$$

is a unital C^*-algebra where the norm is the supremum norm

$$\|f\|_\infty := \sup\{|f(x)| \mid x \in X\} \tag{9.4}$$

such that $(f+g)(x) := f(x) + g(x)$, $(\lambda f)(x) := \lambda f(x)$, $(fg)(x) := f(x)g(x)$, $1(x) := 1$ and $f^*(x) := \overline{f(x)}$ for all $f, g \in C(X), x \in X$ and $\lambda \in \mathbb{C}$. Note, in particular, that $C(X)$ is commutative. The supremum norm is indeed a norm since the compactness of X implies that every continuous function $f : X \to \mathbb{C}$ is bounded, and so $\|f\|_\infty < \infty$ for all $f \in C(X)$. The norm is also submultiplicative because

$$|fg(x)| = |f(x)g(x)| = |f(x)||g(x)| \leq \|f\|_\infty \|g\|_\infty \tag{9.5}$$

which implies that $\|fg\|_\infty \leq \|f\|_\infty \|g\|_\infty$. Similar reasoning on the fact that $|f(x)\overline{f(x)}| = |f(x)|^2$ shows that the supremum norm satisfies the C^*-identity. Finally, the supremum norm is complete because one can show that any Cauchy sequence of continuous functions on X converges uniformly on X such that the limit is also a continuous function on X.

An important theorem for the C^*-algebra of continuous functions on a compact Hausdorff space is the Stone–Weierstrass Theorem. Its proof can be found in Speicher and Weber (2020, Theorem 3.3). First we need the following definition.

Definition 9.5. Let A be an algebra of functions from some set X to the complex numbers $\mathbb{C}$. We say that A **separates points** in X if, for all $x \neq y \in X$, there is a function $f \in A$ such that $f(x) \neq f(y)$.

Consequently, we have that

Theorem 9.6 (Stone–Weierstrass). *Let X be a compact Hausdorff space. If A is a unital *-subalgebra of $C(X)$ such that A separates points in X, then A is dense in $C(X)$. In particular, if A is closed, then $A = C(X)$.*

We also have homomorphisms on C^*-algebras:

Definition 9.7. If A, B are algebras, then a linear map $\phi : A \to B$ such that $\phi(xy) = \phi(x)\phi(y)$ for all $x, y \in A$ is called an **algebra homomorphism**.

If A, B are *-algebras, then if $\phi : A \to B$ is an algebra homomorphism that preserves the involution operation, that is, $\phi(x^*) = \phi(x)^*$ for all $x \in A$, then ϕ is called a ***-homomorphism**.

If A, B are *-algebras, then a bijective *-homomorphism $\phi : A \to B$ is called a ***-isomorphism**.

We use the properties of $\mathcal{B}(\mathcal{H})$ to obtain representations of C^*-algebras:

Definition 9.8. A **representation** of an algebra A is a choice of complex Hilbert space $\mathcal{H}$ and an algebra homomorphism $\pi : A \to \mathcal{B}(\mathcal{H})$.

A ***-representation** of a *-algebra A is a representation of A that preserves the involution operation.

A representation is said to be **faithful** if it is injective.

One of the most important results in the theory of C^*-algebras is the Gelfand–Naimark Theorem (Gelfand and Naimark, 1943). It says that every commutative, unital, C^*-algebra is *-isomorphic to the unital C^*-algebra of continuous functions on some compact Hausdorff space X. Since it is the first of two theorems by Gelfand and Naimark, we will call it Gelfand–Naimark I. Its proof can also be found in Blackadar (2006, Theorem II.2.2.4).

Theorem 9.9 (Gelfand–Naimark I)**.** *Let A be a commutative, unital, C^*-algebra. Then A is *-isomorphic to $C(X)$, where*

$$X := \{\phi : A \to \mathbb{C} \mid \phi \textit{ is a non-zero }{}^*\textit{-homomorphism}\} \tag{9.6}$$

is a compact Hausdorff space. The *-isomorphism is given by $f : A \to C(X)$ where $f(a) : X \to \mathbb{C}$ for all $a \in A$ is such that $f(a)[\phi] = \phi(a)$ for all $\phi \in X$.

We now look at tensor products of C^*-algebras. First, we have the following definition.

Definition 9.10. Let A, B be *-algebras. The **algebraic tensor product**, written $A \odot B$, is defined to be the algebraic tensor product of the vector spaces A and B together with a multiplication operation that is given by

$$(a_1 \odot b_1)(a_2 \odot b_2) = a_1a_2 \odot b_1b_2 \tag{9.7}$$

and an involution operation that is given by

$$(a \odot b)^* = a^* \odot b^* \tag{9.8}$$

In particular, $A \odot B$ is a *-algebra.

To define a tensor product on C^*-algebras A and B such that the result is also a C^*-algebra, we need to be able to define a norm $\|\cdot\|$ that satisfies the C^*-identity on $A \odot B$, and then complete $(A \odot B, \|\cdot\|)$ to obtain a C^*-algebra. Any norm that satisfies the C^*-identity is said to be a **C^*-norm**. Although it can be shown, as a consequence of the Gelfand–Naimark–Segal Theorem, that C^*-norms on algebraic tensor products of C^*-algebras always exist — see the definition of the spatial norm below — one of the main difficulties behind this construction is that in many cases there can be more than one C^*-norm that can be associated with $A \odot B$ (Courtney and Gillaspy, 2023), and so in those cases there are many possible definitions of a tensor product on C^*-algebras A and B such that the result is also a C^*-algebra. Hence, in what follows, we choose only to outline the main ideas that lead to the construction of a particular tensor product called the **minimal tensor product** since this tensor product will be used in the definition of a compact matrix quantum group.

We start with another theorem by Gelfand–Naimark, which is technically a corollary of the Gelfand–Naimark–Segal Theorem. Its proof can be found in Speicher and Weber (2020, Theorem 5.19).

Theorem 9.11 (Gelfand–Naimark II). *Every C^*-algebra A has a faithful *-representation $\pi : A \to \mathcal{B}(\mathcal{H})$ on some Hilbert space $\mathcal{H}$.*

Hence we have faithful *-representations $\pi_A : A \to \mathcal{B}(\mathcal{H}_A)$ and $\pi_B : B \to \mathcal{B}(\mathcal{H}_B)$ for C^*-algebras A and B. One can show (Courtney and Gillaspy, 2023, Corollary 11.15) that this induces a faithful *-representation $\pi_A \odot \pi_B : A \odot B \to \mathcal{B}(\mathcal{H}_A \otimes \mathcal{H}_B)$ such that $(\pi_A \odot \pi_B)(a \odot b) = \pi_A(a) \otimes \pi_B(b)$ for all $a \in A$ and $b \in B$.

Hence we can define a norm $\|\cdot\|_*$ on $A \odot B$, known as the **spatial norm**, that is given by

$$\|x\|_* = \|(\pi_A \odot \pi_B)(x)\|_{\mathcal{B}(\mathcal{H}_A \otimes \mathcal{H}_B)} \tag{9.9}$$

which is a C^*-norm by the fact that the norm on $\mathcal{B}(\mathcal{H}_A \otimes \mathcal{H}_B)$ is a C^*-norm together with the injectivity of $\pi_A \odot \pi_B$.

Takesaki (2002) showed that the spatial norm is independent of the faithful *-representation, and that it is the minimal C^*-norm on $A \odot B$; that is, for all C^*-norms γ on $A \odot B$, we have that

$$\|x\|_* \leq \gamma(x) \tag{9.10}$$

for all $x \in A \odot B$. Hence we call $\|x\|_*$ the **minimum norm**, and write it as $\|x\|_{\text{min}}$.

Consequently, the completion of $A \odot B$, for C^*-algebras A and B, with respect to the minimum norm $\|x\|_{\text{min}}$, is called the **minimal tensor product** and is denoted by $A \otimes_{\text{min}} B$.

One can also show (Courtney and Gillaspy, 2023, Corollary 11.29) that for a pair of *-homomorphisms $\phi_1 : A_1 \to B_1$ and $\phi_2 : A_2 \to B_2$ on *-algebras A_1, A_2, B_1, B_2, the algebraic tensor product $\phi_1 \odot \phi_2 : A_1 \odot A_2 \to B_1 \odot B_2$ extends to a *-homomorphism $\phi_1 \otimes_{\text{min}} \phi_2 : A_1 \otimes_{\text{min}} A_2 \to B_1 \otimes_{\text{min}} B_2$.

We now focus on $M_n(A)$, the set of $n \times n$ matrices with entries in a C^*-algebra A, that is

$$M_n(A) \coloneqq \{(a_{i,j}) \mid a_{i,j} \in A, 1 \leq i, j \leq n\} \tag{9.11}$$

This comes with a natural involution that is given by $(a_{i,j})^* = (a^*_{j,i})$ for all $(a_{i,j}) \in M_n(A)$.

We use the following definition to create a C^*-norm on $M_n(A)$.

Definition 9.12. Suppose that $\phi : A \to B$ is a linear map between *-algebras A and B. Then, for all positive integers n, we define the linear map $\phi^{(n)} : M_n(A) \to M_n(B)$ to be $\phi^{(n)}((a_{i,j})) = (\phi(a_{i,j}))$. The map $\phi^{(n)}$ is called a **matrix amplification** of ϕ.

We create the C^*-norm on $M_n(A)$ as follows. By Gelfand–Naimark II, we have a faithful *-representation $\pi : A \to \mathcal{B}(\mathcal{H})$ for some Hilbert space $\mathcal{H}$. This induces a faithful *-representation $\pi^{(n)} : M_n(A) \to \mathcal{B}(\mathcal{H}^n)$, where $\mathcal{H}^n = \mathcal{H} \oplus \ldots \oplus \mathcal{H}$, since $M_n(\mathcal{B}(\mathcal{H}))$ is *-isomorphic to $\mathcal{B}(\mathcal{H}^n)$. Then we can define a function $\|\cdot\| : M_n(A) \to \mathbb{C}$ by

$$\|(a_{i,j})\| \coloneqq \left\|\pi^{(n)}((a_{i,j}))\right\|_{\mathcal{B}(\mathcal{H}^n)} \tag{9.12}$$

which is a C^*-norm since $\pi^{(n)}$ is injective and the norm on $\mathcal{B}(\mathcal{H}^n)$ is a C^*-norm.

One can show the following inequality, whose proof is outlined in Rørdam et al. (2000, Exercise 1.13).

Lemma 9.13. *Let $(T_{i,j}) \in M_n(\mathcal{B}(H))$. Then*

$$\|T_{i,j}\|_{\mathcal{B}(\mathcal{H})} \leq \|(T_{i,j})\|_{M_n(\mathcal{B}(\mathcal{H}))} \leq \sum_{i,j} \|T_{i,j}\|_{\mathcal{B}(\mathcal{H})} \tag{9.13}$$

Lemma 9.13 implies that the image of $M_n(A)$ under $\pi^{(n)}$, which is *-isomorphic to $M_n(A)$ since $\pi^{(n)}$ is injective, is a closed *-subalgebra of the C^*-algebra $\mathcal{B}(\mathcal{H}^n)$. Hence $M_n(A)$ is itself a C^*-algebra. Moreover, the norm on $M_n(A)$ defined in (9.12) is unique because a *-algebra admits at most one norm that makes it into a C^*-algebra (Lin, 2001, Corollary 1.2.8).

Expressing this result as a theorem, we have that

Theorem 9.14. *If A is a C^*-algebra, then $M_n(A)$ is also a C^*-algebra whose norm is defined in (9.12).*

One can also show the following result.

Lemma 9.15. *Suppose that A is a C^*-algebra and let n be any positive integer. If $\phi : M_n(A) \to M_n(\mathbb{C}) \odot A$ is the map given by*

$$\phi((a_{i,j})) := \sum_{i,j=1}^{n} E_{i,j} \odot a_{i,j} \tag{9.14}$$

where $E_{i,j}$ is the matrix unit with a 1 in the (i,j)-entry, and 0 otherwise, then ϕ is a *-isomorphism.

We use the *-isomorphism between $M_n(A)$ and $M_n(\mathbb{C}) \odot A$ to obtain the following isomorphism.

Proposition 9.16. *Suppose that A is a C^*-algebra and let n be any positive integer. Then there is a unique C^*-norm on the algebraic tensor product $M_n(\mathbb{C}) \odot A$. Hence we write $M_n(\mathbb{C}) \otimes A$, and have that*

$$M_n(\mathbb{C}) \otimes A = M_n(\mathbb{C}) \odot A \cong M_n(A) \tag{9.15}$$

Proof. The *-isomorphism together with the C^*-norm given on $M_n(A)$ defines the existence of a C^*-norm on $M_n(\mathbb{C}) \odot A$, namely

$$\|(\lambda_{i,j}) \odot a\| := \|(\lambda_{i,j} a)\| \tag{9.16}$$

$M_n(\mathbb{C}) \odot A$ is now *-isomorphic to a closed *-subalgebra of the C^*-algebra $M_n(A)$, namely $M_n(A)$ itself, hence it is a C^*-algebra. The result is now immediate because a *-algebra admits at most one norm making it into a C^*-algebra (Lin, 2001, Corollary 1.2.8). □

This means that we can use the *-isomorphism ϕ to identify elements of $M_n(A)$ with elements of $M_n(\mathbb{C}) \otimes A$, namely

$$(a_{i,j}) \leftrightarrow \sum_{i,j=1}^{n} E_{i,j} \otimes a_{i,j} \tag{9.17}$$

We now turn to the concept of **universal C^*-algebras**. We know that we can create C^*-algebras by starting with a *-algebra, finding a C^*-norm for it, and then completing the pair. We would like to be able to create C^*-algebras using sets of generators and relations. We follow the presentation that is given in Speicher and Weber (2020, Chapter 6).

We begin by creating a *-algebra as follows.

Definition 9.17. Let $E = \{x_i \mid i \in I\}$ be a set of symbols over an index set I. Let $E^* = \{x_i^* \mid i \in I\}$ be another set of symbols that is disjoint from E. Then

- A **non-commutative monomial** in $E \cup E^*$ is a word of non-commuting symbols $x_{i_1}^{\alpha_1} \cdots x_{i_m}^{\alpha_m}$ for some positive integer m such that $i_1, \ldots, i_m \in I$ and $\alpha_j \in \{\pm 1\}$ for all $i \in [m]$.
- A **non-commutative polynomial** in $E \cup E^*$ is a formal linear combination of non-commutative monomials in $E \cup E^*$ with coefficients in $\mathbb{C}$.
- We can define a **concatenation operation** on non-commutative monomials in $E \cup E^*$:

$$(x_{i_1}^{\alpha_1} \cdots x_{i_m}^{\alpha_m}) \cdot (x_{j_1}^{\beta_1} \cdots x_{j_m}^{\beta_n}) = x_{i_1}^{\alpha_1} \cdots x_{i_m}^{\alpha_m} x_{j_1}^{\beta_1} \cdots x_{j_m}^{\beta_n} \tag{9.18}$$

  for any non-commutative monomials $x_{i_1}^{\alpha_1} \cdots x_{i_m}^{\alpha_m}$ and $x_{j_1}^{\beta_1} \cdots x_{j_m}^{\beta_n}$ in $E \cup E^*$. This operation is extended linearly to non-commutative polynomials in $E \cup E^*$.
- We can also define an **involution operation** on non-commutative monomials in $E \cup E^*$:

$$(x_{i_1}^{\alpha_1} \cdots x_{i_m}^{\alpha_m})^* = x_{i_1}^{{\alpha_1}^*} \cdots x_{i_m}^{{\alpha_m}^*} \tag{9.19}$$

  where ${\alpha_j}^*$ is 1 if $\alpha_j = *$ and is $*$ if $\alpha_j = 1$ for all $j \in [m]$. This operation is extended antilinearly to non-commutative polynomials in $E \cup E^*$.

We have the following definition.

Definition 9.18. Let E, E^* be as before. The **free *-algebra on the generator set E**, $P(E)$, is the *-algebra generated from the set of non-commutative polynomials in $E \cup E^*$, where addition and scalar multiplication are the standard operations, and multiplication and involution are given by the concatenation and involution operations given in (9.18) and (9.19), respectively.

The *-algebra $P(E)$ is free in the sense that there are no relations amongst the polynomials in $P(E)$.

For any subset R of polynomials in $P(E)$, we can create the *-ideal $I(R)$ that is generated by R. Considering $P(E)$ and $I(R)$ as algebras, we can certainly form the quotient algebra $P(E)/I(R)$. Now, since $P(E)$ and $I(R)$ are *-algebras, the involution operation on $P(E)/I(R)$ given by

$$(a+I)^* \coloneqq a^* + I \tag{9.20}$$

for all $a \in P(E)$ is well-defined, and so in this case $P(E)/I(R)$ is also a (quotient) *-algebra. Hence, we have that

Definition 9.19. The **universal *-algebra with generators *E* and relations *R***, $A(E \mid R)$, is the quotient *-algebra $P(E)/I(R)$.

We now aim to create a C^*-norm on a *-algebra, although this *-algebra isn't quite $A(E \mid R)$. For this, we first need the definition of a C^*-seminorm.

Definition 9.20. A ***C****-**seminorm** on a *-algebra A is a map $\gamma : A \to \mathbb{R}$ such that

1. γ is a seminorm, that is
   - $\gamma(x) \geq 0$ for all $x \in A$
   - $\gamma(\lambda x) = |\lambda|\gamma(x)$ for all $x \in A$ and $\lambda \in \mathbb{C}$
   - $\gamma(x+y) \leq \gamma(x) + \gamma(y)$ for all $x, y \in A$
2. γ is submultiplicative, that is, $\gamma(xy) \leq \gamma(x)\gamma(y)$ for all $x, y \in A$, and
3. γ satisfies the C^*-identity, that is, $\gamma(x^*x) = \gamma(x)^2$ for all $x \in A$.

Note that a norm on an algebra A is a seminorm $\gamma : A \to \mathbb{R}$ such that if $\gamma(x) = 0$ then $x = 0$. Hence a C^*-norm is a C^*-seminorm satisfying this additional property.

We make the following definition.

Definition 9.21. We define a function $\|\cdot\|_{A(E|R)} : A(E \mid R) \to \mathbb{R}$ by

$$\|x\|_{A(E|R)} \coloneqq \sup\{\gamma(x) \mid \gamma \text{ is a } C^*\text{-seminorm on } A(E \mid R)\} \tag{9.21}$$

It is possible that $\|x\|_{A(E|R)}$ is infinity for some $x \in A(E \mid R)$, so we only allow ourselves to consider those E and R such that $\|x\|_{A(E|R)} < \infty$ for all $A(E \mid R)$. In this case, one can show that $\|x\|_{A(E|R)}$ is a

C^*-seminorm: for example, for all $x, y \in A(E \mid R)$ and any C^*-seminorm γ on $A(E \mid R)$, we have that

$$\gamma(xy) \leq \gamma(x)\gamma(y) \leq \|x\|_{A(E|R)} \|y\|_{A(E|R)} \tag{9.22}$$

and so we see that $\|\cdot\|_{A(E|R)}$ is submultiplicative. Also, we see that

$$I_0 := \{x \in A(E \mid R) \mid \|x\|_{A(E|R)} = 0\} \tag{9.23}$$

is a *-ideal. Hence, $A(E \mid R)/I_0$ is a quotient *-algebra having a C^*-norm $\|x\|_{A(E|R)/I_0}$ that is obtained from the C^*-seminorm on $A(E \mid R)$ under the quotient map $A(E \mid R) \to A(E \mid R)/I_0$. Hence, we have that

Definition 9.22. The **universal C^*-algebra with generators E and relations R**, $C^*(E \mid R)$, is the completion of $A(E \mid R)/I_0$ with respect to $\|x\|_{A(E|R)/I_0}$. It is indeed a C^*-algebra, by construction.

The following lemma is very useful for showing the existence of the universal C^*-algebra $C^*(E \mid R)$.

