Group theoretical methods in machine learning

Risi Kondor

Unofficial, corrected version of Ph.D. thesis
Last updated May 2008.

COLUMBIA UNIVERSITY

2008

Contents

Introduction **6**

Overview **8**

I Algebra **10**

1 Groups and representations **11**
1.1 Groups . . . 11
1.1.1 Taxonomy of groups . . . 12
1.1.2 Elementary concepts . . . 13
1.1.3 Decomposing groups . . . 14
1.1.4 G-modules and group algebras . . . 15
1.1.5 Group actions and homogeneous spaces . . . 16
1.1.6 Lie groups . . . 16
1.2 Representation Theory . . . 18
1.2.1 Character theory and Abelian groups . . . 19
1.2.2 G-modules . . . 20
1.2.3 Restricted and induced representations . . . 20
1.3 The symmetric group . . . 21
1.3.1 Representations . . . 22
1.4 The General Linear Group . . . 27
1.4.1 Representations . . . 27

2 Harmonic analysis **30**
2.1 The classical Fourier transforms . . . 30
2.2 Fourier analysis on groups . . . 32
2.3 Spectral analysis on finite groups . . . 34
2.3.1 Factorial designs . . . 34
2.3.2 Analysis of data on the symmetric group . . . 35

3 Fast Fourier transforms **37**
3.1 Fast Fourier transforms on $\mathbb{Z}_n$ . . . 38
3.2 Non-commutative FFTs . . . 39
3.3 Clausen's FFT for the symmetric group . . . 41
3.3.1 Extensions . . . 46
3.4 The $\mathbb{S}_n$`ob` library . . . 47

II Learning on groups **50**

4 Hilbert space learning algorithms **51**
4.1 The statistical learning framework . . . 51
4.1.1 Empirical risk minimization . . . 52
4.2 The Hilbert space formalism . . . 53
4.2.1 Reproducing kernels . . . 54
4.2.2 The representer theorem . . . 55
4.2.3 Regularization theory . . . 56
4.3 Algorithms . . . 57
4.3.1 Gaussian processes . . . 57
4.3.2 Support vector machines . . . 60
4.4 Symmetries . . . 61
4.5 Kernels on groups and homogeneous spaces . . . 64
4.5.1 Positive definite functions . . . 65

5 Learning permutations **67**
5.1 Regularized Risk Minimization . . . 68
5.2 Diffusion Kernels . . . 69
5.3 Immanants and the PerMceptron . . . 73
5.3.1 The "PerMceptron" . . . 73

6 Multi-object tracking **75**
6.1 A band-limited representation . . . 76
6.1.1 Noise model . . . 78
6.1.2 Observations . . . 78
6.1.3 Inference . . . 79
6.2 Efficient computation . . . 79
6.2.1 Performance . . . 79
6.3 Experiments . . . 80

III Invariance **82**

7 Invariant functionals **83**
7.1 The power spectrum and the bispectrum . . . 84
7.2 The skew spectrum . . . 87
7.3 Generalization to homogeneous spaces . . . 89

8 Invariant features for images **92**
8.1 A projection onto the sphere . . . 92
8.2 Bispectral invariants of functions on S_2 . . . 95
8.2.1 Relation to the bispectrum on SO(3) . . . 96
8.3 Experiments . . . 98

Conclusions **100**

Biblography **102**

List of Symbols **106**

Index **108**

Acknowledgements

I would like express my gratitude to everybody who has, directly or indirectly, had a part in helping this thesis come to be. First and foremost, I would like to thank my advisor, Tony Jebara, who has supported me, helped me, guided me, and gave me advice over countless meetings and brainstorming sessions. Thanks to the entire Columbia machine learning crew, especially Andy Howard, Bert Huang, Darrin Lewis, Rafi Pelossof, Blake Shaw, and Pannaga Shivaswamy. A very special thanks to Rocco Servedio, who was always there when I needed help both in scientific matters and the more mundane side of being a Ph.D. student. I would like to thank my entire committee, comprised of Cliff Stein, Christina Leslie and Zoubin Ghahramani in addition to Tony and Rocco, for the amount of time they put into trying to make sense of my thesis, and the valuable comments they made on it.

Further afield, I would first like to thank John Lafferty and Guy Lebanon from my CMU days before I came to Columbia. They might not even remember now, but it was a couple of discussions with Guy and John that gave me the idea of learning on groups in the first place. I would like to thank Dan Rockmore for the discussions I had with him at Dartmouth and Ramakrishna Kakarala for sending me a hard copy of his thesis without ever having met me.

If it were not for Gábor Csányi, who, in addition to being one of my longest-standing friends, is also one of my scientific collaborators (yes, it's time we finally came out with that molecular dynamics paper, Gábor), I might never have stumbled upon the idea of the bispectrum. Balázs Szendrői, János Csirik and Susan Harrington also provided important mathematical (and emotional) support towards the end of my Ph.D..

Finally, I would like to thank DOMA for the inspiring company and the endless stream of coffee.

Introduction

Ever since its discovery in 1807, the Fourier transform has been one of the mainstays of pure mathematics, theoretical physics, and engineering. The ease with which it connects the analytical and algebraic properties of function spaces; the particle and wave descriptions of matter; and the time and frequency domain descriptions of waves and vibrations make the Fourier transform one of the great unifying concepts of mathematics.

Deeper examination reveals that the logic of the Fourier transform is dictated by the structure of the underlying space itself. Hence, the classical cases of functions on the real line, the unit circle, and the integers modulo n are only the beginning: harmonic analysis can be generalized to functions on any space on which a group of transformations acts. Here the emphasis is on the word *group* in the mathematical sense of an algebraic system obeying specific axioms. The group might even be non-commutative: the fundamental principles behind harmonic analysis are so general that they apply equally to commutative and non-commutative structures. Thus, the humble Fourier transform leads us into the depths of group theory and abstract algebra, arguably the most extensive formal system ever explored by humans.

Should this be of any interest to the practitioner who has his eyes set on concrete applications of machine learning and statistical inference? Hopefully, the present thesis will convince the reader that the answer is an emphatic "yes".

One of the reasons why this is so is that groups are the mathematician's way of capturing symmetries, and symmetries are all around us. Twentieth century physics has taught us just how powerful a tool symmetry principles are for prying open the secrets of nature. One could hardly ask for a better example of the power of mathematics than particle physics, which translates the abstract machinery of group theory into predictions about the behavior of the elementary building blocks of our universe.

I believe that algebra will prove to be just as crucial to the science of data as it has proved to be to the sciences of the physical world. In probability theory and statistics it was Persi Diaconis who did much of the pioneering work in this realm, brilliantly expounded in his little book [Diaconis, 1988]. Since then, several other authors have also contributed to the field. In comparison, the algebraic side of machine learning has until now remained largely unexplored. The present thesis is a first step towards filling this gap.

The two main themes of the thesis are (a) learning on domains which have non-trivial algebraic structure; and (b) learning in the presence of invariances. Learning rankings/matchings are the classic example of the first situation, whilst rotation/translation/scale invariance in machine vision is probably the most immediate example of the latter. The thesis presents examples addressing real world problems in these two domains. However, the beauty of the algebraic approach is that it allows us to discuss these matters on a more general, abstract, level, so most of our results apply

equally well to a large range of learning scenarios.

The generality of our approach also means that we do not have to commit to just one learning paradigm (frequentist/Bayesian) or one group of algorithms (SVMs/graphical models/boosting/etc.). We do find that some of our ideas regarding symmetrization and learning on groups meshes best with the Hilbert space learning framework, so in Chapters 4 and 5 we focus on this methodology, but even there we take a comparative stance, contrasting the SVM with Gaussian Processes and a modified version of the Perceptron.

One of the reasons why up until now abstract algebra has not had a larger impact on the applied side of computer science is that it is often perceived as a very theoretical field, where computations are difficult if not impossible due to the sheer size of the objects at hand. For example, while permutations obviously enter many applied problems, calulations on the full symmetric group (permutation group) are seldom viable, since it has $n!$ elements. However, recently abstract algebra has developed a strong computational side [Bürgisser et al., 1997]. The core algorithms of this new computational algebra, such as the non-commutative FFTs discussed in detail in Chapter 3, are the backbone of the bridge between applied computations and abstract theory. In addition to our machine learning work, the present thesis offers some modest additions to this field by deriving some useful generalizations of Clausen's FFT for the symmetric group, and presenting an efficient, expandable software library implementing the transform. To the best of our knowledge, this is the first time that such a library has been made publicly available.

Clearly, a thesis like this one is only a first step towards building a bridge between the theory of groups/representations and machine learning. My hope is that it will offer ideas and inspiration to both sides, as well as a few practical algorithms that I believe are directly applicable to real world problems.

Risi Kondor
London
May 2008

Overview

This thesis aims to be as self-contained as possible and accessible to both the machine learning and the applied harmonic analysis communities. Part I is dedicated to giving a broad overview of the algebra side: we introduce some of the fundamental concepts of group theory and representation theory (Chapter 1), explain why Fourier transforms are important (Chapter 2), and give a brief introduction to the machinery of fast Fourier transforms (Chapter 3). Our intent in these chapters is two-fold: to impress upon the reader the generality and power of modern abstract algebra (as well as its aesthetic appeal); and to lay down some of the technical terms and notations that we use in the rest of the thesis.

While group theory has amassed a huge literature over the past more than one hundred years, it is not necessarily easy to find references which describe all the concepts needed for computational work but do not delve deep into abstract theory. We aim to address this gap between the pure and applied literature by presenting a wide-ranging overview, pulling together threads from many different sources, but steering clear of special cases and technical proofs. There are relatively few original contributions of our own described in this part of the thesis, the notable exceptions being the twisted and sparse Fourier transforms of Section 3.3.1, and the $\mathbb{S}_n$`ob` software library for Fourier analysis on the symmetric group (Section 3.4).

As a converse to Part I, Part II begins with an introduction to the supervised learning paradigm of machine learning and the field of Hilbert space learning algorithms (Chapter 4). We quickly progress to discussing symmetries, which leads to the first connection with group theory. As far as we are aware, the general symmetrization result of Theorem 4.4.3 has never been explicitly stated in the literature before. More generally, in Section 4.5 we discuss how to construct positive definite kernels on group structured domains. This work is original, but it has not been published yet.

Chapter 5 presents a direct application of the above ideas to learning permutations. We develop three different diffusion kernels on the symmetric group and discuss their interpretation in terms of the regularization of functions on permutations. Permutation learning demands a special symmetrization step, and from a computational point of view this is a major bottleneck. The rest of the chapter is taken up by describing two potential ways of getting around this: Bürgisser's result on immanants, our new PerMceptron algorithm. The material of this chapter is original and has not been published elsewhere.

Chapter 6 also involves permutations, but applied to a completely different domain, namely the identity management problem in multi-object tracking. While there are no kernels here, and the setting is online filtering as opposed to batch learning, we find that harmonic analysis on the symmetric group is again the key ingredient to developing a principled approach. This chapter is based on [Kondor et al., 2007], which was the first paper to propose using the low-order Fourier components of the matching distribution for identity management.

Whereas Part II explored problems where groups are the space on which learning takes place, in Part III we discuss invariances, where the group action is what we want to eliminate from the learning process. Chapter 7 introduces the powerful machinery of generalized spectral invariants, including the non-commutative power spectrum and the bispectrum. The foundations of this field were laid down by Kakarala [1992], but we go beyond his work by discussing in detail its generalization to homogeneous spaces (Section 7.3) and introducing the skew spectrum (Section 7.2). This work was published in preprint form in [Kondor, 2007a]. We end the thesis with a chapter describing a specific application of the bispectrum formalism to constructing translation and rotation invariant features for computer vision. The corresponding preprint is [Kondor, 2007b].

Part I

Algebra

Chapter 1

Groups and representations

Group theory is a well developed branch of mathematics with an enormous literature, and it is hardly possible to give the entire field justice in a single introductory chapter. Rather, our goal will be to introduce the main players and set the scene for the rest of the thesis.

For background material the reader can turn to any one of many introductory texts on group theory. As a general reference to abstract algebra we recommend [Lang, 1984], while as a more applied text that still covers almost all the concepts we need, we found [Barut and Rączka, 1977] to be an excellent source. We are also indebted to some classics such as [Weyl, 1946],[Boerner, 1970] and [Fulton and Harris, 1991]. One of the most accessible introductions to representation theory is [Serre, 1977], while the particulars of the representation theory of the symmetric group are described in [Sagan, 2001], [Kerber, 1971], [James, 1978] and [James and Kerber, 1981].

1.1 Groups

The concept of groups is one of the cornerstones of modern mathematics. Technically, a **group** G is a set endowed with an operation $G \times G \to G$ satisfying certain axioms. In line with the majority of the literature we use multiplicative notation for the group operation. Hence, applying the operation to group elements x and y, we denote the resulting group element xy or $x \cdot y$ and refer to xy as the **product** of x and y. The axioms that the group operation must satisfy are the following:

1. For any $x, y \in G$, xy is also an element of G (closure).
2. For any $x, y, z \in G$, $(xy)z = x(yz)$ (associativity).
3. There is a unique element of G, denoted e, and called the **identity element**, for which $ex = xe = x$ for any $x \in G$.
4. For any $x \in G$ there is a corresponding element $x^{-1} \in G$, called the **inverse** of x, which satisfies $xx^{-1} = x^{-1}x = e$.

While seemingly trivial, these axioms give rise to an astonishingly rich theory that has been occupying mathematicians for over a century.

Two groups G and G' are said to be **isomorphic**, denoted $G \cong G'$, if there is a one-to-one mapping $\phi\colon G \to G'$ such that $\phi(x)\,\phi(y) = \phi(xy)$ for all $x, y \in G$. Since the emphasis in group

theory is on structure rather than the identities of the actual group elements, isomorphic groups are often regarded and referred to as the same group.

1.1.1 Taxonomy of groups

Group theory splits into broad areas depending on whether G is finite or infinite, discrete or continuous. The simplest to study are **finite groups**, which only have a finite number of elements. The cardinality of G is called the **order** of G and is denoted $|G|$.

Finite groups can be specified explicitly, by listing their elements and defining the group operation in a multiplication table. For example, Felix Klein's **Viergruppe**, denoted V, has four elements $\{e, a, b, c\}$, which multiply according to

	e	a	b	c
e	e	a	b	c
a	a	e	c	b
b	b	c	e	a
c	c	b	a	e

.

There are several regular series of finite groups. For any $n \geq 1$, the integers $0, 1, 2, \ldots, n-1$ form a group under the operation of addition modulo n. This is called the **cyclic group** of order n, and is denoted $\mathbb{Z}_n$ (sometimes the alternative notations $\mathbb{Z}/n\mathbb{Z}$ or C_n are used). Another important series is formed by the **symmetric groups** $\mathbb{S}_n$, which will be the topic of Section 1.3.

The study of infinite but **countable groups** is a natural extension of the theory of finite groups. Some of the most elementary collections of objects in mathematics are countable groups. The integers, $\mathbb{Z}$, for example, form a group under the usual arithmetic operation of addition. In this particular case using multiplicative notation for the group operation is confusing, so it is customary to change to denoting the group operation by $+$ and denoting the inverse of x by $-x$. Another familiar example of a countable group is the set of non-zero rationals $\mathbb{Q} \setminus \{0\}$ under multiplication.

Group theory also extends to uncountable groups, in which case, in addition to the group structure, one generally also endows G with a topology (formally, a topology is a system of subsets of G, called open sets, that obey certain axioms). A **topological group** is a group G with a topology such that $f(x) = x^{-1}$ is a continuous map $f\colon G \to G$; and $g(x, y) = xy$ is a continous map $g\colon G \times G \to G$. It makes a significant difference in developing the theory if we can appeal to some form of compactness. Recall that a topological space X is **compact** if any collection of open sets that collectively cover X has a finite subcollection that also covers X. A **compact group** is a topological group that is compact. A **locally compact** group is a topological group in which any group element has a compact neighborhood.

By far the most intensely studied topological groups are the Lie groups. The idea here is to introduce a concept of differentiation on the group. A topological group G is a **Lie group** if G is also a differentiable manifold and if the maps $f(x) = x^{-1}$ and $g(x, y) = xy$ are differentiable $G \to G$ and $G \times G \to G$ maps, respectively.

As the most fundamental class of Lie groups, consider any vector space V, and the set of invertible linear transformations on V. Under composition of maps this set forms a Lie group called the **general linear group** of V, denoted $\mathrm{GL}(V)$. When V is of finite dimension n, elements of $\mathrm{GL}(V)$ can be thought of as $n \times n$ matrices. Some of the most important Lie groups are groups of matrices. Such so-called **linear groups** are always subgroups of the corresponding general linear

group. The simplest non-trivial example is SO(3), the group of 3×3 orthogonal matrices with determinant $+1$. Since SO(3) is the group of rotations of $\mathbb{R}^3$, it is no surprise that it turns out to be crucially important in many different contexts in the physical sciences. In general, the group of $n \times n$ orthogonal matrices of unit determinant is denoted SO(n).

A rather prosaic series of Lie groups are the Euclidean spaces $\mathbb{R}^n$ under addition of vectors. The identity is the 0 vector, and the group inherits the familiar differential and topological structures of $\mathbb{R}^n$. As a group, $\mathbb{R}^n$ is commutative but not compact. The simplest example of a compact Lie group is the **circle group** $\mathbb{T}$ of complex numbers of unit modulus, forming a group under multiplication. In fact, provided we remove the origin, the complex plane itself forms a group under multiplication. This group we denote $\mathbb{C}^*$. Note that $\mathbb{C}^*$ is the same as GL($\mathbb{C}$).

A crucial feature of the groups axioms, perhaps at first sight obscured by the multiplicative notation, is that they do not require the group operation to be **commutative**, i.e., in general, $xy \neq yx$. The distinction between commutative and non-commutative groups is a crucial dividing line in group theory. Commutative groups are also called **Abelian groups** and their theory is much simpler than that of their non-Abelian relatives. This is especially true of **locally compact Abelian** (LCA) groups.

While a small finite group, such as $\mathbb{S}_5$, and a Lie group, such as SO(3), are very different objects, one of the revelations of group theory is that many important concepts transfer seamlessly from one domain to the other, or at least have close analogs. In what is to follow, as much as possible, we attempt to give a unified treatment of all the different types of groups described above.

1.1.2 Elementary concepts

Two group elements x and y are said to be **conjugate** if there is a $t \in G$ such that $t^{-1}xt = y$. Conjugacy is an equivalence relation, partitioning G into **conjugacy classes**. Functions on G (i.e., functions $f\colon G \to S$ for some S) that are constant on conjugacy classes are called **class functions**.

If a subset H of G forms a group by itself under the group operation of G (i.e., it satisfies closure and the existence of inverses), then H is said to be a **subgroup** of G, and this is denoted $H \leq G$. Trivially, any group is a subgroup of itself, and the single element group $\{e\}$ is a subgroup of any group. If $H \leq G$ is not one of these trivial cases we say that it is a **proper subgroup** and use the notation $H < G$. For any $t \in G$, the elements $\{\, tht^{-1} \mid h \in H \,\}$ form a subgroup isomorphic to H, called the **conjugate subgroup** and denoted H^t. If $H^t = H$ for any t, then H is said to be a **normal subgroup** of G. This relation is denoted $H \lhd G$. All subgroups of Abelian groups are normal.

A mapping $\phi\colon G \to H$ between two groups that preserves the group structure in the sense that $\phi(x)\,\phi(y) = \phi(xy)$ for any $x, y \in G$, is called a **homomorphism**. A homomorphism that is one-to-one is an isomorphism. It is easy to show that the kernel of a homomorphism (i.e., $\{\, x \in G \mid \phi(x) = e \,\}$) is a normal subgroup of G.

For $H < G$ and $x \in G$ we call the set $xH = \{\, xh \mid h \in H \,\}$ the **left coset** of x and the set $Hx = \{\, hx \mid h \in H \,\}$ the **right coset** of x. The cardinality of any coset is the same as that of H. Any two left cosets of G are either disjoint or the same, hence the set of left cosets provides a partition of G, called the **quotient space** of left cosets and denoted G/H. The quotient space of right cosets is denoted $H\backslash G$. An easy corollary known as Lagrange's theorem is that the order of any subgroup of a finite group must divide the order of the group itself. A set A is called a **transversal** or a set of **coset representatives** with respect to the collection of left cosets of H

if for any two distinct $x, y \in A$, $xH \neq yH$ and $\bigcup_{x \in A} xH = G$. Right transversals are defined analogously.

We can also talk about **two-sided cosets**, which are sets of the form $x_1 H x_2 = \{\, x_1 h x_2 \mid h \in H \,\}$ (with $x_1, x_2 \in G$), and if H_1 and H_2 are both subgroups of G, we can talk about **double cosets** $H_1 x H_2 = \{\, h_1 x h_2 \mid h_1 \in H_1,\ h_2 \in H_2 \,\}$ (with $x \in G$). The latter form the double quotient space $H_1 \backslash G / H_2$.

If H is a normal subgroup, then $xH = Hx$ for any $x \in G$, hence the systems of left and right cosets are the same. In fact, in this case G/H forms a group under the operation $(xH)\,(yH) = (xy)H$. To see that this operation is well defined, consider any x' in the same coset as x and any y' in the same coset as y. In this case $x' = h_1 x$ for some $h_1 \in H$ and $y' = h_2 y$ for some $h_2 \in H$. By normality $y^{-1}x^{-1}x'y' = y^{-1}x^{-1}h_1 x h_2 y = y^{-1}h_3 h_2 y$ for some $h_3 \in H$. Multiplying on the left by xy and again applying normality, $x'y' = xyh_4$ for some $h_4 \in H$ and hence $(x'y')\,H = (xy)\,H$. In other words, no matter what x' and y' we choose to represent the cosets xH and yH, the product coset is always the same.

If A is a set of elements of G, the subgroup **generated** by A is the smallest subgroup of G that contains A. We say that A is a set of **generators** of G if A generates the entire group G.

An isomorphism mapping a group to itself is called an **automorphism**. The set of automorphisms of G forms a group under composition, denoted $\mathrm{Aut}(G)$. We can also talk about the automorphism group of other mathematical objects. For example, the automorphism group of a graph is the group of permutations of its vertices that leave the graph topology (which vertices are connected to which others) invariant. A similar concept is that of **isometry groups**, which is the group of transformations of a metric space which leave the metric invariant.

1.1.3 Decomposing groups

Science is preoccuppied with decomposing complicated objects into simpler ones. In group theory there is more than one way of assembling smaller groups into larger groups.

Given two groups G and H, their **direct product**, denoted $G \times H$, is the group of pairs (g, h), $g \in G,\ h \in H$ with multiplication defined $(g_1, h_1)\,(g_2, h_2) = (g_1 g_2, h_1 h_2)$. The identity of $G \times H$ is (e_G, e_H). A trivial example is $\mathbb{R}^n$, which, as a group, is the direct product of n copies of $\mathbb{R}$. Similarly, the n-dimensional unit torus $\mathbb{T}^n$ is a direct product of n copies of the circle group.

Now let G be a group and let H be a subgroup of $\mathrm{Aut}(G)$. The **semi-direct product** $G \rtimes H$ is the group of all ordered pairs (x, Λ) $(x \in G,\ \Lambda \in H)$ with group multiplication defined

$$\left(x', \Lambda'\right)(x, \Lambda) = \left(x'\Lambda'(x), \Lambda'\Lambda\right).$$

The unit element of $G \rtimes H$ is (e_G, e_H) and the inverse of (x, Λ) is $\left(\Lambda^{-1}(x^{-1}), \Lambda^{-1}\right)$. As an example, take the isometry group $\mathrm{ISO}^+(3)$ of $\mathbb{R}^3$ (excluding reflections), composed of transformations $\boldsymbol{x} \mapsto R\boldsymbol{x} + b$, where R is a rotation matrix and b is a translation vector. This group and its generalizations to n dimensions are also called the **rigid body motions** groups. The effect of (R, b) followed by (R', b') is $\boldsymbol{x} \mapsto R'(R\boldsymbol{x} + b) + b'$, giving the composition rule $(R', b')\,(R, b) = (b' + Rb, R'R)$. The rotation matrices form $SO(3)$, the translation vectors form $\mathbb{R}^3$, and the former is the automorphism group of the latter. Hence $\mathrm{ISO}^+(3) \cong \mathbb{R}^3 \rtimes SO(3)$. Other, less trivial, examples of semi-direct products are considerably more interesting.

1.1.4 G-modules and group algebras

Since much of what is to following will pertain to functions on groups, we introduce some of the corresponding structures early in the discussion. By a function on a group G we simply mean a function $f: G \to S$ mapping each group element to a member of some set S.

Most of the time S will be a field $\mathbb{F}$. Recall that a **field** is a set with two operations $+: \mathbb{F} \times \mathbb{F} \to \mathbb{F}$ and $\cdot: \mathbb{F} \times \mathbb{F} \to \mathbb{F}$, and two special elements $0_{\mathbb{F}}$ and $1_{\mathbb{F}}$, called the additive and multiplicative identity, respectively. A field has to obey the following axioms:

$$\begin{array}{l} \mathbb{F} \text{ is a commutative group with respect to } + \text{ with identity } 0_{\mathbb{F}}; \\ \mathbb{F} \setminus 0_{\mathbb{F}} \text{ is a commutative group with respect to } \cdot \text{ with identity } 1_{\mathbb{F}}; \\ \alpha\,(\beta+\gamma) = \alpha\beta + \alpha\gamma \quad \text{for all} \quad \alpha, \beta, \gamma \in \mathbb{F}. \end{array}$$

The elementary examples of fields are the rationals $\mathbb{Q}$, the reals $\mathbb{R}$, and the complex numbers $\mathbb{C}$. Typically in machine learning S will be the set of real numbers $\mathbb{R}$, or the set of complex numbers $\mathbb{C}$. We denote the set of all functions $f: G \to \mathbb{C}$ by $L(G)$.

A **vector space** V over a field $\mathbb{F}$ is a set endowed with the operations of addition and multiplication by elements of $\mathbb{F}$. A vector space must contain a zero vector 0_V, and must satisfy

$$\begin{array}{ll} x+y \in V; & 0_{\mathbb{F}} x = 0_V; \\ x+y = y+x; & 1_{\mathbb{F}} x = x; \\ 0_V + x = x + 0_V = x; & \alpha(x+y) = \alpha x + \alpha y; \\ \alpha x \in V; & (\alpha+\beta)\, x = \alpha x + \beta x; \end{array}$$

for any $v, w \in V$ and $\alpha, \beta \in \mathbb{F}$. If G is a finite group and V is a vector space over $\mathbb{C}$ of dimension $|\,G\,|$, taking any basis of V and labeling the basis vectors with the group elements $\{e_x\}_{x \in G}$, we identify $L(G)$ with V by mapping each $f \in L(G)$ to the vector $\sum_{x \in G} f(x)\, e_x$.

Another way to relate groups and vector spaces is via G-modules. A **G-module** of a group G is a vector space V over a field $\mathbb{F}$ which also also boasts an operation $\mathcal{G} \times V \to V$ satisfying

$$\begin{array}{l} xv \in V; \\ x(\alpha v + \beta w) = \alpha(xv) + \beta(xw); \\ (xy)v = x(yv); \\ ev = v; \end{array}$$

for all $x, y \in G$, $v, w \in V$ and $\alpha, \beta \in \mathbb{F}$. According to these axioms, each $x \in G$ effects on V a linear map $\phi(x): V \to V$ and these maps compose according to $\phi(x)\,\phi(y) = \phi(xy)$. Hence, a G-module is a vector space V which admits a homomorphism $\phi: G \to GL(V)$. This viewpoint will be important in our discussion of group representations.

A different way to introduce multiplication into vector spaces is provided by algebras. An **algebra** is a vector space V over a field $\mathbb{F}$ endowed with an operation $V \times V \to V$ denoted multiplicatively and satisfying

$$\begin{array}{l} (u+v)w = uw + vw; \\ u(v+w) = uv + uw; \\ (\alpha u)v = u(\alpha v) = \alpha(uv); \end{array}$$

for all $u, v, w \in V$ and $\alpha \in \mathbb{F}$. We are particularly interested in the **group algebra** of G, denoted $\mathbb{F}[G]$, which has a basis $\{e_x\}_{x \in G}$ obeying $e_x e_y = e_{xy}$. Clearly, the group algebra is also a G-module with respect to the multiplication operation defined $xv = e_x\, v$, for $x \in G$ and $v \in \mathbb{F}[G]$. A subspace

U of on algebra V is called a **subalgebra** of V if it is closed with respect to multiplication. V is called an **irreducible algebra** if its only subalgebras are $\emptyset$ and itself. Whenever we need to refer to an inner product on $\mathbb{F}[G]$, we will assume that it is an inner product with respect to which the basis $\{e_x\}_{x\in G}$ is orthonormal.

One of the recurring ideas in this thesis is that if we have a function $f\colon G \to S$, we can talk about **translating** it by $t \in G$, which will yield $f^t\colon G \to S$ given by $f^t(x) = f(t^{-1}x)$. If G is non-Abelian, we must distinguish this so-called **left-translate** from the **right-translate** $f^{(t)}(x) = f(xt^{-1})$. By analogy, if μ is a measure on G, its left-translate is $\mu^t(X) = \mu(\{\, x \in G \mid tx \in X \,\})$. Measures which are invariant to left- (resp. right-) translation by any $t \in G$ are called **Haar measures**. Remarkably, any compact group admits a Haar measure which is both left- and right-invariant and unique up to scaling. The scaling is usually set so that $\int_G \mu(x) = 1$.