Lemma 9.23. *Let $E = \{x_i \mid i \in I\}$ (with E^* understood) be a set of generators and let $R \subseteq P(E)$ be a set of relations. If there is a constant $C > 0$ such that the value of γ on any generator is less than or equal to C for all C^*-seminorms γ on $A(E \mid R)$, then $\|x\|_{A(E|R)} < \infty$ for all $x \in A(E \mid R)$. Hence, by the above construction, $C^*(E \mid R)$ exists.*

Proof. Let γ be any C^*-seminorm on $A(E \mid R)$. Then, since any monomial in $E \cup E^*$ has a positive integer length, we can pick an arbitrary monomial in $E \cup E^*$ having some arbitrary length m. Then, by the submultiplicative property, we have that the value of γ on this monomial is less than or equal to C^m, and so by taking the supremum over all C^*-seminorms on $A(E \mid R)$ we see that any element of $A(E \mid R)$ has a norm that is bounded, as required. □

One can show (Speicher and Weber, 2020, Proposition 6.7) that every universal C^*-algebra $C^*(E \mid R)$ satisfies the following so-called universal property.

Proposition 9.24 (Universal Property). *Let $E = \{x_i \mid i \in I\}$ be a set of generators and let $R \subseteq P(E)$ be a set of relations such that the universal C^*-algebra $C^*(E \mid R)$ exists. Suppose that A is a C^*-algebra having a subset $E' = \{y_i \mid i \in I\}$ that **satisfies the relations** R, that is, all polynomials in R become zero when we replace each x_i by y_i for all $i \in I$. Then there exists a unique *-homomorphism $\phi : C^*(E \mid R) \to A$ such that $x_i \mapsto y_i$ for all $i \in I$.*

Example 9.25. For $n \geq 2$, we have a *-isomorphism of C^*-algebras

$$M_n(\mathbb{C}) \cong C^*(u_{i,j}, 1 \leq i, j \leq n \mid u_{i,j}^* = u_{j,i}, u_{i,k}u_{l,j} = \delta_{k,l}u_{i,j}) \tag{9.24}$$

Proof. Let $C^*(E \mid R) \coloneqq C^*(u_{i,j}, 1 \leq i, j \leq n \mid u_{i,j}^* = u_{j,i}, u_{i,k}u_{l,j} = \delta_{k,l}u_{i,j})$. We first need to show that $C^*(E \mid R)$ exists, that is, that $\|x\| < \infty$ for all $x \in A(E \mid R)$. This is immediate from Lemma 9.23 if we can show that there is some constant C such that $\gamma(u_{i,j}) \leq C$ for all $1 \leq i, j \leq n$ and for all C^*-seminorms γ on $A(E \mid R)$.

Let γ be a C^*-seminorm on $A(E \mid R)$. By the C^*-identity and the relations R we have that

$$\gamma(u_{j,j})^2 = \gamma(u_{j,j}^* u_{j,j}) = \gamma(u_{j,j}u_{j,j}) = \gamma(\delta_{j,j}u_{j,j}) = \gamma(u_{j,j}) \tag{9.25}$$

Hence $\gamma(u_{j,j})$ is either 0 or 1 for all $j \in [n]$ and so, again by the C^*-identity and the relations R, we have that

$$\gamma(u_{i,j})^2 = \gamma(u_{i,j}^* u_{i,j}) = \gamma(u_{j,i}u_{i,j}) = \gamma(\delta_{i,i}u_{j,j}) = \gamma(u_{j,j}) \tag{9.26}$$

Hence $\gamma(u_{i,j})$ is bounded above by 1 for all $1 \leq i, j \leq n$ and for all C^*-seminorms γ on $A(E \mid R)$, as required.

We now prove the *-isomorphism $M_n(\mathbb{C}) \cong C^*(E \mid R)$. The matrix units $E_{i,j} \in M_n(\mathbb{C})$ satisfy the relations R, since $E_{i,j}^* = E_{i,j}^\top = E_{j,i}$ and $E_{i,k}E_{l,j} = \delta_{k,l}E_{i,j}$. Hence, by the universal property, there exists a surjective *-homomorphism $\phi : C^*(E \mid R) \to M_n(\mathbb{C})$ such that $u_{i,j} \mapsto E_{i,j}$ for all $i, j \in [n]$. Looking at the relations R we see that the monomials in $C^*(E \mid R)$ can be at most the elements $u_{i,j}$, but since ϕ is surjective the monomials are exactly the elements $u_{i,j}$. Hence, as a vector space, $C^*(E \mid R)$ has dimension n^2 and so ϕ is also injective. Hence ϕ is a *-isomorphism, as required. □

9.2 Non-Commutative Geometry

In this section, we introduce non-commutative geometry and motivate the existence of compact matrix quantum groups by studying certain properties of compact matrix groups $G(n) \subseteq GL(n)$.

Two ideas were important in the development of non-commutative geometry. The first was the paradigm shift from thinking in terms of points in a space (geometry) to considering functions on spaces (algebra) as the primary method for solving problems in geometry and topology (Budzyński and Kondracki, 1995; Majid, 2000; Sitarz, 2013). Indeed, whilst we have seen in Example 9.4 that the algebra $C(X)$ of continuous, complex-valued functions on a compact Hausdorff space X is a commutative, unital, C^*-algebra, we have also seen that the Gelfand–Naimark Theorem (Gelfand and Naimark, 1943) (Gelfand–Naimark I, Theorem 9.9) shows that every commutative, unital, C^*-algebra is, in fact, an algebra of continuous functions $C(X)$ for some compact Hausdorff space X. In this way the geometric notion of a compact Hausdorff space X is dualised to the algebra of continuous functions on X.

The second important development was the abstraction of spaces entirely by allowing unital C^*-algebras to be non-commutative (Jimbo, 1985; Drinfeld, 1986; Woronowicz, 1987). In doing so, this created a notion of geometry through these algebras, namely by treating non-commutative, unital, C^*-algebras as if they were algebras of continuous functions on (non-existent) *non-commutative spaces*, as an abstract extension of the theorem by Gelfand and Naimark (1943). Consequently, non-commutative, unital, C^*-algebras are the foundation of non-commutative geometry, and, under this framework, the notion of a space (of points) is no longer meaningful.

But with these developments, a new notion of symmetry for non-commutative, unital, C^*-algebras was needed, since, for any compact matrix group $G(n) \subseteq GL(n)$, we have seen in Example 9.4 that the algebra $C(G(n))$ is a commutative, unital, C^*-algebra.

However, we can study the properties of $C(G(n))$ to motivate the definition of a compact matrix quantum group, as given by Woronowicz (1987), to describe this new type of symmetry. Firstly, $G(n)$ is not just a compact Hausdorff space, it is a group, and so it comes with a group law $G(n) \times G(n) \to G(n)$ where $(g_1, g_2) \mapsto g_1 g_2$. We can express this group law in the language of C^*-algebras as follows: define $\Delta : C(G(n)) \mapsto C(G(n) \times G(n))$ under the mapping $f \mapsto ((g_1, g_2) \mapsto f(g_1 g_2))$. Furthermore, since $C(G(n) \times G(n))$ is *-isomorphic to $C(G(n)) \otimes C(G(n))$ under $((g_1, g_2) \mapsto f_1(g_1) f_2(g_2)) \cong f_1 \otimes f_2$, we see that

$$\Delta : C(G(n)) \mapsto C(G(n)) \otimes C(G(n)) \tag{9.27}$$

is the expression of the group law of $G(n)$ in the language of C^*-algebras.

We generalise this construction by replacing $C(G(n))$ by a (potentially non-commutative) unital C^*-algebra A, to define a compact matrix quantum group in the style of Woronowicz (1987).

Definition 9.26. Let A be a unital C^*-algebra, and let $u_{i,j} \in A$, for all $i, j \in [n]$, for some positive integer n. Let u be the $n \times n$ matrix whose (i,j)-entry is $u_{i,j}$, that is, u is an element of the unital C^*-algebra $M_n(A)$. The pair (A, u) is said to be a **compact matrix quantum group** if

1. the elements $u_{i,j}$ generate A,
2. u and $u^\top = (u_{j,i})$ are invertible matrices, and
3. the comultiplication map $\Delta : A \to A \otimes_{\text{min}} A$ defined by

$$\Delta(u_{i,j}) \coloneqq \left(\sum_k u_{i,k} \otimes u_{k,j}\right) \tag{9.28}$$

is a $*$-homomorphism.

Remark 9.27. In motivating Definition 9.26, we did not specify the tensor product in $C(G(n)) \otimes C(G(n))$ for constructing Δ because $C(G(n))$ is a nuclear C^*-algebra (Courtney and Gillaspy, 2023, Proposition 10.10), meaning that all norms that complete the algebraic tensor product $C(G(n)) \odot C(G(n))$ into a C^*-algebra give the same one. However, in general, we have seen that the algebraic tensor product of two C^*-algebras can be completed in many ways to form a C^*-algebra, and so in the general definition of a compact matrix quantum group, we need to say that we are completing with respect to the minimum norm $\|\cdot\|_{\text{min}}$.

The definition of a compact matrix quantum group seems rather abstract. However, with the following proposition, we have some initial examples of compact matrix quantum groups.

Proposition 9.28. *Compact matrix groups $G(n) \subseteq GL(n)$ form a special class of compact matrix quantum groups.*

Proof. Let $G(n) \subseteq GL(n)$ be a compact matrix group. Consider $C(G(n))$, the C^*-algebra of continuous functions $G(n) \to \mathbb{C}$, which is commutative. Firstly, we can define functions $u_{i,j} : G(n) \to \mathbb{C}$ for $i, j \in [n]$ such that $u_{i,j}(g) = g_{i,j}$. By the Stone–Weierstrass Theorem (Theorem 9.6) and the compactness of $G(n)$, the $u_{i,j}$ generate $C(G(n))$, and the matrices $u \coloneqq (u_{i,j})$ and $u^\top$ are invertible since $u^{-1}(g) = u(g^{-1})$.

Moreover, we have that the comultiplication map Δ for $C(G(n))$ is given by

$$\Delta(u_{i,j})(g, h) = u_{i,j}(gh) \tag{9.29}$$

since

$$\left(\sum_k u_{i,k} \otimes u_{k,j}\right)(g, h) = \sum_k u_{i,k}(g)u_{k,j}(h) = \sum_k g_{i,k}h_{k,j} = (gh)_{i,j} = u_{i,j}(gh) \tag{9.30}$$

that is, Δ coincides with matrix multiplication in $G(n)$.

The definition of Δ that is given in (9.29) is a $*$-homomorphism because

$$\Delta(u_{i,j}u_{k,l})(g,h) = u_{i,j}u_{k,l}(gh) \tag{9.31}$$
$$= u_{i,j}(gh)u_{k,l}(gh) \tag{9.32}$$
$$= \Delta(u_{i,j})(g,h)\Delta(u_{k,l})(g,h) \tag{9.33}$$
$$= \Delta(u_{i,j})\Delta(u_{k,l})(g,h) \tag{9.34}$$

and

$$\Delta(u_{i,j}^*)(g,h) = u_{i,j}^*(gh) = \overline{u_{i,j}(gh)} = \overline{\Delta(u_{i,j})(g,h)} = \Delta(u_{i,j})^*(g,h) \tag{9.35}$$

Note that in (9.32) we have used that $C(G(n))$ has a pointwise multiplication, and likewise, in (9.34), we have used that $C(G(n)) \otimes C(G(n))$ has a pointwise multiplication. $\square$

Moreover, Woronowicz (1987, Theorem 1.5) also showed, using Gelfand–Naimark I, that if (A,u) is a commutative compact matrix quantum group for some positive integer n, then $A \cong C(G(n))$ for some compact matrix group $G(n) \subseteq GL(n)$. Combining this with Proposition 9.28, we obtain what is commonly known as the Fundamental Theorem of Compact Matrix Quantum Groups, which is stated as follows.

Theorem 9.29 (Fundamental Theorem of Compact Matrix Quantum Groups)**.** *Let (A,u) be a compact matrix quantum group for some $n \in \mathbb{N}$. Then A is commutative if and only if $A \cong C(G(n))$ for some compact matrix group $G(n) \subseteq GL(n)$.*

Hence, the Fundamental Theorem shows that if $(G(n),u)$ is non-commutative, then we obtain "true" compact matrix quantum groups. Bearing this theorem in mind, convention dictates that we denote A by $C(G(n))$, even if A is non-commutative, and so we often refer to $G(n)$, or sometimes the pair $(G(n),u)$, as the compact matrix quantum group.

We now provide some examples of "true" compact matrix quantum groups to show that they do indeed exist.

Example 9.30. Recall that each element of the symmetric group S_n, considered as a subgroup of $GL(n)$, can be thought of as an $n \times n$ matrix having exactly one entry of 1 in each row and in each column, with all other entries being 0. Writing the elements of $g \in S_n$ as $g_{i,j}$, this condition can be expressed as the following relations:

$$g_{i,j}^2 = g_{i,j} = g_{i,j}^*, \ \sum_{i=1}^{n} g_{i,j} = \sum_{j=1}^{n} g_{i,j} = 1, \ g_{i,j}g_{k,l} = g_{k,l}g_{i,j} \tag{9.36}$$

By the universal property of universal C^*-algebras, we have that

$$C(S_n) \cong C^*(u_{i,j}, 1 \leq i,j \leq n \mid u_{i,j}^2 = u_{i,j} = u_{i,j}^*, \ \sum_{i=1}^{n} u_{i,j} = \sum_{j=1}^{n} u_{i,j} = 1, \ u_{i,j}u_{k,l} = u_{k,l}u_{i,j}) \tag{9.37}$$

Wang (1998) defined the symmetric quantum group as a compact matrix quantum group S_n^+ by **liberating** $C(S_n)$ of its commutativity relations to obtain

$$C(S_n^+) := C^*(u_{i,j}, 1 \leq i,j \leq n \mid u_{i,j}^2 = u_{i,j} = u_{i,j}^*, \ \sum_{i=1}^{n} u_{i,j} = \sum_{j=1}^{n} u_{i,j} = 1) \tag{9.38}$$

As a result, the symmetric quantum group S_n^+ is our first example of a compact matrix quantum group that is not a group.

Example 9.31. We consider two more examples that are obtained by the same liberation procedure. Wang (1995), from the universal C^*-algebra $C(O(n))$ for the orthogonal group $O(n)$

$$C(O(n)) \cong C^*(u_{i,j}, 1 \leq i,j \leq n \mid u_{i,j} = u_{i,j}^*, \ uu^\top = u^\top u = 1, \ u_{i,j}u_{k,l} = u_{k,l}u_{i,j}) \tag{9.39}$$

and from the universal C^*-algebra $C(U(n))$ for the unitary group $U(n)$

$$C(U(n)) \cong C^*(u_{i,j}, 1 \leq i,j \leq n \mid u \text{ is unitary}, \ u_{i,j}u_{k,l} = u_{k,l}u_{i,j}) \tag{9.40}$$

also defined, in a similar way to the symmetric quantum group, the orthogonal quantum group $O(n)^+$

$$C(O(n)^+) := C^*(u_{i,j}, 1 \leq i,j \leq n \mid u_{i,j} = u_{i,j}^*, \ uu^\top = u^\top u = 1) \tag{9.41}$$

and the unitary quantum group $U(n)^+$

$$C(U(n)^+) := C^*(u_{i,j}, 1 \leq i,j \leq n \mid u, u^\top \text{are unitary}) \tag{9.42}$$

As with S_n^+, both $O(n)^+$ and $U(n)^+$ are true compact matrix quantum groups. Note that, by the commutativity of $C(U(n))$, u is unitary implies that $u^\top$ is unitary, but for $C(U(n)^+)$, we also have to choose $u^\top$ to be unitary since choosing u to be unitary does not automatically imply that $u^\top$ is unitary.

To finish this section, we note that there also exists the notion of a compact matrix quantum subgroup, which is defined by using a *-homomorphism between compact matrix quantum groups as follows:

Definition 9.32. Let $(G(n), u)$ and $(H(m), v)$ be compact matrix quantum groups. Then $H(m)$ is said to be a **compact matrix quantum subgroup** of $G(n)$, written $H(m) \subseteq G(n)$, if $n = m$ and if there exists a surjective *-homomorphism $G(n) \to H(m)$ such that $u_{i,j} \mapsto v_{i,j}$ for all $i, j \in [n]$.

9.3 Representation Theory of Compact Matrix Quantum Groups

Analogous to compact matrix groups, compact matrix quantum groups also have a representation theory. We now motivate the definition of a representation of a compact matrix quantum group. Recall that for a compact matrix group $G(n)$, an m-dimensional representation is a choice of m-dimensional vector space V over $\mathbb{C}$ and a continuous group homomorphism $\rho_V : G(n) \to GL(V)$. By choosing a basis of V, we can replace $GL(V)$ by $GL(m)$. The representation ρ_V can itself be represented by functions $v_{i,j} \in C(G(n))$ for $i, j \in [m]$ that map $g \in G(n)$ to the (i, j) entry of $\rho_V(g)$, since ρ_V is continuous. We have the following result.

Lemma 9.33. *If Δ is the comultiplication on $C(G(n))$ defined in (9.29), then the homomorphism property of the representation satisfies*

$$\Delta(v_{i,j})(g, h) = \left(\sum_k v_{i,k} \otimes v_{k,j} \right) (g, h) \tag{9.43}$$

Proof. Indeed, we have that

$$\begin{aligned} \Delta(v_{i,j})(g, h) = v_{i,j}(gh) = [\rho_V(gh)]_{i,j} &= \sum_k [\rho_V(g)]_{i,k} [\rho_V(h)]_{k,j} && (9.44) \\ &= \sum_k v_{i,k}(g) v_{k,j}(h) && (9.45) \\ &= \left(\sum_k v_{i,k} \otimes v_{k,j} \right) (g, h) && (9.46) \end{aligned}$$

as required. □

We generalise Lemma 9.33 to establish the following definition of a representation of a compact matrix quantum group.

Definition 9.34. Let $(G(n), u)$ be a compact matrix quantum group. Then an $\boldsymbol{m}$**-dimensional representation** of $G(n)$ is an element $v \in M_m(C(G(n)))$ such that (9.28) holds. If a representation v of $G(n)$ has a matrix inverse, we say that v is a **non-degenerate** representation, and if v is unitary, that is, $vv^* = 1 = v^*v$, then we call v a **unitary** representation.

Remark 9.35. The matrix u given in the definition of a compact matrix quantum group $(G(n), u)$ is an n-dimensional representation called the **fundamental representation** of $G(n)$, and by Definition 9.26, it is a non-degenerate representation of $G(n)$.

Similar to groups, we can also define the tensor product and complex conjugation of representations, starting with the following definition.

Definition 9.36. Let $v \in M_m(C(G(n)))$ and $w \in M_p(C(G(n)))$. Then the **tensor product** $v \otimes w \in M_{mp}(C(G(n)))$ is simply the Kronecker product of matrices, and the **complex conjugate** is $\bar{v} = (v_{i,j}^*) \in M_m(C(G(n)))$.

Proposition 9.37. *If v, w are representations of a compact matrix quantum group $(G(n), u)$, then both the tensor product and complex conjugate are also representations of $G(n)$.*

Proof. We have that $v \otimes w$ is a representation of $G(n)$ because

$$\Delta(v_{i,j}w_{k,l}) = \Delta(v_{i,j})\Delta(w_{k,l}) = \left(\sum_x v_{i,x} \otimes v_{x,j}\right)\left(\sum_y w_{k,y} \otimes w_{y,l}\right) = \sum_{x,y} v_{i,x}w_{k,y} \otimes v_{x,j}w_{y,l} \quad (9.47)$$

and $\bar{v}$ is a representation of $G(n)$ because

$$\Delta(v_{i,j}^*) = \Delta(v_{i,j})^* = \left(\sum_k v_{i,k} \otimes v_{k,j}\right)^* = \sum_k v_{i,k}^* \otimes v_{k,j}^* \quad (9.48)$$

where in both (9.47) and (9.48) we have used that Δ is a $*$-homomorphism. □

The following lemma will be useful when it comes to considering the fundamental representation of a compact matrix quantum group.

Lemma 9.38. *Let $(G(n), u)$ be a compact matrix quantum group. Then $u^\top$ is unitary if and only if $\bar{u}$ is unitary.*

Proof. We have that

$$\bar{u}^* = (u_{i,j}^*)^* = ((u_{j,i}^*)^*) = (u_{j,i}) = u^\top \quad (9.49)$$

and also

$$(u^\top)^* = (u_{j,i})^* = (u^*_{i,j}) = \bar{u} \tag{9.50}$$

Hence

$$u^\top (u^\top)^* = 1 = (u^\top)^* u^\top \iff \bar{u}\bar{u}^* = 1 = \bar{u}^*\bar{u} \tag{9.51}$$

as required. □

We can also define the analogous concept of equivariance for compact matrix quantum groups.