1.1.5 Group actions and homogeneous spaces

Many of the most important groups arise in the guise of transformations of an underlying object S. Given a set of bijective mappings $\mathcal{T} = \{T_i\colon S \to S\}_i$, we say that $\mathcal{T}$ is a closed set of transformations if for any $T_1, T_2 \in \mathcal{T}$, their composition $T_2 \circ T_1$ is an element of $\mathcal{T}$, and if for any $T \in \mathcal{T}$, the inverse map T^{-1} is an element of $\mathcal{T}$. These operations turn $\mathcal{T}$ into a group.

Formally, we say that a group G **acts** on a set S if to every group element x, we can associate a function $T_x\colon S \to S$ in such a way that $T_e(s) = s$ for all $s \in S$ and $T_{xy} = T_x(T_y(s))$ for all $x, y \in S$ and $s \in S$. When there is no danger of confusion, we use the same symbols to denote group elements and the corresponding action, hence $T_x(s)$ becomes $x(s)$ or simply xs. Note that according to our definition group actions compose from the right, i.e., $(xy)(s) = x(y(s))$. Given $s \in S$, the subset of S that we get by applying every element of G to s, $\{x(s)\}_{s\in G}$ we call the **orbit** of s. The orbits partition S into disjoint subsets.

If there is only one orbit, we say that G acts **transitively** on S. In this case, fixing any $s_0 \in S$, the map $\phi\colon G \to S$ given by $x \mapsto xs_0$ is surjective, (i.e., sweeps out the entire set S) and we say that S is a **homogeneous space** of G. It is easy to see that the group elements fixing s_0 form a subgroup of G. We call this subgroup $H = \{\, h \in G \mid hs_0 = s_0 \,\}$ the **isotropy subgroup** of G. Since $xhs_0 = xs_0$ for any $h \in H$, there is a one-to-one correspondence between S and the left cosets G/H. Left quotient spaces and homogeneous spaces are just two aspects of the same concept.

As a simple example of a homogeneous space, consider the action of the three dimensional rotation group SO(3) on the unit sphere S_2. It is clear that taking any point on S_2, for example the unit vector e_z pointing along the z axis, $\{\, R\, e_z \mid R \in \mathrm{SO}(3) \,\}$ sweeps out the entire sphere, so S_2 is a homogeneous space of SO(3). The isotropy group in this case is the subgroup of rotations about the z axis, which is just SO(2). From an algebraic point of view $S_2 \cong \mathrm{SO}(3)/\mathrm{SO}(2)$, and in fact in general, $S_{n-1} \cong \mathrm{SO}(n)/\mathrm{SO}(n-1)$ for any natural number n.

1.1.6 Lie groups

The differentiable manifold nature of Lie groups imparts on them additional structure. To unravel this, first recall some general concepts pertaining to any n-dimensional differentiable manifold $\mathcal{M}$.

A **chart** at $x \in \mathcal{M}$ is a homeomorphism ϕ from an open neighborhood U of x to an open subset of $\mathbb{R}^n$. Every point of a differentiable manifold is covered by at least one chart. The i'th component of ϕ is a function $[\phi(x)]_i : U \to \mathbb{R}$, which we shall simply denote ϕ_i. We say that a function $f\colon \mathcal{M} \to \mathbb{R}$ is differentiable at x if $f \circ \phi^{-1}$ is differentiable at $\phi(x)$, and use the notation

$\frac{\partial f}{\partial x_i}$ for $\frac{\partial}{\partial x_i}(f \circ \phi^{-1})$. Let $A(x)$ be the space of functions differentiable at x. A **tangent vector** at x is a linear mapping $T\colon A(x) \to \mathbb{R}$ satisfying $T(fg) = T(f)g(x) + f(x)T(g)$ for all $f, g \in A(x)$. This condition is sufficient to establish a close link with differentiation. Specifically, T is a tangent vector if and only if

$$T(f) = \sum_{i=1}^{n} \frac{\partial f}{\partial x_i} T(\phi_i).$$

The collection of tangent vectors at x form an n-dimensional vector space $\mathcal{T}_x$, called the **tangent space**. The natural interpretation of the vectors $T \in \mathcal{T}_x$ is that they specify directions along the manifold at the point x. To make the connection to differentiation more explicit, we will also write $\nabla_T f$ for $T(f)$. A mapping V that assigns to each point $x \in \mathcal{M}$ a tangent vector $V(x) \in \mathcal{T}_x$ we call a **vector field** on $\mathcal{M}$. We can also regard V as a linear operator on differentiable functions on $\mathcal{M}$, in which context we write Vf for the function $(Vf)(x) = \nabla_{V(x)} f(x)$.

The connection between the differential and algebraic structures is provided by invariant vector fields. Let G act on functions on G by $f \mapsto f^y$ such that $f^y(x) = f(y^{-1}x)$ for all $x \in G$. The vector field V is said to be left-invariant on G if $(Vf)^y = V(f^y)$ for any differentiable f. Any $T \in \mathcal{T}_e$ determines a unique left-invariant vector field V_T satisfying $V_T(e) = T$. Now it is possible to show that T also determines a unique curve $\gamma\colon \mathbb{R} \to G$ with the property that $\frac{d}{dt}(f \circ \gamma)(t) = (V_T f)(\gamma(t))$ for all differentiable functions on $\mathcal{M}$. Moreover, the image of this curve will be a one-parameter subgroup of G, and its elements satisfy $\gamma(s+t) = \gamma(s)\gamma(t)$. This justifies the notation e^{tT} for $\gamma(t)$.

Using this notation, given a basis $T_1, T_2, \ldots, T_n$ of $\mathcal{T}_e$, any element of G in a neighborhood of the identity can be written in the form

$$x(\alpha_1, \alpha_2, \ldots, \alpha_n) = \exp\left(\sum_{i=1}^{n} \alpha_i T_i\right) \tag{1.1}$$

for some $\alpha_1, \alpha_2, \ldots, \alpha_n \in \mathbb{R}$. The T_i's are called the infinitesimal generators of the Lie group G. In general for $A, B \in \mathcal{T}_e$ it is not true that $e^A e^B = e^{A+B}$. We can, however, express the product of two group elements given in exponential form by the **Baker-Campbell-Hausdorff formula**

$$\exp A \cdot \exp B = \exp\left(A + B + \frac{1}{2}[A,B] + \frac{1}{12}[A,[A,B]] - \frac{1}{12}[B,[A,B]] - \ldots\right), \tag{1.2}$$

where the **Lie bracket** $[A, B]$ is defined by $[A, B](f) = A(B(f)) - B(A(f))$ and it can be shown that $[A, B] \in \mathcal{T}_e$. The Lie bracket operation turns $\mathcal{T}_e$ into a **Lie algebra** $\mathcal{L}$, which is characteristic of G, and equations (1.1) and (1.2) tell us that at least locally, $\mathcal{L}$ completely determines the structure of G.

When Lie groups arise as groups of matrices (such Lie groups are called linear groups), the generators $T_1, T_2, \ldots, T_n$ are themselves matrices, the exponentiation in (1.1) is just matrix exponentiation, and $[A, B]$ is equal to the **commutator** $AB - BA$. For example, SO(3) is generated by

$$T_x = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}, \qquad T_y = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix}, \qquad \text{and} \qquad T_z = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix},$$

the Lie algebra has a basis $\{I, T_x, T_y, T_z\}$, and is determined by the single relation $[T_a, T_b] = \epsilon_{abc} T_c$, where ϵ_{abc} is $+1$ if (a, b, c) is a cyclic shift of (x, y, z); $\epsilon_{abc} = -1$ if it is a cyclic shift of the reverse permutation (z, y, x) and $\epsilon_{abc} = 0$ in all other cases.

1.2 Representation Theory

Groups are quite abstract mathematical objects. One way to make them more concrete is to model their elements by matrices. This is the fundamental idea behind representation theory. A **matrix representation** ρ of a compact group G over a field $\mathbb{F}$ associates to each element x of G a matrix $\rho(x) \in \mathbb{F}^{d_\rho \times d_\rho}$ in such a way that

$$\rho(xy) = \rho(x)\,\rho(y)$$

for all $x, y \in G$. We also require $\rho(e) = I$. We say that d_ρ is the **order** of the representation. Note that $\rho(x^{-1}) = (\rho(x))^{-1}$. In this thesis we restrict ourselves to the canonical choice of ground field, which is the field of complex numbers, $\mathbb{C}$.

It is possible to extend the concept of representations to the infinite dimensional realm by promoting the $\rho(x)$ to linear operators (see section 1.2.2). While this is important, especially for non-compact groups, in the present work we focus on finite dimensional matrix representations. In the following, unless specified otherwise, by "representation" we will mean a "finite dimensional matrix representation over $\mathbb{C}$".

Two representations ρ_1 and ρ_2 are said to be **equivalent** if there is an invertible square matrix T such that

$$\rho_1(x) = T^{-1}\rho_2(x)\,T \qquad \forall x \in G.$$

For our purposes equivalent representations are regarded as the same. We say that a representation ρ is **reducible** if each matrix $\rho(x)$ has the block structure

$$\rho(x) = \left(\begin{array}{c|c} A(x) & B(x) \\ \hline 0 & C(x) \end{array}\right).$$

In fact, whenever this happens there is always a similarity transformation $\rho \mapsto T^{-1}\rho\,T$ that reduces ρ to a direct sum

$$T^{-1}\rho(x)\,T = \left(\begin{array}{c|c} \rho_1(x) & 0 \\ \hline 0 & \rho_2(x) \end{array}\right) \qquad \forall\, x \in G$$

of smaller representations ρ_1 and ρ_2. We say that ρ is **irreducible** if there is no invertible matrix T that can simultaneously block diagonalize all the $\rho(x)$ matrices in this way. The real significance of irreducible representations is due to the following theorem, variously known as the theorem of complete reducibility, Wedderburn's theorem or Maschke's theorem [Serre, 1977, p.7][Fulton and Harris, 1991, p.7][Barut and Rączka, 1977, p.169].

Theorem 1.2.1 *Any representation ρ of a compact group G may be reduced by appropriate choice of invertible transformation T into a direct sum*

$$\rho(x) = T^{-1}\Big[\bigoplus_i \rho_i(x)\Big]\,T, \qquad x \in G$$

of irreducible representations $\rho_1, \rho_2, \ldots, \rho_N$ of G. Moreover, this decomposition is unique, up to equivalence of representations and changing the order of the terms in the direct sum. Note that the different ρ_i's are not necessarily distinct.

Because of this theorem, given a specific compact group G, there is a lot of interest in finding a complete set $\mathcal{R}_G$ (or just $\mathcal{R}$ for short) of inequivalent irreducible representations of G. There

is considerable freedom in the choice of $\mathcal{R}$, since any $\rho \in \mathcal{R}$ can be replaced by an equivalent representation $\rho'(x) = T^{-1}\rho(x)\,T$. It is possible to show that this freedom always allows one to choose the irreducible representations to be **unitary**, which means that $\rho(x)^{-1} = \rho(x^{-1}) = \rho(x)^{\dagger}$ for all $x \in G$. This will prove to be important in many applications, so we will often assume that $\mathcal{R}$ has been chosen in this way.

The irreducible representations of compact groups are always finite dimensional. If G is finite, then $\mathcal{R}$ is a finite set. If G is compact but not finite, then $\mathcal{R}$ is countable. It is for these reasons (starting with Theorem 1.2.1) that in this thesis we mostly restrict ourselves to working with compact groups.

We can immediately write down two representations for any finite group. The **trivial representation** is just the one dimensional (and hence irreducible) constant representation $\rho_{\mathrm{tr}}(x) = 1$. At the other extreme is the $|\,G\,|$-dimensional **regular representation** based on the action of G on itself. Labeling the rows and columns of the representation matrices by the group elements, the matrix entries of the regular representations are

$$[\rho_{\mathrm{reg}}(x)]_{y,z} = \begin{cases} 1 & \text{if } xy = z \\ 0 & \text{otherwise.} \end{cases}$$

It is possible to show that if we decompose ρ_{reg} into a direct sum of irreducibles, each $\rho \in \mathcal{R}$ is featured at least once (up to equivalence).

1.2.1 Character theory and Abelian groups

Every representation has a corresponding **character** $\chi\colon G \to \mathbb{C}$, which is just the trace of the representation matrices, $\chi(x) = \mathrm{tr}(\rho(x))$. The character corresponding to an irreducible representation we call an **irreducible character**. By the identity $\mathrm{tr}(T^{-1}AT) = \mathrm{tr}(A)$ it is apparent that equivalent representations share the same character. The same identity also shows that the characters are class functions. In fact, the complete set of irreducible characters forms an orthogonal basis for the space of class functions [Serre, 1977, Theorem 6]. This means that the irreducible characters, and the conjugacy classes, can in turn be used to unambiguously label the irreducible representations of a finite group. This is one of the main reasons that characters are important in representation theory.

It is not difficult to prove that for finite groups $\sum_{\rho\in\mathcal{R}} d_\rho^2 = |\,G\,|$. If G is finite and Abelian, then every function on G is a class function, therefore G has exactly $|\,G\,|$ irreducible representations. We deduce that all irreducible representations of G are one dimensional if and only if G is Abelian. In this case the irreducible characters and the irreducible representations are one and the same. Noting that for one dimensional representations $|\,\rho(x)\,| = 1$, both can be regarded as homomorphisms from G to the unit circle in the complex plane. These results also generalize to infinite Abelian groups [Barut and Rączka, 1977, p.159]. For example, the irreducible representations of $\mathbb{R}$ are the functions $\rho_k(x) = e^{-2\pi ikx}$ with $k \in \mathbb{R}$, and the irreducible representations of $\mathbb{Z}_n$ are $\rho_k(x) = e^{-2\pi ikx}$ with $k \in \{0, 1, 2, \ldots, n-1\}$. In the theory of Abelian groups the terms irreducible representation and character are used interchangeably.

The space of characters is called the **dual space** of G. In the case of Abelian groups, the characters themselves form a group under the operation $(\chi_1\chi_2)\,(x) = \chi_1(x)\chi_2(x)$. This group, $\widehat{G}$, is called the **dual group** of G. As stated in a celebrated theorem due to Pontryagin, the double dual $\widehat{\widehat{G}}$ is isomorphic to G, and the isomorphism is canonical in the sense that there is a unique

$\xi \in \widehat{\widehat{G}}$ satisfying $\xi(\chi) = \chi(x)$ for any $\chi \in \widehat{G}$ [Barut and Rączka, 1977, p.163]. The dual of a locally compact Abelian (LCA) group is a discrete LCA group and vica versa. Pontryagin duality is what lies behind the theory of harmonic analysis on LCA groups that we develop in Section 2.2. When G is finite the theory is considerably simpler because in that case $\widehat{G}$ itself is isomorphic to G.

1.2.2 G-modules

A slightly more abstract way to define representations is in terms of G-modules. Recall that a G module over $\mathbb{C}$ is a vector space V over $\mathbb{C}$ supporting a homomorphism $\phi\colon G \to \mathrm{GL}(V)$. A representation, as defined above, is just such a mapping. Hence, each equivalence class of equivalent representations may be identified with a G-module V and its corresponding homomorphism ϕ. With slight abuse of notation we set $\rho = \phi$.

We can go back and forth between our two definitions of representation by noting that given a basis $\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_{d_\rho}$ of V, the matrix entries of the earlier definition can be recovered as $[\rho(x)]_{i,j} = [\rho(x)\boldsymbol{e}_j]_i$. The module theoretic view of representations is more elegant in that it removes the arbitrariness of choosing a particular representation from a class of equivalent ones, and often leads to cleaner proofs.

Defining representations as homomorphisms $\rho\colon G \to \mathrm{GL}(V)$ is also more general than our original definition in that it allows for infinite dimensional representations. For example, letting V be any space with a basis $\{e_x\}_{x\in G}$, we can generalize the concept of regular representation to infinite groups by setting $\phi(x)(e_y) = e_{xy}$. Theorem 1.2.1 will continue to hold for such infinite dimensional representations, as long as G is compact. For non-compact groups, on the other hand, infinite dimensional representations are crucial because some of their irreducibles fall in this category.

Group actions also give rise to G-modules. If a G is acting on a set S, then the **permutation representation** GS associated with S is a G-module with basis vectors labeled by elements of S on which G acts by $g(e_s) = e_{g(s)}$. This is a natural way to construct group representations.

1.2.3 Restricted and induced representations

Given a subgroup H of G, relating representations of G to representations of H will be important in the following. Given a representation ρ of G, its **restriction** to H, denoted $\rho\downarrow^G_H$ is given simply by $\rho\downarrow^G_H(x) = \rho(x),\ x \in H$. As a collection of matrices, $\rho\downarrow^G_H$ is just a subset of ρ; as maps, ρ and $\rho\downarrow^G_H$ are different, though, and this is what is emphasized by the notation. Where unambiguous, we shall refer to $\rho\downarrow^G_H$ as just $\rho\downarrow_H$.

Now let ρ be a representation of H and let $t_1, t_2, \ldots, t_l$ be a transversal for the left cosets of H in G. The **induced representation** denoted $\rho\uparrow^G_H$ is of the block diagonal form

$$\rho\uparrow^G_H(x) = \begin{pmatrix} \tilde\rho(t_1^{-1}xt) & \tilde\rho(t_1^{-1}xt_2) & \ldots & \tilde\rho(t_1^{-1}xt_l) \\ \tilde\rho(t_2^{-1}xt) & \tilde\rho(t_2^{-1}xt_2) & \ldots & \tilde\rho(t_2^{-1}xt_l) \\ \vdots & \vdots & \ddots & \vdots \\ \tilde\rho(t_l^{-1}xt) & \tilde\rho(t_l^{-1}xt_2) & \ldots & \tilde\rho(t_l^{-1}xt_l) \end{pmatrix}, \qquad x \in G,$$

where $\tilde\rho(x) = \rho(x)$ if $x \in H$ and otherwise it is zero. We refer the reader to the literature for proving that $\rho\uparrow^G_H$ is actually a representation. Where unambiguous, we shall refer to $\rho\uparrow^G_H$ as $\rho\uparrow^G$.

1.3 The symmetric group

In Section 1.1.5 we described how groups can be defined by their action on a set S. A prime example of this is the case when S is the set $\{1, 2, \ldots, n\}$ and we consider all $n!$ permutations of its elements. The **permutations** form a group under composition called the **symmetric group** of **degree** n and denoted $\mathbb{S}_n$. Hence, for $\sigma_1, \sigma_2 \in \mathbb{S}_n$, $\sigma_2 \sigma_1$ is the permutation that we get by permuting $\{1, 2, \ldots, n\}$ first according to σ_1 and then according to σ_2.

The symmetric group enjoys a special, canonical status in group theory, and is also the group with the most immediate applications in statistics and machine learning. Any group G acting on a finite set S can be regarded as a subgroup of $\mathbb{S}_{|S|}$. In particular, any finite group G acts on itself by $T_x(y) = xy$, $x \in G$, $y \in G = S$, hence G is a subgroup of $\mathbb{S}_{|G|}$ (Cayley's theorem). From the point of view of applications, the symmetric group is important because of the connection between permutations and rankings or matchings. Some of the most comprehensive texts on the symmetric group, where many of the following results were taken from are [James, 1978],[James and Kerber, 1981] and [Sagan, 2001].

There are several ways to notate elements of $\mathbb{S}_n$. In **two-line notation** we write $1, 2, \ldots, n$ on top and the image of these elements under the permutation on the bottom. For example,

$$\sigma = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 3 & 1 & 4 & 5 \end{bmatrix} \tag{1.3}$$

is an element of $\mathbb{S}_5$. We denote the mapping corresponding to each group element by the same symbol as the one we use to denote the group element itself. Hence, the above permutation is specified by the relations $\sigma(1) = 2$, $\sigma(2) = 3$, $\sigma(3) = 1$, etc.. In terms of these maps the group operation is defined by the relations $\sigma_2(\sigma_1(i)) = (\sigma_2 \sigma_1)(i)$ for all $\sigma_1, \sigma_2 \in \mathbb{S}_n$ and $i = 1, 2, \ldots, n$.

A more compact way to specify permutations is based on the observation that any permutation can be written as a combination of **cycles**. By cycle we mean an ordered subset $s_1, s_2, \ldots, s_k$ of $\{1, 2, \ldots, n\}$, such that $\sigma(s_i) = s_{i+1}$ for $i < k$ and $\sigma(s_k) = s_1$. For example, in **cycle notation** (1.3) becomes $\sigma = (123)(4)(5)$. Cycles of length one can be omitted without ambiguity, so σ can also be denoted simply as (123).

The **cycle-type** of σ is a list of the lengths of all the cycles making up σ. This is usually encoded in a **partition** of the integer n, by which we mean a sequence

$$\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_k)$$

of weakly decreasing natural numbers (i.e., $\lambda_i \geq \lambda_{i+1}$ for $i = 1, 2, \ldots, k-1$) such that $\sum_{i=1}^{k} \lambda_i = n$. We call k the **length** of the the partition λ. In our case, the partition corresponding to (1.3) would be $(3, 1, 1)$. Note that the concept of partitions of the integer n is distinct from the concept of partitions of the set $\{1, 2, \ldots, n\}$. In the following λ will always denote integer partitions of some number n. We will also use the notation $\lambda \vdash n$. It is easy to see that the conjugacy classes of $\mathbb{S}_n$ correspond exactly to the collection of elements of a given cycle type. Hence, the irreducible representations of $\mathbb{S}_n$ can conveniently be labeled by the integer partitions $\lambda \vdash n$.

Of particular interest is the conjugacy class of **transpositions**, which are permutations of the form (i, j). Since any cycle $(s_1, s_2, \ldots, s_k)$ can be written as the product $(s_1, s_2) \cdot (s_2, s_3) \cdot \ldots \cdot (s_{k-1}, s_k)$, the set of transpositions generates the entire symmetric group.

In fact, we can say more. Any transposition (i, j) (assuming without loss of generality $i < j$) can be written as a product of **adjacent transpositions** $\tau_k = (k, k+1)$ as

$$(i, j) = \tau_{j-1} \ldots \tau_{i+1} \tau_i \tau_{i+1} \ldots \tau_{j-1},$$

hence adjacent transpositions alone generate $\mathbb{S}_n$. Taking this minimalistic approach even further, since we can produce any τ_k as

$$\tau_k = (k, k+1) = (1, 2, \ldots, n)^k \cdot (1, 2) \cdot (1, 2, \ldots, n)^{-k},$$

the two group elements $(1, 2)$ and $(1, 2, \ldots, n)$ by themselves generate the whole of $\mathbb{S}_n$.

Related to transpositions is the important concept of the sign of a permutation. Consider the quantity

$$\Delta_\sigma = \prod_{\sigma(i)<\sigma(j)} (i - j) \qquad \sigma \in \mathbb{S}_n.$$

Applying any transposition to σ changes the sign of Δ but not its absolute value. Hence if a given permutation σ can be written as the product of m transpositions, $\mathrm{sgn}(\sigma) \equiv (-1)^m$, called the **sign** of σ, is a well defined quantity. Permutations for which $\mathrm{sgn}(\sigma) = 1$ are called **even** and those for which $\mathrm{sgn}(\sigma) = -1$ are called **odd**.

The function $\mathrm{sgn}(\sigma)$ is our first example of a representation of $\mathbb{S}_n$, called the **alternating representation**. The subgroup of even permutations is called the **alternating group** of degree n, denoted A_n. The alternating groups form a family of non-Abelian finite groups, much like the symmetric groups do.

1.3.1 Representations

The trivial representation $\rho(\sigma) = 1$ and the alternating representation $\rho(\sigma) = \mathrm{sgn}(\sigma)$ are both irreducible representations, in fact, they are the only two one-dimensional ones. By describing $\mathbb{S}_n$ in terms of its action on $(1, 2, \ldots, n)$, we have implicitly also met the n-dimensional representation

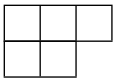
$$[\rho_{\mathrm{def}}(\sigma)]_{i,j} = \begin{cases} 1 & \text{if } \sigma(i) = j \\ 0 & \text{otherwise,} \end{cases}$$

called the **defining representation**. The actual $\rho_{\mathrm{def}}(\sigma)$ matrices are often called **permutation matrices**. The defining representation is not irreducible. In particular, permutation matrices leave the subspace generated by $(1, 1, \ldots, 1)$ invariant, which means that ρ_{def} is a direct sum of the trivial representation and some $n-1$ dimensional representation.

Tableaux, tabloids and permutation modules

The systematic study of the irreducible representations of $\mathbb{S}_n$ was pioneered by the Reverend Alfred Young and has a strong combinatorial flavor. Much of the theory revolves around operations on diagrams called tableaux, tabloids and polytabloids introduced by Young.

We start with a graphical representation for integer partitions consisting of simply laying down $\lambda_1, \lambda_2, \ldots, \lambda_k$ empty boxes in k consecutive rows. This is variously called the **Ferres diagram**, the **frame**, or the **shape** of λ. As an example, the shape of the $(3, 2)$ partition of 5 is

 .

Thanks to the bijection between partitions and irreducible representations of $\mathbb{S}_n$, Ferres diagrams can be used to label the irreducible representations.

A **Young tableau** is a Ferres diagram bijectively populated by the numbers $1, 2, \ldots, n$, such as

$$t = \begin{array}{|c|c|c|}\hline 3 & 2 & 5 \\ \hline 1 & 4 \\ \cline{1-2}\end{array} . \tag{1.4}$$

A **tabloid** is an equivalence class of tableau under permutations of the numerals in each of the rows. For example, the set of tableaux

$$\left\{ \begin{array}{|c|c|c|}\hline 3 & 2 & 5 \\ \hline 1 & 4 \\ \cline{1-2}\end{array}, \begin{array}{|c|c|c|}\hline 3 & 5 & 2 \\ \hline 1 & 4 \\ \cline{1-2}\end{array}, \begin{array}{|c|c|c|}\hline 2 & 3 & 5 \\ \hline 1 & 4 \\ \cline{1-2}\end{array}, \begin{array}{|c|c|c|}\hline 2 & 5 & 3 \\ \hline 1 & 4 \\ \cline{1-2}\end{array}, \begin{array}{|c|c|c|}\hline 5 & 2 & 3 \\ \hline 1 & 4 \\ \cline{1-2}\end{array}, \begin{array}{|c|c|c|}\hline 5 & 3 & 2 \\ \hline 1 & 4 \\ \cline{1-2}\end{array}, \right.$$

$$\left. \begin{array}{|c|c|c|}\hline 3 & 2 & 5 \\ \hline 4 & 1 \\ \cline{1-2}\end{array}, \begin{array}{|c|c|c|}\hline 3 & 5 & 2 \\ \hline 4 & 1 \\ \cline{1-2}\end{array}, \begin{array}{|c|c|c|}\hline 2 & 3 & 5 \\ \hline 4 & 1 \\ \cline{1-2}\end{array}, \begin{array}{|c|c|c|}\hline 2 & 5 & 3 \\ \hline 4 & 1 \\ \cline{1-2}\end{array}, \begin{array}{|c|c|c|}\hline 5 & 2 & 3 \\ \hline 4 & 1 \\ \cline{1-2}\end{array}, \begin{array}{|c|c|c|}\hline 5 & 3 & 2 \\ \hline 4 & 1 \\ \cline{1-2}\end{array} \right\}$$

is a tabloid of shape $(3, 2)$, which we denote

$$\{t\} = \begin{array}{\hline 3 & 2 & 5 \\ \hline 1 & 4 \\ \cline{1-2}\end{array} . \tag{1.5}$$

Clearly, there are $d = n!/(\lambda_1!\lambda_2!\ldots\lambda_k!)$ tabloids of shape λ.

Permutation modules

Members of $\mathbb{S}_n$ act on tableaux by permuting the numerals. This action extends naturally to tabloids, since if t and t' are two members of some tabloid $\{t\}$, then their images $\sigma(t)$ and $\sigma(t')$ for some $\sigma \in \mathbb{S}_n$ will also be members of a common tabloid $\sigma(\{t\})$. In other words the set of tabloids of a given shape λ form a homogeneous space of $\mathbb{S}_n$. Hence the matrices

$$[\rho(\sigma)]_{\{t'\},\{t\}} = \begin{cases} 1 & \text{if } \sigma(\{t\}) = \{t'\} \\ 0 & \text{otherwise,} \end{cases}$$

form a real valued representation of $\mathbb{S}_n$. This is called the **permutation representation** of shape λ. As an example, the defining representation is just the permutation representation of shape $(n-1, 1)$:

$$\begin{array}{|c|c|c|c|}\hline & & & \\ \hline \\ \cline{1-1}\end{array} .$$

The trivial representation is the permutation representation of shape (n):

$$\begin{array}{|c|c|c|c|c|}\hline & & & & \\ \hline\end{array} ;$$

while the regular representation corresponds to the shape $(1, 1, \ldots, 1)$:

$$\begin{array}{|c|}\hline \\ \hline \\ \hline \\ \hline \\ \hline \\ \hline\end{array} .$$

Keeping to our running example, in all the diagrams above we have taken $n = 5$.