Definition 9.39. Let $v \in M_m(C(G(n)))$ and $w \in M_p(C(G(n)))$ be representations of a compact matrix quantum group $G(n)$. A map $\phi : \mathbb{C}^m \to \mathbb{C}^p$ is said to be $G(n)$**-equivariant** (also known as an **intertwiner**) if $\phi v = w\phi$. The set of all such *linear* maps is denoted by $\mathrm{Hom}_{G(n)}(v, w)$, and it is, in fact, a vector space.

Remark 9.40. It is worth noting that the expression $\phi v = w\phi$ given in Definition 9.39 has a different interpretation relative to its sister definition for group equivariance. The representation $v \in M_m(C(G(n)))$ *coacts* on the space $\mathbb{C}^m$ in the following sense: if $x \in \mathbb{C}^m$, then $vx \in \mathbb{C}^m \otimes C(G(n))$. Hence v can be thought of as a map $\mathbb{C}^m \to \mathbb{C}^m \otimes C(G(n))$ or as an element of $M_m(\mathbb{C}) \otimes C(G(n))$. In fact, any representation on a vector space V can be described by a map $V \to V \otimes C(G(n))$. Consequently, $\phi \in \mathrm{Hom}_{G(n)}(v, w)$ if and only if it satisfies the commutative square:

$$\begin{tikzcd}
\mathbb{C}^m \arrow[r, "v"] \arrow[d, "\phi"'] & \mathbb{C}^m \otimes C(G(n)) \arrow[d, "\phi"] \\
\mathbb{C}^p \arrow[r, "w"] & \mathbb{C}^p \otimes C(G(n))
\end{tikzcd} \tag{9.52}$$

where the right-hand arrow maps $x \otimes f \mapsto \phi x \otimes f$.

Definition 9.41. We say that two representations $v \in M_m(C(G(n)))$ and $w \in M_m(C(G(n)))$ are **equivalent** if there is an invertible linear map $\phi : \mathbb{C}^m \to \mathbb{C}^m$ that intertwines v and w.

One can show (Gromada, 2020, Proposition 2.2.3) that every non-degenerate representation is equivalent to a unitary representation. Hence, going forward, we always choose the fundamental representation u of a compact matrix quantum group and its transpose to be unitary. By Lemma 9.38, we therefore have that both u and $\bar{u}$ are unitary.

9.4 Tannaka–Krein Duality for Two-Coloured Representation Categories

Woronowicz's version of Tannaka–Krein duality (Woronowicz, 1988) is an important result in the theory of compact matrix quantum groups. Informally, the duality says that we can construct a compact matrix quantum group just by knowing its fundamental representation category. We will use this duality to derive compact matrix quantum group equivariant neural networks in Section 9.5. We begin with some definitions.

Definition 9.42. Consider the **two-coloured set** $\{\circ, \bullet\}$ consisting of a white point and a black point. For any non-negative integer k, we can construct a **word** w of length k as a string of k colours from $\{\circ, \bullet\}$. If $k = 0$, we define $\varnothing$ to be the empty word. The **concatenation** of two words, w_1, w_2, is the concatenation of their two strings and is written as $w_1 \cdot w_2$. We define a **homomorphism on words**, $w \mapsto \bar{w}$, by first defining it on the individual colours by $\bar{\circ} := \bullet$, $\bar{\bullet} := \circ$, and then applying it elementwise to the word w. If w is a word, then its **involution** w^* is the word read backwards together with its colours inverted.

Example 9.43. If $w_k := \circ \bullet \bullet \circ \bullet \circ$ is a word of length 6 and $w_l := \bullet \circ \bullet \bullet \circ$ is a word of length 5, then $w_k \cdot w_l = \circ \bullet \bullet \circ \bullet \circ \bullet \circ \bullet \bullet \circ$, $\bar{w}_k := \bullet \circ \circ \bullet \circ \bullet$, and $w_k^* := \bullet \circ \bullet \circ \circ \bullet$.

We can use words of colours to create tensor products of the fundamental representation of a compact matrix quantum group as follows.

Definition 9.44. Let $(G(n), u)$ be a compact matrix quantum group, where $u \in M_n(C(G(n)))$. Let $u^\circ := u$ and $u^\bullet := \bar{u}$. Then, for any word w formed from the two-coloured set $\{\circ, \bullet\}$, we define $u^{\otimes w}$ to be the corresponding tensor product of representations u° and $u^\bullet$.

Moreover, if w_k and w_l are words of lengths k, l respectively, then we define $\text{FundRep}_{G(n)}(w_k, w_l)$ to be $\text{Hom}_{G(n)}(u^{\otimes w_k}, u^{\otimes w_l})$, that is, the vector space

$$\{\phi : (\mathbb{C}^n)^{\otimes k} \to (\mathbb{C}^n)^{\otimes l} \mid \phi u^{\otimes w_k} = u^{\otimes w_l} \phi\} \tag{9.53}$$

Example 9.45. If w is the word $\circ \bullet \bullet \circ$, then $u^{\otimes w} = u \otimes \bar{u} \otimes \bar{u} \otimes u$.

We can also construct so-called representation categories using words of colours as follows.

Definition 9.46. A **two-coloured representation category** $\mathcal{C}$ is a collection of subspaces $\mathcal{C}(w_k, w_l)$, where w_k, w_l are two words constructed from $\{\circ, \bullet\}$ of lengths k, l respectively, that are subspaces of the set of all linear maps $(\mathbb{C}^n)^{\otimes k} \to (\mathbb{C}^n)^{\otimes l}$ and which satisfy the following axioms.

1. If $\phi_1 \in \mathcal{C}(w_k, w_l)$, $\phi_2 \in \mathcal{C}(w_q, w_m)$, then $\phi_1 \otimes \phi_2 \in \mathcal{C}(w_k \cdot w_q, w_l \cdot w_m)$.

2. If $\phi_1 \in \mathcal{C}(w_k, w_l)$, $\phi_2 \in \mathcal{C}(w_l, w_m)$, then $\phi_2 \circ \phi_1 \in \mathcal{C}(w_k, w_m)$.

3. If $\phi \in \mathcal{C}(w_k, w_l)$, then $\phi^* \in \mathcal{C}(w_l, w_k)$.

4. For every word w (having some length k), we have $1_n^{\otimes k} \in \mathcal{C}(w, w)$, and

5. The map $R : 1 \mapsto \sum_{i=1}^n e_i \otimes e_i$ is in both $\mathcal{C}(\varnothing, \circ\bullet)$ and $\mathcal{C}(\varnothing, \bullet\circ)$.

Definition 9.47. A **one-coloured representation category** is a two-coloured representation category where $u^\circ = u^\bullet$. Consequently, the words correspond precisely to natural numbers because, for each natural number k, there is only one word of length k in this case.

The following two theorems describe the relationship between two-coloured representation categories and compact matrix quantum groups: namely, the fundamental representation category of a compact matrix quantum group is a two-coloured representation category, and a two-coloured representation category can be used to construct a compact matrix quantum group by viewing it as the fundamental representation category of that compact matrix quantum group. The second half of the previous statement is Woronowicz's version of Tannaka–Krein duality (Woronowicz, 1988).

Theorem 9.48. *If $(G(n), u)$ is a compact matrix quantum group, then* $\mathrm{FundRep}_{G(n)}$ *is a two-coloured representation category.*

Proof. It is easy to show that $\mathrm{FundRep}_{G(n)}$ satisfies the first four axioms of Definition 9.46. For example, the second axiom says that if $\phi_1 \in \mathrm{FundRep}_{G(n)}(w_k, w_l)$ and $\phi_2 \in \mathrm{FundRep}_{G(n)}(w_l, w_m)$, then we have that

$$(\phi_2 \circ \phi_1)u^{\otimes w_k} = \phi_2 u^{\otimes w_l}\phi_1 = u^{\otimes w_m}(\phi_2 \circ \phi_1) \tag{9.54}$$

and so $\phi_2 \circ \phi_1 \in \mathrm{FundRep}_{G(n)}(w_k, w_m)$.

For the fifth axiom, we need to show that the operator $R : 1 \mapsto \sum_{i=1}^n e_i \otimes e_i$ is in both $\mathrm{FundRep}_{G(n)}(\varnothing, \circ\bullet)$ and $\mathrm{FundRep}_{G(n)}(\varnothing, \bullet\circ)$.

For the first case, we have that

$$R \in \mathrm{FundRep}_{G(n)}(\varnothing, \circ\bullet) \iff R = (u \otimes \bar{u})R \tag{9.55}$$

By applying both sides to 1, we have that (9.55) holds if and only if

$$\sum_{i=1}^{n} e_i \otimes e_i = \sum_{i=1}^{n} ue_i \otimes \bar{u}e_i = \sum_{i=1}^{n} \sum_{j,k=1}^{n} e_j \otimes e_k \otimes u_{j,i}u_{k,i}^* = \sum_{j,k=1}^{n} e_j \otimes e_k \otimes (uu^*)_{j,k} \tag{9.56}$$

Hence, by equating coefficients, we see that $R \in \text{FundRep}_{G(n)}(\varnothing, \circ\bullet)$ if and only if $uu^* = 1$. For the second case, a similar calculation shows that $R \in \text{FundRep}_{G(n)}(\varnothing, \bullet\circ)$ if and only if $\bar{u}\bar{u}^* = 1$. Since u and $\bar{u}$ are both unitary, we obtain the result. □

Theorem 9.49 (Woronowicz–Tannaka–Krein Duality, Woronowicz (1988)). *Let $\mathcal{C}$ be a two-coloured representation category. Then there exists a unique compact matrix quantum group $(G(n), u)$ such that* $\text{FundRep}_{G(n)} = \mathcal{C}$. *If $E = \{x_{i,j} \mid 1 \leq i, j \leq n\}$ is a set of generators for the free *-algebra $P(E)$, then the *-ideal $I_{G(n)} \subseteq P(E)$ that determines $G(n)$ is given by*

$$\textit{span}\left\{[\phi x^{\otimes w_k} - x^{\otimes w_l}\phi]_{I,J} \;\middle|\; \phi \in \mathcal{C}(w_k, w_l), I \in [n]^l, J \in [n]^k\right\} \tag{9.57}$$

9.5 Compact Matrix Quantum Group Equivariant Neural Networks

In this section, we present the major contribution of this chapter: we derive a new class of neural networks which we term *compact matrix quantum group equivariant neural networks* using Woronowicz's formulation of Tannaka–Krein duality. In the following definition, when referring to a word of length k_l, we write w_l instead of w_{k_l} to suppress multiple subscripts.

Definition 9.50. Let $(G(n), u)$ be a compact matrix quantum group with fundamental representation $u \in M_n(C(G(n)))$. A **compact matrix quantum group equivariant neural network** for $G(n)$ is a function f_{NN} that is a composition of some L layer functions

$$f_{NN} \coloneqq f_L \circ \ldots \circ f_l \circ \ldots \circ f_1 \tag{9.58}$$

such that the l^{th} layer function is equivariant to $G(n)$ and is a function between representation spaces of $G(n)$ under tensor products and complex conjugations of u

$$f_l : ((\mathbb{C}^n)^{\otimes k_{l-1}}, u^{\otimes w_{l-1}}) \rightarrow ((\mathbb{C}^n)^{\otimes k_l}, u^{\otimes w_l}) \tag{9.59}$$

that is itself a composition

$$f_l \coloneqq \sigma_l \circ \phi_l \tag{9.60}$$

of a learnable, linear function in $\text{FundRep}_{G(n)}(w_{l-1}, w_l)$

$$\phi_l : ((\mathbb{C}^n)^{\otimes k_{l-1}}, u^{\otimes w_{l-1}}) \to ((\mathbb{C}^n)^{\otimes k_l}, u^{\otimes w_l}) \tag{9.61}$$

that is equivariant to $G(n)$, together with a fixed, non-linear activation function

$$\sigma_l : ((\mathbb{C}^n)^{\otimes k_{l-1}}, u^{\otimes w_l}) \to ((\mathbb{C}^n)^{\otimes k_l}, u^{\otimes w_l}) \tag{9.62}$$

that is also equivariant to $G(n)$. By choosing the standard basis of $\mathbb{C}^n$, the learnable, linear functions ϕ_l that are equivariant to $G(n)$ become **matrices** that have **weights** that can be **learned**. As usual, we call these matrices **(equivariant) weight matrices**.

Remark 9.51. A compact matrix quantum group equivariant neural network is well-defined because the fundamental representation category $\text{FundRep}_{G(n)}$ of a compact matrix quantum group $(G(n), u)$ is a two-coloured representation category by Theorem 9.48.

We immediately obtain the following corollary for compact matrix groups $G(n) \subseteq GL(n)$.

Corollary 9.52. *Let f_{NN} be a compact matrix quantum group equivariant neural network for a compact matrix* ***group*** *$(G(n), u)$. If all of the words that are used in the network consist only of the white point $\circ$, then f_{NN} is, in fact, a compact matrix group equivariant neural network for $G(n)$. Moreover, every compact matrix group equivariant neural network is a compact matrix quantum group equivariant neural network.*

9.6 Easy Compact Matrix Quantum Groups

Having just derived the existence of a compact matrix quantum group equivariant neural network, we show in this section that we can characterise the weight matrices that appear in these neural networks for the so-called "easy" compact matrix quantum groups. A compact matrix quantum group is said to be **easy** if its two-coloured representation category is determined under a functor by a so-called two-coloured category of partitions. We first explore two-coloured categories of partitions, then define the functor, construct the easy compact matrix quantum group, and finally characterise its equivariant weight matrices.

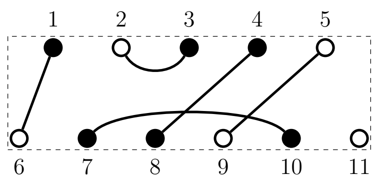
Our first goal is to define a two-coloured category of partitions. We need a few preliminary definitions, but we start from the position of having defined, in Section 3.2.1, a set partition π of $[l + k]$ and its associated (k, l)–partition diagram d_π for all non-negative integers l and k. In this chapter, we sometimes call (k, l)–partition diagrams **one-coloured** to distinguish them from the two-coloured partition diagrams that follow. As before, we represent one-coloured (k, l)–partition diagrams with white vertices in this chapter.

Definition 9.53. A **(two-coloured)** (w_k, w_l)**–partition diagram** is a (k, l)–partition diagram together with two-coloured words w_k, w_l of lengths k, l respectively such that the vertices on the top row have the same colours as the word w_l, from left to right, and the vertices on the bottom row have the same colours as the word w_k, from left to right. We use the same notation d_π to refer to the two-coloured version because w_k and w_l will be known in making reference to the diagram. We define the **two-coloured partition vector space** $P_{w_k}^{w_l}(n)$ to be the $\mathbb{C}$–linear span of the set of all (w_k, w_l)–partition diagrams.

Example 9.54. Suppose that π is the set partition of $[5 + 6]$ given by

$$\{1, 6 \mid 2, 3 \mid 4, 8 \mid 5, 9 \mid 7, 10 \mid 11\} \tag{9.63}$$

Then



(9.64)

is the two-coloured (w_k, w_l)–partition diagram d_π where w_k is the word $\circ \bullet \bullet \circ \bullet \circ$ and w_l is the word $\bullet \circ \bullet \bullet \circ$.

We can also define the following three $\mathbb{C}$–(bi)linear operations on two-coloured set partition diagrams

$$\textbf{composition: } \bullet : P_{w_l}^{w_m}(n) \times P_{w_k}^{w_l}(n) \to P_{w_k}^{w_m}(n) \tag{9.65}$$

$$\textbf{tensor product: } \otimes : P_{w_k}^{w_l}(n) \times P_{w_q}^{w_m}(n) \to P_{w_k \cdot w_q}^{w_l \cdot w_m}(n) \tag{9.66}$$

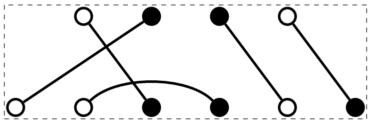
$$\textbf{involution: } * : P_{w_k}^{w_l}(n) \to P_{w_l}^{w_k}(n) \tag{9.67}$$

They are defined in full as follows.

Definition 9.55 (Composition)**.** Let $d_{\pi_1} \in P_{w_k}^{w_l}(n)$ and $d_{\pi_2} \in P_{w_l}^{w_m}(n)$. First, we concatenate the diagrams, written $d_{\pi_2} \circ d_{\pi_1}$, by putting d_{π_1} below d_{π_2}, concatenating the edges in the middle row of vertices, and then removing all connected components that lie entirely in the middle row of the concatenated diagrams. Let $c(d_{\pi_2}, d_{\pi_1})$ be the number of connected components that are removed from the middle row in $d_{\pi_2} \circ d_{\pi_1}$. Then the composition is defined, using infix notation, as

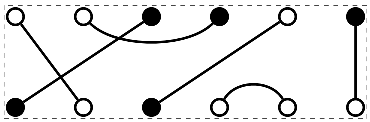
$$d_{\pi_2} \bullet d_{\pi_1} := n^{c(d_{\pi_2}, d_{\pi_1})}(d_{\pi_2} \circ d_{\pi_1}) \tag{9.68}$$

Example 9.56. If d_{π_2} is the two-coloured $(\circ \circ \bullet \bullet \circ \bullet, \circ \bullet \bullet \circ)$–partition diagram



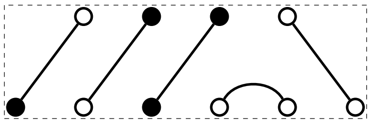
(9.69)

and d_{π_1} is the two-coloured $(\bullet \circ \bullet \circ \circ \circ, \circ \circ \bullet \bullet \circ \bullet)$–partition diagram



(9.70)

then we have that $d_{\pi_2} \circ d_{\pi_1}$ is the $(\bullet \circ \bullet \circ \circ \circ, \circ \bullet \bullet \circ)$–partition diagram

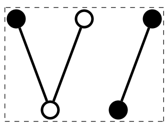


(9.71)

and so $d_{\pi_2} \bullet d_{\pi_1}$ is the diagram (9.71) multiplied by n, since one connected component was removed from the middle row of $d_{\pi_2} \circ d_{\pi_1}$.

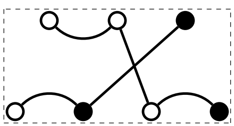
Definition 9.57 (Tensor Product)**.** Let $d_{\pi_1} \in P_{w_k}^{w_l}(n)$ and $d_{\pi_2} \in P_{w_q}^{w_m}(n)$. Then $d_{\pi_1} \otimes d_{\pi_2}$ is defined to be the $(w_k \cdot w_q, w_l \cdot w_m)$–partition diagram that is obtained by horizontally placing d_{π_1} to the left of d_{π_2} without any overlapping of vertices.

Example 9.58. If d_{π_1} is the two-coloured $(\circ \bullet, \bullet \circ \bullet)$–partition diagram



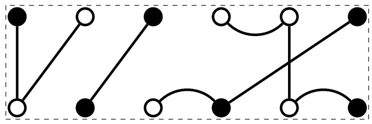
(9.72)

and d_{π_2} is the two-coloured $(\circ \bullet \circ \bullet, \circ \circ \bullet)$–partition diagram



(9.73)

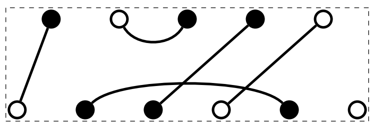
then we have that $d_{\pi_1} \otimes d_{\pi_2}$ is the $(\circ \bullet \circ \bullet \circ \bullet, \bullet \circ \bullet \circ \circ \bullet)$–partition diagram



(9.74)

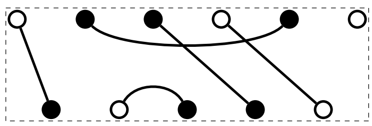
Definition 9.59 (Involution)**.** Let $d_\pi \in P_{w_k}^{w_l}(n)$. Then $d_\pi^* \in P_{w_l}^{w_k}(n)$ is defined by reflecting d_π in the horizontal axis.

Example 9.60. Recalling the two-coloured $(\circ \bullet \bullet \circ \bullet \circ, \bullet \circ \bullet \bullet \circ)$–partition diagram d_π from Example 9.54 (reprinted below for ease)



(9.75)

we have that d_π^* is the $(\bullet \circ \bullet \bullet \circ, \circ \bullet \bullet \circ \bullet \circ)$–partition diagram



(9.76)

We can now define a category for two-coloured partition diagrams as follows:

Definition 9.61. $\mathcal{P}(n)$ is the category whose objects are the two-coloured words and, for any pair of objects w_k and w_l, the morphism space $\mathrm{Hom}_{\mathcal{P}(n)}(w_k, w_l)$ is defined to be $P_{w_k}^{w_l}(n)$.

The vertical composition of morphisms is the composition given in Definition 9.55, the tensor product of morphisms is the tensor product given in Definition 9.57, and the unit object is the empty word $\varnothing$.

Proposition 9.62. *$\mathcal{P}(n)$ is a strict $\mathbb{C}$–linear monoidal category.*

Proof. $\mathcal{P}(n)$ is a strict monoidal category because the bifunctor on objects reduces to the concatenation operation on words, which is associative, and the bifunctor on morphisms is the tensor product of linear combinations of two-coloured partition diagrams that is given in Definition 9.57, which is associative by definition.

$\mathcal{P}(n)$ is $\mathbb{C}$–linear because the morphism space between any two objects is by definition a vector space, and the composition of morphisms is $\mathbb{C}$–bilinear by definition. For the same reason, the bifunctor is also $\mathbb{C}$–bilinear. $\square$

We achieve our first goal with the following definition.

Definition 9.63. A **two-coloured category of partitions** $\mathcal{K}(n)$ is any subcategory of $\mathcal{P}(n)$ such that

1. if w_k, w_l are objects in $\mathcal{K}(n)$, then $\mathrm{Hom}_{\mathcal{K}(n)}(w_k, w_l)$ is a subspace of $\mathrm{Hom}_{\mathcal{P}(n)}(w_k, w_l)$,
2. the identity partition diagram is an element of $\mathrm{Hom}_{\mathcal{K}(n)}(\circ, \circ)$,
3. the top-row pair partition diagram corresponding to the set partition $\{1, 2\}$ of $\{1, 2\}$, when superimposed with the word $\circ\bullet$, is in $\mathrm{Hom}_{\mathcal{K}(n)}(\varnothing, \circ\bullet)$, and, when superimposed with the word $\bullet\circ$, is in $\mathrm{Hom}_{\mathcal{K}(n)}(\varnothing, \bullet\circ)$, and
4. the morphism spaces are closed under the vertical, horizontal and involution operations.

By Proposition 9.62, it is immediate that a two-coloured category of partitions $\mathcal{K}(n)$ is a strict $\mathbb{C}$–linear monoidal category.

We now introduce the critical functor for our purposes: we show that this functor takes two-coloured (w_k, w_l)–partition diagrams d_π that live in a two-coloured category of partitions $\mathcal{K}(n)$ to linear maps $\phi_\pi : (\mathbb{C}^n)^{\otimes k} \to (\mathbb{C}^n)^{\otimes l}$ that live in a two-coloured representation category. By choosing the standard basis for $\mathbb{C}^n$, we obtain $n^l \times n^k$ matrices instead, which are denoted by D_π in what follows.

Definition 9.64. Suppose that d_π is a (w_k, w_l)–partition diagram, where w_k, w_l are two-coloured words of lengths k, l respectively. We define D_π as follows.

Associate the indices $i_1, i_2, \dots, i_l$ with the vertices in the top row of d_π and $j_1, j_2, \dots, j_k$ with the vertices in the bottom row of d_π. Then, if $S_\pi((I, J))$ is defined to be the set

$$\left\{(I, J) \in [n]^{l+k} \mid \text{if } x, y \text{ are in the same block of } \pi, \text{then } i_x = i_y\right\} \tag{9.77}$$

(where we have momentarily replaced the elements of J by $i_{l+m} := j_m$ for all $m \in [k]$), we define

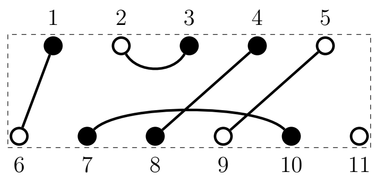
$$D_\pi := \sum_{I \in [n]^l, J \in [n]^k} \delta_{\pi,(I,J)} E_{I,J} \tag{9.78}$$

where

$$\delta_{\pi,(I,J)} := \begin{cases} 1 & \text{if } (I, J) \in S_\pi((I, J)) \\ 0 & \text{otherwise} \end{cases} \tag{9.79}$$

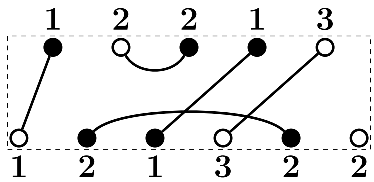
and $E_{I,J}$ is the $n^l \times n^k$ matrix having a 1 in the (I, J) position and is 0 elsewhere. We extend the definition of the map $d_\pi \mapsto D_\pi$ linearly to $P_{w_k}^{w_l}(n)$.

Example 9.65. If d_π is the two-coloured $(\circ \bullet \bullet \circ \bullet \circ, \bullet \circ \bullet \bullet \circ)$–partition diagram



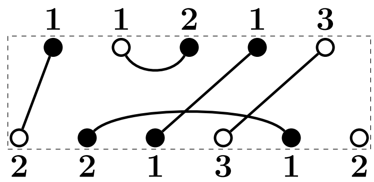
(9.80)

given in Example 9.54, then, if $n = 3$, for example, we see from



(9.81)

that the $(1, 2, 2, 1, 3 \mid 1, 2, 1, 3, 2, 2)$-entry of D_π is 1, whereas, we see from



(9.82)

that the $(1, 1, 2, 1, 3 \mid 2, 2, 1, 3, 1, 2)$-entry of D_π is 0.

Consequently, we obtain the following key result, which makes it possible for us to construct a two-coloured representation category from any two-coloured category of partitions.

Theorem 9.66. *For any two-coloured category of partitions $\mathcal{K}(n)$, we can form a category $\mathcal{C}_\mathcal{K}(n)$ whose objects are the same as those in $\mathcal{K}(n)$, and, for any pair of objects w_k and w_l, the morphism space $\mathrm{Hom}_{\mathcal{C}_\mathcal{K}(n)}(w_k, w_l)$ is the image of $\mathrm{Hom}_{\mathcal{K}(n)}(w_k, w_l)$ under $d_\pi \mapsto D_\pi$; that is, $\mathrm{Hom}_{\mathcal{C}_\mathcal{K}(n)}(w_k, w_l)$ is defined to be*

$$\{D_\pi \mid d_\pi \in \mathrm{Hom}_{\mathcal{K}(n)}(w_k, w_l)\} \tag{9.83}$$

Then the map $d_\pi \mapsto D_\pi$ defines a strict $\mathbb{C}$–linear monoidal functor

$$\Theta : \mathcal{K}(n) \to \mathcal{C}_\mathcal{K}(n) \tag{9.84}$$

such that $\mathcal{C}_\mathcal{K}(n)$ is a two-coloured representation category.

Proof. We prove this theorem in a series of steps. We first consider the image $\mathcal{C}_{\mathcal{P}}(n)$ of the two-coloured partition category $\mathcal{P}(n)$ under $d_\pi \mapsto D_\pi$.

Step 1: We need to show that $\mathcal{C}_{\mathcal{P}}(n)$ is actually a category.

The objects and the morphisms have been defined in the statement of the theorem. For every triple of objects, the composition rule is given by matrix multiplication, which is associative. Furthermore, the identity morphism for every object w_k of length k is the $n^k \times n^k$ identity matrix. Note also that the unit object is $\varnothing$, the empty word.

Step 2: We need to show that $\Theta : \mathcal{P}(n) \to \mathcal{C}_{\mathcal{P}}(n)$ is a functor.

- Θ maps the objects of $\mathcal{P}(n)$ to the objects of $\mathcal{C}_{\mathcal{P}}(n)$ since Θ maps each word w_k to itself, by definition.
- It is enough to show that $\Theta(g)\Theta(f) = \Theta(g \bullet f)$ on arbitrary basis elements of arbitrary morphism spaces where the codomain of f is the domain of g because the morphism spaces are vector spaces.

  Let $f = d_{\pi_1}$ be a (w_k, w_l)–partition diagram, and let $g = d_{\pi_2}$ be a (w_l, w_m)–partition diagram. Then, by Definition 9.55, we have that

  $$d_{\pi_2} \bullet d_{\pi_1} = n^{c(d_{\pi_2}, d_{\pi_1})} d_{\pi_3} \tag{9.85}$$

  where d_{π_3} be the composition $d_{\pi_2} \circ d_{\pi_1}$ expressed as a (w_k, w_m)–partition diagram. Then

  $$\Theta(g \bullet f) = \Theta(d_{\pi_2} \bullet d_{\pi_1}) = n^{c(d_{\pi_2}, d_{\pi_1})} D_{\pi_3} \tag{9.86}$$

  We also have that

  $$\Theta(g)\Theta(f) = D_{\pi_2} D_{\pi_1} = \left(\sum_{I \in [n]^m, K \in [n]^l} \delta_{\pi_2,(I,K)} E_{I,K} \right) \left(\sum_{L \in [n]^l, J \in [n]^k} \delta_{\pi_1,(L,J)} E_{L,J} \right) \tag{9.87}$$

  $$= \sum_{I \in [n]^m, K \in [n]^l, J \in [n]^k} \delta_{\pi_2,(I,K)} \delta_{\pi_1,(K,J)} E_{I,K} E_{K,J} \tag{9.88}$$

  For fixed I, J, consider

  $$\sum_{K \in [n]^l} \delta_{\pi_2,(I,K)} \delta_{\pi_1,(K,J)} \tag{9.89}$$

This is equal to

$$n^{c(d_{\pi_2},d_{\pi_1})}\delta_{\pi_3,(I,J)} \tag{9.90}$$

Indeed, for fixed I, J, $\delta_{\pi_3,(I,J)}$ is 1 if and only if both $\delta_{\pi_2,(I,K)}$ and $\delta_{\pi_1,(K,J)}$ are 1 for some $K \in [n]^l$ since d_{π_3} is the composition $d_{\pi_2} \circ d_{\pi_1}$. The number of such K is determined only by the vertices that appear in connected components that are removed from the middle row of $d_{\pi_2} \circ d_{\pi_1}$, since, for fixed I, J, only these vertices can be freely chosen if we want both $\delta_{\pi_2,(I,K)}$ and $\delta_{\pi_1,(K,J)}$ to be 1. However, since the entries in each connected component must be the same for both $\delta_{\pi_2,(I,K)}$ and $\delta_{\pi_1,(K,J)}$ to be 1, this implies that the number of $K \in [n]^l$ such that both $\delta_{\pi_2,(I,K)}$ and $\delta_{\pi_1,(K,J)}$ are 1 is $n^{c(d_{\pi_2},d_{\pi_1})}$.

Hence (9.88) becomes

$$\sum_{I\in[n]^m, J\in[n]^k} n^{c(d_{\pi_2},d_{\pi_1})}\delta_{\pi_3,(I,J)}E_{I,J} = n^{c(d_{\pi_2},d_{\pi_1})}D_{\pi_3} \tag{9.91}$$

and so, by (9.86), we have that $\Theta(g)\Theta(f) = \Theta(g \bullet f)$, as required.

- For each object w_k in $\mathcal{P}(n)$, the identity morphism 1_{w_k} is the two-coloured (w_k, w_k)–partition diagram d_π, where π is the set partition

$$\{1, k+1 \mid 2, k+2 \mid \cdots \mid k, 2k\} \tag{9.92}$$

of $[2k]$, and its image under $d_\pi \mapsto D_\pi$ is the $n^k \times n^k$ identity matrix, as required.

Step 3: We need to show that $\mathcal{C}_{\mathcal{P}}(n)$ is a strict $\mathbb{C}$–linear monoidal category.

We first consider the tensor product of arbitrary basis elements of arbitrary morphism spaces. Let d_π be a (w_k, w_l)–partition diagram, and let d_τ be a (w_m, w_q)–partition diagram. Then, defining d_ω to be the $(w_k \cdot w_q, w_l \cdot w_m)$–partition diagram, where ω is the set partition $\pi \cup \tau$ of $[l+m+k+q]$, we see that

$$D_\pi \otimes D_\tau = \left(\sum_{I\in[n]^l, J\in[n]^k} \delta_{\pi,(I,J)}E_{I,J}\right) \otimes \left(\sum_{X\in[n]^m, Y\in[n]^q} \delta_{\tau,(X,Y)}E_{X,Y}\right) \tag{9.93}$$

$$= \sum_{(I,X)\in[n]^{l+m}, (J,Y)\in[n]^{k+q}} \delta_{\omega,(I,X),(J,Y))}E_{(I,X),(J,Y)} \tag{9.94}$$

$$= D_\omega \tag{9.95}$$

since $S_\pi((I,J)) \cup S_\tau((X,Y)) = S_\omega((I,X),(J,Y))$.

The bifunctor on objects of $\mathcal{C}_{\mathcal{P}}(n)$ reduces to the concatenation operation on words, which is associative. Also, since the bifunctor on morphisms in $\mathcal{C}_{\mathcal{P}}(n)$ is the tensor product of linear combinations of images of two-coloured partition diagrams, it is associative because the tensor product of images of two-coloured partition diagrams is associative, both by the above calculation and because taking unions of set partitions is associative. Hence, $\mathcal{C}_{\mathcal{P}}(n)$ is a strict monoidal category.

$\mathcal{C}_{\mathcal{P}}(n)$ is $\mathbb{C}$–linear because the morphism space between any two objects is by definition a vector space, and the composition of morphisms is $\mathbb{C}$–bilinear by definition. For the same reason, the bifunctor is also $\mathbb{C}$–bilinear.

Step 4: We need to show that $\Theta : \mathcal{P}(n) \to \mathcal{C}_{\mathcal{P}}(n)$ is a strict $\mathbb{C}$–linear monoidal functor.

1. Θ preserves the tensor product on objects, since $\Theta(w_k) = w_k$ for any object in $\mathcal{P}(n)$, and the tensor product in both categories is given by the concatenation of words.

2. It is enough to show that $\Theta(f) \otimes \Theta(g) = \Theta(f \otimes g)$ on arbitrary basis elements of arbitrary morphism spaces as the morphism spaces are vector spaces. This result is now immediate from the calculation given in (9.93).

3. It is clear that Θ sends the unit object $\varnothing$ in $\mathcal{P}(n)$ to $\varnothing$, which is the unit object in $\mathcal{C}_{\mathcal{P}}(n)$.

4. This is immediate because the map $d_\pi \mapsto D_\pi$ is $\mathbb{C}$–linear by Definition 9.64.

Step 5: We need to show that $\mathcal{C}_{\mathcal{P}}(n)$ is a two-coloured representation category.

The tensor product, concatenation and identity axioms have already been shown in previous steps. For the involution, if D_π is an element of $\mathrm{Hom}_{\mathcal{C}_{\mathcal{P}}(n)}(w_k, w_l)$ and is the image of a (w_k, w_l)–partition diagram d_π under Θ, then

$$(D_\pi)^* = \left(\sum_{I \in [n]^l, J \in [n]^k} \delta_{\pi,(I,J)} E_{I,J} \right)^* = \sum_{J \in [n]^k, I \in [n]^l} \delta_{\pi^*,(J,I)} E_{J,I} = D_{\pi^*} \tag{9.96}$$

As D_{π^*} is the image of the (w_l, w_k)–partition diagram d_{π^*} under Θ, this implies that $(D_\pi)^*$ is in $\mathrm{Hom}_{\mathcal{C}_{\mathcal{P}}(n)}(w_l, w_k)$, as required.

Finally, we see that the map $R : 1 \mapsto \sum_{i=1}^{n} e_i \otimes e_i$ is in both $\mathrm{Hom}_{\mathcal{C}_{\mathcal{P}}(n)}(\varnothing, \circ\bullet)$ and $\mathrm{Hom}_{\mathcal{C}_{\mathcal{P}}(n)}(\varnothing, \bullet\circ)$ since it is the image under $d_\pi \mapsto D_\pi$ of the top-row pair partition diagram corresponding to the set partition $\{1, 2\}$ of $\{1, 2\}$ superimposed, on the one hand, with the word $w_2 := \circ\bullet$, and on the other hand, with the word $w_2 := \bullet\circ$.

Step 6: We need to show that $d_\pi \mapsto D_\pi$ defines a strict $\mathbb{C}$–linear monoidal functor $\Theta : \mathcal{K}(n) \to \mathcal{C}_\mathcal{K}(n)$ such that $\mathcal{C}_\mathcal{K}(n)$ is a two-coloured representation category.

This is now immediate by Definition 9.63, the definition of a two-coloured category of partitions, and since the functor on $\mathcal{P}(n)$ is strict $\mathbb{C}$–linear monoidal whose image is a two-coloured representation category. In particular, the morphism spaces for $\mathcal{K}(n)$ are closed under composition, tensor product and involution, and so these are inherited by their image under Θ. Also, the map $R : 1 \mapsto \sum_{i=1}^n e_i \otimes e_i$ is in $\mathrm{Hom}_{\mathcal{C}_\mathcal{K}(n)}(\varnothing, \circ\bullet)$ and $\mathrm{Hom}_{\mathcal{C}_\mathcal{K}(n)}(\varnothing, \bullet\circ)$, since the top row pair partition diagram corresponding to the set partition $\{1, 2\}$ of $\{1, 2\}$, when superimposed with the word $\circ\bullet$, is in $\mathrm{Hom}_{\mathcal{K}(n)}(\varnothing, \circ\bullet)$, and, when superimposed with the word $\bullet\circ$, is in $\mathrm{Hom}_{\mathcal{K}(n)}(\varnothing, \bullet\circ)$. $\square$

Hence we can use Woronowicz–Tannaka–Krein duality, as given in Theorem 9.49, to not only construct the easy compact matrix quantum group but also to characterise the equivariant weight matrices that appear in the neural network that was derived in Definition 9.50, as follows.

Theorem 9.67. *A two-coloured category of partitions $\mathcal{K}(n)$ determines a unique compact matrix quantum group $(G(n), u)$. We call these compact matrix quantum groups **easy**, and they can be constructed as the universal C^*-algebra*

$$C(G(n)) = C^*(u_{i,j}, 1 \leq i, j \leq n \mid D_\pi u^{\otimes w_k} = u^{\otimes w_l} D_\pi \textit{ for all } d_\pi \in \mathcal{K}(n)(w_k, w_l)) \tag{9.97}$$

Moreover, the weight matrix W that appears in a $G(n)$-equivariant linear layer function of the form (9.61) is

$$W = \sum_{\pi \mid d_\pi \in \mathcal{K}(n)(w_{l-1}, w_l)} \lambda_\pi D_\pi \tag{9.98}$$

9.7 Examples

We noted at the beginning of this chapter that easy compact matrix quantum groups have been studied extensively in the literature and many classes of them have been characterised in full. We present a number of important easy compact matrix quantum groups below, characterise their equivariant weight matrices, and refer the reader to the referenced works cited throughout the following text for the other known cases.

A General Procedure for Calculating the Weight Matrix of an Equivariant Linear Layer Function from $((\mathbb{C}^n)^{\otimes k}, u^{\otimes w_k}) \rightarrow ((\mathbb{C}^n)^{\otimes l}, u^{\otimes w_l})$ for an Easy Compact Matrix Quantum Group $(G(n), u)$.

Assume that $\mathcal{K}(n)$ is the two-coloured category of partitions that defines the easy compact matrix quantum group $(G(n), u)$ as in Theorem 9.67.

Perform the following steps:

1. Calculate all of the two-coloured (w_k, w_l)–partition diagrams d_π that live in $\mathcal{K}(n)$. These diagrams form a basis of the morphism space $\mathrm{Hom}_{\mathcal{K}(n)}(w_k, w_l)$.
2. Apply the function $d_\pi \mapsto D_\pi$ to each two-coloured (w_k, w_l)–partition diagram d_π to obtain its associated spanning set matrix D_π.
3. Attach a weight $\lambda_\pi \in \mathbb{C}$ to each matrix D_π.
4. Finally, calculate $\sum \lambda_\pi D_\pi$ to give the overall weight matrix.