To paint a more algebraic picture of permutation representations, we define the **row stabilizer** R_t of a tableau t of shape λ to be the subgroup of $\mathbb{S}_n$ which leaves the rows of t invariant, i.e., only permutes numerals within each row. Clearly, $R_t \cong \mathbb{S}_{\lambda_1} \times \mathbb{S}_{\lambda_2} \times \ldots \times \mathbb{S}_{\lambda_k}$, which is called the **Young subgroup** of shape λ. Now we identify each tabloid with the group algebra element

$$\{\boldsymbol{t}\} = \sum_{\sigma \in R_t} \sigma \boldsymbol{e}.$$

The elements corresponding to tabloids of shape λ form a submodule of $\mathbb{C}[\mathbb{S}_n]$, called the **permutation module** and denoted M^λ.

Irreducible modules

The starting point for constructing the irreducible representations of $\mathbb{S}_n$ are the permutation modules. Unfortunately, the M^λ themselves are in general not irreducible. However, each M^λ contains a submodule S^λ, called the **Specht module**, that is irreducible. It is possible to prove that the collection of Specht modules $\{S^\lambda\}_\lambda$ forms a complete set of irreducible G-modules for G. Thus, we get a complete set of irreducible representations.

To get a handle on the Specht modules, Young introduced yet another type of diagram, called a polytabloid. Analogously to the row stabilizer R_t, we can also talk about the **column stabilizer** C_t of a tabloid. A **polytabloid** is a linear combination of tabloids that is antisymmetric with respect to this subgroup:

$$\boldsymbol{e_t} = \sum_{\pi \in C_\mathbf{t}} \operatorname{sgn}(\pi)\, \pi(\{t\}).$$

The corresponding element of the group algebra is then

$$\boldsymbol{e_t} = \sum_{\pi \in C_\mathbf{t},\ \sigma \in R_t} \operatorname{sgn}(\pi)\, \pi(t)\sigma(t),$$

and the polytabloid corresponding to our running example (1.5) is

$$\boldsymbol{e_t} = \begin{array}{|ccc|}\hline 3 & 2 & 5 \\ 1 & 4 & \\ \hline \end{array} - \begin{array}{|ccc|}\hline 1 & 2 & 5 \\ 3 & 4 & \\ \hline \end{array} - \begin{array}{|ccc|}\hline 3 & 4 & 5 \\ 1 & 2 & \\ \hline \end{array} + \begin{array}{|ccc|}\hline 1 & 4 & 5 \\ 3 & 2 & \\ \hline \end{array} \ .$$

The Specht module is the submodule of the group algebra spanned by the set of polytabloids of shape λ.

To provide some examples, $M^{(n)}$ corresponding to the trivial representation is one dimensional and hence irreducible, so in this case $S^{(n)} = M^{(n)}$. Regarding $M^{(n-1,1)}$, the module of the defining representation, we have already noted that it contains a copy of the trivial module $S^{(n)}$. In fact, we will shortly be able to prove that $M^{(n-1,1)} = S^{(n)} \oplus S^{(n-1,1)}$. At the opposite extreme to these two cases, $M^{(1,1,\ldots,1)}$, the module of the regular representation, decomposes into a direct sum of S^λ's, where λ runs over all partitions of n. Its own Specht module, $S^{(1,1,\ldots,1)}$, is the module of the sign representation.

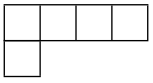
Young's standard representation

While the polytabloids span S^λ, they don't immediately give rise to an irreducible representation because they are not linearly independent. Once again, the combinatorial properties of Young tableaux come to the rescue. Young defined **standard tableaux** to be tableaux in which the numbers in each row and each column are increasing. Hence,

1	3	4
2	5	

is a standard tableau, but none of the tableaux in the tabloid of (1.4) are standard. Considering standard tableaux of a given shape, the corresponding polytabloids both span S^λ and are linearly independent. The corresponding irreducible representation is called the **standard representation**. We compute the matrix coefficients of the standard representation by letting permutations act on standard polytabloids and expressing the result as the linear combination of other standard polytabloids. The standard representation is only one of several equivalent named representations, however, and from a computational point of view, not necessarily the most attractive.

The standard tableaux based labeling scheme gives us a way to compute the dimensionality of each irreducible representation. As an example, recall that the defining representation ρ_{def} is the permutation representation of shape



(for $\mathbb{S}_5$). Clearly, there are $n-1$ possible standard tableaux of this shape. We have seen that ρ_{def} contains one copy of the trivial representation. Now we see that the remaining $n-1$ dimensional representation is irreducible.

For a general shape λ, the number of standard tableaux of shape λ is given by the so-called hook length formula of Frame, Robinson and Hall [Fulton, 1997, p. 53]. Given any box in a Ferres diagram, the corresponding **hook** consists of that box, all the boxes in the same row to its right and the all boxes in the same column below it. The length of a hook is the total number of boxes in such an inverted L-shape. The hook length formula states that the number of standard tableaux of shape λ is

$$f^\lambda = \frac{n!}{\prod_i l_i},$$

where l_i is the length of hook centered on box i, and the product extends over all boxes of the Ferres diagram of λ. Table 1.1 shows some applications of this formula to a few specific shapes. Note the general rule that for $k << n$ if there are $n-k$ boxes in the first row, then f^λ is order n^k.

Young's orthogonal representation

Historically the standard representation was discovered first, but in some respects **Young's orthogonal representation** (YOR) is simpler. In YOR it is again convenient to label the dimensions of ρ_λ by standard tableaux of shape λ, but this time we do not need to invoke polytabloids or the group algebra. Instead, restricting ourselves to the adjacent transpositions $\{\tau_1, \tau_2, \ldots, \tau_{n-1}\}$, we can specify the matrix entries of $\rho_\lambda(\tau_k)$ explicitly. These matrices turn out to be very sparse: the only non-zero entries in any row, indexed by the standard tableau t, are the diagonal entry

$$[\rho_\lambda(\tau_k)]_{t,t} = 1/d_t(k, k+1), \tag{1.6}$$

λ		f^λ
(n)		1
$(n-1,1)$		$n-1$
$(n-2,2)$		$\frac{n(n-3)}{2}$
$(n-2,1,1)$		$\frac{(n-1)(n-2)}{2}$
$(n-3,3)$		$\frac{n(n-1)(n-5)}{6}$
$(n-3,2,1)$		$\frac{n(n-2)(n-4)}{3}$

λ		f^λ
$(n-3,1,1,1)$		$\frac{(n-1)(n-2)(n-3)}{6}$
$(n-4,4)$		$\frac{n(n-1)(n-2)(n-7)}{24}$
$(n-4,3,1)$		$\frac{n(n-1)(n-3)(n-6)}{8}$
$(n-4,2,2)$		$\frac{n(n-1)(n-4)(n-5)}{12}$
$(n-4,2,1,1)$		$\frac{n(n-2)(n-3)(n-5)}{8}$

Table 1.1: The number of standard tableaux of shape λ for some specific $\lambda \vdash n$. Note that by the bijection between partitions and irreducible representations of the symmetric group, f^λ is also the dimensionality of ρ_λ. For concreteness the diagrams are drawn as if $n = 8$.

and, if $\tau_k(t)$ happens to be also a standard tableau, the single off-diagonal entry

$$[\rho_\lambda(\tau_k)]_{t,\tau_k(t)} = \sqrt{1 - 1/d_t(k,k+1)^2}. \tag{1.7}$$

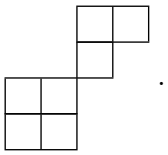
Here $d_t(k,k+1)$ is a special signed distance defined on Young tableaux. If x is a numeral in t, then the **content** of x, denoted $c(x)$ is the column index minus the row index of the cell where x is to be found. The distance $d_t(i,j)$ is then defined as $c(j) - c(i)$. Since adjacent transpositions generate the whole of $\mathbb{S}_n$, (1.6) and (1.7) are sufficient to fully define the representation.

While at first sight seemingly cryptic, the orthogonal representation has several advantages. As its name implies, the different rows of its representation matrices are orthogonal. From a practical point of view, the fact that representation matrices of transpositions are easy to compute and are sparse is important. Finally, we can readily see that considering the subgroup of $\mathbb{S}_n$ holding n fixed, the action of the transpositions $\tau_1, \tau_2, \ldots, \tau_{n-2}$ generating this subgroup on the $1, 2, \ldots, n-1$ part of the tableau is exactly the same as in a representation of $\mathbb{S}_{n-1}$. This property will be the key to constructing a fast Fourier transform on $\mathbb{S}_n$ in Section 3.3. YOR is described in detail in [Boerner, 1970].

The Murnaghan-Nakayama rule

A little more combinatorial wizardry gives an explicit formula for the irreducible characters. Recall that characters are constant on conjugacy classes. Hence, we can talk about $\chi_\lambda(\mu)$, the character corresponding to the partition λ evaluated at any permutation of cycle type μ, where μ is another partition of n.

There is a natural partial order on partitions induced by inclusion. We set $\lambda' \leq \lambda$ if $\lambda'_i \leq \lambda_i$ for each row $i = 1, 2, \ldots, k$ of λ. In case the length k' of λ' is less than k, we just set $\lambda'_i = 0$ for $k' < i \leq k$. We can generalize Ferres diagrams by defining the so-called **skew shapes**, denoted λ/λ' as the difference of their respective diagrams. For example, the skew shape $(4,3,2,2)\,/\,(2,2)$ is



A skew shape is said to be a **border strip** if it is both connected (i.e, one can get from any box to any other box via a series of boxes of which any two consecutive ones share an edge) and it

does not contain any 2×2 block of boxes. The weight of a border strip λ/λ' is defined to be $\mathrm{wt}\,(\lambda/\lambda') = (-1)^{\mathrm{length}(\lambda/\lambda')-1}$.

Now for a partition $\mu \vdash n$ of length k, a μ-border strip tableau of shape $\lambda \vdash n$ is a sequence of skew shapes

$$\lambda = \lambda^{(k)} \geq \lambda^{k-1} \geq \ldots \geq \lambda^{(0)} = \emptyset$$

such that each $\lambda^{(i)}/\lambda^{(i-1)}$ is a border strip and has exactly μ_i boxes. The whole sequence can be encoded in a tableau T of shape λ filled in with the numbers $1, 2, \ldots, k$, according to which stage in the sequence each box is eliminated. The weight of such a so-called **μ-border strip tableau** T is then defined

$$\mathrm{wt}(T) = \prod_{i=1}^{k} \mathrm{wt}\big(\lambda^{(i)}/\lambda^{(i-1)}\big).$$

Murnaghan and Nakayama's remarkable formula states that if λ and μ are partitions of n, then

$$\chi_\lambda(\mu) = \sum_T \mathrm{wt}(T), \tag{1.8}$$

where the sum is taken over all μ-border strip tableaux of shape λ. This formula will be of use to us in evaluating kernels on the symmetric group in Section 5.3.

1.4 The General Linear Group

The representation theories of the classical Lie groups such as GL_n, $\mathrm{SU}(n)$, $\mathrm{SO}(n)$, are closely related. Unfortunately, an introduction to the general theory goes well beyond the scope of this work. We restrict ourselves to giving a highly specialized but elementary description of the representation theory of the one group that will be important to us in the future, namely the general linear group GL_n.

Recall that $\mathrm{GL}_n \equiv \mathrm{GL}(\mathbb{C}^n)$ is the group of invertible linear maps of $\mathbb{C}^n$, hence its elements can be conveniently represented by $n \times n$ complex matrices $M = (m_{ij})_{i,j=1}^n$. Departing from our usual notation, we denote the elements of GL_n by M, instead of x.

1.4.1 Representations

There are multiple, at first sight very different approaches to describing the irreducible representations of the general linear group. In keeping with the spirit of the previous sections and bearing in mind our computational goals, we opt for a fundamentally combinatorial approach. An added benefit of this will be the appearance of remarkable parallels with the representation theory of $\mathbb{S}_n$. In order to keep this section brief, most of the results we state without proof.

The first point of contact with the representation theory of $\mathbb{S}_n$ is that the irreducible representations of GL_n are labeled by partitions λ. Now, however, $m = |\,\lambda\,|$ is unconstrained, accommodating the fact that the actual number of irreducible representations is infinite. Only the length of λ is upper bounded by n. The exact statement, much less trivial to prove than its analog for $\mathbb{S}_n$, is that there is a bijection between the irreducible representations of GL_n and partitions of length at most n of natural numbers $m \geq 1$.

Specht modules also have a close analog. Again, we start by constructing a larger, reducible module, V^λ, with basis vectors e_t labeled by Young tableaux of shape λ. However, now the tableaux

are filled with the integers $1, 2, \ldots, n$ as opposed to $1, 2, \ldots, m = |\,\lambda\,|$. Naturally, repetitions are permitted. GL_n acts on V^λ by

$$\rho(M)\colon e_t \mapsto \sum_{i,j=1}^{n} M_{i,j}\, e_{t[i,j]}$$

where $t\,[i,j]$ denotes the tableau derived from t be replacing each occurrence of i by j. In exact analogy with the $\mathbb{S}_n$ case, we introduce the Young symmetrizer, and let it act on basis vectors to form equivalence classes corresponding to the analogs of polytabloids. The resulting module, called the **Weyl module** will be our irreducible GL_n-module.

Matrices of the form

$$M = \begin{pmatrix} m_{1,1} & m_{1,2} & \ldots & m_{1,n-1} & 0 \\ m_{2,1} & m_{2,2} & \ldots & m_{2,n-1} & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ m_{n-1,1} & m_{n-1,2} & \ldots & m_{n-1,n-1} & 0 \\ 0 & 0 & \ldots & 0 & 1 \end{pmatrix}$$

form a subgroup of GL_n isomorphic to GL_{n-1}. Whenever we talk of the latter as a subgroup of the former, we shall be referring to this particular embedding.

The actual irreducible representations $D_\lambda\colon \mathrm{GL}_n \to \mathrm{GL}(V^\lambda)$ we undertake to find by constructing a Gelfand-Tsetlin basis adapted to the tower of subgroups

$$\mathrm{GL}_n \;>\; \mathrm{GL}_{n-1} \times \mathbb{C}^* \;>\; \mathrm{GL}_{n-2} \times (\mathbb{C}^*)^2 \;>\; \ldots \;>\; (\mathbb{C}^*)^n\,.$$

The key is again the way that $D_\lambda \downarrow_{\mathrm{GL}_{n-1}}$ splits into irreducible representations of GL_{n-1}, and this turns out to be only slightly more complicated than in the $\mathbb{S}_n$ case. Let us introduce the notation $\lambda' \leq \lambda$ to denote that $\lambda'_i \leq \lambda_i$ for each row $i = 1, 2, \ldots, k$ of λ. If the length of λ' is less than k, we just set $\lambda_i = 0$ for $i > k$. With this notation,

$$D_\lambda = \bigoplus_{\lambda' \leq \lambda} D_{\lambda'} \times \mathbb{C}^{*|\,\lambda\,|-|\,\lambda'\,|}\,,$$

where $D_{\lambda'}$ are the irreducible representations of GL_{n-1}, and the sum includes the empty partition $\lambda' = ()$. In contrast to the $\mathbb{S}_n$ case, the graph of this branching rule is not a tree (Figure 1.1). Since

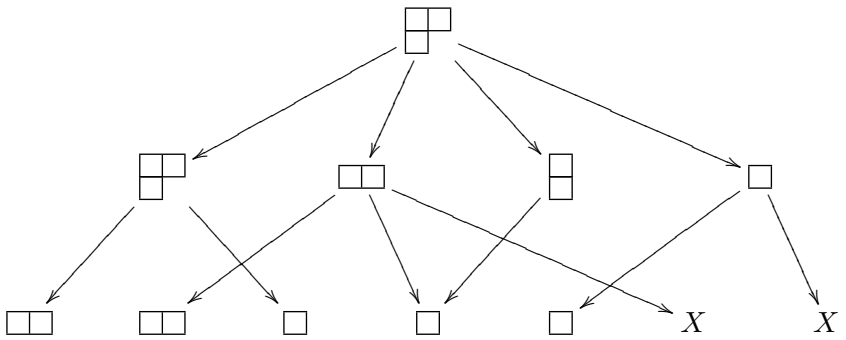


Figure 1.1: The branching structure of the irreducible representations $(2, 1)$ of GL_3

$\mathrm{GL}_1 = \mathbb{C}^*$ is Abelian, the irreducible representations at the bottom of the branching diagram are

one-dimensional. The basis vectors of V^λ are labeled by paths in the diagram from top to bottom, and such paths can in turn be labeled by Young tableaux of shape λ, putting the number i in every slot of the tableau which gets eliminated at level i, as we descend the path. The numbers in the resulting tableaux are strictly increasing in each column and weakly increasing in each row. These are the so-called **semi-standard tableaux**. We see that semi-standard tableaux of shape λ are in bijection with the basis vectors of V^λ.

Chapter 2

Harmonic analysis

Fourier analysis is a unifying theme, spanning almost all branches of mathematics. Part of what makes it so powerful is its ability to relate the analytical to the algebraic properties of abstract spaces. One of the goals of this thesis is to show that this can be generalized to the non-commutative realm, and that generalized Fourier transforms can help in solving statistical and learning problems.

We begin with a review of the classical Fourier transforms, a careful discussion of their algebraic interpretation, and a discussion of how that generalizes to compact non-Abelian groups. We then present two examples that show how these seemingly abstract ideas relate to concrete problems in statistical data analysis.

A hugely entertaining and very informative book on classical Fourier analysis with many insights into the history and development of the subject is [Körner, 1988]. Of the many books written on abstract harmonic analysis [Rudin, 1962] is a true classic, but it only deals with the locally compact Abelian case. Very closely in line with our algebraic approach are Audrey Terras's books [Terras, 1999] and [Terras, 1985].

One of the best sources of inspiration for applications of these ideas is the sequence of papers [Maslen and Rockmore, 1997][Rockmore, 1997] and [Healy et al., 1996], while without question the key book on applying ideas from finite group theory to statistics is [Diaconis, 1988].

2.1 The classical Fourier transforms

The three classical Fourier transforms are defined on the space of functions on the unit circle (equivalently, periodic functions on the real line with period 2π),

$$\widehat{f}(k) = \frac{1}{2\pi}\int_0^{2\pi} e^{-ikx}\, f(x)\, dx, \qquad k \in \mathbb{Z}; \tag{2.1}$$

functions on the real line,

$$\widehat{f}(k) = \int e^{-2\pi ikx}\, f(x)\, dx, \qquad k \in \mathbb{R}; \tag{2.2}$$

and functions on $\{0, 1, 2, \ldots, n-1\}$,

$$\hat{f}(k) = \sum_{x=0}^{n-1} e^{-2\pi ikx/n}\, f(x), \qquad k \in \{0, 1, 2, \ldots, n-1\}\,. \tag{2.3}$$

The first of these cases is sometimes called the **Fourier series**, and the third one the **discrete Fourier transform**. In all three cases we assume that f may be complex valued.

The corresponding inverse transforms are

$$f(x) = \sum_{k=-\infty}^{\infty} \hat{f}(k)\, e^{ikx},$$

$$f(x) = \int e^{2\pi ikx}\, \widehat{f}(k)\, dk, \tag{2.4}$$

$$f(x) = \frac{1}{n} \sum_{k=0}^{n-1} e^{2\pi ikx/n}\, \widehat{f}(k).$$

We will sometimes use the notation $\mathcal{F}\colon f \mapsto \widehat{f}$ for the forward transform and $\mathcal{F}^{-1}\colon \widehat{f} \mapsto f$ for the inverse transform. Note that cases (2.1) and (2.2) require some restrictions on f, namely integrability for the forward transform, and certain continuity conditions for the existence of the inverse. These conditions are quite technical, and not of primary concern to us here, so we refer the reader to the literature for details [Körner, 1988].

Instead, we focus on the properties shared by all three transforms. The first of these is that with respect to the appropriate inner products

$$\langle f, g\rangle = \frac{1}{2\pi} \int_0^{2\pi} f(x)\, g(x)^*\, dx, \qquad \langle \widehat{f}, \widehat{g}\rangle = \sum_{k=-\infty}^{\infty} \widehat{f}(k)\, \widehat{g}(k)^*,$$

$$\langle f, g\rangle = \int f(x)\, g(x)^*\, dx, \qquad \langle \widehat{f}, \widehat{g}\rangle = \int \widehat{f}(k)\, \widehat{g}(k)^*\, dx,$$

$$\langle f, g\rangle = \sum_{x=0}^{n-1} f(x)\, g(x)^*, \qquad \langle \widehat{f}, \widehat{g}\rangle = \sum_{k=0}^{n-1} \widehat{f}(k)\, \widehat{g}(k)^*,$$

all three transforms are unitary, i.e., $\langle f, g\rangle = \langle \widehat{f}, \widehat{g}\rangle$. This is sometimes referred to as **Parseval's theorem** or **Plancherel's theorem**.

The second property is related to the behavior of $\widehat{f}$ under translation. If f is a function on the real line, its **translate** by $t \in \mathbb{R}$ is $f^t(x) = f(x-t)$. In the other two cases $f^t(x) = f([x-t]_{2\pi})$ and $f^t = f([x-t]_n)$, respectively, where $[\,\cdot\,]_z$ denotes modulo z. The translation property in the three different cases reads $\widehat{f^t}(k) = e^{itk}\, \widehat{f}(k)$, $\widehat{f^t}(k) = e^{2\pi itk}\, \widehat{f}(k)$, and $\widehat{f^t}(k) = e^{2\pi itk/n}\, \widehat{f}(k)$.

As an extension of the translation property by linearity we have the **convolution theorems**. Convolution in the three different cases is defined respectively

$$(f * g)(x) = \int f([x-y]_{2\pi})\, g(y)\, dy,$$

$$(f * g)(x) = \int f(x-y)\, g(y)\, dy,$$

$$(f * g)(x) = \sum_{y=0}^{n-1} f([x-t]_n)\, g(y)\, dy,$$

and the convolution theorem reads $\mathcal{F}(f * g)(k) = \widehat{f}(k) \cdot \widehat{g}(k)$.

When the underlying space has a natural differential structure, the Fourier transform also relates to that in a canonical way: on the unit circle (parametrized by angle) we have $\widehat{\partial f}(k) = ik\,\widehat{f}(k)$, while on the real line $\widehat{\partial f}(k) = 2\pi ik\,\widehat{f}(k)$. This is one of the reasons why Fourier analysis is central to the study of partial differential equations as well.

2.2 Fourier analysis on groups

From an algebraic point of view the striking feature of the exponential factor appearing in (2.1) is the multiplicative property $e^{ik(x_1+x_2)} = e^{ikx_1}e^{ikx_2}$. Similar relations hold for (2.2) and (2.3). Noting that in each case the domain of f is a group G (namely, $\mathbb{T} = \mathbb{R}/\mathbb{Z}$, $\mathbb{R}$ and $\mathbb{Z}_n$), a group theorist will immediately notice that these factors are just the irreducible characters of G. Hence, all three classical Fourier transforms are special cases of the general formula for the Fourier transform on a locally compact Abelian group:

$$\widehat{f}(\chi) = \mathcal{F}(f)(\chi) = \int_G \chi(x)\, f(x)\, d\mu(x), \tag{2.5}$$

where χ ranges over the characters of G, and μ is the Haar measure on G. Since the irreducible characters of an LCA group form a dual group $\widehat{G}$, the inverse transform is given by a similar integral over the dual group:

$$f(x) = \mathcal{F}^{-1}(\widehat{f})(x) = \int_{\widehat{G}} \chi(x^{-1})\, \widehat{f}(\chi)\, d\widehat{\mu}(\chi). \tag{2.6}$$

Taking advantage of the fact that the irreducible characters of an Abelian group form an orthonormal basis for the group algebra, (2.5) and (2.6) provide a straightforward extension of harmonic analysis to locally compact Abelian groups, preserving many of the properties of the classical Fourier transforms.

The difficulty in extending harmonic analysis to non-commutative groups is that in the non-Abelian case the irreducible characters (while still orthogonal) are no longer sufficient to form a basis of the group algebra $\mathbb{C}[G]$. Harmonic analysis is not just any projection of functions to an orthonormal basis, however. The convolution theorem, in particular, tells us that $\mathcal{F}$ is special because it corresponds to a decomposition of $L(G)$ (the class of all complex valued functions on G) into a sum of spaces which are closed under convolution. This is a purely algebraic property, put into even starker relief by noting that when G is compact, $L(G)$ may be identified with the group algebra $\mathbb{C}[G]$, where convolution becomes just multiplication: $fg = f * g$. In other words, Fourier transformation is nothing but a decomposition of the group algebra into irreducible sub-algebras.

Algebras that have a unique decomposition into a sum of irreducible sub-algebras are called **semi-simple**. Once again, compactness is key: it is possible to show that the group algebra of any compact group is semi-simple. The V_η irreducible subalgebras in the corresponding decomposition

$$\mathbb{C}[G] \cong \bigoplus_\eta V_\eta$$

are called **isotypals**. When G is Abelian, each V_η is one-dimensional, and the form of the Fourier transform (2.5) is essentially uniquely determined. When G is non-Abelian, however, the V_η subspaces may be multidimensional and demand further interpretation.

Since V_η is irreducible, it must be isomorphic to $\mathrm{GL}(V)$ for some vector space V over $\mathbb{C}$. Denoting projection onto V_η by $\downarrow_\eta$, for any two basis vectors e_x and e_y of $\mathbb{C}[G]$ we must have

$e_x\downarrow_\eta \cdot e_y\downarrow_\eta = (e_xe_y)\downarrow_\eta = e_{xy}\downarrow_\eta$. In other words, the map $\phi_\eta\colon G \to \mathrm{GL}(V)$ given by $\phi_\eta\colon x \mapsto e_x\downarrow_\eta$ is a representation. Since V_η is irreducible, ϕ_η must be an irreducible representation. One can also show that each irreducible of G is featured in the sum exactly once. In summary, we have an isomorphism of algebras

$$\mathbb{C}[G] \cong \bigoplus_{\rho\in\mathcal{R}} \mathrm{GL}(V_\rho). \tag{2.7}$$

Giving an explicit basis to V_ρ allows us to express $e_x\downarrow_\rho$ in matrix form $\rho(x)$, and corresponds to choosing one specific representation from a class of equivalent ones. The elements of V_ρ then take the form of d-dimensional complex matrices, and the projection of a general $f \in \mathbb{C}[G]$ onto the V_ρ-isotypal can be expressed explicitly as

$$f\downarrow_\rho = \sum_{x\in G} \langle f, e_x\rangle\, e_x\downarrow_\rho = \sum_{x\in G} f(x)\,\rho(x).$$

By the **Fourier transform** of f we mean this collection of matrices

$$\widehat{f}(\rho) = \sum_{x\in G} f(x)\,\rho(x) \qquad\qquad \rho\in\mathcal{R}. \tag{2.8}$$

The Fourier transform $\mathfrak{F}\colon f \mapsto \widehat{f}$ is unitary with respect to the norms

$$\|\, f\,\|^2 = \frac{1}{|\,G\,|}\sum_{x\in G} |\, f(x)\,|^2 \qquad \text{and} \qquad \|\,\widehat{f}\,\|^2 = \frac{1}{|\,G\,|^2}\sum_{\rho\in\mathcal{R}} d_\rho \|\,\widehat{f}(\rho)\,\|^2_{\mathrm{Frob}}. \tag{2.9}$$

and the inverse transform is

$$f(x) = \frac{1}{|\,G\,|}\sum_{\rho\in\mathcal{R}} d_\rho \,\mathrm{tr}\left[\,\widehat{f}(\rho)\,\rho(x^{-1})\,\right]. \tag{2.10}$$

In the continuous but compact case, as usual, we have analogous formulae except that summation is replaced by integration with respect to Haar measure.

Defining convolution on a group as $(f*g)(x) = \sum_{y\in G} f(xy^{-1})g(y)$, by construction, the Fourier transform satisfies the convolution theorem $\widehat{f*g}(\rho) = \widehat{f}(\rho)\widehat{g}(\rho)$, and consequently also the left- and right-translation properties $\widehat{f^t}(\rho) = \rho(t)\,\widehat{f}(\rho)$ and $\widehat{f^{(t)}}(\rho) = \widehat{f}(\rho)\,\rho(t)$. From a computational point of view, these propeties are a large part of the reason that there is so much interest in Fourier transforms.

It is interesting to note that while as algebras the V_ρ are irreducible, as G-modules under left-translation $g(v) = e_g v$ or right-translation $g^{\mathrm{R}}(v) = ve_g$ (with $g \in G$ and $v \in V_\rho$) they are not. Indeed, since $\rho\colon G \to V_\rho$ is a d_ρ-dimensional irreducible representation, the G-module $Gv = \{\, g(v) \mid g\in G\,\}$ generated by any $v \in V_\rho$ is d_ρ-dimensional, while V_ρ-itself is d_ρ^2 dimensional. Naturally, the same holds for the right G-modules $vG = \{\, g^{(R)}(v) \mid g\in G\,\}$. On the other hand, $\{\, g_1vg_2 \mid g_1, g_2 \in G\,\}$ spans the whole of V_ρ. Thus, setting an orthonormal basis $\{\nu_1, \nu_2, \ldots, \nu_{d_\rho}\}$ for Gv induces a decomposition of V_ρ into right G-modules $W_1 = \nu_1 G,\ W_2 = \nu_2 G,\ \ldots, W_{d_\rho} = \nu_{d_\rho} G$,

$$V_\rho = \bigoplus_{i=1}^{d_\rho} W_i; \tag{2.11}$$

while setting an orthonormal basis $\{\nu'_1, \nu'_2, \ldots, \nu'_{d_\rho}\}$ for vG induces a decomposition

$$V_\rho = \bigoplus_{i=1}^{d_\rho} U_i \tag{2.12}$$

into left G-modules $U_1 = G\nu'_1,\ U_2 = G\nu'_2,\ \ldots, U_{d_\rho} = G\nu'_{d_\rho}$. In contrast to (2.7), however, these decompositions are not unique, but depend on the choice of v and the $\{\nu\}$ and $\{\nu'\}$ bases. Once again, this corresponds to the choice of representation: in matrix terms W_i is the space spanned by the i'th row of $\widehat{f}(\rho)$, while U_i is the space spanned by its i'th column. While (2.7) captures the fundamental structure of the group algebra, (2.11) and (2.12) are only a consequence of how we choose to represent its elements.