A Procedure for Calculating the (I, J)-entry of each Equivariant Spanning Set Matrix D_π from $((\mathbb{C}^n)^{\otimes k}, u^{\otimes w_k}) \rightarrow ((\mathbb{C}^n)^{\otimes l}, u^{\otimes w_l})$ for an Easy Compact Matrix Quantum Group $(G(n), u)$.

Assume that d_π is a two-coloured (w_k, w_l)–partition diagram in the two-coloured category of partitions $\mathcal{K}(n)$ that uniquely determines $(G(n), u)$.

Perform the following steps:

1. Place the indices I on the top row of d_π and the indices J on the bottom row of d_π.
2. If all of the vertices in each block in d_π have been overlaid with the same number, then the (I, J)-entry of D_π is 1, otherwise it is 0.

Note in the following that it is enough to provide the spanning set of matrices (9.83) for an easy compact matrix quantum group, since, by (9.98), the weight matrix is a weighted linear combination of these matrices.

9.7.1 One-Coloured Partition Categories

One-coloured partition categories were introduced initially by Banica and Speicher (2009), studied further in Banica et al. (2010), Weber (2013) and Raum and Weber (2014, 2015), and then characterised fully by Raum and Weber (2016).

They can be divided into four cases: group, non-crossing, half-liberated, and the rest. The two most important cases are the group and non-crossing ones: we refer the reader either to Banica et al. (2010) or to Gromada (2020) for results relating to the other cases. The corresponding compact matrix quantum groups $G(n)$ are known in the literature as **easy orthogonal compact matrix quantum groups**, or sometimes **Banica–Speicher quantum groups**, because $S_n \subseteq G(n) \subseteq O(n)^+$.

Group: Banica and Speicher (2009) provided a full characterisation: there are only six compact matrix groups such that $S_n \subseteq G(n) \subseteq O(n)^+$. Consequently, this results in characterisations of six compact matrix group equivariant neural networks, of which two — the symmetric group S_n (Ravanbakhsh et al., 2017; Zaheer et al., 2017; Maron et al., 2019a; Godfrey et al., 2023; Pearce-Crump, 2024) and the orthogonal group $O(n)$ (Pearce-Crump, 2023a) — were discussed separately in Chapters 5 and 6. Said differently, we have found characterisations of the weight matrices that appear in four new compact matrix group equivariant neural networks.

Theorem 9.68 (Banica and Speicher (2009) [Theorem 2.8])**.** *There are only six easy orthogonal compact matrix groups such that $S_n \subseteq G(n) \subseteq O(n)^+$ — the groups and the spanning sets that determine the weight matrices of Theorem 9.67 are given below. In what follows, when we refer to the image of a (k,l)–partition diagram, we mean the image under the map $d_\pi \mapsto D_\pi$ given in Definition 9.64.*

- *The symmetric group S_n: the image of all (k,l)–partition diagrams.*
- *The orthogonal group $O(n)$: the image of all (k,l)–partition diagrams whose blocks come in pairs.*
- *The hyperoctahedral group H_n (the symmetry group of the hypercube): the image of all (k,l)–partition diagrams having blocks of even size.*
- *The bistochastic group B_n (the group of orthogonal matrices having sum 1 in each row and in each column): the image of all (k,l)–partition diagrams having blocks of size one or two.*
- *The modified symmetric group $S_n' := \mathbb{Z}_2 \times S_n$: the image of all (k,l)–partition diagrams where the number of blocks of odd size is even.*
- *The modified bistochastic group $B_n' := \mathbb{Z}_2 \times B_n$: the image of all (k,l)–partition diagrams having an even number of blocks of size one and any number of blocks of size two.*

Remark 9.69. We have seen in Section 6.1 that we can improve upon the spanning set for the symmetric group S_n by removing all (k,l)–partition diagrams that have more than n blocks from the set of all (k,l)–partition diagrams before taking their image. The resulting set of matrices is the diagram basis for S_n.

Non-Crossing: In order to state the classifications for the non-crossing case, we first need the following definition.

Definition 9.70. A set partition diagram d_π corresponding to a set partition π of $[l+k]$ is said to be **crossing** if there exist four integers $1 \leq x_1 < x_2 < x_3 < x_4 \leq l+k$ satisfying:

1. x_1 and x_3 are in the same block
2. x_2 and x_4 are in the same block, and
3. x_1 and x_2 are not in the same block.

Otherwise, d_π is said to be **non-crossing**.

The six easy orthogonal compact matrix groups that were found by Banica and Speicher (2009) were **liberated** by the same authors to create six compact matrix quantum groups — which are true quantum groups — that fall into the non-crossing case. As before, liberation means that crossings are not allowed in any of the set partition diagrams. Weber (2013) found a seventh using this method. We call these compact matrix quantum groups **free** since their categories of partitions $\mathcal{K}(n)$ only contain non-crossing set partition diagrams. We have the following result.

Theorem 9.71. *There are only seven free easy orthogonal compact matrix quantum groups. We list the quantum groups and the spanning sets that determine the weight matrices of Theorem 9.67 below.*

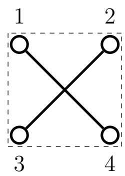
The first six are the symmetric quantum group S_n^+, the orthogonal quantum group $O(n)^+$, the hyperoctahedral quantum group H_n^+, the bistochastic quantum group B_n^+, the modified symmetric quantum group $S_n'^+$, and the modified bistochastic quantum group $B_n'^+$, whose spanning sets are determined by the same sets of set partition diagrams that were given in Theorem 9.68 for their "sister" groups, but with all crossing set partition diagrams removed.

The seventh is the freely modified bistochastic quantum group $B_n^{\#+}$. To obtain its corresponding spanning set, we consider the same set of set partition diagrams as for $B_n'^+$, but now temporarily label each vertex with a colour from $\{\circ, \bullet\}$ in an alternating fashion, starting with $\bullet$ in the top right corner and moving counterclockwise, ultimately finishing in the bottom right corner. We only retain those set partition diagrams where the blocks of size two pair one $\circ$ with one $\bullet$, and take its image under $d_\pi \mapsto D_\pi$, removing the colours in the process.

In fact, we obtain a stronger classification for the non-crossing case due to the following theorem.

Theorem 9.72 (Banica and Speicher (2009) [Theorem 3.8]). *The spanning sets for the seven compact matrix quantum groups in the non-crossing case are bases when $n \geq 4$.*

Remark 9.73. By Theorem 9.67, we note that each easy compact matrix quantum group $(G(n), u)$ that appears in Theorems 9.68 and 9.71 is *defined by* the two-coloured category of partitions $\mathcal{K}(n)$ that is associated with it. That is, Banica and Speicher (2009) actually showed that there are only six one-coloured categories of partitions that contain the swap partition diagram, that is, the $(2, 2)$–partition diagram



(9.99)

and together with Weber (2013) they showed that there are only seven one-coloured categories of partitions that are non-crossing. With these restrictions on the possible one-coloured categories of partitions in place they could immediately apply Woronowicz–Tannaka–Krein duality to construct the easy compact matrix quantum groups from each of these one-coloured categories of partitions. In fact, we have, again by Theorem 9.67, that each compact matrix quantum group is precisely the universal C^*-algebra

$$C(G(n)) = C^*(u_{i,j}, 1 \leq i, j \leq n \mid D_\pi u^{\otimes w_k} = u^{\otimes w_l} D_\pi \text{ for all } d_\pi \in \mathcal{K}(n)(w_k, w_l)) \tag{9.100}$$

Example 9.74. We calculate the weight matrices that appear in the compact matrix quantum group equivariant neural networks for all of the easy orthogonal compact matrix quantum groups that appear in the group and non-crossing case when $n = k = l = 2$. Recall from Theorems 9.68 and 9.71 that the weight matrices are formed as a weighted linear combination of the image under $d_\pi \mapsto D_\pi$ of some subset of all of the possible $(2, 2)$–partition diagrams.

There are $B(4) = 15$ $(2, 2)$–partition diagrams in total, where $B(4)$ is the fourth Bell number. They are shown in Figure 9.1. We number the diagrams from $i = 1$ to 15, going from left to right and then top to bottom, assigning a weight w_i to each diagram. Figure 9.2 shows the images of the fifteen $(2, 2)$–partition diagrams under the map $d_\pi \mapsto D_\pi$. We apply the characterisations of Theorems 9.68 and 9.71 to obtain the following results.

For the group case, we have that

- **The Symmetric Group** S_n: The weight matrix is a weighted linear combination of the images of all fifteen $(2, 2)$–partition diagrams.

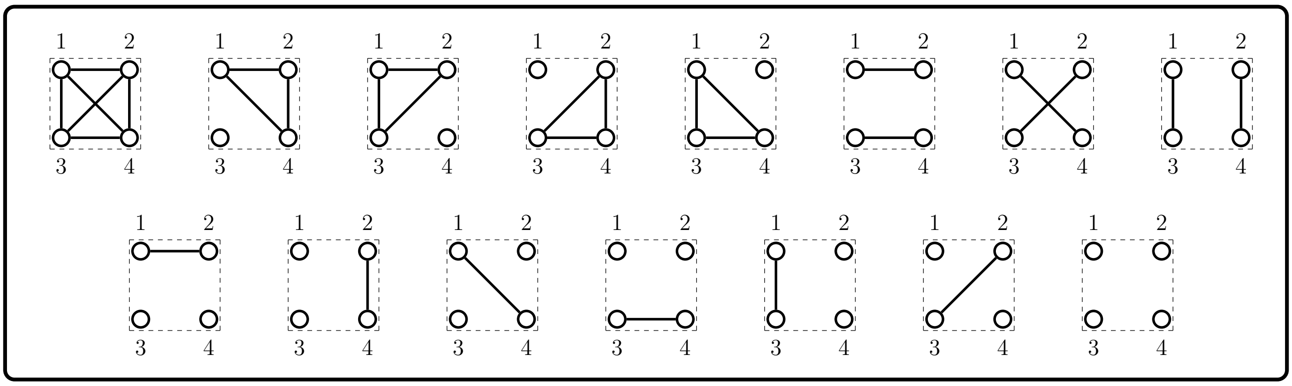


Figure 9.1: The fifteen $(2,2)$–partition diagrams.

- **The Orthogonal Group** $O(n)$: The weight matrix is a weighted linear combination of the images of diagrams six, seven and eight.

- **The Hyperoctahedral Group** H_n: The weight matrix is (coincidentally) the same as for the orthogonal group in this case, since the $(2,2)$–partition diagrams with blocks of even size are the same as the $(2,2)$–partition diagrams whose blocks come in pairs.

- **The Bistochastic Group** B_n: The weight matrix is a weighted linear combination of the images of diagrams six through fifteen, inclusive.

- **The Modified Symmetric Group** S_n': The weight matrix is (coincidentally) the same as for the symmetric group, since the number of blocks of odd size in any $(2,2)$–partition diagram is either $0, 2$ or 4.

- **The Modified Bistochastic Group** B_n': The weight matrix is (coincidentally) the same as for the bistochastic group, since the number of blocks of odd size in any $(2,2)$–partition diagram is either $0, 2$ or 4.

and for the non-crossing case, we have that

- **The Symmetric Quantum Group** S_n^+: The weight matrix is a weighted linear combination of the images of all diagrams except for seven, which is the only crossing diagram.

- **The Orthogonal Quantum Group** $O(n)^+$: The weight matrix is a weighted linear combination of the images of diagrams six and eight.

- **The Hyperoctahedral Quantum Group** H_n^+: The weight matrix is the same as for $O(n)^+$.

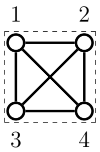
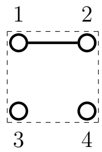
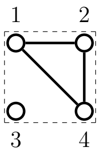
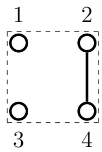
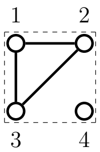
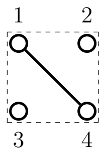
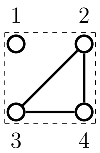
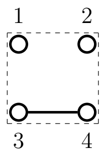
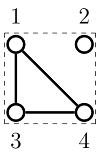
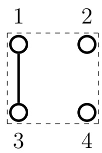
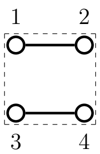
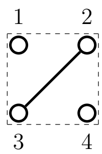
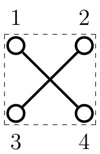
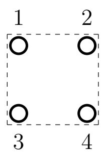
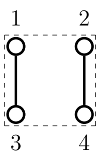
<table>
<tr><th>d_π</th><th>D_π</th><th>d_π</th><th>D_π</th></tr>
<tr><td></td><td>$\begin{array}{cc} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right] \end{array}$</td><td></td><td>$\begin{array}{cc} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc|cc} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{array}\right] \end{array}$</td></tr>
<tr><td></td><td>$\begin{array}{cc} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc|cc} 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{array}\right] \end{array}$</td><td></td><td>$\begin{array}{cc} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc|cc} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ \hline 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{array}\right] \end{array}$</td></tr>
<tr><td></td><td>$\begin{array}{cc} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc|cc} 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{array}\right] \end{array}$</td><td></td><td>$\begin{array}{cc} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc|cc} 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ \hline 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{array}\right] \end{array}$</td></tr>
<tr><td></td><td>$\begin{array}{cc} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ \hline 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right] \end{array}$</td><td></td><td>$\begin{array}{cc} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc|cc} 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ \hline 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{array}\right] \end{array}$</td></tr>
<tr><td></td><td>$\begin{array}{cc} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{array}\right] \end{array}$</td><td></td><td>$\begin{array}{cc} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc|cc} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{array}\right] \end{array}$</td></tr>
<tr><td></td><td>$\begin{array}{cc} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc|cc} 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{array}\right] \end{array}$</td><td></td><td>$\begin{array}{cc} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc|cc} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ \hline 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{array}\right] \end{array}$</td></tr>
<tr><td></td><td>$\begin{array}{cc} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right] \end{array}$</td><td></td><td>$\begin{array}{cc} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc|cc} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{array}\right] \end{array}$</td></tr>
<tr><td></td><td>$\begin{array}{cc} & \begin{array}{cccc} 1,1 & 1,2 & 2,1 & 2,2 \end{array} \\ \begin{array}{c} 1,1 \\ 1,2 \\ 2,1 \\ 2,2 \end{array} & \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right] \end{array}$</td><td></td><td></td></tr>
</table>

Figure 9.2: For $n = 2$, we display the images under the map $d_\pi \mapsto D_\pi$ for each of the $(2, 2)$–partition diagrams.

- **The Bistochastic Quantum Group** B_n^+: The weight matrix is a weighted linear combination of the images of diagrams six and eight through fifteen, inclusive.
- **The Modified Symmetric Quantum Group** $S_n'^+$: The weight matrix is the same as for S_n^+.
- **The Modified Bistochastic Quantum Group** $B_n'^+$: The weight matrix is the same as for B_n^+.
- **The Freely Modified Bistochastic Quantum Group** $B_n^{\#+}$: We colour in the diagrams that appear in the weight matrix for $B_n'^+$, and retain only the diagrams where the blocks of size two pair one $\circ$ with one $\bullet$. Consequently, we remove diagrams eleven and fourteen from the set, and so the weight matrix is a weighted linear combination of the images of diagrams six, eight, nine, ten, twelve, thirteen, and fifteen.

9.7.2 Two-Coloured Partition Categories

Two-coloured partition categories are much richer than one-coloured partition categories because the generators $u_{i,j}$ of the corresponding compact matrix quantum groups are no longer self-adjoint. They were first investigated by Freslon and Weber (2016). Unfortunately, at the time of writing, a full characterisation of these categories is unknown. However, Tarrago and Weber (2016, 2018) classified all two-coloured partition categories in the group and non-crossing cases. In particular, there are seven series of two-coloured partition categories in the group case and twelve series of two-coloured partition categories in the non-crossing case. The corresponding compact matrix quantum groups $G(n)$ are known as **easy unitary compact matrix quantum groups**, since they satisfy $S_n \subseteq G(n) \subseteq U(n)^+$. We focus on the characterisation for only the two most important series: the unitary group $U(n)$ and the unitary quantum group $U(n)^+$, referring the reader to Tarrago and Weber (2016, Theorem 5.3) for the characterisations of the other ones.

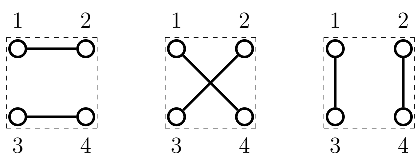
Tarrago and Weber (2016, 2018) showed that the two-coloured category of partitions for the unitary group $U(n)$ has, for fixed words w_k and w_l, a morphism space that is spanned by all two-coloured (w_k, w_l)–partition diagrams whose blocks come in pairs such that if two vertices of a block are in the same row, then they have different colours, otherwise they have the same colours. Tarrago and Weber (2016, 2018) also showed that the two-coloured category of partitions for the unitary quantum group $U(n)^+$ has, for words w_k and w_l, a morphism space that is spanned by all *non-crossing* two-coloured (w_k, w_l)–partition diagrams satisfying the same conditions as those for the unitary group. As before, we obtain the $n^l \times n^k$ weight matrices of Theorem 9.67 for these compact matrix quantum groups by taking the images of the two-coloured partition diagrams that span the respective morphism spaces under the map $d_\pi \mapsto D_\pi$ given in Definition 9.64.

This introduces the rather interesting point for the unitary group $U(n)$. If we pick two words w_k and w_l, then the $n^l \times n^k$ weight matrix corresponding to these words that appears in a compact matrix quantum group equivariant neural network for the unitary group $U(n)$ is the $\mathbb{C}$–linear span of the image of all (w_k, w_l)–partition diagrams that live in the two-coloured partition category for $U(n)$. However, if we consider compact matrix *group* equivariant neural networks for $U(n)$ instead, as the words can only be formed from white points, then not only must we have that k equals l, by the characterisation of the two-coloured partition category for $U(n)$, but also we must have that the $n^k \times n^k$ weight matrix is the image of all permutations in the symmetric group S_k, expressed as (k, k)–partition diagrams. We see that this is the classic version of Schur–Weyl duality.

Example 9.75. We calculate the weight matrices that appear in compact matrix quantum group equivariant neural networks for the unitary group $U(n)$ and unitary quantum group $U(n)^+$ when $n = k = l = 2$.

The easiest way to do this is to consider the one-coloured partition diagrams that come in pairs and then superimpose colours onto each row of vertices to see which two-coloured partition diagrams match the classification result.

The one-coloured $(2, 2)$–partition diagrams that come in pairs are



(9.101)

We label the diagrams as 1 to 3 from left to right. Note that the matrices obtained under $d_\pi \mapsto D_\pi$ are the same no matter the colours on the vertices, hence we can reuse the matrices for diagrams six, seven and eight from Figure 9.2 in order to calculate the weight matrices.

We now consider pairs of words. As there are only 4 words of length 2, this implies that there are 16 such pairs of words. We obtain the following result for the weight matrices for $U(2)$, where for each cell, we take weighted linear combinations of the matrices D_π for each set partition diagram in the cell.

$$\begin{array}{c|cccc} (I,J) & \circ\circ & \circ\bullet & \bullet\circ & \bullet\bullet \\ \hline \circ\circ & 2,3 & \varnothing & \varnothing & 2,3 \\ \circ\bullet & \varnothing & 1,3 & 1,2 & \varnothing \\ \bullet\circ & \varnothing & 1,2 & 1,3 & \varnothing \\ \bullet\bullet & 2,3 & \varnothing & \varnothing & 2,3 \end{array} \tag{9.102}$$

Similarly, for $U(2)^+$, the weight matrices are weighted linear combinations of the images of the following set partition diagrams, where we have removed the second diagram from (9.102).

$$\begin{array}{c|cccc} (I,J) & \circ\circ & \circ\bullet & \bullet\circ & \bullet\bullet \\ \hline \circ\circ & 3 & \varnothing & \varnothing & 3 \\ \circ\bullet & \varnothing & 1,3 & 1 & \varnothing \\ \bullet\circ & \varnothing & 1 & 1,3 & \varnothing \\ \bullet\bullet & 3 & \varnothing & \varnothing & 3 \end{array} \tag{9.103}$$

Note that the top left hand cell of (9.102) provides a classification of the weight matrix appearing in the compact matrix *group* equivariant neural network for $U(2)$ when $n = k = l = 2$.