2.3 Spectral analysis on finite groups

Engineers are accustomed to the idea that looking at data in the frequency domain is often much more revealing than looking at it as a time series. Similarly, in natural science, from earthquake prediction to astronomy, Fourier transformation is well entrenched as the first step in approaching various types of data.

In general, by spectral analysis we mean the principle of analyzing data in terms of its projection to orthogonal subspaces. In the context of data on groups, the natural choice of subspaces are the isotypals. We now give two examples of this technique, one relating to the direct product of Abelian groups, and one relating to the symmetric group.

2.3.1 Factorial designs

Interestingly, one of the pioneering applications of Fourier analysis on groups other than the classical $\mathbb{R}^d, \mathbb{T}^d, \mathbb{Z}_n$ came from statistics. Without explicitly calling it as such, the British statistician Frank Yates not only developed a systematic approach to analyzing data on $\mathbb{Z}_2^n$ by Fourier transformation, but also proposed what amounts to a fast transform algorithm for this case [Yates, 1937]. In the following we closely follow the summary of this work in the survey article [Rockmore, 1997].

Assume that we are interested in studying the effect of sunlight(s), weed killer(w) and fertilizer(f) on the yield of wheat. As a simplification, we assume that each of these three factors has only two possible values: high and low. In general, the problem of analyzing the combined effect of k binary factors on some variable of interest is called a 2^k **factorial design**. In our example $k = 3$, so we denote the observed yield corresponding to each of the 8 possibilities $f_{+++}, f_{++-}, f_{+-+}, \ldots$. As statisticians, we are interested in different linear combinations of these observations. The zeroth order effect is the grand mean

$$\mu = \frac{1}{8}\left(f_{+++} + f_{++-} + f_{+-+} + f_{+--} + f_{-++} + f_{-+-} + f_{--+} + f_{---}\right),$$

giving an indication of the expected yield independent of each of the factors. The three first order effects μ_{s}, μ_{w} and μ_{f} tell us how the yield changes relative to the grand mean due to each factor in isolation. For example,

$$\mu_{\mathrm{s}} = \frac{1}{4}\left(f_{+++} + f_{++-} + f_{+-+} + f_{+--}\right) - \frac{1}{4}\left(f_{-++} + f_{-+-} + f_{--+} + f_{---}\right).$$

The second order effects $\mu_{\rm sw}$, $\mu_{\rm wf}$ and $\mu_{\rm sf}$ give information about how the first order effects are modified by the interaction between pairs of factors, for example,

$$\mu_{\rm sw} = \frac{1}{2}\left(f_{+++} + f_{++-}\right) - \frac{1}{2}\left(f_{+-+} + f_{+--}\right) - \frac{1}{2}\left(f_{-++} + f_{-+-}\right) + \frac{1}{2}\left(f_{--+} + f_{---}\right).$$

Finally, the lone third order effect

$$\mu_{\rm swf} = \frac{1}{8}\left(f_{+++} - f_{++-} + f_{+--} - f_{+-+} + f_{--+} - f_{-++} + f_{-++} - f_{---}\right)$$

captures third order interactions and completes the picture. The grand mean, the three first order effects, the three second order effects and the single third order effect together give a complete representation of the data, in the sense that the original data can be reconstructed from them.

Going from the original data to n'th order effects can be seen as a linear transformation

$$\begin{pmatrix} \mu \\ \mu_{\rm s} \\ \mu_{\rm w} \\ \mu_{\rm f} \\ \mu_{\rm sw} \\ \mu_{\rm wf} \\ \mu_{\rm sf} \\ \mu_{\rm swf} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{pmatrix} \cdot \begin{pmatrix} f_{+++} \\ f_{++-} \\ f_{+-+} \\ f_{+--} \\ f_{-++} \\ f_{-+-} \\ f_{--+} \\ f_{---} \end{pmatrix}. \tag{2.13}$$

We recognize the transform matrix as the character table of $\mathbb{Z}_2^3$, and the vector of μ's as the Fourier transform of f over this group,

$$\mu_k = \sum_{x \in \mathbb{Z}_2^3} \chi_k(x)\, f(x).$$

The ordering of the k indices implied by (2.13) establishes an isomorphism $x \mapsto \chi_k$ from $\mathbb{Z}_2^3$ to its dual, and if we define a notion of "norm" on $\mathbb{Z}_2^3$ from the identity $(+,+,+)$ based on how many $-$ components a group element has, and maps this to the dual, then the zeroth order effect will have norm 0, the first order effects will have norm 1, etc., just as one analyzes functions on $\mathbb{R}^n$ in terms of a hierarchy of Fourier components of increasing frequency.

2.3.2 Analysis of data on the symmetric group

It is because of the connection to rankings that of all non-Abelian groups the symmetric group has enjoyed the most attention from the statistics community. The pioneering figure in developing a systematic approach to the analysis of data on $\mathbb{S}_n$ has been the astonishingly versatile mathematician Persi Diaconis. In this section we sketch an outline of this procedure based on his monograph [Diaconis, 1988]. We will return to a more detailed discussion of the connection between rankings and $\mathbb{S}_n$ in chapter 5.

Given n people, movies, flavors of ice cream, etc., denoted $x_1, x_2, \ldots, x_n$, any ranking of these n items can be represented by listing them in the order $x_{\sigma(1)}, x_{\sigma(2)}, \ldots, x_{\sigma(n)}$ for the appropriate $\sigma \in \mathbb{S}_n$. There is a one-to-one relationship between rankings and permutations. The ranking data we analyze, such as the number of people voting for a particular ordering of the n items, we regard as a function $f(\sigma)$ on the $\mathbb{S}_n$.

The easily interpretable Fourier components are those corresponding to small permutation representations. For concreteness, as in section 1.3, we assume that $n=5$ and use Ferres diagrams to label the irreducible representations. As always, the transform at the trivial representations returns the grand mean

$$\widehat{f}(\ydiagram{5}) = \sum_{\sigma\in\mathbb{S}_n} f(\sigma)$$

which we take as the zeroth order effect.

The next level of analysis involves averaging f over all permutations that rank element i in position j. This gives rise to a 5×5 matrix with elements

$$M^{(1)}_{i,j} = \sum_{\sigma:\sigma(j)=i} f_\sigma\,,$$

which can easily be written in terms of the defining representation as

$$M^{(2)} = \sum_{\sigma\in\mathbb{S}_n} \rho_{\rm def}(\sigma)\, f(\sigma)\,.$$

As we have seen in (1.3.1), $\rho_{\rm def} = \rho_{\ydiagram{5}} \oplus \rho_{\ydiagram{4,1}}$, so M is immediate from the Fourier transform of f. Similarly to the $\mathbb{Z}_2^n$ case, the "pure" first order effect is given by the Fourier transform at an irreducible representation, in this case $\ydiagram{4,1}$.

The second order summary we define as the $\binom{n}{2}\times\binom{n}{2}$ matrix giving the averages over permutations which rank elements $\{i_1,i_2\}$ in positions $\{j_1,j_2\}$:

$$M^{(2)}_{\{i_1,i_2\},\{j_1,j_2\}} = \sum_{\sigma:\{\sigma(i_1),\sigma(i_2)\}=\{j_1,j_2\}} f(\sigma).$$

A few moments of reflection show that this is equal to

$$M^{(2)}_{\{i_1,i_2\},\{j_1,j_2\}} = \sum_{\sigma\in\mathbb{S}_n} \rho_{\ydiagram{3,2}}(\sigma)\, f(\sigma)$$

weighted by the $\ydiagram{3,2}$ permutation representation. Since $\rho_{\ydiagram{3,2}} = \rho_{\rm def}\oplus\rho_{\ydiagram{3,1,1}}$, the matrix M is easy to calculate from the Fourier transform. Once again, the irreducible $\rho_{\ydiagram{3,1,1}}$ captures the pure second order effect. The rest of the analysis we leave to the reader's imagination.

Chapter 3

Fast Fourier transforms

Given the central role of harmonic analysis in pure mathematics, it is not surprising that Fourier transforms are also a crucial device in engineering applications. Fourier transforms would not be nearly as ubiquitous in real world system, though, were it not for a family of powerful algorithms collectively called **Fast Fourier Transforms** (FFT's).

The Fast Fourier Transform revolutionized signal processing when it first appeared in a paper by Cooley and Tukey [1965]. The motivation for developing the FFT at the time was no less than to build a global observation system that would enable the United States to monitor whether the Soviet Union is adhering to the global nuclear test ban treaty. Given how important the FFT has proved to be in engineering and signal processing, the reader will not be surprised to hear that it had, in fact, already been discovered by Gauss, with the rather more modest intention of saving himself time in performing certain tedious astronomical calculations [Gauss, 1886].

When G is a finite group of order n, the Fourier transform is a unitary transformation $\mathcal{F}\colon \mathbb{C}^n \to \mathbb{C}^n$, so the naive way to implement it is by multiplying an $n \times n$ transformation matrix by an $n \times 1$ data vector, resulting in a total complexity of $O(n^2)$. In contrast, FFTs typically accomplish the same transformation in $O(n(\log n)^p)$ operations for some small integer p.

All FFTs build on a few deep ideas from representation theory, in which light the Gauss-Cooley-Tukey algorithm is just a special case of a more general and powerful approach, called the matrix separation of variables method. We describe this connection in detail, because studying the internal structure of FFTs reveals a lot about the structure of the group algebra, and is key to developing applications of non-commutative algebra to real-world problems. This is highlighted by our discussion of sparse, partial and twisted FFTs, which go beyond Clausen's original FFT for the symmetric group, and are original contributions to the field.

For more detailed and wide ranging discussions of the still juvenile field of fast non-commutative Fourier transforms, we recommend [Rockmore, 1997], [Maslen and Rockmore, 1997], [Clausen and Baum, 1993], and references cited therein.

3.1 Fast Fourier transforms on $\mathbb{Z}_n$

Given a prime factorization $p_1 p_2, \ldots p_l$ of n, the Cooley-Tukey algorithm computes the discrete Fourier transform of length n,

$$\hat{f}(k) = \sum_{j=0}^{n-1} \omega^{jk} f(j), \qquad k = 0, 1, 2, \ldots, n-1, \qquad \text{with} \quad \omega = e^{i2\pi/n}, \tag{3.1}$$

in $O(n \sum_{i=1}^{k} p_i)$ operations.

To see how the algorithm works, consider the case $n = pq$ for integers $p, q \geq 2$ (not necessarily prime) and the indexing scheme

$$j = j_2 + j_1 q, \qquad k = k_1 + k_2 p,$$

with $j_1, k_1 \in \{0, 1, 2, \ldots, p-1\}$ and $j_2, k_2 \in \{0, 1, 2, \ldots, q-1\}$, which allow us to factor (3.1) in the form

$$\hat{f}(k_1 + k_2 p) = \sum_{j_2=0}^{q-1} \omega^{j_2(k_1+k_2 p)} \sum_{j_1=0}^{p-1} (\omega^q)^{j_1 k_1} f(j_2 + j_1 q). \tag{3.2}$$

Note that the factor $(\omega^q)^{j_1 k_2 p}$ has dropped out, since $\omega^{pq} = 1$. The key observation is that (3.2) is really q separate DFTs of length p of the functions $f_{j_2}(j_1) = f(j_2 + j_1 q)$:

$$\hat{f}_{j_2}(k_1) = \sum_{j_1=0}^{p-1} (\omega^q)^{j_1 k_1} f_{j_2}(j_1);$$

followed by p DFTs of length q of the functions $g_{k_1}(j_2) = \omega^{j_2(k_1)} \tilde{f}_{j_2}(k_1)$:

$$\widehat{g}_{k_1}(k_2) = \sum_{j_2=0}^{q-1} (\omega^p)^{j_2 k_2} g_{k_1}(j_2).$$

Finally, the full transform is assembled according to $\hat{f}(k_1 + k_2 p) = \widehat{g}_{k_1}(k_2)$. Each of the q DFTs of length p take p^2 operations and each of the p DFTs of length q take q^2 operations, so computing the transform this way reduces its complexity from $O(p^2 q^2)$ to $O(pq(p+q))$. Clearly, for $n = p_1 p_2, \ldots p_l$ this idea can be extended to an l-level scheme of $p_1\, p_2\, \ldots\, p_{l-1}$ DFTs of length p_l followed by $p_1\, p_2\, \ldots\, p_{l-2} p_l$ DFTs of length p_{l-1}, etc., giving an overall complexity of $O(n \sum_{i=1}^{k} p_i)$. For $n = 2^m$, as promised, this achieves $O(n \log n)$.

It is important to note that the Cooley-Tukey trick is not the only device for constructing FFTs on $\mathbb{Z}_n$. In particular, the reader might ask what do if n is a prime. In this case a different algorithm due to Rader comes to the rescue. Rader's algorithm is based on the convolution theorem. Observe that if n is prime, then $\mathbb{Z}_n \setminus \{0\}$ forms a cyclic group of order $n-1$ under multiplication modulo n. This group we denote G. Although G is isomorphic to $\mathbb{Z}_{n-1}$, we do not label it as such, because we still want to label its elements according to their role in $\mathbb{Z}_n$ under the additive structure. Now let a be any generator of G (any element other than 1 will do). As j traverses $0, 1, 2, \ldots, n-2$, a^j traverses $1, 2, \ldots, n-1$, but not in numerical order. Furthermore, if $\cdot$ denotes

arithmetic multiplication modulo n, then $(a^{-j}) \cdot (a^k) = a^{k-j}$. One way to rewrite the DFT (3.1) then is

$$\begin{aligned} \hat{f}(0) &= \sum_{j=0}^{n-1} f(j), \\ \hat{f}(a^k) &= f(0) + \sum_{j=0}^{n-2} \omega^{a^{k-j}} f(a^{-j}). \end{aligned}$$

The second summation above is just the convolution over $\mathbb{Z}_{n-1}$ of the function $\check{f}(j) = f(a^{-j})$ with the function $g(i) = \omega^{a^i}$. By the convolution theorem, in Fourier space this can be computed in just $n-1$ operations. Since $n-1$ is not prime, we can use the Cooley-Tukey FFT to transform g and $\check{f}$ to Fourier space, perform convolution there, and use an inverse FFT to transform back.

3.2 Non-commutative FFTs

To find how the Cooley-Tukey algorithm might be generalized to non-commutative groups, we start with the following observations.

1. The decomposition of (3.1) into smaller Fourier transforms hinges on the fact that $H = \{0, q, 2q, \ldots, (p-1)q\}$ is a subgroup of $G = \mathbb{Z}_n$ isomorphic to $\mathbb{Z}_p$.

2. The corresponding cosets are $yH = \{\, x \in G \mid x \bmod q = y \,\}$, thus $Y = \{0, 1, 2, \ldots, q-1\}$ forms a transversal. The indexing scheme $j = j_2 + j_1 q$ corresponds to decomposing $x \in G$ in the form $x = yh$. In this light, the inner summation in (3.2) runs over H, while the outer summation runs over Y.

3. The factors ω^{jk} are the representations $\rho_k(j) = \omega^{jk}$ of $\mathbb{Z}_n$, and the factorization $\omega^{(j_2+j_1 q)k} = \omega^{j_2 k}\omega^{j_1 q k}$ generalizes to the non-commutative case in the form $\rho(x) = \rho(t)\rho(h)$ $(t \in T,\ h \in H)$, which is now of course a matrix equation.

What does not generalize to the non-Abelian setting is the indexing scheme for k, since the identification between the k indices and group elements is based on Pontryagin duality, which does not hold for non-commutative groups. The irreducible representations of non-commutative groups do not form a group, so it is more difficult to determine how to combine the components of the "sub-transforms" into components of the global Fourier transform.

Non-commutative fast Fourier transforms based on the so-called matrix separation of variables idea start with the analog of (3.1), which is

$$\hat{f}(\rho) = \sum_{y \in Y} \rho(y) \sum_{h \in H} \left(\rho\downarrow_H\right)(h)\, f(yh) \qquad \rho \in \mathcal{R}_G. \tag{3.3}$$

This does not by itself give a fast Fourier transform, because it does not take advantage of any relationships between the different irreducibles. In particular, the inner summations are not true sub-transforms, since they are expressed in terms of the irreducibles of G and not of H.

However, the theorem of complete reducibility tells us that $\rho\downarrow_H$ is always expressible as

$$\left(\rho\downarrow_H\right)(h) = T^\dagger \left[\bigoplus_{\rho' \in \mathcal{R}_H(\rho)} \rho'(h) \right] T,$$

where $\mathcal{R}_H(\rho)$ is some sequence of irreducible representations of H (possibly with repeats) and T is a unitary matrix. We are particularly interested in the special case when T is a unit matrix, in which case we say that $\mathcal{R}_G$ and $\mathcal{R}_H$ are **adapted representations**.

Definition 3.2.1 *Let $\mathcal{R}_G$ be a set of matrix representations of a finite group G and let H be a subgroup of G. We say that $\mathcal{R}_G$ is H-**adapted** if there is a set $\mathcal{R}_H$ of irreducible representations of H such that for all $\rho \in \mathcal{R}_G$, for some appropriate sequence $\mathcal{R}_H(\rho)$ of irreducible representations of H (possibly with repeats), $(\rho \downarrow H)(h)$ is expressible as*

$$(\rho \downarrow H)(h) = \bigoplus_{\rho' \in \mathcal{R}_H(\rho)} \rho'(h), \qquad \forall\, h \in H. \tag{3.4}$$

A set of representations of G is said to be adapted to a chain of subgroups $H_k < \ldots < H_2 < H_1 < G$ if it is adapted to each subgroup in the chain.

An equivalent concept is that of **Gel'fand-Tsetlin bases**. Given a G-module V, we say that a basis B is a Gel'fand-Tsetlin basis for V relative to $H < G$ if the matrix representation induced by B is adapted to H. When $\mathcal{R}_G$ satisfies (3.4), we can rewrite the inner summation in (3.3) as

$$\bigoplus_{\rho' \in \mathcal{R}_H(\rho)} \sum_{h \in H} \rho'(h)\, f(yh),$$

which, introducing the functions $f_y \colon H \to \mathbb{C}$ defined $f_y(h) = f(yh)$, reduces into

$$\bigoplus_{\rho' \in \mathcal{R}_H(\rho)} \sum_{h \in H} \rho'(h)\, f_y(h) = \bigoplus_{\rho'} \widehat{f}_y(\rho'),$$

where $\widehat{f_y}$ are now Fourier transforms over H. This leads to an algorithm which is analogous to the Cooley-Tucker FFT:

1. For each $y \in Y$, compute the Fourier transform $\{\widehat{f}_y(\rho)\}_{\rho' \in \mathcal{R}_H}$ (over H) of f_y.
2. If $\rho \in \mathcal{R}_G$ decomposes into irreducibles of H in the form $\rho = \rho'_1 \oplus \rho'_2 \oplus \ldots \oplus \rho'_k$, then assemble the matrix $\tilde{f}_y(\rho) = \widehat{f}_y(\rho'_1) \oplus \widehat{f}_y(\rho'_2) \oplus \ldots \oplus \widehat{f}_y(\rho'_k)$ for each $y \in Y$.
3. Finally, complete the transform by computing the sums

$$\hat{f}(\rho) = \sum_{y \in Y} \rho(y)\, \tilde{f}_y(\rho), \qquad \rho \in \mathcal{R}_G. \tag{3.5}$$

If instead of just $H < G$, we have a chain of subgroups $1 < G_1 < G_2 < \ldots < G$, we can apply the above decomposition scheme recursively to first compute $|\,G\,|\,/\,|\,G_1\,|$ Fourier transforms over G_1, then from these compute $|\,G\,|\,/\,|\,G_2\,|$ Fourier transforms over G_2, etc., all the way up to G. Maslen and Rockmore [1997] provide a detailed analysis of fast Fourier transforms of this type.

Before closing this section we remark that assuming that all our irreducible representations are unitary, Fourier transforms based on the matrix separation of variables principle automatically give us inverse transforms as well. At a fundamental level, if we represent the FFT as a computation on a lattice, where each edge of weight a between x and y stands for "multiply the value at x by a and

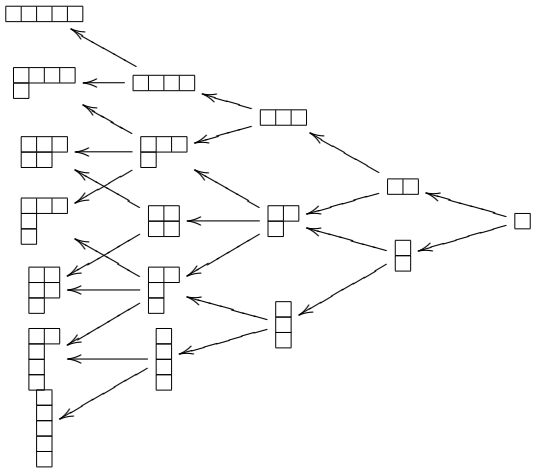


Figure 3.1: The Bratelli diagram for $\mathbb{S}_5$. The first column lists a complete set of irreducible representations of $\mathbb{S}_5$, the second column lists a complete set of irreducible representations of $\mathbb{S}_4$, etc.. The arrows indicate how the representations of $\mathbb{S}_k$ decompose when restricted to $\mathbb{S}_{k-1}$. From the FFT's point of view the crucial aspect of this diagram is that each irreducible representation of $\mathbb{S}_{k-1}$ get "reused" multiple times in constructing the larger representations.

add it to y", we can exploit unitarity to show that to get the lattice of the inverse transform, all that we need to do is invert the direction of the edges ad renormalize. More explicitly, the inverse transform uses the recursive structure as the forward transform, but instead of (3.5), at each step it computes the sub-transforms $\widehat{f}_y$ from $\widehat{f}$ by forming the sums

$$\widehat{f}(\rho') = \frac{|\,H\,|}{d_{\rho'}} \sum_{\rho \in \mathcal{R}} \sum_{i=1}^{m(\rho,\rho')} \frac{d_\rho}{|\,G\,|} \big[\rho(y)^\dagger\, \widehat{f}(\rho)\big]_{[\rho',i]},$$

where $m(\rho, \rho')$ is the multiplicity of ρ in $\mathcal{R}_H(\rho')$ and $[M]_{[\rho',i]}$ stands for "extract the i'th block corresponding to ρ' from the matrix M". In the case of the symmetric group, to be discussed next, this formula simplifies because each of the multiplicities is either 0 or 1.

3.3 Clausen's FFT for the symmetric group

Historically, the first non-commutative FFT was proposed by Clausen in 1989 for Fourier transformation on the symmetric group [Clausen, 1989]. Clausen's algorithm, which we now describe, follows the matrix separation of variables idea tailored to the chain of subgroups

$$\mathbb{S}_1 < \mathbb{S}_2 < \ldots < \mathbb{S}_{n-1} < \mathbb{S}_n. \tag{3.6}$$

Here and in the following whenever for $k < n$ we refer to $\mathbb{S}_k$ as a subgroup of $\mathbb{S}_n$, we identify it with the subgroup permuting the letters $1, 2, \ldots, k$ amongst themselves and fixing $k+1, \ldots, n$.

The structure of Young's orthogonal representation described in Section 1.3.1 reveals how each irreducible representation ρ_λ of $\mathbb{S}_n$ splits when restricted to $\mathbb{S}_{n-1}$. The key observations are that in the standard tableaux indexing the dimensions of ρ_λ, the number n must always be at an inner

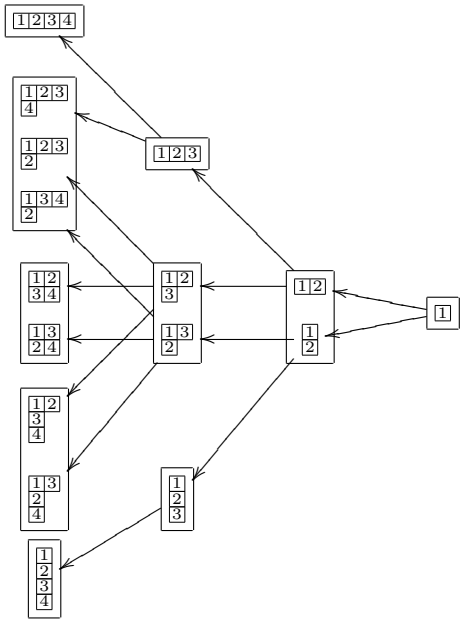
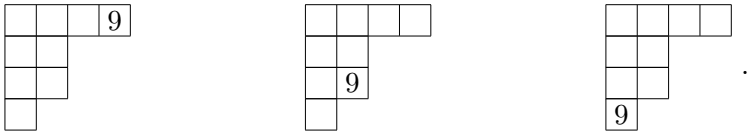


Figure 3.2: An extended Bratelli diagram for $\mathbb{S}_3$ showing the standard tableaux of each irreducible representation. Assuming Young's orthogonal representation, the arrows indicate how the basis vectors of each representation of $\mathbb{S}_k$ get promoted to basis vectors of various representations of $\mathbb{S}_{k+1}$.

corner, and that the only adjacent transposition which "mixes" dimensions labeled by standard tableaux which have n at different inner corners is $\tau_{n-1} = (n-1, n)$. For example, the standard tableaux of shape $\lambda = (4, 2, 2, 1)$ fall into three groups based on which of the following three positions the number 9 occupies:



Since $\sigma \in \mathbb{S}_{n-1}$ does not involve τ_{n-1}, the representation matrices $\rho_\lambda{\downarrow_{\mathbb{S}_{n-1}}}(\sigma)$ must be block diagonal, with each block corresponding to one of the possible positions of n in the standard tableaux. Closer inspection shows that each such block is in fact identical to $\rho_{\lambda^-}(\sigma)$, where λ^- is the Young shape derived from λ by removing the box containing n. In summary, YOR is adapted to $\mathbb{S}_n > \mathbb{S}_{n-1}$ and

$$\rho_\lambda \downarrow_{\mathbb{S}_{n-1}}(\sigma) = \bigoplus_{\lambda^- \in R(\lambda)} \rho_{\lambda^-}(\sigma), \qquad \sigma \in \mathbb{S}_{n-1}, \tag{3.7}$$

where $R(\lambda) = \{\, \lambda^- \vdash n-1 \mid \lambda^- \leq \lambda \,\}$. While for systems of representations other than YOR, (3.7) might involve conjugation by a matrix T, the general branching behavior still remains the same. This is called **Young's rule** and is captured in the so-called **Bratelli diagram** (Figure 3.1).

From a computational point of view the fact that YOR is adapted to the chain (3.6) is a major advantage. For concreteness in the following we assume that all the representation matrices are expressed in YOR. An appropriate choice of coset representatives for $\mathbb{S}_n/\mathbb{S}_k$ are then the **contiguous**

cycles defined

$$[\![i,n]\!](j) = \begin{cases} j+1 & \text{for } \; i \le j \le n-1 \\ i & \text{for } \; j = n \\ j & \text{otherwise} \end{cases} \qquad\qquad i \in \{1,2,\ldots,n\}\,.$$

Note that $[\![i,n]\!]$ is expressible as the $n-i$-fold product $\tau_i\,\tau_{i+1}\ldots\tau_{n-1}$ and that in YOR the $\rho_\lambda(\tau_j)$ matrices are very sparse.

Combining these elements with the general matrix separation of variables technique described in the previous section yields the following theorem. Note that here and in the following, we use the simplified notation $\widehat{f}(\lambda)$ for $\widehat{f}(\rho_\lambda)$.

Theorem 3.3.1 *[Clausen] Let f be a function $\mathbb{S}_n \to \mathbb{C}$ and let*

$$\widehat{f}(\lambda) = \sum_{\sigma\in\mathbb{S}_n} \rho_\lambda(\sigma)\, f(\sigma) \qquad\qquad \lambda \vdash n \tag{3.8}$$

be its Fourier transform with respect to Young's orthogonal representation. Now for $i = 1,2,\ldots,n$ define $f_i\colon \mathbb{S}_{n-1} \to \mathbb{C}$ as $f_i(\sigma') = f([\![i,n]\!]\,\sigma')$ for $\sigma' \in \mathbb{S}_{n-1}$, and let

$$\widehat{f_i}(\lambda^-) = \sum_{\sigma'\in\mathbb{S}_{n-1}} \rho_{\lambda^-}(\sigma')\, f_i(\sigma') \qquad\qquad \lambda^- \vdash n-1 \tag{3.9}$$

be the corresponding Fourier transforms, again with respect to Young's orthogonal representation. Then up to reordering of rows and columns,

$$\widehat{f}(\lambda) = \sum_{i=1}^{n} \rho_\lambda([\![i,n]\!]) \bigoplus_{\lambda^-\in R(\lambda)} \widehat{f_i}(\lambda^-), \tag{3.10}$$

where $R(\lambda) = \{\, \lambda^- \vdash n-1 \mid \lambda^- \le \lambda \,\}$.