Chapter 10

Conclusion

10.1 Summary of Thesis Achievements

We have looked at the problem of finding principled approaches for constructing group equivariant neural networks whose layers are tensor power spaces of $\mathbb{R}^n$. Not only have we successfully addressed this challenge, but we have also gone far beyond our original objectives in the process. In particular, our work firmly establishes weight tying as a fundamental paradigm for constructing equivariant neural networks.

We can divide our contributions into three overarching themes.

The first major contribution of our work is two-fold. First, we have discovered a novel approach for obtaining the weight matrices between tensor power layer spaces for many groups, without needing to decompose the layer spaces into direct sums of irreducible representations. Second, we have shown that set partition diagrams play a crucial role in equivariant deep learning. In particular, we saw that Schur–Weyl duality between the partition algebra and the symmetric group, that was simultaneously found by Martin (1990, 1994, 1996) and Jones (1994), could be adapted to give a characterisation of the permutation equivariant weight matrices in Chapter 3. In Chapter 4, we added in a jellyfish representing the determinant map to obtain this characterisation for the alternating group. For the orthogonal and symplectic groups, we saw in Chapter 5 that set partition diagrams where the vertices come in pairs give the characterisation, and for the special orthogonal group, we saw how these diagrams together with those that pair some vertices and attach a jellyfish to the remaining unpaired vertices give the characterisation. Similar to the symmetric group, these results came from adapting a number of Schur–Weyl dualities that Brauer (1937) found. In Chapter 9, we also saw that set partition diagrams could be used to give characterisations of the equivariant weight matrices for a number of other groups, the most important of which are the hyperoctahedral, bistochastic and unitary groups.

The second major contribution of our work is the realisation that certain subsets of set partition diagrams can be given the structure of a strict $\mathbb{R}$–linear monoidal category. We saw that this was not merely a restructuring of our previous results, but that this structure came with additional benefits. In particular, the strict $\mathbb{R}$–linear monoidal functor relationships that we established between the partition categories and the representation categories for the tensor power layer spaces enabled us to treat the $\mathbb{R}$–linear span of the partition diagrams as if they were the weight matrices themselves. This approach resulted in an algorithm for performing the linear equivariant layer functions that reduced the time complexity exponentially compared with a naïve matrix multiplication. In Chapter 7, we used the fact that the strict $\mathbb{R}$–linear monoidal functors respect the structure of the tensor product to factor the original weight matrix into a Kronecker product of smaller sized matrices (up to permutations), which can be executed faster owing to the parallelism that is inherent in a Kronecker product of matrices. In Chapter 8, we applied these ideas to form a strict $\mathbb{R}$–linear monoidal category from bilabelled graph diagrams, which we used to characterise the weight matrices that appear in neural networks that are equivariant to the automorphism group of a graph.

The third major contribution of our work is the construction of a whole new class of equivariant neural networks. These neural networks are equivariant to compact matrix quantum groups, which, unlike groups, are algebras that generalise the space of continuous functions on compact matrix groups. Where groups are used to describe symmetries in classical, commutative geometries, compact matrix quantum groups are often used to describe symmetries in non-commutative geometries. In particular, we used Woronowicz's version of Tannaka–Krein duality to derive the existence of compact matrix quantum group equivariant neural networks. We saw that this duality made it possible for us to reconstruct a compact matrix quantum group from its fundamental representation category. We also saw that the equivariant weight matrices for the easy compact matrix quantum groups could be characterised using one- and two-coloured set partition diagrams.

10.2 Applications

As our work is primarily a theoretical contribution, we do not expect profound societal impact in the short term. However, in the medium term, a number of applications may well emerge from the theory having high levels of impact. Equivariant deep learning models in general have been used in a wide range of applications, including, but not restricted to, particle physics (Bogatskiy et al., 2020; Villar et al., 2021; Bogatskiy et al., 2022; Gong et al., 2022), protein structure prediction (Eismann et al., 2021;

Jumper et al., 2021; Ganea et al., 2022), biological and medical imaging (Bekkers et al., 2018; Lafarge et al., 2021; Suk et al., 2024), molecular and quantum chemistry (Thomas et al., 2018; Finzi et al., 2020; Fuchs et al., 2020; Miller et al., 2020; Satorras et al., 2021; Schütt et al., 2021; Batatia et al., 2022; Batzner et al., 2022; Hoogeboom et al., 2022; Liao and Smidt, 2023), quantum error correction decoders (Egorov et al., 2023), reinforcement learning (van der Pol et al., 2020; Wang et al., 2022a,b,c), computer vision (Marcos et al., 2017; Worrall et al., 2017; Esteves et al., 2019; Deng et al., 2021; Chatzipantazis et al., 2023; Kaba et al., 2023), object dynamics prediction (Guttenberg et al., 2016) and modelling compositional structures in natural language (Gordon et al., 2020; Petrache and Trivedi, 2024).

We suspect that the two most promising applications of our work could be in learning functions on point clouds and predicting properties of molecular structures, both of which involve rotation and permutation symmetries (Dym and Maron, 2021). The design philosophy of the state-of-the-art models in these domains (Thomas et al., 2018; Weiler et al., 2018a; Fuchs et al., 2020; Liao and Smidt, 2023) relies on $SO(3)$-equivariant weight matrices that are constructed from a truncated basis of spherical harmonics, which are the irreducible representations of $SO(3)$. We propose that replacing these weight matrices with the $SO(3)$-equivariant weight matrices that map between tensor power space representations, which we characterised in full in Chapter 5, could improve the efficiency of both the training and inference processes. Additionally, the permutation equivariant neural networks that we characterised in Chapter 3 have already proven effective in learning from both set-structured and graph-structured data (Zaheer et al., 2017; Hartford et al., 2018; Maron et al., 2019a). We also anticipate that the compact matrix quantum group equivariant neural networks that we introduced in Chapter 9 may have valuable applications in particle physics. Indeed, Woronowicz (1987) originally introduced compact matrix quantum groups to model symmetries in particle physics that could not be described by groups.

However, we recognise that achieving these applications will be challenging owing to the limitations of current hardware, particularly since the neural networks that we have studied consist of layers that are tensor power space representations. In particular, significant engineering efforts will be needed because storing high-order tensors in memory is not a straightforward task. This has already been demonstrated by Kondor et al. (2018), who had to develop custom CUDA kernels in order to implement their tensor product based neural networks. However, we anticipate that with the increasing availability of computing power, higher-order group equivariant neural networks will become more prevalent in practical applications. We are beginning to see such neural networks being used in practical applications: for example, third-order tensor power spaces appeared in the architecture for AlphaFold2 (Jumper et al., 2021). Furthermore, with the need to create algorithms that employ parallelism in order to take advantage of the graphics processing units that continue to be improved, we expect that the techniques that we presented in Chapter 7 could result in new algorithms that improve both the training and inference times of equivariant neural networks that use tensor power space representations as their layers. Finally, we also believe that the ideas that we have explored could benefit those working on machine learning models for quantum computers as these models, by design, are built using tensor power spaces.

10.3 Universality and Expressibility

As a consequence of the universal approximation theorems for multi-layer perceptrons (Cybenko, 1989; Hornik et al., 1989; Pinkus, 1999), it is a natural question in equivariant deep learning to explore the expressibility and approximation power of invariant and equivariant models. The use of high-order tensor power representations as layer spaces in the group equivariant neural networks that we have considered improve their potential to be highly expressive, that is, they could increase the networks' capacity to model more complex relationships while maintaining equivariance. However, although universality of equivariant neural networks is highly desirable, it is not always guaranteed: for example, Xu et al. (2019) showed that the message passing model (Gilmer et al., 2017), which is commonly used in graph neural networks, is not universal. We review the existing literature on universality for invariant and equivariant networks before concluding on the consequences of these results for the neural networks that we have studied in this thesis.

A number of authors have focused on the universality question for invariant networks. Two pioneering permutation equivariant models, Deep Sets (Zaheer et al., 2017) and PointNet (Qi et al., 2017), played a key role in inspiring research in this area as they each showed that their models are universal approximators for invariant continuous functions on sets. Yarotsky (2022) demonstrated that one can trivially construct invariant universal neural networks for finite permutation groups by performing group averaging on a non-invariant universal network; however, this construction is impractical since the size of the symmetric group S_n is $n!$. Maron et al. (2019b) showed that, for any subgroup $G(n)$ of the symmetric group S_n, any continuous function that is invariant to $G(n)$ of the form $\mathbb{R}^n \to \mathbb{R}$ can be approximated to arbitrary precision by a neural network that is invariant to $G(n)$. They used high-order tensor power spaces of $\mathbb{R}^n$ to prove their result and showed that, to achieve universality, the order of these spaces depends on the group $G(n)$. They also proved that if $G(n)$ is the alternating group A_n, then any neural network that is invariant to A_n that uses tensor power spaces of order at most $\frac{n-2}{2}$ in its construction cannot approximate arbitrary continuous A_n-invariant functions. A corollary of the main result in Maron et al. (2019b) implies that continuous permutation invariant functions on graphs can also be approximated to arbitrary precision by a permutation invariant network.

Proving universality results for equivariant neural networks has been more difficult. Segol and Lipman (2020) showed that PointNet (Qi et al., 2017), while universal for approximating invariant continuous functions, is not universal for approximating equivariant continuous functions. However, the same authors showed that the Deep Sets model (Zaheer et al., 2017) is equivariant universal. Yarotsky (2022) proved that any group equivariant, continuous map on completely reducible representations can be approximated by a single layer perceptron consisting of invariant generators as inputs and equvariant generators as coefficients in the outputs. Keriven and Peyré (2019) extended the work of Maron et al. (2019b) by showing that certain single layer equivariant vector-valued graph neural networks are universal approximators of equivariant vector-valued continuous functions on graphs. Dym and Maron (2021) constructed a general approach for proving the universality of rotation equivariant models for three dimensional point clouds, and showed that two popular equivariant models for learning from these point clouds, Tensor Field Networks (Thomas et al., 2018) and $SE(3)$-Transformers (Fuchs et al., 2020), are equivariant universal. Ravanbakhsh (2020) studied the approximation power of permutation equivariant neural networks that use high-order tensor power spaces of $\mathbb{R}^n$, providing strong bounds on the orders of the tensor powers that can be used in order to guarantee equivariant universality of the resulting networks. Finally, Finzi et al. (2021) showed that equivariant neural networks built out of direct sums of $O(n)$-equivariant linear maps between tensor power spaces, combined with non-linearities that act on each of the tensors separately, are not equivariant universal.

The invariant and equivariant universality results that appeared in some of these works (Zaheer et al., 2017; Keriven and Peyré, 2019; Maron et al., 2019b; Ravanbakhsh, 2020; Segol and Lipman, 2020; Finzi et al., 2021) apply immediately to their corresponding instances of the group equivariant neural networks that we have studied in this thesis. However, as noted in Keriven and Peyré (2019), extending existing methods to obtain universality results for equivariant continuous functions whose output lives in a high-order tensor power space remains a significant challenge. We hope to address this problem in future work.

10.4 Future Work

We started our research at a time when one of the major unanswered questions in deep learning was how to construct neural networks using principled approaches. There is still much more research to be done on this question, but we have certainly made significant contributions in this direction. Our work comes at a time when other machine learning researchers are beginning to look to category theory as a way of creating neural network architectures based on fundamental principles (Cornish, 2024; Gavranović et al., 2024), and our category theoretic results give us confidence in thinking that this could be a fruitful direction for the future.

In principle, designing equivariant neural networks is a promising idea. However, despite their use in a wide range of applications, the question remains as to whether a truly mainstream application can be identified for these networks. Discovering such an application could significantly increase interest in the field. A more immediate question is whether equivariant neural networks work effectively at scale. There is a general, albeit largely untested, belief that it is difficult to scale equivariant models. To address this issue, some researchers have begun to explore methods to approximate equivariance as opposed to enforcing it exactly (Romero and Lohit, 2022; Wang et al., 2022d; Petrache and Trivedi, 2023; Hofgard et al., 2024; Huang et al., 2024), while others are abandoning equivariance altogether in favour of more traditional data augmentation techniques (Wang et al., 2023; Abramson et al., 2024). Despite these various approaches, the original question remains unresolved and is largely unexplored in the literature. At the time of writing, one empirical work (Brehmer et al., 2024) suggests that exact equivariant models could perform well when scaled with greater computational resources and larger datasets. However, this study was limited to a single $E(3)$-equivariant dataset and compared only one non-equivariant model with an $E(3)$-equivariant model, largely due to the prohibitive financial costs that are associated with large scale experiments. It is clear that a thorough, systematic investigation, supported by solid theoretical foundations, is needed to settle this debate.

Regarding our specific contributions, we think that there is potential for the ideas that we explored in Chapters 3 and 8, namely on the relationship between set partitions and equivariance to the symmetric group and on the relationship between bilabelled graphs and equivariance to the automorphism group of a graph, to be extended to create deep learning models for data that lives on other topological domains. In this case, we can think of the symmetric group as being the automorphism group of a graph consisting solely of a set of elements, and the set partitions as being bilabelled graphs where the underlying graph is a subset of the set of these elements. In topological deep learning, there exist so-called higher-order domains that extend sets and graphs, such as simplicial complexes, cellular complexes, combinatorial complexes and hypergraphs (Hajij et al., 2023; Papamarkou et al., 2024; Papillon et al., 2024), that are formed not only from vertices and edges but also include other types of relations, such as faces and volumes. Potentially, bilabelled diagrams for these higher-order domains could be defined and used to create deep learning models that are equivariant to the automorphism group of instances of these domains.

Finally, we are most excited by what we explored in Chapter 9 on compact matrix quantum groups because we think that it contains the kernel of what could be an important idea for creating neural network architectures for other algebraic structures. In the earlier chapters of our work, we began with an algebraic structure, specifically a group, and focused on identifying the weight matrices that can appear in a neural network that is equivariant to that group. In the last chapter, where the algebraic structure was a compact matrix quantum group, instead of starting with the algebraic structure, we derived it from the category of linear maps that happened to be equivariant to this algebraic structure. We think that this idea could be more general than what we have pursued so far. Consequently, we hope to explore this line of research further in the future.

Bibliography

Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J. Ballard, Joshua Bambrick, Sebastian W. Bodenstein, David A. Evans, Chia Chun Hung, Michael O'Neill, David Reiman, Kathryn Tunyasuvunakool, Zachary Wu, Akvilė Žemgulytė, Eirini Arvaniti, Charles Beattie, Ottavia Bertolli, Alex Bridgland, Alexey Cherepanov, Miles Congreve, Alexander I. Cowen-Rivers, Andrew Cowie, Michael Figurnov, Fabian B. Fuchs, Hannah Gladman, Rishub Jain, Yousuf A. Khan, Caroline M. R. Low, Kuba Perlin, Anna Potapenko, Pascal Savy, Sukhdeep Singh, Adrian Stecula, Ashok Thillaisundaram, Catherine Tong, Sergei Yakneen, Ellen D. Zhong, Michal Zielinski, Augustin Žídek, Victor Bapst, Pushmeet Kohli, Max Jaderberg, Demis Hassabis, and John M. Jumper. Accurate structure prediction of biomolecular interactions with AlphaFold 3. *Nature*, 630(8016):493–500, 2024.

Sir Michael Atiyah. Duality in Mathematics and Physics. In *Conference Lecture at Institut de Mathematica de la Universidad de Barcelona (IMUB)*, pages 69–91, 2007.

James Atwood and Don Towsley. Diffusion-Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, pages 1993–2001, 2016.

Dave Bacon, Isaac L. Chuang, and Aram W. Harrow. Efficient Quantum Circuits for Schur and Clebsch-Gordan Transforms. *Physical Review Letters*, 97(17), 2006a.

Dave Bacon, Isaac L. Chuang, and Aram W. Harrow. The Quantum Schur Transform: I. Efficient Qudit Circuits. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1235—1244, 2006b.

Teodor Banica. *Introduction to Quantum Groups*. Springer Cham, 2023.

Teodor Banica and Roland Speicher. Liberation of orthogonal Lie groups. *Advances in Mathematics*, 222(4):1461–1501, 2009.

Teodor Banica, Stephen Curran, and Roland Speicher. Classification results for easy quantum groups. *Pacific Journal of Mathematics*, 247(1):1–26, 2010.

Hélène Barcelo and Arun Ram. Combinatorial Representation Theory. *New Perspectives in Geometric Combinatorics*, 38:23–90, 1999.

Jeffrey M. Barnes, Georgia Benkart, and Tom Halverson. McKay Centralizer Algebras. *Proceedings of the London Mathematical Society*, 112(2):375–414, 2016.

Ilyes Batatia, David P. Kovacs, Gregor Simm, Christoph Ortner, and Gabor Csanyi. MACE: Higher Order Equivariant Message Passing Neural Networks for Fast and Accurate Force Fields. In *Advances in Neural Information Processing Systems*, volume 35, pages 11423–11436, 2022.

Simon Batzner, Albert Musaelian, Mario Geiger Lixin Sun, Jonathan P. Mailoa, Mordechai Kornbluth, Nicola Molinari, Tess E. Smidt, and Boris Kozinsky. E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *Nature Communications*, 13:2453, 2022.

Erik J. Bekkers, Maxime W. Lafarge, Mitko Veta, Koen A. J. Eppenhof, Josien P. W. Pluim, and Remco Duits. Roto-Translation Covariant Convolutional Networks for Medical Image Analysis. In *Medical Image Computing and Computer Assisted Intervention–MICCAI 2018: 21st International Conference, Granada, Spain, September 16-20, 2018, Proceedings, Part I*, pages 440–448. Springer International Publishing, 2018.

Georgia Benkart. Commuting Actions – a Tale of Two Groups. In *Lie Algebras and Their Representations: A Symposium on Lie Algebras and Representation Theory, January 23 - 27, 1995, Seoul National University, Seoul, Korea*, volume 194, pages 1–46, 1996.

Georgia Benkart. Connecting the McKay correspondence and Schur–Weyl duality. In *Proceedings of the International Congress of Mathematicians, August 13 - 21, 2014, Seoul, Korea*, volume I, pages 633–656, 2014.

Georgia Benkart. Poincaré Series for Tensor Invariants and the McKay Correspondence. *Advances in Mathematics*, 290: 236–259, 2016.

Georgia Benkart and Tom Halverson. Partition algebras $P_k(n)$ with $2k > n$ and the fundamental theorems of invariant theory for the symmetric group S_n. *Journal of the London Mathematical Society*, 99(2):194–224, 2019a.

Georgia Benkart and Tom Halverson. Partition Algebras and the Invariant Theory of the Symmetric Group. In *Recent Trends in Algebraic Combinatorics*, volume 16 of *Association for Women in Mathematics Series*, pages 1–41. Springer, 2019b.

Georgia Benkart and Dongho Moon. Tensor Product Representations of Temperley–Lieb Algebras and Chebyshev Polynomials. In *The Tenth International Conference on Representations of Algebras and Related Topics, July 15 – August 10, 2002, Fields Institute, Toronto, Canada*, volume 45, pages 57–80, 2005.

Georgia Benkart and Dongho Moon. Walks on Graphs and Their Connections with Tensor Invariants and Centralizer Algebras. *Journal of Algebra*, 509:1–39, 2018.

Georgia Benkart, Tom Halverson, and Nate Harman. Dimensions of irreducible modules for partition algebras and tensor power multiplicities for symmetric and alternating groups. *Journal of Algebraic Combinatorics. An International Journal*, 46 (1):77–108, 2017.

Jyotishman Bhowmick, Debashish Goswami, Adam Skalski, and Piotr M. Soltan. Quantum gauge symmetries in noncommutative geometry. *Journal of Noncommutative Geometry*, 8(2):433–471, 2014.

Lawrence C. Biedenharn and James D. Louck. A Pattern Calculus for Tensor Operators in the Unitary Groups. *Communications in Mathematical Physics*, 8(2):89–131, 1968.