Clausen's FFT proceeds by recursively breaking down Fourier transformation over $\mathbb{S}_n$ into smaller transforms over $\mathbb{S}_{n-1}, \mathbb{S}_{n-2}, \ldots$, and computing each $\mathbb{S}_k$-transform from the k independent $\mathbb{S}_{k-1}$-transforms below it by

$$\widehat{f}(\lambda) = \sum_{i=1}^{k} \rho_\lambda([\![i,k]\!]) \bigoplus_{\lambda^-\in R(\lambda)} \widehat{f_i}(\lambda^-), \qquad \lambda \vdash k, \tag{3.11}$$

with $R(\lambda) = \{\, \lambda^- \vdash n-1 \mid \lambda^- \le \lambda \,\}$. This prescribes a breadth-first traversal of the tree of left $\mathbb{S}_k$ cosets (figure 3.3). The pseudocode for Clausen's algorithm (to be called with $k = n$) is the following:

```
function FFT(k, f) {
  for i ← 1 to k { f̂_i ← FFT(k − 1, f_i) ;}
  for each λ ⊢ k do {
    f̂(λ) ← 0_{d_λ × d_λ};
    for i ← 1 to k {
      f̂(λ) ← f̂(λ) + ρ_λ([[i, k]]) ⊕_{λ⁻ ∈ R(λ)} f̂_i(λ⁻) ;
    }
  }
  return f̂;
}
```

The ideas behind Clausen's FFT can easily be adapted to computing the inverse Fourier transform

$$f(\sigma) = \frac{1}{n!} \sum_{\lambda \vdash n} d_\lambda \operatorname{tr}\left[\widehat{f}(\rho_\lambda)\, \rho_\lambda(\sigma)^\dagger\right] \qquad \sigma \in \mathbb{S}_n. \tag{3.12}$$

The key is that each stage of computing $\mathbb{S}_k$ transforms from $\mathbb{S}_{k-1}$ transforms is by itself a unitary transformation, hence the FFT can be reversed simply by applying the conjugate transpose transformation in each stage, and reversing the order of the stages. The pseudocode for the inverse FFT (to be called with $k = n$ and $\sigma = e$) is the following:

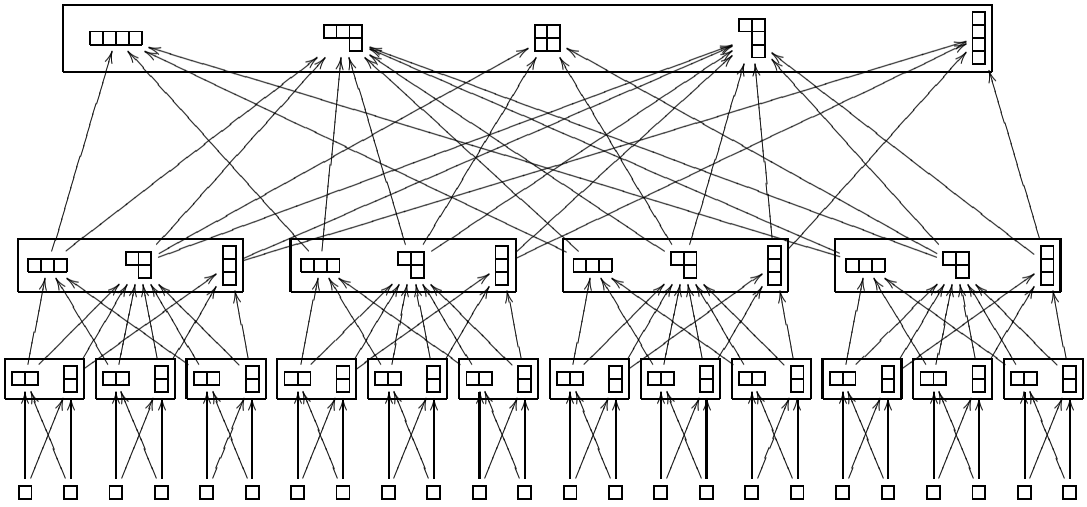


Figure 3.3: Fast Fourier transform on $\mathbb{S}_4$. The full transform is assembled from 4 FFTs over $\mathbb{S}_3$, which are, in turn, each assembled from 3 FFTs over $\mathbb{S}_2$.

```
function iFFT(k, f̂, σ){
  if k = 1 { f(σ) ← f̂; }
  else {
    for i ← 1 to k {
      for each λ⁻ ⊢ k − 1 do { f̂_i(λ⁻) = 0; }
      for each λ ⊢ k do {
        M ← ρ_λ([[i, k]]^{-1}) · f̂(λ);
        c ← 1;
        for each λ⁻ ∈ R(λ) do {
          f̂_i(λ⁻) ← f̂_i(λ⁻) + (d_λ / (k d_{λ⁻})) M(c : c + d_{λ⁻} − 1, c : c + d_{λ⁻} − 1);
          c ← c + d_λ;
      }}
      iFFT(k − 1, f̂_i, σ · [[i, k]]);
}}}
```

The inverse FFT appears to be more complicated than the forward FFT, but that is mostly due to the fact that decomposing M into blocks of specified sizes is more difficult to notate than assembling it from similar blocks. The notation $M(a:b,\ c:d)$ stands for the block of elements in M from row a to row b and from column c to column d (inclusive). The factor $\frac{d_\lambda}{k\,d_{\lambda^-}}$ is required to compensate for the $d_\lambda/n!$ multipliers in the inner products in (3.12). Note that in YOR $\rho_\lambda([\![i,k]\!]^{-1}) = [\rho_\lambda([\![i,k]\!])]^\top$.

Given a computational model where copying information is free and computational complexity is measured in terms of scalar operations consisting of a single multiplication followed by a single addition, the cost of Clausen's algorithm can be computed from the following considerations:

1. In YOR $\rho_\lambda(\tau_j)$ has at most 2 non-zero entries in each row, so for a general matrix M the multiplication $\rho_\lambda(\tau_j)M$ takes $2\,d_\lambda^2$ operations.

2. Using the factorization $[\![i,n]\!] = \tau_i\,\tau_{i+1}\ldots\tau_{n-1}$, computing $\rho_\lambda([\![i,k]\!])\bigoplus_{\lambda^-\in R(\lambda)}\widehat{f}_i(\lambda^-)$ in (3.11) thus takes $2(k-i)d_\lambda^2$ time.

3. By unitarity $\prod_{\lambda\vdash k} d_\lambda^2 = k!$, but at level k the FFT needs to compute $n!/k!$ independent Fourier transforms on $\mathbb{S}_k$, and in each of them the index i ranges from 1 to k.

Multiplying all these factors together yields a total complexity of

$$2n!\sum_{k=1}^{n}\sum_{i=1}^{k}(k-i) = 2n!\sum_{k=1}^{n}\frac{k(k-1)}{2} = \frac{(n+1)\,n\,(n-1)}{3}\,n!\,,$$

in accordance with our expectation that FFTs on finite groups generally run in $O(|G|\log^p|G|)$ time for some small integer p.

It is important to note that Clausen's algorithm is not the fastest known algorithm for Fourier transformation on $\mathbb{S}_n$. The so-called double-coset method yields a somewhat more efficient, albeit more complicated algorithm [Maslen and Rockmore, 2000], while David Maslen has a specialized algorithm which improves on even that result [Maslen, 1998]. Clausen's FFT, however, has the important distinctions of being the archetypal example of the matrix separation of variables technique,

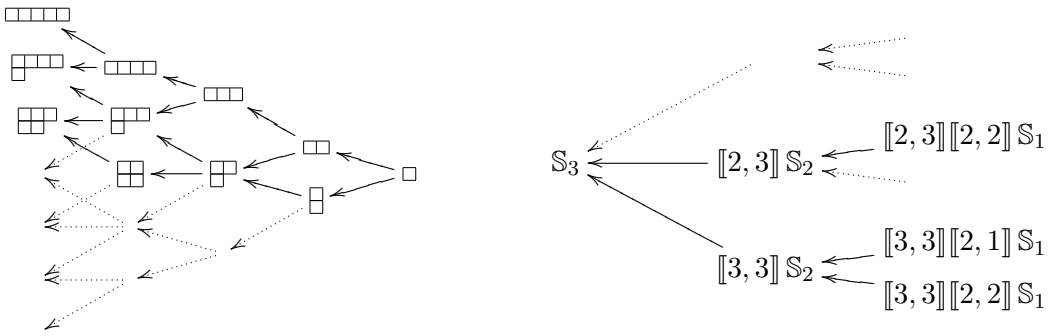


Figure 3.4: Two forms of sparsity: partial Fourier transforms restrict us to a subtree of the Bratelli diagram, while sparse functions restrict us to a subtree of the coset tree.

and of having an intuitive, flexible structure that can easily be adapted to specialized transforms, as we now investigate.

3.3.1 Extensions

The appealing internal structure of Clausen's algorithm lends itself to specialization to certain sub-classes of function of practical interest.

If the support of $f\colon \mathbb{S}_n \to \mathbb{C}$ is confined to a subset S of the full symmetric group, then the FFT can be made much faster by only traversing the appropriate subtree of cosets. For example, the right hand pane of of Figure 3.4 shows which transforms need to be computed when $f\colon \mathbb{S}_3 \to \mathbb{C}$ is confined to the set of permutations $S = \{(2,3),(1,2),e\}$.

More interestingly, we can also take advantage of situations where f is **band-limited** in the sense that $\widehat{f}$ is restricted to a subset of irreducibles labeled by $R_f \subset \{\lambda \vdash n\}$. In this case the savings come from only having to compute Fourier components labeled by the partitions $\bigcup_{k=1,2,\ldots,n} \{\, \lambda' \vdash k \mid \exists\, \lambda \in R \;\text{ such that }\; \lambda' \leq \lambda \,\}$, i.e., the subtree of the Bratelli diagram terminating in leaves in R. In many cases of practical interest, as in Figure 3.4, R will form a relatively compact block, inducing a subtree which is much smaller than the full Bratelli diagram.

Sparsity and band-limited representations are most helpful when they occur in tandem, because in that case a specialized FFT taking advantage of both can run in time less than $n!$. Indeed, while the FFT is always much faster than a naive transform, considering that 12! is already close to 500 million, any algorithm that scales with $n!$ quickly becomes infeasible with increasing n. Appropriately confining Clausen's algorithm both in the "time" and "frequency" domains can yield a polynomial time algorithm.

As a further extension of these ideas, we consider alternative ways of partitioning $\mathbb{S}_n$ into cosets. While the original algorithm splits $\mathbb{S}_n$ into the left cosets $\{[\![1,n]\!]\,\mathbb{S}_{n-1}, [\![2,n]\!]\,\mathbb{S}_{n-1}, \ldots, [\![n,n]\!]\,\mathbb{S}_{n-1}\}$, we can equally well take $\{\,\mathbb{S}_{n-1}\,[\![1,n]\!],\, \mathbb{S}_{n-1}\,[\![2,n]\!], \ldots,\, \mathbb{S}_{n-1}\,[\![n,n]\!]\}$, or, at a slight computational penalty, for fixed j, we could use the two-sided cosets

$$\{\,[\![1,n]\!]\,\mathbb{S}_{n-1}\,[\![j,n]\!],\; [\![2,n]\!]\,\mathbb{S}_{n-1}\,[\![j,n]\!], \ldots,\; [\![n,n]\!]\,\mathbb{S}_{n-1}\,[\![j,n]\!]\}\,.$$

The corresponding recursion rules are summarized in the following theorem.

Theorem 3.3.2 *Let f be a function $\mathbb{S}_n \to \mathbb{C}$ and let*

$$\widehat{f}(\lambda) = \sum_{\sigma \in \mathbb{S}_n} \rho_\lambda(\sigma)\, f(\sigma) \qquad \lambda \vdash n \tag{3.13}$$

be its Fourier transform with respect to Young's orthogonal representation. Let j be a fixed integer $1 \leq j \leq n$. Now for $i = 1, 2, \ldots, n$ define $f_i^L, f_i^R \colon \mathbb{S}_{n-1} \to \mathbb{C}$ as $f_i^L(\sigma) = f([\![i, n]\!]\, \sigma\, [\![j, n]\!])$ and $f_i^R(\sigma) = f([\![j, n]\!]\, \sigma\, [\![i, n]\!])$. Let

$$\widehat{f}_i^L(\lambda^-) = \sum_{\sigma' \in \mathbb{S}_{n-1}} \rho_{\lambda^-}(\sigma')\, f_i^L(\sigma') \qquad \text{and} \qquad \widehat{f}_i^R(\lambda^-) = \sum_{\sigma' \in \mathbb{S}_{n-1}} \rho_{\lambda^-}(\sigma')\, f_i^R(\sigma') \qquad \lambda \vdash n-1$$

be the corresponding Fourier transforms, again with respect to Young's orthogonal representation. Then up to reordering of rows and columns,

$$\widehat{f}(\lambda) = \left[\sum_{i=1}^{n} \rho_\lambda([\![i, n]\!]) \left[\bigoplus_{\lambda^- \in R(\lambda)} \widehat{f}_i^L(\rho_{\lambda^-}) \right] \right] \cdot \rho_\lambda([\![j, n]\!]), \tag{3.14}$$

and

$$\widehat{f}(\lambda) = \rho_\lambda([\![j, n]\!]) \cdot \left[\sum_{i=1}^{n} \left[\bigoplus_{\lambda^- \in R(\lambda)} \widehat{f}_i^R(\rho_{\lambda^-}) \right] \rho([\![i, n]\!]) \right], \tag{3.15}$$

where $R(\lambda) = \{\, \lambda^- \vdash n-1 \mid \lambda^- \leq \lambda \,\}$.

Clausen-type transforms which use this result to run on trees of two-sided cosets we call **twisted Fourier transforms**. Twisted transforms will be critical to the multi-object tracking application discussed in Chapter 6.

3.4 The $\mathbb{S}_n$`ob` library

While Clausen's FFT is well known in computational harmonic analysis, we have not been able to find any actual computer implementations of the algorithm, much less of the above described extensions, which, to our knowledge, have never been described in the literature. The $\mathbb{S}_n$`ob` C++ library is an attempt to fill this gap by providing a fast, highly structured and extendible set of library functions for Fourier transformation and other algebraic operations on the symmetric group. The name $\mathbb{S}_n$`ob` comes from "an `ob`ject oriented library for computations on $\mathbb{S}_n$".

The development version of $\mathbb{S}_n$`ob` was released into the public domain in August 2006, and since then has been downloadable from `http://www.cs.columbia.edu/~risi/SnOB` in source code format under the GNU Public License. The package is fully documented and has a clean, object oriented application programming interface. In its current public form $\mathbb{S}_n$`ob` can

1. Compute the Fourier transform and inverse Fourier transform of functions on $\mathbb{S}_n$ using Clausen's algorithm;

2. Compute partial Fourier transforms, sparse Fourier transforms, and twisted transforms of any structure;

3. Compute the representation matrices of $\mathbb{S}_n$ in Young's orthogonal representation as well as the corresponding characters;

4. Perform various operations on the group elements of $\mathbb{S}_n$.

Additional code has been developed to extend $\mathbb{S}_n$`ob` to operations on homogeneous spaces of $\mathbb{S}_n$, and to wreath product groups, but this code has not yet been made publicly available.

Classes

$\mathbb{S}_n$`ob` follows a strictly object oriented approach with a system of classes corresponding to algebraic objects:

```
Partition
Cycles
StandardTableau
Sn::Sn
Sn::Element
Sn::Irreducible
Sn::Function
Sn::FourierTransform
Sn::Ftree.
```

All the library functions are implemented as methods of these classes. For example, to compute a forward Fourier transform, one first constructs an `Sn::Sn` object with n as an argument to represent the group $\mathbb{S}_n$, then constructs an `Sn::Function` object to hold $f\colon \mathbb{S}_n \to \mathbb{C}$, and finally gets $\widehat{f}$ in the form of an `Sn::FourierTransform` object by calling the `FFT` method of `Sn::Function`. The reason that all these classes are defined within the "super-class" `Sn` is to allow for future expansions of the system to other finite groups, and to be able to define general methods applicable to any of these groups, corresponding to, for example, forming direct, semi-direct and wreath products.

Possibly the most powerful class in the current implementation is `Sn::Ftree`, which is used to construct coset trees for sparse/partial Fourier transforms, as well as run calculations on them.

Implementation

To make $\mathbb{S}_n$`ob` efficient, considerable effort has gone into optimizing the core low-level functions, such as the function implementing the recursive step in Clausen's algorithm. To avoid any overhead associated with objects and `STL` containers, these functions are implemented in pure C.

Since memory is often as much of bottleneck as computation time, $\mathbb{S}_n$`ob` is also organized to minimize its memory footprint, while some data structures are cached to facilitate further computations. In particular, before any computations on $\mathbb{S}_n$ can take place, the user must construct an `Sn` object, which automatically constructs objects corresponding to the whole cascade $\mathbb{S}_{n-1}, \mathbb{S}_{n-2}, \ldots, \mathbb{S}_1$, as well as each of their irreducible representations. Typically this only needs to be done once, and the objects are retained until the program terminates. The YOR coefficients are computed on demand, but once computed, are stored in the corresponding `Sn::Irreducible` object.

At this time $\mathbb{S}_n$`ob` does not explicitly take advantage of parallel architectures. The floating point precision, as well as the base field ($\mathbb{R}$ or $\mathbb{C}$) is specified in a master header file.

On a desktop PC $\mathbb{S}_n$`ob` can compute full FFTs for n up to about 10 or 11, the limiting factor being simply fitting f into memory. Computation time is still on the order of seconds. With partial FFTs meaningful computations can be performed with n in the 30–40 range.

Part II

Learning on groups

Chapter 4

Hilbert space learning algorithms

In artificial intelligence by (supervised) learning we mean the problem of inferring a relationship, sometimes called the **hypothesis**, between x and y values presented in (x, y) pairs called **examples**. The job of a learning algorithm is to formulate such a hypothesis after seeing m examples, $\{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$, called the **training set**. The quality of the hypothesis is determined by how well it predicts the y values corresponding to further x's. A collection of m' (x, y) pairs intended for measuring this is called the **testing set**.

Note the essential asymmetry between x and y. With nomenclature going back to the neural networks era, the x's are usually referred to as **inputs** and the y's as **outputs**. The inputs come from the **input space** $\mathcal{X}$, the outputs come from the **output space** $\mathcal{Y}$. Both of these spaces are presumed to be known in advance.

In this section we show how learning can be formulated in a statistical framework and introduce kernel methods, which are currently some of the most popular algorithms for solving learning problems of this type. Since kernel methods are based on the machinery of reproducing kernel Hilbert spaces, we take some time to review this formalism. Our contribution to this area concerns the form that kernel methods take under symmetry: either when the examples have hidden symmetries captured by a group, or when the input/output spaces are themselves groups. These cases are investigated in sections 4.4 and 4.5.

4.1 The statistical learning framework

A simple minded approach to learning might be to look up each testing input x in the training set and return the corresponding y. Such an approach is unlikely to lead to good results, however, because in most machine learning problems the probability of finding an exact match of x in the training set is very small. Even if we did find an exact match, predicting the same y as we saw in the training set might not be optimal, since the relationship between the x's and the y's is not necessarily deterministic. Machine learning tries to infer more general relationships between x and y than what can be accomplished by database lookup. The ability to make good predictions on inputs that we have not explicitly seen in the training set is called **generalization**, and the perilous case of finding a hypothesis that does well on the training data but doesn't generalize well is called **overfitting**. The ultimate performance measure of a machine learning algorithms is its generalization ability.

The 1990's saw learning problems of this kind being put on a firm mathematical footing by

formulating them in a probabilistic framework. The framework assumes that there is a single probability distribution $\mathcal{D}$ on $\mathcal{X}\times\mathcal{Y}$ which both training and testing examples are drawn from, each draw being statistically independent of the others. The distribution $\mathcal{D}$ is not known, and we make no attempt to infer it. Our goal is merely to "learn" a function $f\colon \mathcal{X} \to \mathcal{Y}$ which tells us what to guess for y given x. Specifically, for a given **loss function** $L\colon \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$, our objective is to minimize $\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[L(f(x), y)\right]$, where $\mathbb{E}_{(x,y)\sim\mathcal{D}}$ denotes expectation with respect to $\mathcal{D}$.

Learning problems of the form we just described abound in practical applications. Conceptually the simplest type are binary **classification** problems, where $\mathcal{Y} = \{-1, +1\}$ and **regression**, where $\mathcal{Y} = \mathbb{R}^n$. In the former case the loss function might be the **zero-one loss**

$$L(\hat{y}, y) = \begin{cases} 1 & \text{if } \hat{y} \neq y \\ 0 & \text{otherwise.} \end{cases}$$

whereas for regression we may use the **squared error loss** $L(\hat{y}, y) = (\hat{y} - y)^2$. In recent years statistical machine learning has developed a wide range of variations on these basic cases to suit more elaborate scenarios.

While the probabilistic formulation puts learning squarely in the realm of probability theory and statistics, in general the approach favored by these long-standing disciplines is quite different from that of machine learning. In supervised machine learning we focus strictly on prediction, as opposed to model building or trying to estimate the distribution $\mathcal{D}$. We make no attempt to predict what x's are likely to appear in the testing set, only to predict the y value once we know x. We are not even interested in estimating the conditional probability of y given x. All that our algorithm has to do is to give one specific estimate $\hat{y}$ for each testing input x.

4.1.1 Empirical risk minimization

It is a natural idea to solve learning problems by searching some space $\mathcal{F}$ for the hypothesis

$$f_{\text{emp}} = \arg\inf_{f\in\mathcal{F}} \left[\frac{1}{m} \sum_{i=1}^{m} L(f(x_i), y_i)) \right], \tag{4.1}$$

which minimizes the loss on the training set, i.e., the **training error**. The quantity in square brackets is also called the **empirical risk**, and the general philosophy of learning this way is called the principle of **empirical risk minimization** (ERM).

Empirical risk minimization seems like a promising inductive principle because $\mathcal{E}_{\text{train}}[f] = \frac{1}{m}\sum_{i=1}^{m} L(f(x_i), y_i))$ is an unbiased estimator (over random draws of the training set) of the **true error** $\mathcal{E}_{\mathcal{D}}[f] = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[L(f(x), y)\right]$. This makes it plausible that by driving down $\mathcal{E}_{\text{train}}[f]$ we should be able to find a hypothesis with close to optimal $\mathcal{E}_{\mathcal{D}}[f]$ as well. Unfortunately, for the particular choice $f = f_{\text{emp}}$, the unbiased approximation argument breaks down because f itself depends on the choice of training set. In fact, in terms of the excess generalization error $\mathcal{E}_{\text{gen}}[f] = \mathcal{E}_{\mathcal{D}}[f] - \mathcal{E}_{\text{emp}}[f]$, we are likely to find that f_{emp} is one of the worst possible choices in all of $\mathcal{F}$, since it has explicitly been selected to minimize the second term in $\mathcal{E}_{\text{gen}}$.

In less abstract terms, this anomaly is a manifestation of the fact that ERM by itself is almost guaranteed to overfit. Narrowly focusing on the training data alone, it is likely to return a hypothesis which describes the training set very well, but performs poorly on testing examples.

What the above argument is telling us is that we cannot hope to achieve effective learning with ERM without introducing some form of **capacity control**. Typically, this means one of two

things: either explicitly restricting $\mathcal{F}$ to a carefully chosen and suitably small class of well-behaved hypotheses on which overfitting is unlikely to occur; or adding an extra term $\Omega[f]$ to (4.1), which penalizes f based on some a priori measure of how likely f is to overfit. In this thesis we take the second route and focus on algorithms which minimize the so-called **regularized risk**

$$R_{\text{reg}}[f] = \frac{1}{m}\sum_{i=1}^{m} L(f(x_i), y_i)) + \lambda\Omega[f], \tag{4.2}$$

where $\lambda \in \mathbb{R}$ is a tunable so-called **regularization parameter**. The **regularizer** $\Omega[f]$ is often some measure of the smoothness of f, and its exact form typically depends on the specific learning problem at hand. The Hilbert space algorithms discussed in the following set $\Omega[f] = \langle f, f\rangle$ for some appropriately defined inner product between hypotheses. As we shall see in the next section, this leads to an elegant and powerful formalism. The loss function, on the other hand, is related to the choice of algorithm, which we discuss in Section 4.3.

At a conceptual level what is important is that (4.2) expresses a compromise between between smoothness and good performance on the training data. The idea that generalization ability will emerge as a consequence of this type of trade-off is at the very heart of modern machine learning.

4.2 The Hilbert space formalism

One of the most fruitful ideas to have emerged in machine learning is to formulate regularized risk minimization problems in Hilbert spaces, and identify $\Omega[f]$ with the squared Hilbert space norm $\| f \|^2 = \langle f, f\rangle$. The simple and highly regular nature of Hilbert spaces make them particularly well suited to learning on group structured input or output spaces.

Formally, a (real) **Hilbert space** is a complete linear inner product space. Let us unravel this definition a little. First, an **inner product** on a vector space V over $\mathbb{R}$ is a function $V \times V \to \mathbb{R}$ denoted $\langle v, w\rangle$, satisfying

$$\begin{aligned}
&\langle u+v, w\rangle = \langle u, w\rangle + \langle v, w\rangle\,,\\
&\langle \alpha u, v\rangle = \alpha\,\langle u, v\rangle\,,\\
&\langle u, v\rangle = \langle v, u\rangle,\\
&\langle u, u\rangle \geq 0 \;\text{ with equality only when } u = 0_V,
\end{aligned}$$

for all $u, v, w \in V$ and $\alpha \in \mathbb{F}$. Such an inner product gives rise to the **norm** $\| u \| = \sqrt{\langle u, u\rangle}$, and this provides V with a topology and a distance metric $d(x, x') = \| x - x' \|$.

A sequence $a_1, a_2, \ldots$ in a metric space is said to **converge** to $a \in M$ if $\lim_{i\to\infty} d(a_i, a) = 0$ and is called a **Cauchy sequence** if $\lim_{\min(i,j)\to\infty} d(a_i, a_j) = 0$. The space M is said to be a **complete metric space** if every Cauchy sequence in M converges to some $a \in M$.

A **Hilbert space** is a vector space $\mathcal{H}$ endowed with an inner product such that with respect to the norm induced by the inner product it is a complete metric space. The power of the Hilbert space formalism comes from the simultaneous presence of three different but closely related structures: the vector space, the inner product, and the topology.

It is important to note that one can also define Hilbert spaces over fields other than $\mathbb{R}$. Since machine learning almost exclusively deals with real Hilbert spaces, however, in the following whenever we refer to "Hilbert space" we will assume that the base field is $\mathbb{R}$. The simplest examples of such Hilbert spaces are the Euclidean spaces $\mathbb{R}^n$. In fact, any finite dimensional Hilbert space

over $\mathbb{R}$ is isomorphic to $\mathbb{R}^n$. In this sense, when $\mathcal{H}$ is finite dimensional, calling it a "Hilbert space" amounts to little more than glorifying an already familiar structures with a new name.

Things become more interesting when $\mathcal{H}$ is infinite dimensional. In this case we can still define a basis for $\mathcal{H}$, and if we label the basis vectors with elements of some countably or uncountably infinite set S, we can expand any $v \in \mathcal{H}$ as

$$v = \sum_{s \in S} \alpha_s \, e_s, \qquad \alpha_s \in \mathbb{R},$$

or the corresponding integral in the uncountable case. This makes it natural to view $\mathcal{H}$ as a space of functions $v(s) = \alpha_s$. As function spaces, Hilbert spaces are very well behaved. Almost all of linear algebra carries over, even to the infinite dimensional setting, making Hilbert spaces very attractive from the computational point of view. Hilbert spaces generated by a kernel function, which we now discuss, are particularly "computer-friendly".

4.2.1 Reproducing kernels

In the machine learning literature the concept of kernels has become so inseparable from that of Hilbert spaces that Hilbert space algorithms are often referred to as "kernel methods". Given a mapping Φ from some set $\mathcal{X}$ to a Hilbert space $\mathcal{H}$, the **kernel** k is simply the pull-back of the Hilbert space inner product to $\mathcal{X}$:

$$k(x, x') = \big\langle \Phi(x), \Phi(x') \big\rangle \,.$$

Clearly, k must be symmetric, $k(x, x') = k(x', x)$. Furthermore, any linear combination $f = \sum_{i=1}^m \alpha_i \Phi(x_i)$ (with $x_1, x_2, \ldots, x_m \in \mathcal{X}$ and $\alpha_1, \alpha_2, \ldots, \alpha_m \in \mathbb{R}$) must satisfy

$$\langle f, f \rangle = \sum_{i=1}^m \sum_{j=1}^m \alpha_i \, \alpha_j \, \langle \Phi(x_i), \Phi(x_j) \rangle \geq 0,$$

hence

$$\sum_{i=1}^m \sum_{j=1}^m \alpha_i \, \alpha_j \, k(x_i, x_j) \geq 0. \tag{4.3}$$

A symmetric function $k \colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ satisfying this condition for any choice of x_i's, α_i's and m is called a **positive semi-definite kernel** on $\mathcal{X}$. To streamline the terminology in the following we refer to positive semi-definite kernels simply as **positive definite kernels** or just kernels. Kernels which satisfy (4.3) with strict inequality we distinguish by calling them **strictly positive kernels**.

A related concept is that of conditionally positive definite kernels. A symmetric function $k \colon \mathcal{X} \times \mathcal{X} \to \mathbb{F}$ is said to be **conditionally positive definite** if it satisfies (4.3) under the additional restriction on the coefficients $\sum_{i=1}^n \alpha_i = 0$. Confusingly, instead of using the term conditionally-positive, some authors refer to such a kernel by saying that $-k$ is **negative definite**.

Remarkably, symmetry and positive definiteness are not only necessary, but also sufficient for the existence of a Hilbert space corresponding to k. We show this constructively, by building the so-called **reproducing kernel Hilbert space** (**RKHS**) corresponding to a given $k \colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ positive definite kernel. The construction proceeds in stages:

1. Define the functions $k_x(\cdot) = k(x, \cdot)$, and form a vector space V with basis labeled by $\{k_x\}_{x \in \mathcal{X}}$.
2. Define an inner product between basis vectors by $\langle k_x, k_{x'} \rangle = k(x, x')$ and extend this by linearity to the rest of V.
3. Finally, complete V to $\mathcal{H}$ by adjoining to it the limits of all Cauchy sequences and extending the inner product by continuity.

On a more abstract level, any Hilbert space in which the **evaluation maps** $\phi_x \colon f \mapsto f(x)$ are continuous is a reproducing kernel Hilbert space, since in such spaces by the Riesz representation theorem there must exist a unique $k_x \in \mathcal{H}$ satisfying $\phi_x(f) = \langle f, k_x \rangle$ for any $f \in \mathcal{H}$. The function defined $k(x, x') = k_x(x')$ is then symmetric and positive definite, and the RKHS it induces is $\mathcal{H}$. The reproducing property

$$f(x) = \langle f, k_x \rangle \tag{4.4}$$

alone is sufficient to define the concept of reproducing kernel Hilbert space [Schölkopf and Smola, 2002, chapter 2]. The close link between function evaluation and inner products is what makes Hilbert space algorithms computationally efficient.

4.2.2 The representer theorem

There is a long history in analysis and approximation theory of using the sequence of balls

$$B_r = \{\, f \in \mathcal{H} \mid \| f \| < r \,\}$$

to capture classes of functions of increasing complexity. In accordance with this principle, in the Hilbert space learning setting we define the regularized risk as

$$R_{\text{reg}}[f] = \frac{1}{m} \sum_{i=1}^{m} L(f(x_i), y_i)) + \lambda \| f \|^2 . \tag{4.5}$$

In certain learning algorithms $\| f \|$ is raised to a power different than 2, but for now we restrict ourselves to this simplest choice. In addition to mathematical elegance, this choice also has important computational advantages, thanks to the celebrated **representer theorem**:

Theorem 4.2.1 *(Kimeldorf and Wahba [1971]) Let $\mathcal{X}$ be an input space, $\mathcal{Y}$ an output space, $T = \{(x_i, y_i)\}_{i=1}^m$ a training set, $L \colon \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ a loss function, and $\mathcal{H}$ a reproducing kernel Hilbert space induced by some positive definite kernel $k \colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. Then the minimizer f_T of any regularized risk functional of the from (4.5) with $\lambda > 0$ is expressible as a linear combination*

$$f_T(x) = \sum_{i=1}^{m} \alpha_i \, k_{x_i}(x), \qquad \alpha_i \in \mathbb{R}. \tag{4.6}$$

Much of the power of Hilbert space learning algorithms stems from the fact that Theorem 4.2.1 reduces searching the dauntingly large space $\mathcal{H}$ to just finding the optimal values of the m coefficients $\alpha_1, \alpha_2, \ldots, \alpha_m$.

The statistics community generally regards Hilbert space algorithms as nonparametric methods because we are not fitting a fixed model with a finite number of pre-determined parameters. The number of α's in Equation (4.6) is, of course, finite, in fact, there are as many of them as we have training data points, but the form of (4.6) is dependent on the input data via the x_i indices. Even the best computers we have today would find it difficult to deal with a variational problem that cannot be reduced to operations on a finite number of variables.

4.2.3 Regularization theory

To complete our survey of the Hilbert space formalism it is important to ask what features of f are captured by the regularization schemes induced by different choices of k. When $\mathcal{X}$ is Lebesgue measurable, the clearest way to do this is to compare the Hilbert space inner product $\langle f, f \rangle_{\mathcal{F}}$ (note the subscript $\mathcal{F}$ emphasizing that this is the inner product induced by k) with the usual L_2-type inner product

$$\langle f, g \rangle_{L_2(\mathcal{X})} = \int f(x)\, g(x)\, dx\,.$$

Given a kernel k and corresponding RKHS $\mathcal{F}$, we define the **kernel operator** $\mathcal{K}\colon L_2(\mathcal{X}) \to L_2(\mathcal{X})$ as

$$\left(\mathcal{K}(f)\right)(x) = \int k(x, x')\, f(x')\, dx'\,,$$

and assume that it is invertible. By the RKHS property of $\mathcal{F}$, for any pair of functions $\alpha_1, \alpha_2 \in L_2(\mathcal{X})$ with finite support, $\langle \mathcal{K}\alpha_1, \mathcal{K}\alpha_2 \rangle_{\mathcal{F}} = \langle \alpha_1, \mathcal{K}\alpha_2 \rangle_{L_2(\mathcal{X})}$. In particular, for any $f, g \in \mathcal{F}$, $\langle f, g \rangle_{\mathcal{F}} = \left\langle \mathcal{K}^{-1} f, \mathcal{K}\mathcal{K}^{-1} g \right\rangle_{L_2(\mathcal{X})}$. If we then define the so-called **regularization operator** $\Upsilon = \mathcal{K}^{-1/2}$, we have

$$\langle f, g \rangle_{\mathcal{F}} = \langle \Upsilon f, \Upsilon g \rangle_{L_2(\mathcal{X})} = \int (\Upsilon f)(x)\, (\Upsilon g)(x)\, dx\,,$$

which explicitly tells us that using k is equivalent to L_2 regularization of Υf [Girosi et al., 1995][Schölkopf and Smola, 2002].

As an example, consider what is probably the most popular kernel on $\mathbb{R}^n$, the Gaussian (RBF) kernel

$$k(x, x') = e^{-\| x - x' \|^2 / (2\sigma^2)} \tag{4.7}$$

with length scale parameter (or variance parameter) σ. Letting $\hat{f}$ denote the Fourier transform of f and $\hat{\Upsilon}$ the frequency space regularization operator $\hat{\Upsilon}\hat{f} = \widehat{\Upsilon f}$, Girosi et al. [1995] show that in this case

$$\left(\hat{\Upsilon}\hat{f}\right)(\omega) = e^{|\omega|^2 \sigma^2} \hat{f}(\omega).$$

This is perhaps the most natural regularization scheme possible, penalizing high frequency components in f by a factor exponential in $|\omega|^2$. An alternative description of Υ in terms of derivatives of f (following [Yuille and Grzywacz, 1988]) gives

$$\| \Upsilon f \|_{\mathcal{F}}^2 = \int_{\mathcal{X}} \sum_{i=0}^{\infty} \frac{\sigma^{2i}}{i!\, 2^n} \left\| \left(O^i f\right)(x) \right\|_{L_2}^2 dx \tag{4.8}$$

where for i even $O^i = \Delta^{i/2}$, and for i odd $O^i = \nabla \Delta^{(i-1)/2}$, and Δ is the Laplacian operator. Again, this corresponds well to our intuitive notion that functions which have a large amount of energy in their high order derivatives are not smooth. The beginnings of the connection between learning theory and the spectral world view of Chapter 2 are already starting to emerge.

Unfortunately, the Gaussian kernel having such an easily interpretable regularization scheme is more of an exception than a rule. The best theoretical motivation behind many kernels frequently used in machine learning, especially kernels defined on structured objects like strings, graphs, etc., is that high kernel values correspond to some intuitive sense of "similarity" between the pair of objects. This is a much vaguer notion than rigorous arguments phrased in terms of regularization theory.

4.3 Algorithms

While the choice of inner product (i.e., kernel) determines the regularization scheme, the choice of loss function L in (4.5) determines the algorithm. Maybe somewhat surprisingly, and stretching our concept of empirical risk minimization a little, this loss function might not be the same as the loss function in the formula for the training and test errors.

To distinguish between the two, we switch to using ℓ to denote the loss involved in how we measure training, test, and expected error, and reserve L for the loss term in the regularized risk (4.5). One reason for this bifurcation is purely algorithmic: in general, convex loss functions lead to efficiently solvable optimization problems, while ℓ is often non-convex and not even continuous, as in the case of the zero-one loss for classification. The other reason is statistical: an appropriate choice of loss function can lead to more robust estimators.

4.3.1 Gaussian processes

Probably the simplest conceivable loss function is the squared error loss

$$L(y, \widehat{y}) = (\widehat{y} - y)^2$$

that we already encountered in Section 4.1 in the context of regression. While in some situations more sophisticated loss functions do have advantages over the squared error loss, in simple real-valued regression problems, despite our earlier arguments regarding the distinction between ℓ and L, it is not unreasonable to choose both losses to be the squared error loss. Two things that are particularly attractive about the regularized risk minimization problem (4.5) corresponding to this loss are that: (a) its solution can be expressed in closed form; (b) it has an attractive Bayesian interpretation as the maximum a posteriori estimate of a Gaussian process. For an excellent introduction to Gaussian processes see [Mackay, 1997].

Given a countable or uncountable set $\mathcal{X}$, a corresponding set of real valued random variables $\mathcal{G} = \{Y_x\}_{x \in \mathcal{X}}$ is said to form a **Gaussian process** if for any finite $\{x_1, x_2, \ldots, x_m\} \subset \mathcal{X}$, the random variables $Y_{x_1}, Y_{x_2}, \ldots, Y_{x_m}$ are jointly Gaussian distributed. Similarly to finite collections of jointly Gaussian distributed variables, $\mathcal{G}$ is completely specified by its mean $\mu(x) = \mathrm{E}[Y_x]$ and covariance function $k(x, x') = \mathrm{Cov}[Y_x, Y_{x'}]$. From elementary arguments in probability, k must be positive definite. Conversely, any $\mu : \mathcal{X} \mapsto \mathbb{R}$ and any symmetric and positive definite $k \colon \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$ gives rise to a corresponding Gaussian Process over $\mathcal{X}$.

Another way to regard $\mathcal{G}$ is as a probability measure over functions f in the reproducing kernel Hilbert space $\mathcal{H}$ induced by k. We now show that assuming zero mean, the distribution

$$p(f) \propto e^{-\langle f, f \rangle / 2}. \tag{4.9}$$

is equivalent to $\mathcal{G}$, in the sense that it reproduces the marginals, i.e. that for any $x_1, x_2, \ldots, x_m$ and $t = (t_1, t_2, \ldots, t_m)^\top$

$$p\left(f(x_1) = t_1 , \ldots , f(x_m) = t_m \right) = \frac{1}{(2\pi)^{m/2} \, |K|^{1/2}} \exp\left(-\frac{1}{2} \, t^\top K^{-1} \, t \right), \tag{4.10}$$

where K is the so-called **Gram matrix** with elements $K_{ij} = K(x_i, x_j)$. To this end we decompose $\mathcal{H}$ into $V = \mathrm{span}\{k_{x_1}, k_{x_2}, \ldots, k_{x_m}\}$ and its orthogonal complement V^c and note that any f obeying

$f(x_1) = t_1, \ldots, f(x_m) = t_m$ may be written as

$$f = f_V + f_\perp = \left(\sum_{i=1}^{m} \alpha_i \, k_{x_i} \right) + f_\perp$$

with $f_\perp \in V^c$. The vector of coefficients $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_m)^\top$ can be found from

$$f(x_j) = \sum_{i=1}^{m} \alpha_i \, k_{x_i}(x_j) = \sum_{i=1}^{m} \alpha_i \, k(x_i, x_j) = t_j$$

leading to the matrix equation $K\alpha = t$. We can marginalize p to V just as in the finite dimensional case by

$$p_V(f_V) = p(\, f(x_1) = t_1, \, \ldots \,) \propto \int_{V^c} p(f_V + f_\perp) \, df_\perp = e^{\, -\langle f_V, f_V \rangle / 2} \tag{4.11}$$

and expand

$$\langle f_V, f_V \rangle = \sum_{i=1}^{m} \sum_{j=1}^{m} \left[K^{-1} t \right]_i \left[K^{-1} t \right]_j \left\langle k_{x_i}, k_{x_j} \right\rangle = t^\top K^{-1} K K^{-1} t = t^\top K^{-1} t$$

proving (4.10).

In the following we make extensive use of Dirac's bra-ket notation. Functions $f \in \mathcal{H}$ will be denoted by "kets" $|f\rangle$ and members of the dual-space by the "bras" $\langle f|$. Hence the inner product becomes $\langle f, g \rangle = \langle f | g \rangle$. In contrast, linear combinations of the form $\sum_{ij} |f_i\rangle \, F_{i,j} \, \langle f_j|$ for some matrix F, or more generally, $\int \int d_x d'_x \, |f_x\rangle \, F(x, x') \, \langle f_{x'}|$ where F is now a function are bilinear operators on $\mathcal{H}$.

For inference Gaussian Processes are used as a Bayesian tool for estimating functions $f \colon \mathcal{X} \mapsto \mathbb{R}$ from observations $D = ((x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m))$ with $x_i \in \mathcal{X}$ and $y_i \in \mathbb{R}$. For notational simplicity assuming that the mean is zero, we assert a prior over functions

$$p(f) \propto e^{\, -\langle f | f \rangle / 2}$$

by specifying the kernel (covariance function) corresponding to the inner product. The noise model for the data is typically also Gaussian:

$$p(\, y \,|\, x, f \,) = \frac{1}{\sqrt{2\pi\sigma^2}} \, e^{\, -(y - f(x))^2 / (2\sigma^2)}.$$

The posterior can then be computed via Bayes rule

$$p(f|D) = \frac{p(D|f) \, p(f)}{\int p(D|f) \, p(f) \, df} \tag{4.12}$$

with $p(D|f) = \prod_{i=1}^{m} p(\, y_i |\, x_i, f \,)$. Since all distributions here are Gaussian, we need not worry about normalizing factors and we can write the negative log posterior straight away as

$$-\log p(f|D) \;=\; \frac{1}{2} \, \langle f | f \rangle + \frac{1}{2\sigma^2} \sum_{i=1}^{m} (f(x_i) - y_i)^2. \tag{4.13}$$

At this point the reproducing property of $\mathcal{H}$ becomes crucial, allowing us to write

$$-\log p(f|D) \;=\; \frac{1}{2}\langle f|f\rangle + \frac{1}{2\sigma_0^2}\sum_{i=1}^{m}(\langle f|k_{x_i}\rangle - y_i)^2 \tag{4.14}$$

and to complete the square

$$-2\log p(f|D) = \Big[\,\langle f| - \langle u|\,\hat{S}\,\Big]\;\hat{S}^{-1}\;\Big[\,|f\rangle - \hat{S}\,|u\rangle\,\Big]$$

with

$$u = \frac{1}{\sigma^2}\sum y_i\,k_{x_i} \quad \text{and} \quad \hat{S}^{-1} = I + \frac{1}{\sigma^2}\sum |k_{x_i}\rangle\,\langle k_{x_i}|\,.$$

To express $\hat{S}$, we extend k_{x_1}, k_{x_2} to a countable basis of $\mathcal{H}$ and define the matrix S^{-1} with elements

$$S^{-1}_{i,j} = \langle k_{x_i}|\,\hat{S}^{-1}\,\big|k_{x_j}\big\rangle = K(I + \tfrac{1}{\sigma^2}K).$$

This matrix can readily be inverted with ordinary matrix algebra to give another matrix $S = K^{-1}(I + \frac{1}{\sigma^2}K)^{-1}$. Writing

$$\sum_i\sum_j \langle k_{x_a}|k_{x_i}\rangle\,S_{i,j}\,\big\langle k_{x_j}\big|\,\hat{S}^{-1}\,|k_{x_b}\rangle = \sum_i \langle k_{x_a}|k_{x_i}\rangle\,[SS^{-1}]_{i,b} = \langle k_{x_a}|k_{x_b}\rangle \tag{4.15}$$

then shows that

$$\hat{S} = \sum_i\sum_j |k_{x_i}\rangle\,S_{i,j}\,\big\langle k_{x_j}\big|$$

is the correct operator inverse of $\hat{S}^{-1}$.

We can now read off the posterior mean

$$\mathrm{E}[|f\rangle] = \hat{S}\,|u\rangle = \frac{1}{\sigma^2}\sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{l=1}^{m}|k_{x_i}\rangle\,S_{i,j}\,\big\langle k_{x_j}\big|k_{x_l}\big\rangle\,y_l = \sum_{i=1}^{m}|k_{x_i}\rangle\,[(\sigma^2 I + K)^{-1}y]_i$$

and the posterior variance

$$\mathrm{Var}[|f\rangle] = \hat{S} = \sum_i\sum_j |k_{x_i}\rangle\,S_{i,j}\,\big\langle k_{x_j}\big|\,.$$

It is sometimes helpful to describe the posterior in terms of the mean and variance of $f(x)$ for fixed x. Note that this is not the whole story, though: the posterior will also have a new covariance structure (effectively a new K) which this does not shed light on. Computing the mean is easy:

$$\mathrm{E}[f(x)] = \mathrm{E}\,[\langle k_x|f\rangle] = \langle k_x|\,\mathrm{E}[|f\rangle] = \sum_{i=1}^{m}\langle k_x|k_{x_i}\rangle\,[(\sigma^2 I + K)^{-1}y]_i = k^\top(\sigma^2 I + K)^{-1}y \tag{4.16}$$

with $k = (K(x,x_1), K(x,x_2), \ldots, K(x,x_m))^\top$. Computing the variance requires somewhat more work, since now that the inner product and covariance do not match any more, marginalization is not quite as simple as applying (4.11) to a one dimensional subspace. Instead, we opt to take

advantage of the property that any finite subset of variables is jointly Gaussian distributed. In particular, the distribution over $y_x^* = (y_1, y_2, \ldots, y_m, f(x))$ is

$$p(y^*) \propto e^{-y^{*\top} \overline{K}^{-1} y^*} \tag{4.17}$$

with covariance matrix

$$\overline{K}^* = \left[\begin{array}{c|c} K + \sigma^2 I & k \\ \hline k^\top & \kappa \end{array}\right]$$

where $\kappa = K(x, x)$. Given $y_1, y_2, \ldots, y_m$, the variance of $f(x)$ can be read off from (4.17) as $\left(K^{*-1}_{m+1,m+1}\right)^{-1}$, which, using the technique of partitioned inverses can be written as

$$\mathrm{Var}[f(x)] = \left(K^{*-1}_{m+1,m+1}\right)^{-1} = \kappa - k^\top K^{-1} k. \tag{4.18}$$

From a practitioner's point of view it is significant that (4.16) and (4.18) only depend on x through k and κ. For given data $((x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m))$ only a single $m \times m$ matrix has to be inverted. Afterwards, to probe the posterior at various points is only a matter of matrix/vector multiplications.

As a learning algorithm, a Gaussian process is "trained" on $D = ((x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m))$, and predicts the label corresponding to x according to the maximum a posteriori (MAP) estimate

$$\widehat{y} = \mathrm{E}[f(x)] = k^\top (\sigma^2 I + K)^{-1} y.$$

However, since the MAP estimator of f is the minimizer of the negative log posterior (4.13), this is equivalent to solving the regularized risk minimization problem

$$f_{\mathrm{MAP}} = \arg \inf_{f \in \mathcal{F}} \left[\sum_{i=1}^{m} L(f(x_i), y_i) + \lambda \langle f | f \rangle \right]$$

with squared error loss $L(\widehat{y}, y) = (\widehat{y} - y)^2$ and regularization parameter $\lambda = \sigma^2$. This is sometimes also called **ridge regression** and provides a nice example of the way that Bayesian and frequentist methods can mesh together, forging a link between two very different learning paradigms. Thanks to this connection, considerations of regularization theory and the symmetries that we examine towards the end of this chapter also carry over to the Bayesian realm.

4.3.2 Support vector machines

The algorithm which first directed the machine learning community's interest to Hilbert space methods and which is still by far the most famous as well as most widely used kernel method is the **Support Vector Machine** (SVM). To the extent that such a thing as a universal black-box algorithm can ever exist, the SVM comes very close to being one. When Vapnik developed SVM's, his fundamental insight was the relationship between robustness and sparsity [Vapnik, 1995].

Robustness is achieved by the choice of loss function. The easiest case to develop is binary classification, where we learn a real-valued function $f \colon \mathcal{X} \to \mathbb{R}$, and predict according to its sign, $\widehat{y} = \mathrm{sgn}(f(x))$. In this case the SVM loss function is the so-called **hinge loss**

$$L(\widehat{y}, y) = (1 - \hat{y}y)_+ \tag{4.19}$$

with $(y)_+ \equiv \max(0, y)$. The idea behind the hinge loss is to not only penalize examples which are incorrectly classified (i.e., $\widehat{y} = -y$) but also examples which are correctly classified, but with not enough confidence in the sense that $\widehat{y}y < 1$. On the other hand, examples with $\widehat{y}y \geq 1$ drop out of the loss term of the regularized risk completely, and hence will feature in the kernel expansion (4.6) with zero weight.

Because the SVM is formulated in Hilbert space, $f(x) = \langle f, k_x \rangle$, and in this sense it is just a linear classifier, albeit in Hilbert space. In fact, in the limit $C \to \infty$, if the SVM has a solution at all it will guarantee a "cordon sanitaire" called the margin of width $1/\| f \|$ on either side of the separating hyperplane $\langle f, k_x \rangle = 0$. Such **large margin algorithms** are surprisingly robust, and show impressive generalization ability both in theoretical analyses and in practical applications.

We cannot possibly hope to cover here the entire arsenal of algorithms that machine learning has developed inspired by the success of the RKHS, regularized risk minimization, and large margin ideas. By the generality of the Hilbert space framework, however, the group theoretical ideas that we discuss in the following apply equally to any of these algorithms.

4.4 Symmetries

Symmetries are one of the primary motivations for studying groups in the first place. It is not surprising then that the first points of contact between Hilbert space learning methods and representation theory should be the implementation of symmetries of the input space $\mathcal{X}$. In this section we discuss the case when we have a group G acting on $\mathcal{X}$ by $x \mapsto T_g(x)$ for each $g \in G$, and we are interested in finding kernels inducing RKHS of functions which are invariant to this action in the sense that $f(T_g(x)) = f(x)$ for all $g \in G$. In the next section we discuss the case when $\mathcal{X}$ itself is a group or a homogeneous space and we want to find a kernel that is compatible with its algebraic structure.

As a motivating example consider invariant kernels between images. Image classification, object recognition, and optical character recognition have been key applications of Machine Learning from the start. Images are typically represented as $n \times n$ arrays of real numbers corresponding to pixel intensities. The key to applying one of the Hilbert space learning algorithms to this domain is defining an appropriate kernel between such arrays. The baseline choice is to treat the array as a n^2-dimensional vector, and simply use the Gaussian kernel between such vectors. Surprisingly, even such a rather ad-hoc choice of kernel can lead to reasonably good results.

The Gaussian kernel makes no attempt to account for the natural invariances of the data, however. From our prior knowledge about the nature of images we know that translating an image by a small number of pixels ought not to affect what the algorithm "perceives" the image as, even if the representation of the image as a vector in $\mathbb{R}^{n^2}$ changes drastically. Ideally, we would like to set up equivalence classes of vectors in $\mathbb{R}^{n^2}$ that represent translated versions of the same image and let our algorithm operate on these equivalence classes. Taking this idea to the extreme, consider all k horizontal cyclic translations (meaning that the image wraps around at the edges) and all n vertical cyclic translations. Clearly, the combination of the two form a group isomorphic to $G = \mathbb{Z}_n \times \mathbb{Z}_n$. By working in an RKHS where $f(T_g(x)) = f(x)$ for all translations $g \in G$, the learning algorithm is forced to treat all n^2 translated versions of a given image as the same. Effectively we have reduced the size of $\mathcal{X}$ by identifying all the vectors corresponding to translated versions of the same image. The question that we address in this section is how to modify the Gaussian RBF or some other kernel so as to achieve this.

Our starting point is a notion of translation invariance of the kernel itself. This is much weaker than the our ultimate goal, which is to make the RKHS invariant to the action of G.

Definition 4.4.1 *Let G be a group acting on the space $\mathcal{X}$ by $x \mapsto T_g(x)$, $g \in G$. We say that a positive definite kernel $k\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is invariant with respect to this action if $k(x, x') = k(T_g(x), T_g(x'))$ for all $x, x' \in \mathcal{X}$ and $g \in G$ [Krein, 1950].*

Many popular kernels possess some invariance of this form. A trivial example is the translation invariance of kernels on $\mathbb{R}^n$, amounting to $k(x+z, x'+z) = k(x, x')$. Here the group is $\mathbb{R}^n$ itself, acting by $T_z(x) = x + z$. For example, the Gaussian kernel is clearly translation invariant. A different type of invariance is invariance with respect to permuting the coordinates of x and x'. This is exactly how $\mathbb{Z}_k \times \mathbb{Z}_k$ acts on $\mathcal{X}$ in our example of translated images. It is easy to see that the (spherical) Gaussian kernel is invariant to this action as well.

Invariant RKHSs

The following two theorems connect the notions of invariant kernels and invariant RKHSs.

Theorem 4.4.2 *Let G be a group acting on the space $\mathcal{X}$ by $x \mapsto T_g(x)$ (with $g \in G$) and let k be a positive definite kernel on $\mathcal{X}$ inducing an RKHS $\mathcal{H}$ of functions f satisfying $f(x) = f(T_g(x))$ for $x \in \mathcal{X}$ and $g \in G$. Then k is invariant with respect to the $x \mapsto T_g(x)$ action.*

Proof. Defining $k_x(\cdot) = k(x, \cdot)$, by the properties of RKHSs $k_x \in \mathcal{H} \quad \forall x \in \mathcal{X}$. Hence $k(x, x') = k_x(x') = k_x(T_g(x')) = k_{T_g(x')}(x) = k_{T_g(x')}(T_g(x)) = k(T_g(x), T_g(x'))$ for any $x \in \mathcal{X}$ and $g \in G$. ■

The invariance of the kernel by itself is not sufficient to ensure the invariance of $\mathcal{H}$. We now show that to induce an invariant RKHS we also need to symmetrize the kernel. Symmetrized kernels are not new to machine learning, but to the best of our knowledge the following theorem has never been stated in full generality before. Note that G is now assumed to be a finite group.

Theorem 4.4.3 *Let G be a finite group acting on the space $\mathcal{X}$ by $x \mapsto T_g(x)$, (with $g \in G$) and let k be a positive definite kernel on $\mathcal{X}$ with RKHS $\mathcal{H}$ that is invariant to this action. Then*

$$k^G(x, x') = \frac{1}{|G|} \sum_{g \in G} k(x, T_g(x')) \tag{4.20}$$

is a positive definite function. Furthermore, the RKHS $\mathcal{H}^G$ induced by k^G is the subspace of $\mathcal{H}$ of functions satisfying $f(x) = f(T_h(x))$ for all $x \in \mathcal{X}$ and $h \in G$.

Proof. By the invariance property

$$k^G(x, x') = \frac{1}{|G|} \sum_{g \in G} k(x, T_g(x')) = \frac{1}{|G|^2} \sum_{g \in G} \sum_{h \in G} k(T_h(x), T_{hg}(x')).$$

Because G is a group, we can rewrite the above as

$$k^G(x, x') = \frac{1}{|G|^2} \sum_{h \in G} \sum_{g \in G} k(T_h(x), T_g(x')).$$

Recall that the RKHS of k is spanned by the representer functions $\{k_x = k(x,\cdot)\}_{x\in\mathcal{X}}$ and that $\langle k_x, k_{x'}\rangle_{\mathcal{H}} = k(x,x')$. To prove that k^G is positive definite all that we need is that for any $m\in\mathbb{N}$, $x_1, x_2, \ldots, x_m \in \mathcal{X}$ and $\alpha_1, \alpha_2, \ldots, \alpha_m \in \mathbb{R}$

$$\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\,\alpha_j\,k^G(x,x') = \frac{1}{|\,G\,|^2}\left\langle \sum_{i=1}^{m}\sum_{g\in G}\alpha_i\,k_{T_g(x_i)},\ \sum_{j=1}^{m}\sum_{h\in G}\alpha_j\,k_{T_h(x_j)}\right\rangle_{\mathcal{H}}$$
$$= \frac{1}{|\,G\,|^2}\left\| \sum_{i=1}^{m}\sum_{g\in G}\alpha_i\,k_{T_g(x_i)}\right\|^2_{\mathcal{H}} \geq 0.$$

Clearly, $\mathcal{H}$ can be identified with a subspace of $\mathcal{H}$. Let $\mathcal{H}^{\rm inv}$ be the subspace of $\mathcal{H}$ satisfying $f(x) = f(T_h(x))\ \ \forall x\in\mathcal{X},\ h\in G$. The functions

$$k^G_{x'}(x) = \frac{1}{|\,G\,|}\sum_{g\in G} k(x', T_g(x)) \qquad x'\in\mathcal{X}$$

spanning $\mathcal{H}^G$ satisfy this invariance, hence $\mathcal{H}^G \subset \mathcal{H}^{\rm inv}$. On the other hand, for any $f^{\rm fin}\in\mathcal{H}^{\rm inv}$ that has the additional property that it can be expressed as a finite linear combination $f^{\rm fin}(x) = \sum_{i=1}^m \alpha_i\,k(x_i,x)$,

$$f^{\rm fin}(x) = \frac{1}{|\,G\,|}\sum_{g\in G} f^{\rm fin}(T_g(x)) = \frac{1}{|\,G\,|}\sum_{g\in G}\sum_{i=1}^{m}\alpha_i\,k(x_i,T_g(x)) = \frac{1}{|\,G\,|}\sum_{g\in G}\sum_{i=1}^{m}\alpha_i\,k(T_{g^{-1}}(x_i),x),$$

showing that $f^{\rm fin}\in\mathcal{H}^G$. The space of such functions expressible as finite linear combinations is dense in $\mathcal{H}^{\rm inv}$, hence $\mathcal{H}^{\rm inv}\subset\mathcal{H}^G$. Together with the earlier result, this proves that $\mathcal{H}^G = \mathcal{H}^{\rm inv}$.