Bruce Blackadar. *Operator Algebras*, volume 122 of *Encyclopaedia of Mathematical Sciences*. Springer-Verlag, Berlin, 2006.

Matthew Bloss. G-colored partition algebras as centralizer algebras of wreath products. *Journal of Algebra*, 265(2):690–710, 2003.

Matthew Bloss. The Partition Algebra as a Centralizer Algebra of the Alternating Group. *Communications in Algebra*, 33: 2219–2229, 2005.

Ben Blum-Smith and Soledad Villar. Machine learning and invariant theory. *Notices of the American Mathematical Society*, 70 (8):1205–1213, 2023.

Alexander Bogatskiy, Brandon Anderson, Jan Offermann, Marwah Roussi, David Miller, and Risi Kondor. Lorentz Group Equivariant Neural Network for Particle Physics. In *International Conference on Machine Learning*, pages 992–1002. PMLR, 2020.

Alexander Bogatskiy, Sanmay Ganguly, Thomas Kipf, Risi Kondor, David W. Miller, Daniel Murnane, Jan T. Offermann, Mariel Pettee, Phiala Shanahan, Chase Shimmin, and Savannah Thais. Symmetry Group Equivariant Architectures for Physics, 2022. `arXiv:2203.06153`.

Bela Bollobas. *Modern Graph Theory*. Springer, 1998.

N. Bourbaki. *General Topology: Chapters 1–4*. Springer Berlin, Heidelberg, 1995.

Chris Bowman, Maud De Visscher, and Rosa Orellana. The partition algebra and the Kronecker product (Extended abstract). In *DMTCS Proceedings, 25th International Conference on Formal Power Series and Algebraic Combinatorics (FPSAC 2013)*, pages 289– 300, 2013.

Richard Brauer. On Algebras Which Are Connected with the Semisimple Continuous Groups. *Annals of Mathematics*, 38: 857–872, 1937.

Johann Brehmer, Sönke Behrends, Pim de Haan, and Taco Cohen. Does equivariance matter at scale?, 2024. `arXiv:2410.23179`.

Michael M. Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric Deep Learning: Going Beyond Euclidean Data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Velicković. Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges, 2021. `arXiv:2104.13478`.

William P. Brown. An Algebra Related to the Orthogonal Group. *Michigan Mathematical Journal*, 3(1):1 – 22, 1955.

William P. Brown. The Semisimplicity of the Brauer Algebra. *Annals of Mathematics*, 63(2):324–335, 1956.

Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral Networks and Locally Connected Networks on Graphs. In *International Conference on Learning Representations*, 2014.

Robert Budzyński and Witold Kondracki. Quantum Spaces: Notes and Comments on a Lecture by S. L. Woronowicz. *Banach Center Publications*, 34(1):37–45, 1995.

William Burnside. On the Continuous Group that is defined by any given Group of Finite Order. *Proceedings of the London Mathematical Society*, s1-29(1):207–224, November 1897a.

William Burnside. On the Continuous Group that is defined by any given Group of Finite Order. (Second Paper). *Proceedings of the London Mathematical Society*, s1-29(1):546–565, November 1897b.

William Burnside. On Group-Characteristics. *Proceedings of the London Mathematical Society*, s1-33(1):146–161, November 1900a.

William Burnside. On some Properties of Groups of Odd Order. *Proceedings of the London Mathematical Society*, s1-33:162–184, 1900b.

William Burnside. On the representation of a group of finite order as an irreducible group of linear substitutions and the direct establishment of the relations between the group-characteristics. *Proceedings of the London Mathematical Society*, s2-1: 117–123, 1904.

Martijn Caspers. Quantum symmetry. *Snapshots of Modern Mathematics from Oberwolfach*, 9, 2020.

Tullio Ceccherini-Silberstein, Fabio Scarabotti, and Filippo Tolli. *Representation Theory of the Symmetric Groups*. Cambridge University Press, 2010.

Arthur Chassaniol. Study of quantum symmetries for vertex-transitive graphs using intertwiner spaces. `arXiv:1904.00455`, 2019.

Evangelos Chatzipantazis, Stefanos Pertigkiozoglou, Edgar Dobriban, and Kostas Daniilidis. SE(3)-Equivariant Attention Networks for Shape Reconstruction in Function Space. In *The Eleventh International Conference on Learning Representations*, 2023.

Taco Cohen and Max Welling. Group Equivariant Convolutional Networks. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 2990–2999, New York, USA, 20–22 Jun 2016. PMLR.

Taco Cohen and Max Welling. Steerable CNNs. In *International Conference on Learning Representations*, 2017.

Taco Cohen, Mario Geiger, Jonas Köhler, and Max Welling. Convolutional Networks for Spherical Signals, 2017. `arXiv:1709.04893`.

Taco Cohen, Mario Geiger, Jonas Köhler, and Max Welling. Spherical CNNs. In *International Conference on Learning Representations*, 2018.

Taco Cohen, Mario Geiger, and Maurice Weiler. A General Theory of Equivariant CNNs on Homogeneous Spaces. In *Advances in Neural Information Processing Systems*, volume 32, 2019a.

Taco Cohen, Maurice Weiler, Berkay Kicanaoglu, and Max Welling. Gauge Equivariant Convolutional Networks and the Icosahedral CNN. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR, 09–15 Jun 2019b.

Laura Colmenarejo, Rosa Orellana, Franco Saliola, Anne Schilling, and Mike Zabrocki. An Insertion Algorithm on Multiset Partitions with Applications to Diagram Algebras. *Journal of Algebra*, 557:97–128, 2020.

Jonathan Comes. Jellyfish Partition Categories. *Algebras and Representation Theory*, 23:327–347, 2020.

Alain Connes. *Noncommutative Geometry*. Elsevier Science, January 1995.

Rob Cornish. Stochastic Neural Network Symmetrisation in Markov Categories, 2024. `arXiv:2406.11814`.

Kristin Courtney and Elizabeth Gillaspy. Notes on C*-Algebras, 2023. Online notes from GOALS summer school.

Charles W. Curtis. *Pioneers of Representation Theory: Frobenius, Burnside, Schur, and Brauer*. Number 15 in History of Mathematics. American Mathematical Society, 1999.

George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2: 303–314, 1989.

Pim de Haan, Taco Cohen, and Max Welling. Natural Graph Networks. In *Advances in Neural Information Processing Systems*, volume 33, pages 3636–3646, 2020.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In *Advances in Neural Information Processing Systems*, 2016.

Congyue Deng, Or Litany, Yueqi Duan, Adrien Poulenard, Andrea Tagliasacchi, and Leonidas J. Guibas. Vector Neurons: A General Framework for SO(3)-Equivariant Networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12200–12209, 2021.

V. G. Drinfeld. Quantum groups. *Zapiski Nauchnykh Seminarov POMI (In Russian)*, 155:18–49, 1986.

David K. Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P. Adams. Convolutional Networks on Graphs for Learning Molecular Fingerprints. In *Advances In Neural Information Processing Systems*, volume 28, 2015.

Nadav Dym and Haggai Maron. On the Universality of Rotation Equivariant Point Cloud Networks. In *International Conference on Learning Representations*, 2021.

Richard D. P. East, Guillermo Alonso-Linaje, and Chae-Yeun Park. All you need is spin: SU(2) equivariant variational quantum circuits based on spin networks, 2023. `arXiv:2309.07250`.

Evgenii Egorov, Roberto Bondesan, and Max Welling. The END: An Equivariant Neural Decoder for Quantum Error Correction, 2023. `arXiv:2304.07362`.

Samuel Eilenberg and Saunders Mac Lane. General Theory of Natural Equivalences. *Transactions of the American Mathematical Society*, 58(2):231–294, 1945.

Stephan Eismann, Raphael J. L. Townshend, Nathaniel Thomas, Milind Jagota, Bowen Jing, and Ron O. Dror. Hierarchical, Rotation-Equivariant Neural Networks to Select Structural Models of Protein Complexes. *Proteins: Structure, Function, and Bioinformatics*, 89(5):493–501, 2021.

Carlos Esteves, Christine Allen-Blanchette, Ameesh Makadia, and Kostas Daniilidis. Learning SO(3) Equivariant Representations with Spherical CNNs. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–68, 2018.

Carlos Esteves, Yinshuang Xu, Christine Allen-Blanchette, and Kostas Daniilidis. Equivariant Multi-View Networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

Ben Finkelshtein, Chaim Baskin, Haggai Maron, and Nadav Dym. A Simple and Universal Rotation Equivariant Point-Cloud Network. In *Proceedings of Topological, Algebraic, and Geometric Learning Workshops 2022*, volume 196 of *Proceedings of Machine Learning Research*, pages 107–115. PMLR, 25 Feb–22 Jul 2022.

Marc Finzi, Samuel Stanton, Pavel Izmailov, and Andrew Gordon Wilson. Generalizing Convolutional Neural Networks for Equivariance to Lie Groups on Arbitrary Continuous Data. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3165–3176. PMLR, 13–18 Jul 2020.

Marc Finzi, Max Welling, and Andrew Gordon Wilson. A Practical Method for Constructing Equivariant Multilayer Perceptrons for Arbitrary Matrix Groups. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 3318–3328. PMLR, 18–24 Jul 2021.

Amaury Freslon. On the Partition Approach to Schur-Weyl Duality and Free Quantum Groups. *Transformation Groups*, 22: 707–751, 2017.

Amaury Freslon. *Compact Matrix Quantum Groups and Their Combinatorics*. Cambridge University Press, 2023.

Amaury Freslon and Moritz Weber. On the representation theory of partition (easy) quantum groups. *Journal für die reine und angewandte Mathematik (Crelles Journal)*, 2016(720):155–197, 2016.

F. G. Frobenius. Über Gruppencharaktere. *Sitzungsberichte der Königlich Preußischen Akademie der Wissenschaften zu Berlin*, pages 985–1021, 1896. Gesammelte Abhandlungen, III, 1-37.

F. G. Frobenius. Über die Darstellung der endlichen Gruppen durch lineare Substitutionen. *Sitzungsberichte der Königlich Preußischen Akademie der Wissenschaften zu Berlin*, pages 994–1015, 1897. Gesammelte Abhandlungen, III, 82-103.

F. G. Frobenius. Über die Charaktere der symmetrischen Gruppe. *Sitzungsberichte der Königlich Preußischen Akademie der Wissenschaften zu Berlin*, pages 516–534, 1900. Gesammelte Abhandlungen, III, 148-166.

F. G. Frobenius. Über die charakteristischen Einheiten der symmetrischen Gruppe. *Sitzungsberichte der Königlich Preußischen Akademie der Wissenschaften zu Berlin*, pages 328–358, 1903. Gesammelte Abhandlungen, III, 244-274.

R. Frucht. Herstellung von Graphen mit vorgegebener abstrakter Gruppe. *Compositio Mathematica*, 6:239–250, 1938.

Fabian B. Fuchs, Daniel E. Worrall, Volker Fischer, and Max Welling. SE(3)-transformers: 3D roto-translation equivariant attention networks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, 2020.

Octavian-Eugen Ganea, Xinyuan Huang, Charlotte Bunne, Yatao Bian, Regina Barzilay, Tommi Jaakkola, and Andreas Krause. Independent SE(3)-Equivariant Models for End-to-End Rigid Protein Docking. In *International Conference on Learning Representations*, 2022.

Bruno Gavranović, Paul Lessard, Andrew Joseph Dudzik, Tamara Von Glehn, João Guilherme Madeira Araújo, and Petar Veličković. Position: Categorical Deep Learning is an Algebraic Theory of All Architectures. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 15209–15241. PMLR, 21–27 Jul 2024.

I. M. Gelfand and M. A. Naimark. On the imbedding of normed rings into the ring of operators in Hilbert space. *Matematiceskij sbornik*, 54(2):197–217, 1943.

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural Message Passing for Quantum Chemistry. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1263–1272. PMLR, 06–11 Aug 2017.

Charles Godfrey, Michael G. Rawson, Davis Brown, and Henry Kvinge. Fast computation of permutation equivariant layers with the partition algebra. In *ICLR 2023 Workshop on Physics for Machine Learning*, 2023.

Shiqi Gong, Qi Meng, Jue Zhang, Huilin Qu, Congqiao Li, Sitian Qian, Weitao Du, Zhi-Ming Ma, and Tie-Yan Liu. An Efficient Lorentz Equivariant Graph Neural Network for Jet Tagging. *Journal of High Energy Physics*, 2022(7):1–22, 2022.

Roe Goodman and Nolan R. Wallach. *Symmetry, Representations and Invariants*. Springer, 2009.

Jonathan Gordon, David Lopez-Paz, Marco Baroni, and Diane Bouchacourt. Permutation Equivariant Models for Compositional Generalization in Language. In *International Conference on Learning Representations*, 2020.

Marco Gori, Gabriele Monfardini, and Franco Scarselli. A new model for learning in graph domains. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 2, pages 729–734, 2005.

Samuel Greydanus, Misko Dzamba, and Jason Yosinski. Hamiltonian Neural Networks. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Daniel Gromada. *Compact Matrix Quantum Groups and Their Representation Categories*. PhD thesis, Universität des Saarlandes, 2020.

Daniel Gromada. Group-theoretical graph categories. *Journal of Algebraic Combinatorics*, 55:591–627, 2022.

Cheryl Grood. Brauer Algebras and Centralizer Algebras for SO(2n,C). *Journal of Algebra*, 222(2):678–707, 1999.

Nicholas Guttenberg, Nathaniel Virgo, Olaf Witkowski, Hidetoshi Aoki, and Ryota Kanai. Permutation-equivariant neural networks applied to dynamics prediction, 2016. `arXiv:1612.04530`.

Mustafa Hajij, Ghada Zamzmi, Theodore Papamarkou, Nina Miolane, Aldo Guzmán-Sáenz, Karthikeyan Natesan Ramamurthy, Tolga Birdal, Tamal K. Dey, Soham Mukherjee, Shreyas N. Samaga, Neal Livesay, Robin Walters, Paul Rosen, and Michael T. Schaub. Topological Deep Learning: Going Beyond Graph Data, 2023. `arXiv:2206.00606`.

Tom Halverson. Characters of the Partition Algebras. *Journal of Algebra*, 238:502–533, 2001.

Tom Halverson. Set-Partition Tableaux, Symmetric Group Multiplicities, and Partition Algebra Modules. In *Proceedings of the 31st International Conference on "Formal Power Series and Algebraic Combinatorics", July 1 - 5, 2019, University of Ljubljana, Slovenia*, 2019.

Tom Halverson and Theodore N. Jacobson. Set-Partition Tableaux and Representations of Diagram Algebras. *Algebraic Combinatorics*, 3(2):509–538, 2020.

Tom Halverson and Arun Ram. Partition algebras. *European Journal of Combinatorics*, 26(6):869–921, 2005.

Tom Halverson and Arun Ram. Gems from the Work of Georgia Benkart. *Notices of the American Mathematical Society*, 69(3): 375–384, 2022.

Tom Halverson and Mike Reeks. Gelfand Models for Diagram Algebras. *Journal of Algebraic Combinatorics*, 41(2):229–255, 2015.

William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems*, 2017.

Phil Hanlon and David Wales. On the Decomposition of Brauer's Centralizer Algebras. *Journal of Algebra*, 121(2):409–445, 1989.

Aram W. Harrow. *Applications of Coherent Classical Communication and the Schur Transform to Quantum Information Theory*. PhD thesis, Massachusetts Institute of Technology, 2005.

Jason S. Hartford, Devon R. Graham, Kevin Leyton-Brown, and Siamak Ravanbakhsh. Deep Models of Interactions Across Sets. In *Proceedings of the 35th International Conference on Machine Learning*, pages 1914–1923. PMLR, 2018.

Yuka Hashimoto, Zhao Wang, and Tomoko Matsui. C*-algebra Net: A New Approach Generalizing Neural Network Parameters to C*-algebra. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 8523–8534. PMLR, 17–23 Jul 2022.

Ryuichiro Hataya and Yuka Hashimoto. Noncommutative C^*-algebra Net: Learning Neural Networks with Powerful Product Structure in C^*-algebra, 2023. `arXiv:2302.01191`.

Hamed Hessam, Masoud Khalkhali, Nathan Pagliaroli, and Luuk S. Verhoeven. From noncommutative geometry to random matrix theory. *Journal of Physics A: Mathematical and Theoretical*, 55(41):413002, 2022.

Jordan Hoffmann, Simon Schmitt, Simon Osindero, Karen Simonyan, and Erich Elsen. AlgebraNets, 2020. `arXiv:2006.07360`.

Elyssa Hofgard, Rui Wang, Robin Walters, and Tess Smidt. Relaxed Equivariant Graph Neural Networks, 2024. `arXiv:2407.20471`.

Emiel Hoogeboom, Víctor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant Diffusion for Molecule Generation in 3D. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 8867–8887. PMLR, 17–23 Jul 2022.

Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer Feedforward Networks are Universal Approximators. *Neural Networks*, 2:359–366, 1989.

Mengwei Hu. Presentations of Diagram Categories. *The PUMP Journal of Undergraduate Research*, 3:1–25, 2019.

Ningyuan Huang, Ron Levie, and Soledad Villar. Approximately Equivariant Graph Networks. In *Advances in Neural Information Processing Systems*, volume 36, 2024.

Truong Son Hy, Shubhendu Trivedi, Horace Pan, Brandon M Anderson, and Risi Kondor. Covariant Compositional Networks for Learning Graphs. In *Proceedings of the International Workshop on Mining and Learning with Graphs (MLG)*, 2019.

Gordon James. *The Representation Theory of the Symmetric Group*. Cambridge University Press, 1981.

Michio Jimbo. A q-Difference Analogue of $\mathrm{U}(g)$ and the Yang-Baxter Equation. *Letters in Mathematical Physics*, 10:63–69, July 1985.

V. F. R. Jones. The Potts model and the symmetric group. In *Subfactors: Proceedings of the Taniguchi Symposium on Operator Algebras (Kyuzeso, 1993)*, pages 259–267. World Scientific, 1994.

Stephen P. Jordan. Permutational Quantum Computing. *Quantum Information & Computation*, 10(5):470–497, 2010.

John M. Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A. A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W. Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, 2021.

Sékou-Oumar Kaba, Arnab Kumar Mondal, Yan Zhang, Yoshua Bengio, and Siamak Ravanbakhsh. Equivariance with Learned Canonicalization Functions. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 15546–15566. PMLR, 23–29 Jul 2023.

Michał Kaczor and Paweł Jakubczyk. A new approach to the construction of Schur-Weyl states, 2020. `arXiv:2004.14799`.

Nicolas Keriven and Gabriel Peyré. Universal Invariant and Equivariant Graph Neural Networks. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Piotr Kicki, Mete Ozay, and Piotr Skrzypczynski. A Computationally Efficient Neural Network Invariant to the Action of Symmetry Subgroups, 2020. `arXiv:2002.07528`.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

Joachim Kock. *Frobenius Algebras and 2-D Topological Quantum Field Theories*. London Mathematical Society Student Texts. Cambridge University Press, 2003.

Risi Kondor. *Group Theoretical Methods in Machine Learning*. PhD thesis, Columbia University, 2008.

Risi Kondor. N-body Networks: a Covariant Hierarchical Neural Network Architecture for Learning Atomic Potentials, 2018. `arXiv:1803.01588`.

Risi Kondor and Shubhendu Trivedi. On the Generalization of Equivariance and Convolution in Neural Networks to the Action of Compact Groups. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2747–2755. PMLR, 10–15 Jul 2018.

Risi Kondor, Zhen Lin, and Shubhendu Trivedi. Clebsch–Gordan Nets: a Fully Fourier Space Spherical Convolutional Neural Network. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Ioannis Konstas, Vassilios Stathopoulos, and Joemon M. Jose. On Social Networks and Collaborative Recommendation. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY, USA, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, volume 25, 2012.

Hari Krovi. An efficient high dimensional quantum Schur transform. *Quantum*, 3:122, 2019.

Wataru Kumagai and Akiyoshi Sannai. Universal Approximation Theorem for Equivariant Maps by Group CNNs, 2020. `arXiv:2012.13882`.

Maxime W. Lafarge, Erik J. Bekkers, Josien P. W. Pluim, Remco Duits, and Mitko Veta. Roto-translation equivariant convolutional networks: Application to histopathology image analysis. *Medical Image Analysis*, 68:101849, 2021.

Martín Larocca, Frédéric Sauvage, Faris M. Sbahi, Guillaume Verdon, Patrick J. Coles, and Marco Cerezo. Group-invariant quantum machine learning. *PRX Quantum*, 3:030341, 2022.

Aaron D. Lauda and Joshua Sussan. An Invitation to Categorification. *Notices of the American Mathematical Society*, 69(1): 11–21, 2022.

Yann LeCun, Bernhard Boser, John Denker, Donnie Henderson, Richard Howard, Wayne Hubbard, and Lawrence Jackel. Handwritten digit recognition with a back-propagation network. *Advances in Neural Information Processing Systems*, 2, 1989.

Gustav I. Lehrer and Ruibin B. Zhang. The Brauer category and invariant theory. *Journal of the European Mathematical Society*, 17:2311–2351, 2012.

Gustav I. Lehrer and Ruibin B. Zhang. Invariants of the special orthogonal group and an enhanced Brauer category. *L'Enseignement Mathématique*, 63:181–200, 2018.

Tom Leinster. *Basic Category Theory*. Cambridge University Press, 2014.

Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. Predicting Positive and Negative Links in Online Social Networks. In *Proceedings of the 19th International Conference on World Wide Web*, page 641–650, New York, NY, USA, 2010.

Yi-Lun Liao and Tess Smidt. Equiformer: Equivariant Graph Attention Transformer for 3D Atomistic Graphs. In *The Eleventh International Conference on Learning Representations*, 2023.

Lek-Heng Lim. Tensors in computations. *Acta Numerica*, 30:555–764, 2021.

Lek-Heng Lim and Bradley J. Nelson. What is an equivariant neural network? *Notices of the American Mathematical Society*, 70(4):619–625, 2023.

Huaxin Lin. *An Introduction to the Classification of Amenable C*-Algebras*. World Scientific, 2001.

James D. Louck. *Unitary Symmetry and Combinatorics*. World Scientific, 2008.

L. Lovász. Operations with structures. *Acta Mathematica Academiae Scientiarum Hungarica*, 18:321–328, 1967.

Laura Maaßen. *Representation Categories of Compact Matrix Quantum Groups*. Ph.D. thesis, RWTH Aachen University, Aachen, Germany, 2021.

Saunders Mac Lane. *Categories for the Working Mathematician*. Springer New York, NY, 1998.

Shahn Majid. Non-Commutative Geometry and Quantum Groups. *Philosophical Transactions: Mathematical, Physical and Engineering Sciences*, 358(1765):89–109, January 2000.

Laura Mančinska and David E. Roberson. Quantum isomorphism is equivalent to equality of homomorphism counts from planar graphs. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 661–672, 2020.

Diego Marcos, Michele Volpi, Nikos Komodakis, and Devis Tuia. Rotation Equivariant Vector Field Networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5048–5057, 2017.

Haggai Maron, Heli Ben-Hamu, Nadav Shamir, and Yaron Lipman. Invariant and Equivariant Graph Networks. In *International Conference on Learning Representations*, 2019a.

Haggai Maron, Ethan Fetaya, Nimrod Segol, and Yaron Lipman. On the Universality of Invariant Networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 4363–4371. PMLR, 09–15 Jun 2019b.

Haggai Maron, Or Litany, Gal Chechik, and Ethan Fetaya. On Learning Sets of Symmetric Elements. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 6734–6744. PMLR, 13–18 Jul 2020.

Paul P. Martin. Representations of Graph Temperley–Lieb Algebras. *Publications of the Research Institute for Mathematical Sciences*, 26(3):485–503, 1990.

Paul P. Martin. Temperley–Lieb Algebras for Non–Planar Statistical Mechanics – the Partition Algebra Construction. *Journal of Knot Theory and Its Ramifications*, 03(01):51–82, 1994.

Paul P. Martin. The Structure of the Partition Algebras. *Journal of Algebra*, 183:319–358, 1996.

Benjamin Kurt Miller, Mario Geiger, Tess E. Smidt, and Frank Noé. Relevance of rotationally equivariant convolutions for predicting molecular properties. *arXiv preprint arXiv:2008.08461*, 2020.

Christopher Morris, Gaurav Rattan, Sandra Kiefer, and Siamak Ravanbakhsh. SpeqNets: Sparsity-aware Permutation-equivariant graph networks. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, pages 16017–16042, 17–23 Jul 2022.

James R. Munkres. *Topology*. Pearson, 2018.

Quynh T. Nguyen, Louis Schatzki, Paolo Braccia, Michael Ragone, Patrick J. Coles, Frédéric Sauvage, Martín Larocca, and Marco Cerezo. Theory for Equivariant Quantum Neural Networks. *PRX Quantum*, 5, May 2024.

Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. Learning Convolutional Neural Networks for Graphs. In *International Conference on Machine Learning*, pages 2014–2023. PMLR, 2016.

Horace Pan and Risi Kondor. Permutation Equivariant Layers for Higher Order Interactions. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 5987–6001. PMLR, 28–30 Mar 2022.

Theodore Papamarkou, Tolga Birdal, Michael M. Bronstein, Gunnar E. Carlsson, Justin Curry, Yue Gao, Mustafa Hajij, Roland Kwitt, Pietro Lio, Paolo Di Lorenzo, Vasileios Maroulas, Nina Miolane, Farzana Nasrin, Karthikeyan Natesan Ramamurthy, Bastian Rieck, Simone Scardapane, Michael T. Schaub, Petar Veličković, Bei Wang, Yusu Wang, Guowei Wei, and Ghada Zamzmi. Position: Topological Deep Learning is the New Frontier for Relational Learning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 39529–39555. PMLR, 21–27 Jul 2024.

Mathilde Papillon, Sophia Sanborn, Mustafa Hajij, and Nina Miolane. Architectures of Topological Deep Learning: A Survey of Message-Passing Topological Neural Networks, 2024. `arXiv:2304.10031`.

Edward Pearce-Crump. A Multigraph Approach for Performing the Quantum Schur Transform, 2022. `arXiv:2204.10694`.

Edward Pearce-Crump. Brauer's Group Equivariant Neural Networks. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 27461–27482. PMLR, 23–29 Jul 2023a.

Edward Pearce-Crump. How Jellyfish Characterise Alternating Group Equivariant Neural Networks. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 27483–27495. PMLR, 23–29 Jul 2023b.

Edward Pearce-Crump. Compact Matrix Quantum Group Equivariant Neural Networks, 2023c. `arXiv:2311.06358`.

Edward Pearce-Crump. Connecting Permutation Equivariant Neural Networks and Partition Diagrams. In *Proceedings of the 27th European Conference on Artificial Intelligence*, volume 392, pages 1511–1518. IOS Press Ebooks, 2024.

Edward Pearce-Crump and William J. Knottenbelt. A Diagrammatic Approach to Improve Computational Efficiency in Group Equivariant Neural Networks, 2024a. `arXiv:2412.10837`.

Edward Pearce-Crump and William J. Knottenbelt. Graph Automorphism Group Equivariant Neural Networks. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 40051–40077. PMLR, 21–27 Jul 2024b.

Mircea Petrache and Shubhendu Trivedi. Approximation-Generalization Trade-offs under (Approximate) Group Equivariance. In *Advances in Neural Information Processing Systems*, volume 36, pages 61936–61959, 2023.

Mircea Petrache and Shubhendu Trivedi. Position Paper: Generalized grammar rules and structure-based generalization beyond classical equivariance for lexical tasks and transduction, 2024. `arXiv:2402.01629`.

Allan Pinkus. Approximation theory of the MLP model in neural networks. *Acta Numerica*, 8:143–195, 1999.

Piotr Podlés. Symmetries of quantum spaces. Subgroups and quotient spaces of quantum SU(2) and SO(3) groups. *Communications in Mathematical Physics*, 170:1–20, 1995.

Omri Puny, Derek Lim, Bobak Kiani, Haggai Maron, and Yaron Lipman. Equivariant Polynomials for Graph Neural Networks. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 28191–28222. PMLR, 23–29 Jul 2023.

Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017.

Michael Ragone, Paolo Braccia, Quynh T. Nguyen, Louis Schatzki, Patrick J. Coles, Frédéric Sauvage, Martin Larocca, and Marco Cerezo. Representation Theory for Geometric Quantum Machine Learning, 2022. `arXiv:2210.07980`.

Jad Rahme, Samy Jelassi, Joan Bruna, and S. Matthew Weinberg. A Permutation-Equivariant Neural Network Architecture For Auction Design. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(6):5664–5672, 2021.

Raghunathan Ramakrishnan, Pavlo O. Dral, Matthias Rupp, and O. Anatole von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data*, 1, 2014.

Sven Raum and Moritz Weber. The combinatorics of an algebraic class of easy quantum groups. *Infinite Dimensional Analysis, Quantum Probability and Related Topics*, 17(3), 2014.

Sven Raum and Moritz Weber. Easy quantum groups and quantum subgroups of a semi-direct product quantum group. *Journal of Noncommutative Geometry*, 9:1261–1293, 2015.

Sven Raum and Moritz Weber. The Full Classification of Orthogonal Easy Quantum Groups. *Communications in Mathematical Physics*, 341:751–779, 2016.

Siamak Ravanbakhsh. Universal Equivariant Multilayer Perceptrons. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 7996–8006. PMLR, 2020.

Siamak Ravanbakhsh, Jeff Schneider, and Barnabás Póczos. Equivariance Through Parameter-Sharing. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 2892–2901, 06–11 Aug 2017.

Emily Riehl. *Category Theory in Context*. Dover Publications, 2017.

David W. Romero and Suhas Lohit. Learning Partial Equivariances from Data. In *Advances in Neural Information Processing Systems*, volume 35, pages 36466–36478, 2022.

Mikael Rørdam, Flemming Larsen, and Niels J. Laustsen. *An Introduction to K-Theory for C*-Algebras*, volume 49 of *London Mathematical Society Student Texts*. Cambridge University Press, Cambridge, 2000.

David J. Rowe, M. J. Carvalho, and J. Repka. Dual pairing of symmetry and dynamical groups in physics. *Reviews of Modern Physics*, 84(2):711–757, 2012.

David Ruhe, Johannes Brandstetter, and Patrick Forré. Clifford Group Equivariant Neural Networks. In *Advances in Neural Information Processing Systems*, volume 36, pages 62922–62990, 2023a.

David Ruhe, Jayesh K. Gupta, Steven De Keninck, Max Welling, and Johannes Brandstetter. Geometric Clifford Algebra Networks. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 29306–29337. PMLR, 23–29 Jul 2023b.

Kieran Ryan. *Representation-theoretic approaches to several problems in probability*. PhD thesis, Queen Mary University of London, 2021.

Bruce E. Sagan. *The Symmetric Group: Representations, Combinatorial Algorithms, and Symmetric Functions*. Springer, 2000.

Víctor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E(n) Equivariant Graph Neural Networks. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 9323–9332. PMLR, 18–24 Jul 2021.

Alistair Savage. *String Diagrams and Categorification*, pages 3–36. Springer International Publishing, 2021.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The Graph Neural Network Model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

Louis Schatzki, Martin Larocca, Quynh T. Nguyen, Frédéric Sauvage, and Marco Cerezo. Theoretical guarantees for permutation-equivariant quantum neural networks. *npj Quantum Information*, 10(1):12, 2024.

Issai Schur. *Über Eine Klasse Von Matrizen und Die Sich Einer Gegeben Matrix Zurodenen Lassen*. PhD thesis, Berlin, 1901. Gesammelte Abhandlungen, I, 1-73.

Issai Schur. Neue Begründung der Theorie der Gruppencharaktere. *Sitzungsberichte der Königlich Preußischen Akademie der Wissenschaften zu Berlin*, pages 406–432, 1905. Gesammelte Abhandlungen, I, 143-169.

Issai Schur. Über die rationalen Darstellungen der allgemeinen linearen Gruppe. *Sitzungsberichte der Königlich Preußischen Akademie der Wissenschaften zu Berlin*, pages 58–75, 1927. Gesammelte Abhandlungen, III, 68-85.

Kristof Schütt, Oliver Unke, and Michael Gastegger. Equivariant Message Passing for the Prediction of Tensorial Properties and Molecular Spectra. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 9377–9388. PMLR, 18–24 Jul 2021.

Travis Scrimshaw. Cellular Subalgebras of the Partition Algebra. *Journal of Combinatorial Algebra*, 8(1/2):147–207, 2024.

Ed Segal. Group Representation Theory. Course Notes for Imperial College London, 2014.

Nimrod Segol and Yaron Lipman. On universal equivariant set networks. In *International Conference on Learning Representations*, 2020.

Dan Shiebler, Bruno Gavranović, and Paul Wilson. Category Theory in Machine Learning, 2021. `arXiv:2106.07032`.

Andrzej Sitarz. A Friendly Overview of Noncommutative Geometry. *Acta Physica Polonica. B*, 44(12), 2013.

Roland Speicher and Moritz Weber. ISEM24 C*-Algebras and Dynamics Lecture Notes. Course Notes for Universität des Saarlandes, 2020.

Shlomo Sternberg. *Group Theory and Physics*. Cambridge University Press, 1994.

Julian Suk, Pim de Haan, Phillip Lippe, Christoph Brune, and Jelmer M. Wolterink. Mesh neural networks for SE(3)-equivariant hemodynamics estimation on the artery wall. *Computers in Biology and Medicine*, 173, 2024.

Masamichi Takesaki. *Theory of Operator Algebras. I*, volume 124 of *Encyclopaedia of Mathematical Sciences*. Springer-Verlag, Berlin, 2002.

Pierre Tarrago and Moritz Weber. Unitary Easy Quantum Groups: The Free Case and the Group Case. *International Mathematics Research Notices*, 2017(18):5710–5750, 2016.

Pierre Tarrago and Moritz Weber. The classification of tensor categories of two-colored noncrossing partitions. *Journal of Combinatorial Theory, Series A*, 154:464–506, 2018.

Erik Henning Thiede, Truong Son Hy, and Risi Kondor. The general theory of permutation equivarant neural networks and higher order graph variational encoders, 2020. `arXiv:2004.03990`.

Erik Henning Thiede, Wenda Zhou, and Risi Kondor. Autobahn: Automorphism-based Graph Neural Nets. In *Advances in Neural Information Processing Systems*, volume 34, pages 29922–29934, 2021.

Nathaniel Thomas, Tess Smidt, Steven Kearnes, Lusann Yang, Li Li, Kai Kohlhoff, and Patrick Riley. Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds, 2018. `arXiv:1802.08219`.

Vladimir Turaev and Alexis Virelizier. *Monoidal Categories and Topological Field Theory*. Birkhäuser, 2017.

Elise van der Pol, Daniel Worrall, Herke van Hoof, Frans Oliehoek, and Max Welling. MDP Homomorphic Networks: Group Symmetries in Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 4199–4210, 2020.

Walter D. van Suijlekom. *Noncommutative Geometry and Particle Physics*. Springer Netherlands, 2014.

Petar Velicković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. In *International Conference on Learning Representations*, 2018.

A. M. Vershik and A. Yu. Okounkov. A New Approach to the Representation Theory of the Symmetric Groups II. *Zapiski Seminarod POMI (In Russian) v.307*, 1996.

A. M. Vershik and A. Yu. Okounkov. A New Approach to the Representation Theory of the Symmetric Groups II. *Journal of Mathematical Sciences*, 2005.

Soledad Villar, David W. Hogg, Kate Storey-Fisher, Weichi Yao, and Ben Blum-Smith. Scalars are universal: Equivariant machine learning, structured like classical physics. In *Advances in Neural Information Processing Systems*, 2021.

Dan Voiculescu. Symmetries of some reduced free product C^*-algebras. In *Operator Algebras and Their Connections with Topology and Ergodic Theory (Bușteni, 1983)*, volume 1132 of *Lecture Notes in Mathematics*, pages 556–588. Springer, Berlin, 1985.

Dan Voiculescu, Kenneth J. Dykema, and Alexandru Nica. *Free Random Variables*. American Mathematical Society, 1992.

Michael Walter. Symmetry and Quantum Information. Course Notes for the University of Amsterdam, 2018.

Dian Wang, Mingxi Jia, Xupeng Zhu, Robin Walters, and Robert Platt. On-Robot Learning with Equivariant Models. In *6th Annual Conference on Robot Learning*, 2022a.

Dian Wang, Robin Walters, and Robert Platt. SO(2)-Equivariant Reinforcement Learning. In *International Conference on Learning Representations*, 2022b.

Dian Wang, Robin Walters, Xupeng Zhu, and Robert Platt. Equivariant Q Learning in Spatial Action Spaces. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 1713–1723. PMLR, 08–11 Nov 2022c.

Rui Wang, Robin Walters, and Rose Yu. Approximately Equivariant Networks for Imperfectly Symmetric Dynamics. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 23078–23091. PMLR, 17–23 Jul 2022d.

Shuzhou Wang. Free products of compact quantum groups. *Communications in Mathematical Physics*, 167(3):671–692, 1995.

Shuzhou Wang. Quantum symmetry groups of finite spaces. *Communications in Mathematical Physics*, 195(1):195–211, 1998.

Yuyang Wang, Ahmed A. Elhag, Navdeep Jaitly, Joshua M. Susskind, and Miguel Ángel Bautista. Generating Molecular Conformer Fields, 2023. `arXiv:2311.17932`.

Moritz Weber. On the classification of easy quantum groups. *Advances in Mathematics*, 245:500–533, 2013.

Moritz Weber. Introduction to compact (matrix) quantum groups and Banica-Speicher (easy) quantum groups. *Proceedings Mathematical Sciences*, 127:881–933, 2017.

Moritz Weber. Quantum symmetry. *Snapshots of Modern Mathematics from Oberwolfach*, 5, 2020.

Maurice Weiler and Gabriele Cesa. General E(2)-Equivariant Steerable CNNs. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Maurice Weiler, Mario Geiger, Max Welling, Wouter Boomsma, and Taco Cohen. 3D Steerable CNNs: Learning Rotationally Equivariant Features in Volumetric Data. In *Advances in Neural Information Processing Systems*, volume 31, 2018a.

Maurice Weiler, Fred A. Hamprecht, and Martin Storath. Learning Steerable Filters for Rotation Equivariant CNNs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018b.

Hans Wenzl. On the Structure of Brauer's Centralizer Algebras. *Annals of Mathematics*, 128(1):173–193, 1988.

Hermann Weyl. *The Classical Groups: Their Invariants and Representations*. Princeton University Press, 1939.

Peter Woit. *Quantum Theory, Groups and Representations*. Springer International Publishing, 2017.

Stanisław L. Woronowicz. Compact Matrix Pseudogroups. *Communications in Mathematical Physics*, 111:613–665, 1987.

Stanisław L. Woronowicz. Tannaka-Krein duality for compact matrix pseudogroups. Twisted SU(N) groups. *Inventiones mathematicae*, 93(1):35–76, 1988.

Stanisław L. Woronowicz. A remark on compact matrix quantum groups. *Letters in Mathematical Physics*, 21:35–39, 1991.

Stanisław L. Woronowicz. Compact quantum groups. symétries quantiques, 1998.

Daniel E. Worrall, Stephan J. Garbin, Daniyar Turmukhambetov, and Gabriel J. Brostow. Harmonic Networks: Deep Translation and Rotation Equivariance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How Powerful are Graph Neural Networks? In *International Conference on Learning Representations*, 2019.

Dmitry Yarotsky. Universal approximations of invariant maps by neural networks. *Constructive Approximation*, 55(1):407–474, 2022.

Alfred Young. On Quantitative Substitutional Analysis. *Proceedings of the London Mathematical Society*, s1-33(1):97–145, 1900.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R. Salakhutdinov, and Alexander J. Smola. Deep Sets. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Chao Zhang, Stephan Liwicki, William Smith, and Roberto Cipolla. Orientation-Aware Semantic Segmentation on Icosahedron Spheres. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3532–3540, 2019.

Han Zheng, Zimu Li, Junyu Liu, Sergii Strelchuk, and Risi Kondor. Speeding Up Learning Quantum States Through Group Equivariant Convolutional Quantum Ansätze. *PRX Quantum*, 4:020327, 2023.