It remains to prove that for any $f_1, f_2\in\mathcal{H}^G,\ \langle f_1,f_2\rangle_{\mathcal{H}} = \langle f_1,f_2\rangle_{\mathcal{H}^G}$. Assume that f_1 and f_2 are expressible as finite linear combinations of the representers of $\mathcal{H}^G$ as $f_j(x) = \sum_{i=1}^{m_j}\alpha_i^{(1)}k^G(x_i^{(j)},x)$ for some $\{x_i^{(1)}\}_{i=1}^{m_1}$, $\{\alpha_i^{(1)}\}_{i=1}^{m_1}$, $\{x_i^{(2)}\}_{i=1}^{m_2}$ and $\{\alpha_i^{(2)}\}_{i=1}^{m_2}$. Then they are also expressible in terms of the representers of $\mathcal{H}$ as $f_j(x) = \frac{1}{|G|}\sum_{i=1}^{m_j}\sum_{g\in G}\alpha_i^{(j)}k(T_g(x_i^{(j)}),x)$. Now

$$\langle f_1,f_2\rangle_{\mathcal{H}^G} = \sum_{i=1}^{m_1}\sum_{j=1}^{m_2}\alpha_i^{(1)}\,\alpha_j^{(2)}\,k^G(x_i^{(1)},x_j^{(2)}) = \frac{1}{|\,G\,|}\sum_{i=1}^{m_1}\sum_{j=1}^{m_2}\sum_{g\in G}\alpha_i^{(1)}\,\alpha_j^{(2)}\,k(x_i^{(1)},T_g(x_j^{(2)}))$$

and

$$\langle f_1,f_2\rangle_{\mathcal{H}} = \frac{1}{|\,G\,|^2}\sum_{i=1}^{m_1}\sum_{j=1}^{m_2}\sum_{g\in G}\sum_{h\in G}\alpha_i^{(1)}\,\alpha_j^{(2)}\,k^G(T_g(x_i^{(1)}),T_h(x_j^{(2)})) =$$
$$\frac{1}{|\,G\,|}\sum_{i=1}^{m_1}\sum_{j=1}^{m_2}\sum_{g\in G}\alpha_i^{(1)}\,\alpha_j^{(2)}\,k^G(x_i^{(1)},T_g(x_j^{(2)}))$$

give the same value for the inner product. Since such linear combinations are dense in $\mathcal{H}^G$ and the evaluation functional is continuous, this result then extends to the rest of $\mathcal{H}^G$. ■

4.5 Kernels on groups and homogeneous spaces

As opposed to kernels on sets acted on by a group G, we now consider kernels on G itself which in some sense match the algebraic structure of G. Our starting point is the following definition, which is equivalent to Definition 4.4.1 when we consider G as acting on itself by $z(x) = zx$ for $x, z \in G$.

Definition 4.5.1 *Given a group G, we say that a positive (semi-)definite kernel $k\colon G \times G \to \mathbb{R}$ is **left-invariant** on G if $k(x, y) = k(zx, zy)$ for all $x, y, z \in G$.*

Now that we are working on a group, however, we have a second notion of invariance, as well.

Definition 4.5.2 *Given a group G, we say that a positive (semi-)definite kernel $k\colon G \times G \to \mathbb{R}$ is **right-invariant** on G if $k(x, y) = k(xz, yz)$ for all $x, y, z \in G$.*

If k is both left- and right-invariant, we say that it is **bi-invariant**. Naturally, on an Abelian group, there is no distinction between left- and right-invariance: any left-invariant kernel will also be right-invariant and vica versa. For example, the Gaussian kernel is a (left-, right-, bi-) invariant kernel on the group $\mathbb{R}^n$.

If k is a right-invariant kernel on a group G, then $k(x, y) = k(xy^{-1}, e)$, hence it is fully specified by $r(x) = k(x, e)$. By symmetry $r(x) = k(x, e) = k(e, x) = k(x^{-1}, e) = r(x^{-1})$. Similarly, any left-invariant kernel is specified by $r(x) = k(x, e)$. Functions $r\colon G \to \mathbb{F}$ that give rise to positive definite kernels in this way are called **positive definite functions** on G. The topic of the next subsection is characterizing the class of such functions. For now we concentrate on the algebraic properties of the kernel.

Theorem 4.5.3 *Let r be a positive definite function on a group G and let $k(x, y) = k(xy^{-1}, e) = r(xy^{-1})$ be the corresponding right invariant kernel. Then the following statements are equivalent:*

1. *k is bi-invariant;*
2. *r is a class function;*
3. *$r(xy) = r(yx)$ for any $x, y \in G$;*
4. *$k(x^{-1}, y^{-1}) = k(x, y)$.*

Proof. For (1)⇒(2) note that $r(y^{-1}xy) = k(y^{-1}xy, e) = k(x, y^{-1}y) = k(y, e) = r(x)$. For (2)⇒(3), $r(xy) = r(yxyy^{-1}) = r(yx)$. For (3)⇒(4), $k(x, y) = r(xy^{-1}) = r(y^{-1}x) = k(y^{-1}, x^{-1}) = k(x^{-1}, y^{-1})$ Finally for (4)⇒(1), $k(zx, zy) = k(x^{-1}z^{-1}, y^{-1}z^{-1}) = k(x^{-1}, y^{-1}) = k(x, y)$. ■

Once we have a kernel on G, we can define the inner product $\langle e_x, e_y \rangle = k(x, y)$ on $\mathbb{C}[G]$, and identify it with the RKHS of k by $k_x(y) = \langle e_x, e_y \rangle$. This also provides the natural way to induce kernels on homogeneous spaces of G, since xH left cosets are naturally identified with the group algebra elements

$$e_{xH} = \frac{1}{|H|} \sum_{h \in H} e_{xh}.$$

Accordingly, overloading notation, the kernel on G/H becomes

$$k(xH, x'H) = \langle e_{xH}, e_{x'H} \rangle = \frac{1}{|H|^2} \sum_{h \in H} \sum_{h' \in H} k(xh, x'h'). \tag{4.21}$$

Clearly, this is a positive definite function on G/H. If k is right invariant on G, the above simplifies to

$$k(xH, x'H) = \frac{1}{|\,H\,|} \sum_{h \in H} k(x, x'h).$$

Note the similarity to (4.20) under the substitutions $\mathcal{X} \mapsto G$, $G \mapsto H$ and $T_g(x) \mapsto hx$. The algebraic approach provides a coherent framework for establishing kernels on a variety of objects. However, at the core of it all is the theory of positive definite functions, and its correspondence to regularization.

4.5.1 Positive definite functions

As early as 1923, Mathias proved that if $f\colon \mathbb{R} \to \mathbb{R}$ is a function with Fourier transform $\widehat{f}$, then f is positive definite if and only if $\widehat{f}(x) > 0$ [Mathias, 1923]. However, results of this type are more commonly associated with the name of Bochner, who generalized Mathias's result to functions which do not necessarily have a Fourier Transform. Bochner proved that if f is a continuous positive definite function on $\mathbb{R}$, then there exists a bounded monotone increasing function V on $\mathbb{R}$ such that

$$f(x) = \int e^{ikx} dV(k)$$

[Bochner, 1932]. This result was extended to $\mathbb{R}^n$ in [Bochner, 1933]; and further generalized to measurable but not necessarily continuous functions in [Riesz, 1933].

The generalization of Bochner's theorem to locally compact Abelian groups was provided by Weil [1940], Povzner [1940] and Raikov [1940]. The irreducible representations of Abelian groups are one dimensional, hence in this case continuous characters and irreducible representations coincide. The Weil-Povzner-Raikov theorem states that any continuous positive definite function on G is expressible as

$$f(x) = \int_{\hat{G}} \chi(x)\, d\mu(\chi)$$

for some positive bounded measure μ on $\hat{G}$.

All these results show that positive definiteness is closely related to the form of the Fourier transform. It should not come as a surprise then that positive definite functions on non-commutative groups can also be characterized by their Fourier transform.

Theorem 4.5.4 *A right-invariant (resp. left-invariant) symmetric function $k\colon G \times G \to \mathbb{R}$ is a positive definite kernel on the finite group G if and only if letting $r(x) = k(x, e)$, as ρ ranges over $\mathcal{R}$, the matrices $\widehat{r}(\rho)$ are all positive definite.*

Proof. Assume without loss of generality that k is right-invariant. Then k is positive definite if and only if

$$Q = \sum_{x \in G} \sum_{y \in G} a(x)\, r(xy^{-1})\, a(y) > 0 \tag{4.22}$$

for any non-zero $a\colon G \to \mathbb{C}$. Now by the Fourier inversion formula

$$Q = \sum_{x \in G} \sum_{y \in G} a(x)\, a(y)\, \frac{1}{|\,G\,|} \sum_{\rho \in \mathcal{R}} d_\rho \operatorname{tr}\left[\widehat{r}(\rho)\, \rho(yx^{-1})\right], \tag{4.23}$$

which, by $\rho(yx^{-1}) = \rho(y)\,\rho(x^{-1})$ and $\mathrm{tr}(AB) = \mathrm{tr}(BA)$, we can also write as

$$Q = \frac{1}{|\,G\,|} \sum_{\rho \in \mathcal{R}} d_\rho \,\mathrm{tr}\left[\left(\sum_{y \in G} a(y)\rho(y) \right) \widehat{r}(\rho) \left(\sum_{x \in G} a(x)\rho(x^{-1}) \right) \right]. \tag{4.24}$$

Assuming, without loss of generality that all ρ are unitary representations,

$$Q = \frac{1}{|\,G\,|} \sum_{\rho \in \mathcal{R}} d_\rho \,\mathrm{tr}\left[\widehat{a}(\rho)^\dagger \widehat{r}(\rho) \widehat{a}(\rho) \right]. \tag{4.25}$$

Clearly, if each $\widehat{r}(\rho)$ is positive definite, then $Q > 0$. To prove the converse, note that since $\{\bigoplus_{\rho \in \mathcal{R}} \rho(x)\}_{x \in G}$ form a basis for $\bigoplus_{\rho \in \mathcal{R}} \mathbb{R}^{d_\rho \times d_\rho}$, the $\widehat{a}(\rho)$ are general matrices. Hence if $Q > 0$ for any a, then each $\widehat{r}(\rho)$ must be positive definite. ■

There is special interest in the case that k is bi-invariant. By theorem 4.5.3 in this case r is a class function, and hence, as stated in section (1.2.1), it is a sum of characters. Now from the Fourier inversion formula it is immediate that the Fourier transform of $\widehat{\chi}_\rho(\rho) = I$ and $\widehat{\chi}_\rho(\rho') = 0$ for all other irreducibles $\rho' \in \mathcal{R} \setminus \rho$. We thus have the following corollary showing that bi-invariant kernels are determined by setting $|\,\mathcal{R}\,|$ real coefficients.

Corollary 4.5.5 *A bi-invariant symmetric function $k\colon G \times G \to \mathbb{R}$ is a positive definite kernel on finite group G if and only if the Fourier transform of its associated function $r(x) = k(x, e)$ is of the form $\widehat{r}(\rho) = r_\rho I$ for some real coefficients $r_\rho \geq 0$. Furthermore, $r(x) = \sum_{\rho \in \mathcal{R}} r_\rho \chi_\rho(x)$.*

In the next chapter we will use Theorem 4.5.4 and Corollary 4.5.5 to construct kernels on the symmetric group for learning permutations, and in Chapter 6 we will find that the same results dictate the form of our noise model of matchings in multi-object tracking.

Chapter 5

Learning permutations

As a concrete example of learning on non-commutative groups, we develop the theory of learning permutations. In this scenario training data consists of a set of multi-instances $X = (x_1, x_2, \ldots, x_n)$, $x_i \in \mathcal{X}$ and associated permutations $\sigma \in \mathbb{S}_n$, with the interpretation that according to some criterion the correct way to order $x_1, x_2, \ldots, x_n$ is $x_{\sigma(1)}, x_{\sigma(2)}, \ldots, x_{\sigma(n)}$. The task of permutation learning is to learn this association between sets of elements and permutations. In testing we are presented with m' testing multi-instances and our job is to predict the corresponding $\widehat{\sigma}_1, \widehat{\sigma}_2, \ldots, \widehat{\sigma}_{m'} \in \mathbb{S}_n$.

Permutation learning is related to, but more general than **ranking**. While the exact definition of ranking varies, in general, by learning a ranking most authors mean learning a rule to predict for any x and x' in $\mathcal{X}$, whether x should be ranked higher than x' (denoted $x > x'$) or the other way round [Freund et al., 2003] [Clemençon et al., 2005] [Rudin et al., 2005]. A ranking algorithm is typically trained on a collection of ranked list $x_1 > x_2 > \ldots > x_k$ of fixed or varying lengths, or just a collection of pairwise relationships of the form $x_i > x'_i$.

Ranking is subtly different from permutation learning because it implicitly assumes that given a set $S = \{x_1, x_2, \ldots, x_k\}$, whether x_i is ranked above or below x_j is independent of the other members of S. In permutation learning we do not enforce this a priori. Consider, for example, sports tournaments, where S denotes a set of players playing together in a match. While in general whether player i finishes before player j largely just depends on their relative playing ability, their performance is also influenced by which other players are present. Hence, the relative performance of x_i and x_j depends on the entire set S.

Other permutation learning scenarios do not have any ranking character at all. One class of such problems is learning the temporal order of events. As an example, take the academic hurdles faced by Ph.D. students in a Computer Science department. Each student is required to pass a certain number of core courses, but which order she takes the courses in is her choice. After some years, patterns might start to emerge in students' choices. Which order a particular student will favor might be predictable from covariates such as her background, specialization area and so forth. However, even for fixed covariates, the induced distribution over permutations can be non-trivial. Some students might prefer starting with the easy courses and progressing to the harder ones, while others might prefer to take the opposite route. Such bi-modal distributions over permutations cannot be captured by a simple ranking model.

As far as we are aware, permutation learning has never been addressed in full generality, and certainly not in a consistent algebraic framework. However, the related problem of rank aggregation and voting schemes does have a considerable literature, both in the theoretical computer science

literature (see, e.g., [Ailon et al., 2005]) and in statistics [Critchlow, 1985][Diaconis, 1988].

5.1 Regularized Risk Minimization

In permutation learning we assume that individual instances x_i come from the input space $\mathcal{X}$, and as usual let $\mathbb{R}[\mathbb{S}_n]$ denote the group algebra of the symmetric group. Following the methodology developed in Chapter 4, we formulate our problem as that of learning a function $f\colon \mathcal{X}^n \to \mathbb{R}[\mathbb{S}_n]$ which predicts the permutation corresponding to a testing input X as $\hat{\sigma} = \arg\max_{\sigma\in\mathbb{S}_n} \langle f(X), e_\sigma\rangle$.

Given a training set $(X_1,\sigma_1),(X_2,\sigma_2),\ldots,(X_m,\sigma_m)$, where each X_i is a multi-instance $X_i = (x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)})$, we find f by minimizing the regularized risk

$$R[f] = \frac{1}{m}\sum_{i=1}^{m} L(f(X_i),\sigma_i) + \| f \|^2_{\mathcal{F}} \tag{5.1}$$

over an appropriately chosen Hilbert space $\mathcal{F}$.

The loss function L is chosen to reflect the fact that we are choosing $\hat{\sigma}$ above all other permutations. As in multiclass classification, the natural generalization of hinge loss to use in this case is

$$L(f(X_i),\sigma_i) = \max\big(0\,,\,1 - \big(f(X_i)\cdot e_{\sigma_i} - \max_{\sigma\,\in\,\mathbb{S}_n\setminus\sigma_i} f(X_i)\cdot e_\sigma\big)\big), \tag{5.2}$$

corresponding to a concept of margin

$$\min_{\sigma\neq\sigma_i}\Big[f\big(x_1^{(i)},x_2^{(i)},\ldots,x_n^{(i)},\sigma\big) - f\big(x_1^{(i)},x_2^{(i)},\ldots,x_n^{(i)},\sigma_i\big)\Big]. \tag{5.3}$$

It remains to define the Hilbert space $\mathcal{F}$ by choosing an appropriate kernel on $\mathcal{X}^n\times\mathbb{S}_n$. In accordance with the general methodology of input/output kernels we look for this in the form $k((X,\sigma),(X',\sigma'))) = k_{\mathbb{S}_n}(\sigma,\sigma')\,k_{\mathcal{X}^n}(X,X')$. The first factor in this product will be a right invariant kernel the construction of which we describe in the next section, while $k_{\mathcal{X}^n}$ is meant to capture similarity between the two ordered sets $(x_1,x_2,\ldots,x_n)$ and $(x'_1,x'_2,\ldots,x'_n)$. Since this is an essentially pairwise comparison, it is natural to define $k_{\mathcal{X}^n}$ itself as the product kernel $k_{\mathcal{X}^n}(X,X') = \prod_{i=1}^n k_{\mathcal{X}}(x_i,x'_i)$ leading to

$$k\left((x_1,\ldots,x_n,\sigma),\big(x'_1,\ldots,x'_n,\sigma'\big)\right) = k_{\mathbb{S}_n}\big(\sigma,\sigma'\big)\,\prod_{i=1}^{n} k_{\mathcal{X}}\big(x_i,x'_i\big). \tag{5.4}$$

We assume that $k_{\mathcal{X}}$ is some suitable pre-defined kernel between whatever objects we are trying to learn the permutations of.

The right invariance of $k_{\mathbb{S}_n}$ guarantees that that our algorithm will be invariant to simultaneously permuting $(x_1,x_2,\ldots,x_n)$ and $(x'_1,x'_2,\ldots,x'_n)$ by the same permutation. In the context of permutation learning this is certainly desirable. However, in permutation learning we also have an additional relabeling symmetry that the RKHS $\mathcal{F}_0$ induced by (5.4) does not satisfy. Assume that the permutation corresponding to the multi-instance $(x_1,x_2,\ldots,x_n)$ is σ. Now consider the pre-permuted multi-instance $(x_{\tau(1)},x_{\tau(2)},\ldots,x_{\tau(n)})$ for some $\tau\in\mathbb{S}_n$. To match the semantics we attach to σ, the permutation corresponding to the latter must be $\sigma\tau^{-1}$. This restricts the RKHS $\mathcal{F}$ to the subspace of functions satisfying

$$f\big(x_{\tau(1)},x_{\tau(2)},\ldots,x_{\tau(n)},\sigma\tau^{-1}\big) = f(x_1,x_2,\ldots,x_n,\sigma) \qquad \forall\,\tau\in\mathbb{S}_n. \tag{5.5}$$

Equivalently, each $f \in \mathcal{F}$ must be invariant to the action of $\mathbb{S}_n$ on $\mathcal{X}^n \times \mathbb{S}_n$ by

$$T_\tau \colon (x_1, x_2, \ldots, x_n, \sigma) \mapsto \left(x_{\tau(1)}, x_{\tau(2)}, \ldots, x_{\tau(n)}, \sigma\tau^{-1}\right) \qquad \tau \in \mathbb{S}_n. \tag{5.6}$$

According to our general result in Theorem 4.4.3, the subspace of $\mathcal{F}_0$ satisfying (5.5) is an RKHS induced by the kernel

$$k\left((x_1, \ldots, x_n, \sigma), (x'_1, \ldots, x'_n, \sigma')\right) = \frac{1}{n!} \sum_{\tau \in \mathbb{S}_n} k_{\mathbb{S}_n}(\sigma, \sigma'\tau^{-1}) \prod_{i=1}^{n} k_{\mathcal{X}}(x_i, x'_{\tau(i)}). \tag{5.7}$$

Applying (5.5) allows us to rewrite this as

$$\frac{1}{n!} \sum_{\tau \in \mathbb{S}_n} k_{\mathbb{S}_n}(e, \sigma'\tau^{-1}) \prod_{i=1}^{n} k_{\mathcal{X}}(x_{\sigma^{-1}(i)}, x'_{\tau(i)}),$$

then applying right invariance gives

$$\frac{1}{n!} \sum_{\tau \in \mathbb{S}_n} k_{\mathbb{S}_n}(\tau, \sigma') \prod_{i=1}^{n} k_{\mathcal{X}}(x_{\sigma^{-1}(i)}, x'_{\tau(i)}),$$

to which we can again apply (5.5) to get

$$\frac{1}{n!} \sum_{\tau \in \mathbb{S}_n} k_{\mathbb{S}_n}(\tau, e) \prod_{i=1}^{n} k_{\mathcal{X}}(x_{\sigma^{-1}(i)}, x'_{\tau\sigma'^{-1}(i)}).$$

Thus, (5.7) is equivalent to

$$k\left((x_1, \ldots, x_n, \sigma), (x'_1, \ldots, x'_n, \sigma')\right) = \frac{1}{n!} \sum_{\tau \in \mathbb{S}_n} k_{\mathbb{S}_n}(\tau, e) \prod_{i=1}^{n} M_{i,\tau(i)}, \tag{5.8}$$

where M is the matrix $M_{i,j} = k_{\mathcal{X}}(x_{\sigma^{-1}(i)}, x_{\sigma'^{-1}(i)})$. This is the form of the kernel that we employ in permutation learning, and because of the summation over $n!$ terms, it immediately presents us with a computational challenge. We address this issue in Section 5.3.

5.2 Diffusion Kernels

While Section 4.5 provided a general characterization of invariant positive definite kernels on groups, it did not address the question of what kernel exactly is appropriate to use in specific learning problems. Beyond just algebra, answering this question requires introducing knowledge about the way the structure of the group is reflected in the data.

Section 4.2.3 suggests that one of the best ways of inducing a regularization scheme by a kernel is to start from the differential properties of the underlying space. While on a discrete space, especially an abstract one such as a finite group, we might not have the same concepts of partial derivatives as on $\mathbb{R}^n$, some aspects of our discussion in 4.2.3 are so fundamental that they have natural analogs.

In particular, the diffusion kernels formalism [Kondor and Lafferty, 2002][Smola and Kondor, 2003] reveals that the Gaussian kernel is just a special case of the more general concept of kernels

induced by local similarities captured by a Laplacian. Whereas (4.8) expresses the regularization operator corresponding to the Gaussian kernel in terms of differential operators on $\mathbb{R}^n$, when working on $\mathbb{S}_n$ or any other finite group G, we need a discrete notion of neighborliness. In effect, this means turning G into an undirected graph. Letting $x \sim y$ denote the fact that $x, y \in G$ are neighbors, and letting d_x denote the number of edges incident on x, Kondor and Lafferty [2002] show that if we define the **graph Laplacian** as the $|G| \times |G|$ matrix

$$\Delta_{x,y} = \begin{cases} 1 & \text{if} \quad x \sim y \\ -d_x & \text{if} \quad x = y \\ 0 & \text{otherwise}\,, \end{cases}$$

then the **diffusion kernel**

$$k(x,y) = \left[e^{\beta\Delta}\right]_{x,y} = \left[\lim_{n\to\infty}\left(1 + \frac{\beta\Delta}{n}\right)^n\right]_{x,y} \tag{5.9}$$

is not only positive definite, but also an attractive and natural way to capture similarity and regularize functions.

In particular, regarding Δ as an operator on functions on G by $(\Delta f)(x) = \sum_{y\in G} \Delta_{x,y} f(y)$, we have

$$\langle f, \Delta f\rangle = -\frac{1}{2}\sum_{x\sim y}(f(x) - f(y))^2$$

suggesting that $-\Delta$ in some sense captures the extent to which f violates the graph structure. If $v_1, v_2, \ldots, v_{|G|}$ are the eigenvectors of Δ with corresponding eigenvalues $\nu_1, \nu_2, \ldots, \nu_{|G|}$, then the kernel operator may be expressed in the form

$$\mathcal{K} = \sum_{i=1}^{|G|} e^{\beta\nu_i}\, v_i\, v_i^\top,$$

and the regularization operator is

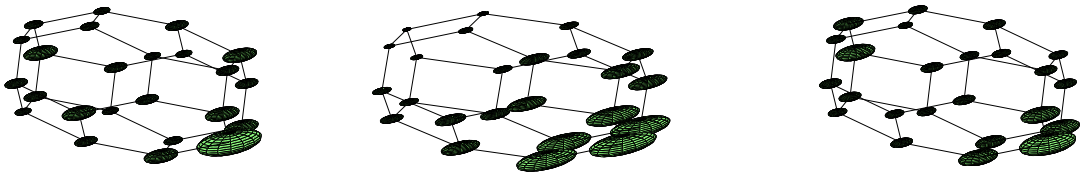
$$\Upsilon = \sum_{i=1}^{|G|} e^{-\beta\nu_i/2}\, v_i\, v_i^\top.$$

This is the exact discrete analog of (4.8), showing that the diffusion kernel penalizes functions according to how much any energy they have in the "high frequency" modes on the graph, i.e., the ones which violate many edges.

There is also a more mechanistic, intuitive analogy between diffusion on continuous spaces (the solution to which is the Gaussian kernel) and diffusion on graphs, which we shall not go into here. The reader is referred to [Kondor and Lafferty, 2002] for details.

The most natural graphs on finite groups are the so called **Cayley graphs**. Such graphs are induced by a set $W \subset G$, and the rule defining the graph topology is that two group elements x and y are neighbors if and only if $y = zx$ for some $z \in W$. We shall assume that W is a generating set for G, since otherwise the induced Cayley graph would not be connected.

The graph Laplacian of a Cayley graph is right invariant in the sense that $\Delta_{x,y} = \Delta_{xz,yz}$, and is thus characterized by a function $u(xy^{-1}) = \Delta_{x,y}$. Matrices that have this type of symmetry are called G-**circulants**. As the following theorem shows, Cayley graphs induce right-invariant diffusion kernels, so the results of Section 4.5 apply.



(a) (b) (c)

Figure 5.1: Three diffusion kernels on the permutation polytope of $\mathbb{S}_4$: (a) k_{transp} with $\beta = 0.5$; (b) k_{adj} with $\beta = 2.0$; and (c) k_{cyc} with $\beta = 1$. The size of the sphere at each vertex corresponds to $k(e, \sigma)$.

Theorem 5.2.1 *Let Δ be a G-circulant matrix with $\Delta_{x,y} = u(xy^{-1})$. Then $K = e^{\beta\Delta}$ is a G-circulant matrix $K_{x,y} = r(y^{-1}x)$ with $\widehat{r}(\rho) = \exp(\beta\,\widehat{u}(\rho))$ for $\rho \in \mathcal{R}$.*

Proof. In terms of the regular representation ρ^{reg} of G, Δ can be written in the form $\Delta = \sum_{z\in G} u(z)\,\rho^{\text{reg}}(z)$, where

$$[\rho^{\text{reg}}(z)]_{x,y} = \begin{cases} 1 & \text{if } \; x = zy \\ 0 & \text{otherwise}\,. \end{cases}$$

By total reducibility, for some invertible matrix T, Δ decomposes in the form

$$T^{-1}\Delta T = \bigoplus_{\rho} \sum_{z\in G} u(z)\,\rho(z),$$

where the direct sum runs over irreducible representations of G with possible repeats. By Corollary 1, section 2.4 of Serre [1977], each $\rho \in \mathcal{R}$ is featured in this sum with multiplicity at least one. We recognize each inner sum as the Fourier Transform of u at the irreducible representation ρ. Now $\exp\left(\beta\,T^{-1}\Delta T\right) = T^{-1}\exp(\beta\Delta)T$, hence $T^{-1}KT = \bigoplus_\rho \exp(\beta\,\widehat{u}(\rho)) = \bigoplus_\rho \widehat{r}(\rho)$. ■

In the special case that W is a union of conjugacy classes, u is a class function, hence it can be written as a weighted sum of characters, and its Fourier transform will consist of diagonal matrices. Thus by Corollary 4.5.5 we have the following result.

Corollary 5.2.2 *The diffusion kernel on the Cayley graph of a finite group G induced by a generating set W is bi-invariant if and only if W is a weighted sum of conjugacy classes.*

Three kernels on $\mathbb{S}_n$

It is natural to regard two permutations $\sigma_1, \sigma_2 \in \mathbb{S}_n$ as "neighbors" if they only differ by a transposition, $\sigma_2 = \sigma_1 \cdot (i, j)$. This motivates looking at diffusion kernels induced by one of the following three generating sets:

1. $\mathcal{T}_{\text{transp}}$, the set of all transpositions;
2. $\mathcal{T}_{\text{adj}}$, the set of adjacent transpositions;

id	partition
A	(4)
B	(3, 1)
C	(2, 2)
D	(2, 1, 1)
E	(1, 1, 1, 1)

	E	D	C	B	A
A	1	1	1	1	1
B	3	1	-1	0	-1
C	2	0	2	-1	0
D	3	-1	-1	0	1
E	1	-1	1	1	-1

id	partition
A	(5)
B	(4, 1)
C	(3, 2)
D	(3, 1, 1)
E	(2, 2, 1)
F	(2, 1, 1, 1)
G	(1, 1, 1, 1, 1)

	G	F	E	D	C	B	A
A	1	1	1	1	1	1	1
B	4	2	0	1	-1	0	-1
C	5	1	1	-1	1	-1	0
D	6	0	-2	0	0	0	1
E	5	-1	1	-1	-1	1	0
F	4	-2	0	1	1	0	-1
G	1	-1	1	1	-1	-1	1

Table 5.1: The character tables of $\mathbb{S}_4$ and $\mathbb{S}_5$. The right hand tables show $\chi_\lambda(\mu)$, λ indexing the rows and μ the columns according to the scheme laid out in the left hand tables. Reproduced from [James and Kerber, 1981].

3. $\mathcal{T}_{\text{cyc}} = \mathcal{T}_{\text{adj}} \cup \{[1, n]\}$, the set of cyclically adjacent transpositions.

Figure 5.1 shows the corresponding kernels, labeled k_{transp}, k_{adj} and k_{cyc}, on the permutation polytope of $\mathbb{S}_4$. The permutation polytope of $\mathbb{S}_n$ is the convex hull of the points $((\sigma(1), \sigma(2), \ldots, \sigma(n)))_{\sigma \in \mathbb{S}_n}$. This is an $n-1$-dimensional polygon contained in the hyperplane $x_1 + x_2 + \ldots + x_n = n(n+1)/2$, hence the largest symmetric group for which it can be plotted in three dimensions is $\mathbb{S}_4$. The edges of the permutation polytope connect permutations that differ by a single adjacent transposition.

From a purely algebraic point of view, k_{transp} is by far the most natural choice. Indeed, one sense in which the symmetric group is "symmetric" is that it treats the objects $1, 2, \ldots, n$ that it operates on on an exactly equal footing, having no sense of which object is adjacent to which others. Unlike $\mathcal{T}_{\text{adj}}$ and $\mathcal{T}_{\text{cyc}}$, $\mathcal{T}_{\text{transp}}$ is a conjugacy class, so by Corollary 5.2.2, k_{transp} is bi-invariant, leading to several attractive properties.

One potential problem with k_{transp} is that the diameter of $\mathbb{S}_n$ in terms of transpositions is relatively small. The corresponding graph distance on $\mathbb{S}_n$ is called the Cayley distance. It is easy to see that if the cycle type of $\sigma_1^{-1}\sigma_2$ is λ , then $d_{\text{Cayley}}(\sigma_1, \sigma_2) = \sum_i (\lambda_i - 1)$. Hence, the maximum distance on G is $d_{\text{Cayley}}(e, [1, 2, \ldots, n]) = n - 1$. Fixing σ_1, many other permutations will be the same distance from σ_1, limiting the discriminative power of this kernel.

Adjacent transpositions are a natural metric to use between rankings. Clearly, exchanging adjacent items in a ranking is less of a change than exchanging more distant items. Finally, k_{cyc} is a variation on k_{adj} for problems with cyclic symmetry. An example would be the seating arrangement of people around a circular table.

The kernels k_{adj} and k_{cyc} are best computed by explicit matrix exponentiation or Fourier transformation. In contrast, by virtue of bi-invariance, k_{transp} can be expressed in terms of characters. Since k_{transp} is induced by the indicator function of transpositions, which have cycle type $(2, 1, 1, \ldots)$ by Corollary 5.2.2, have the explicit formula

$$k_{\text{transp}}(\sigma_1, \sigma_2) = \frac{1}{|G|} \sum_{\rho \in \mathcal{R}} d_\rho \exp\left(\frac{\beta n(n-1)}{d_\rho} \chi_\rho((2, 1, 1, \ldots))\right) \chi_\rho(\mu), \tag{5.10}$$

where μ is the cycle type of $\sigma_2^{-1}\sigma_1$.

The characters may simply be looked up in a table, [see, e.g., James and Kerber, 1981]. By way of example, the character tables of $\mathbb{S}_4$ and $\mathbb{S}_5$ are given in Table 5.1. Alternatively, the characters may be computed using a general theorem called the Murnaghan-Nakayama rule (1.8). For the special case of $\chi_\rho((2, 1, 1, \ldots))$ a short calculation gives the following particularly simple result.

Lemma 5.2.3 *Let p_ρ be the number of standard tableaux of shape ρ where $\boxed{2}$ is to the right of the $\boxed{1}$, and let q_ρ be the number of standard tableaux where it is below it. Then $\chi_\rho((2,1,1,\ldots)) = p_\rho - q_\rho$.*

5.3 Immanants and the PerMceptron

The complexity of permutation learning is determined by the number of training examples m, and the "degree" of the examples, n, determining which symmetric group we are working on. While m may be in the thousands or tens of thousands, n is severely limited by the fact that both the evaluation time of the symmetrized kernel (5.7) and the number of constraints in (5.3) scale with $n!$. In this section we describe two computational tricks that ease this burden to some extent.

The first of these applies to the case when $k_{\mathbb{S}_n}$ is bi-invariant. Recall that in this case $r(x) = k(x,e)$ may be expressed as a weighted sum of characters $r(x) = \sum_{\rho\in\mathcal{R}} r_\rho \chi_\rho(x)$, thus (5.8) becomes

$$k\left((X,\sigma),\left(X',\sigma'\right)\right) = \frac{1}{n!}\sum_{\lambda\vdash n} r_\lambda \sum_{\tau\in\mathbb{S}_n} \chi_\lambda(\tau)\prod_{i=1}^{n} M_{i,\tau(i)}. \tag{5.11}$$

The inner summation appearing in this expression is called the **λ-immanant** of M and denoted $\mathrm{Im}_\lambda(M)$. The two well known examples of immanants are the determinant, which corresponds to the alternating character of $\mathbb{S}_n$, and the permanent, which corresponds to the constant character.

The cost of computing immanants varies greatly with λ. The determinant can be computed in time $O(n^3)$ by matrix diagonalization, while the best currently known exact algorithm for computing the permanent is due to Ryser [1963] and runs in $O(n2^n)$ time. Recently, Jerrum et al. [2001] have proposed a polynomial time approximation algorithm for computing the permanent of a non-negative matrix based on Markov Chain Monte Carlo sampling.

Remarkably, immanants are closely related to representations of the general linear group. In particular, for $M \in \mathrm{GL}(n)$,

$$\mathrm{Im}_\lambda(M) = \sum_t [\, D_\lambda(M)\,]_{t,t}\,,$$

where D_λ is now the irreducible representation of $\mathrm{GL}(\mathbb{C}^n)$ of shape λ, and the sum runs over those semi-standard tableaux of shape λ which are also standard tableaux [Bürgisser, 2000, Lemma 4]. This reduction is significant because Bürgisser has recently presented an $O(n^2(\mathrm{mult}(\lambda)+\log n)s_\lambda d_\lambda)$ algorithm for computing $D_\lambda(M)$, where s_λ and d_λ are respectively the number of semi-standard and standard tableaux of shape λ, and $\mathrm{mult}(\lambda)$ is the maximum multiplicity of any irreducible representation of $\mathrm{GL}(\mathbb{C}^{n-2})$ in the decomposition of $D_\lambda \downarrow_{\mathrm{GL}(\mathbb{C}^{n-2})}$. This improves significantly on the complexity of computing symmetrized kernels, especially when the coefficients α_λ drop off fast enough to allow us to cut off the expansion of $k_{\mathbb{S}_n}(\sigma,e)$ after a relatively small number of characters.

5.3.1 The "PerMceptron"

Carrying out a full QP optimization to solve (5.1) is often prohibitively expensive. The size of the problem is $m \cdot n!$, and using a loss function such as (5.2), for any given training example (X_i,σ_i) we can expect many α_σ^i coefficients in the kernel expansion

$$f(X,\sigma) = \sum_{i=1}^{m}\sum_{\sigma\in\mathbb{S}_n} \alpha_\sigma^i\, k((X_i,\sigma_i),(X,\sigma)) \tag{5.12}$$

to be non-zero, i.e., the solution is not going to be sparse. In addition, in the general case when we cannot use the immanants trick, each kernel evaluation (5.7) takes $O(n!)$ time. We propose to circumvent these computational limitations by essentially converting our problem to the on-line setting.

It is well-known that the venerable perceptron can not only be "kernelized" but in a certain sense also approximates the solution of batch large-margin algorithms [Freund and Schapire, 1999]. Recently, the Perceptron has been enjoying renewed attention, in particular, various extensions to the multi-class setting have been proposed [Crammer and Singer, 2003]. For permutation learning the following very simple variant is already a viable learning algorithm:

```
Let  f ← 0;
For each training example (X_i, σ_i) do {
  Predict σ̂ = arg max_{σ∈S_n} f(X_i, σ);
  If σ̂ ≠ σ_i
     then f ← f + k((X_i, σ_i), · ) − k((X_i, σ̂), · ) ;
     else do nothing ;
}
```

The current hypothesis f can be stored in the form of an expansion

$$f(X,\sigma) = \sum_{j=1} \sum_{\sigma \in \mathbb{S}_n} \alpha_j \, k((X_j,\sigma_j),(X,\sigma)) \tag{5.13}$$

similar to (5.12). A perceptron may cycle through the training data multiple times, these are called training epochs. It is also possible to test the final hypothesis on an independent testing set, so the perceptron can function as a batch algorithms as well. The advantages over SVM-type algorithms are that sparsity is guaranteed a forteriori and no expensive QP needs to be solved.

However, at each step of the algorithm, to compute the arg max involved in prediction, $h_i(\sigma) = f(X_i,\sigma)$ must be evaluated for each $\sigma \in \mathbb{S}_n$. Recalling that each kernel evaluation takes time $O(n!)$, this puts us back in the $O((n!)^2)$ regime. Once more, the algebraic properties of $\mathbb{S}_n$ come to the rescue. The crucial observation is that plugging in (5.8) to (5.13) gives

$$h_i(\sigma) = f(X_i,\sigma) = \frac{1}{n!} \sum_j \alpha_j \sum_{\tau \in \mathbb{S}_n} k_{\mathbb{S}_n}(\tau, e) \prod_{l=1}^{n} M^{(j)}_{l,\tau(l)}, \qquad [M^{(j)}]_{a,b} = k_{\mathcal{X}}(x^{(i)}_{\sigma^{-1}(a)}, x^{(j)}_{\sigma_j^{-1}(b)}),$$

and assuming right invariance, $k_{\mathbb{S}_n}(\sigma,\sigma') = r(\sigma\sigma'^{-1})$, and using $r(\sigma) = r(\sigma^{-1})$ this is just a convolution

$$h_i(\sigma) = \frac{1}{n!} \sum_{\tau \in \mathbb{S}_n} r(\sigma\tau^{-1}) \, g(\tau) = (r * g)(\sigma) \tag{5.14}$$

of r with $g(\tau) = \sum_j \alpha_j \prod_{l=1}^{n} M^{(j)}_{l,\tau(l)}$. Convolution can be computed fast by transforming to Fourier space, multiplying the appropriate components, and transforming back. Using Clausen's FFT from Section 3.3, this reduces the complexity of a single perceptron update step from $O(n!^2)$ to $O(n^3 n!)$. In keeping with the tradition of giving ranking-related algorithms silly names such as **pranking** [Crammer and Singer, 2002] and **cranking** [Lebanon and Lafferty, 2002], we call the resulting fast algorithm the **perMceptron**. To our knowledge the Permceptron is unique amongst kernel-based learning algorithms in that for computational efficiency it does not compute kernel values individually, but in batch.

Chapter 6

Multi-object tracking

Multi-object tracking systems associate tracks $r_1, r_2, \ldots, r_n$ with real world objects $o_1, o_2, \ldots, o_n$ called targets. When the objects are well separated and good quality observations are available, following which track corresponds to which target is relatively easy. However, when two objects come very close, are occluded, or observations are not available, the association between tracks and objects becomes uncertain. This is what is referred to as the identity management or data association problem in multi-object tracking.

Most tracking systems in practical use today are probabilistic, in the sense that at time t, each $r_i(t)$ stands for a probability distribution describing our belief of where the corresponding object is likely to be in space. It is then natural to address the identity management problem in a similarly probabilistic fashion, by maintaining a probability distribution p over the $n!$ possible matchings between objects and tracks. Equivalently, we may regard p as a probability distribution over permutations of the objects $o_1, o_2, \ldots, o_n$. Missing observations, occlusion, random noise, etc. tend to smear out $p(\sigma)$, while positively observing some target at a particular track makes the distribution more concentrated again. The central challenge in maintaining p is the factorial blow-up in the number of permutations.

Multi-object tracking has many applications in vision, security, control, observational problems in the natural sciences and even in sports [Bar-Shalom and Formann, 1988][Stone et al., 2000]. The method we propose is particularly well suited to problems where location information is relatively abundant, but identity information is scarce. A typical example is air traffic control, where aircraft are more or less continuously visible on radar, but in the absence of transponders their identity is only revealed when the pilot reports by radio. A similar application is tracking animals released into the wild. Information about the animals might include sightings, or indirect evidence for their presence, such as killed prey, but only a small fraction of these observations reveal the identity of the specific animal. The task is to extrapolate to the rest of the observations and reconstruct the paths of individual animals. The identity management problem is also critical in sensor networks, such as "wired" office buildings. Security cameras, motion detectors, or even light switches can all detect the presence of people, but positively identifying an individual person is only possible when he or she comes in contact with potentially more intrusive sensors, such as RFID transmitters, or an identity card reader.

Sensor networks are the motivating application for recent papers on the information-form data association filter [Schumitsch et al., 2005][Shin et al., 2006], which is perhaps closest in spirit to our approach. In particular, the combination of identity observations and a continuous noise process

operating in the background is similar to our framework. However, in a certain sense the two approaches are complementary: our preliminary experiments indicate that the data association filter performs better when the association distribution is concentrated on a small set of permutations, while our method, based on smoothness arguments, performs better when the distribution is more vague. The data association filter scales very favorably with n, but it is limited to an n^2 dimensional projection of the distribution over matchings. In contrast, we can choose from a whole spectrum of representations at different levels of complexity, but the interesting ones scale worse and hence limit us to about $n \leq 30$. It is possible that in the future the two approaches can be fused, combining their respective strengths.

The present chapter is based on [Kondor et al., 2007] and all the results presented in the following pages are original material. After this thesis was defended, Huang et al. [2008] have published a paper that takes this work even farther.

6.1 A band-limited representation

An online multi-object tracking system following n targets needs to maintain n separate distributions p_i describing the spatial location of each track, and a single distribution p over the $n!$ possible matchings of targets to tracks. In real-world systems these two types of approximation problems are closely coupled. For now, however, we focus purely on the identity management side.

Matchings between targets and tracks are readily identified with permutations $\sigma \in \mathbb{S}_n$, if we let $\sigma(i) = j$ denote that target i is located at track r_j. The identity management problem then becomes an online estimation problem on $\mathbb{S}_n$. Assuming that the system starts out in a fully observed state (say, without loss of generality, each target i is at track r_i), our job is to design an algorithm which maintains an internal representation of the association distribution updated by (a) observations linking targets to particular tracks; (b) the effects of some random noise process modeling our increasing uncertainty in-between observations. The simplest way of doing this is to just maintain a $n \times n$ matrix describing the probabilities of finding object i at track j. Such an approach, however, vastly oversimplifies the problem, since it ignores all higher order interactions between the targets.

Assume, for example that we have just three targets: A, B and C, and corresponding tracks labeled r_1, r_2 and r_3. Consider first what happens if we find that A and B approach each other so closely that we are no longer able to tell which one of them is associated with track r_1 and which one is associated with track r_2. In this case p must reflect that all information about the relative associations of A and B have been lost. Even if A and B (and hence, tracks r_1 and r_2) subsequently separate, we will not know whether track r_1 is now following A and track r_2 is following B or the other way around. To make things even worse, now imagine that a short time later the same thing happens to C and whichever object is at track r_2. Now we can't be certain about the locations of any of the targets. However, receiving observations can disambiguate the situation.

For example, if we observe C at track r_3, we are back to the situation we had before the second encounter: we know where C is, but we have complete uncertainty about the relative association of A and B to tracks r_1 and r_2. On the other hand, observing B at track r_3 not only disambiguates the association of that target, but also allows us to infer that C must be at track r_2 and A at track r_1. It is this type of inference which is not possible without maintaining a rich, ideally exact, representation of p.

Unfortunately, due to the extremely rapid growth of the factorial function, for n greater than

about 10 or 11, maintaining an exact representation of p in practical systems would be prohibitively expensive. Inspired by the theory of non-commutative harmonic analysis we developed in Chapters 2 and 5, our novel way of addressing this problem will be to project p from $\mathbb{R}^{|n!|}$ to a carefully chosen subspace W satisfying at least the following criteria:

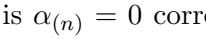
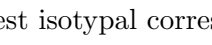
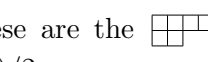
1. It should be possible to reconstruct p from its projection p_W with minimal loss of information.
2. The projection must be symmetric in the sense that each component of $p(\sigma)$ suffers the same loss when projected to W.
3. It should be possible to apply observations and noise updates directly to p_W, without reconstructing the full distribution p.

Unsurprisingly, the optimal choice of W turns out to be a sum of isotypals (see Section 2.2). We argue that a good projection of p is one which is relatively insensitive to noise: we want to retain the robust "low frequency" components, while we are not too concerned about throwing out the more noisy "high frequency" part of p. Whatever the exact form of the noise process is, in tracking it is almost always expressible as pairwise exchanges of targets. In the absence of factors breaking the symmetry, the probability of an exchange is the same for any pair of targets. In other words, the noise process is just a random walk on the Cayley graph or transpositions, bringing us back to the framework of Section 5.2.

Since transpositions are a conjugacy class, the Fourier transform of Δ will be of the form $\widehat{\Delta}(\lambda) = \alpha_\lambda I$, and the eigenspectrum of Δ will be highly degenerate: each partition λ will have d_λ orthogonal eigenvectors collectively spanning the λ-isotypal. The actual α_λ values are given by the formula

$$\alpha_\lambda = \binom{n}{2}\left(1 - \frac{\chi_\lambda((2,1,1,\ldots))}{d_\lambda}\right). \tag{6.1}$$

Our band-limited representation will consist of a projection of p onto the isotypals with low α_λ. In general, these follow the following sequence:

1. The lowest α is $\alpha_{(n)} = 0$ corresponding to the  isotypal which is one dimensional and is associated with the trivial representation.
2. The next lowest isotypal corresponds to  and is $n-1$ dimensional.
3. Following these are the  and isotypals of dimensionality $n(n-3)/2$ and $(n-1)(n-2)/2$.
4. The , and isotypals of dimensionality $n(n-1)(n-5)/6$, $n(n-2)(n-4)/3$ and $\binom{n-1}{3}$.
5. etc.

For concreteness in drawing the above diagrams we assumed $n = 8$. Depending on how much computational time we can afford, we typically project to the isotypals in the first two, first three or first four sets. Since $\rho_{(n)}$ and $\rho_{(n-1,1)}$ together make up the defining representation, maintaining just the and isotypals corresponds to the traditional method of approximating p by its marginals $p_{i,j} = \mathbb{P}[p(i) = j]$.

6.1.1 Noise model

We model the uncertainty seeping into p by assuming that at each time t with some small probability β the targets corresponding to tracks i and j get exchanged, and this probability β is independent of i and j. If previously the correct assignment was σ, then the new assignment will be $(i,j)\,\sigma$. Considering all possible exchanges, the corresponding transition matrix is $I-\beta\Delta$, where Δ is again the Laplacian. Since this noise process is assumed to operate in the background in a continuous manner, we take the limiting case of infintesimally small time intervals, leading to the update equation $p_{t'} = \exp\left(-\beta\,(t'-t)\,\Delta\right) p_t$, where

$$e^{-\beta(t'-t)\Delta} := \lim_{k\to\infty}\left(I - \frac{\beta\,(t'-t)}{k}\,\Delta\right)^k \tag{6.2}$$

is the matrix exponential. This, however, is just a diffusion process on the Cayley graph [Kondor and Lafferty, 2002]. In particular, the Fourier space updates of p take on the particularly simple form

$$\widehat{p}_{t'}(\rho_\lambda) = e^{-\beta\alpha_\lambda(t'-t)}\;\widehat{p}_t(\rho_\lambda).$$

In our bandwidth-limited Fourier space scheme this update equation is not only very efficient to implement, but it is also exact: diffusion does not take us outside the subspace of distributions restricted to W.

6.1.2 Observations

The simplest type of observations consist of receiving information that with probability π, target o_i is at track r_j:

$$p\left(\,O_{i\to j}\,|\,\sigma\,\right) = \begin{cases} \pi & \text{if } \;\sigma(i)=j, \\ (1-\pi)/(n-1) & \text{if } \;\sigma(i)\neq j. \end{cases} \tag{6.3}$$

Fixing j, $p\left(\,O_{i\to j}\,|\,\sigma\,\right)$ is to be regarded as a probability distribution over which target i is observed at track j. The posterior over permutations is given by Bayes' rule:

$$p\left(\,\sigma\,|\,O_{i\to j}\,\right) = \frac{p\left(\,O_{i\to j}\,|\,\sigma\,\right)\;p(\sigma)}{\sum_{\sigma'\in\mathbb{S}_n} p\left(\,O_{i\to j}\,|\,\sigma'\,\right)\;p(\sigma')}\,, \tag{6.4}$$

giving rise to the update rule

$$p'(\sigma) = \begin{cases} \frac{1}{Z}\,\pi\cdot p(\sigma) & \text{for } \;\sigma(i)=j, \\ \frac{1}{Z}\,\frac{1-\pi}{n-1}\,p(\sigma) & \text{for } \;\sigma(i)\neq j, \end{cases} \tag{6.5}$$

where Z is the normalizer $Z = \pi\cdot p([\sigma(i)=j]) + \frac{1-\pi}{n-1}\cdot(1-p([\sigma(i)\neq j]))$. In summary, observation updates hinge on (a) marginalizing p over sets of the form $C_{i\to j} = \{\,\sigma\in\mathbb{S}_n \mid \sigma(i)=j\,\}$ and (b) rescaling the projections of p onto subspaces corresponding to $C_{i\to j}$ and $C_{i\to j'},\;j'\neq j$.

A naive implementation of this update requiring explicit enumeration of group elements would run in time $O(n!)$. The next section will present a much faster algorithm. The observation updates couple the different Fourier components. This means when restricted to W the update step can only be approximate: an exact update would push some energy into some of the "higher frequency" Fourier components which we do not maintain in our bandwidth-limited representation of p. This is the only source of approximation error in our framework.

6.1.3 Inference

The third component to our identity management system is a procedure for reading off the predicted association between targets and tracks. In the simplest case this consists of returning the marginal probabilities $p([\sigma(i)=j])$ for a fixed object i as the track variable j ranges over $1, 2, \ldots, n$, or for a fixed track j as i ranges over $1, 2, \ldots, n$.

A more challenging task is to find the optimal association $\tilde{\sigma} = \arg\max_{\sigma \in \mathbb{S}_n} p(\sigma)$. Again, the techniques presented in the next section lead to a search algorithm which can beat the apparent $O(n!)$ complexity of this operation.

6.2 Efficient computation

We generalize Clausen's FFT by modifying (3.10) to

$$\widehat{f}(\lambda) = \sum_{j=1}^{n} \rho([\![j,n]\!]) \left[\bigoplus_{\lambda^-} \widehat{f}_{i\to j}(\rho_{\lambda^-})\right] \rho([\![i,n]\!]^{-1}),$$

where $1 \le i \le n$, and $f_{i\to j}(\sigma') = f([\![j,n]\!]\,\sigma'\,[\![i,n]\!]^{-1})$, leading to what we call "twisted" FFTs. The connection to the identity management problem is that the two-sided coset $[\![j,n]\!]\,\mathbb{S}_{n-1}\,[\![i,n]\!]^{-1}$ to which $f_{i\to j}$ corresponds is exactly the set $C_{i\to j}$ introduced in 6.1.2. In particular, since $\rho_{(n-1)}$ is the trivial representation of $\mathbb{S}_{n-1}$, the Fourier component $\widehat{p}_{i\to j}(\rho_{n-1})$ is equal to the marginal $p([\sigma(i)=j])$.

Our Bayesian update step can then be performed by transforming to the $\widehat{p}_{i\to j}(\rho_{\lambda^-})$ matrices from the original $\widehat{p}(\rho_\lambda)$, reading off $\widehat{p}_{i\to j}(\rho_{(n-1)})$, rescaling the $\widehat{p}_{i\to j}$ according to (6.5), and then transforming back to $\widehat{p}$. For band-limited functions all these computations can be restricted to the subtree of the Bratelli diagram leading to those Fourier components $\widehat{p}(\rho_\lambda)$ which we actually maintain, and the total complexity of the update step becomes $O(D^2 n)$. This algorithmic viewpoint of information passing back and forth between the $\widehat{p}(\rho_\lambda)$ matrices and the $\widehat{p}_{i\to j}(\rho_{\lambda^-})$ matrices also sheds light on the nature of the coupling between the Fourier components: this arises because in the Bratelli diagram each partition of $n-1$ is linked to multiple partitions of n.

As regards the inference step, our proposed solution is a "branch and bound" type search over the cosets that the FFT successively decomposes $\mathbb{S}_n$ into (Figure 6.1). The algorithm is based on the fact that the total energy $\| p(\sigma\mathbb{S}_k) \|^2$ in any $\mathbb{S}_k$-coset is the sum of the total energies of the constituent $\mathbb{S}_{k-1}$-cosets, and that by unitarity this energy is fast to compute. We cannot give a theoretical complexity guarantee for this algorithm, but in practical applications it is expected to be much faster than $O(n!)$.

6.2.1 Performance

In summary, our computational strategy is based on carefully avoiding complete Fourier transforms or any other $O(n!)$ computations. This makes our framework viable up to about $n = 30$. To indicate performance, on a 2.6 GHz PC, a single observation update takes 59ms for $n = 15$ when W is spanned by the first 4 Fourier components, and 3s when maintaining the first 7 components. For $n = 30$ and 4 components an update takes approximately 3s. Recall that the third and fourth Fourier components have $O(n^4)$ scalar components. When storing and updating this many variables is no longer feasible, we are reduced to the just the first two components with 1 and $(n-1)^2$ scalar

```
find_largest_in_subgroup(σ𝕊_k){
  if k = 1 {
    if ‖ p(σ) ‖² > ‖ p(σ̃) ‖² { σ̃ ← σ; }
  }
  else {
    order {η_1, η_2, ..., η_k} so that ‖ p(ση_1 𝕊_{k-1}) ‖² ≥
            ‖ p(ση_2 𝕊_{k-1}) ‖² ≥ ... ≥ ‖ p(ση_k 𝕊_{k-1}) ‖²;
    i ← 1;
    while ‖ p(ση_i 𝕊_{k-1}) ‖² > ‖ p(σ̃) ‖² {
      find_largest_in_subgroup(ση_i 𝕊_{k-1});
      i ← i + 1;
    }
  }}
```

Figure 6.1: Fast algorithm for finding the maximum probability permutation. The recursive procedure is called with the argument $e\mathbb{S}_n$, and $\tilde{\sigma}$ is initialized to any permutation. For $\sigma \in \mathbb{S}_n$, $\| p(\sigma\mathbb{S}_k) \|^2$ stands for $\frac{1}{k!}\sum_{\tau\in\mathbb{S}_k} \| p(\sigma\tau) \|^2$, which can be efficiently computed by partial Fourier transform methods, particularly when $\widehat{p}$ is sparse. Finally, $\eta_i \equiv [\![i, k]\!]$.

entries, respectively. At this point we lose the advantage in terms of detail of representation over other, simpler algorithms. Our spectral method is most likely to be of use in the $8 \leq n \leq 30$ range.

6.3 Experiments

We performed experiments on a dataset of (x, y) radar positions of aircraft within an approximately 30 mile diameter area about New York's John F. Kennedy International Airport on a given day in May, 2006, updated at few second intervals. The data is available in streaming format from `http://www4.passur.com/jfk.html`. The presence of three major airports in the area (JFK, La Guardia and Newark), together with specific local procedures, such as the routing of aircraft over the Hudson river for noise abatement, and simultaneous approaches at closely spaced parallel runways at JFK, result in aircraft frequently crossing paths in the (x, y) plane, making this dataset particularly interesting for our purposes.

The number of simultaneously tracked aircraft was $n = 6, 10$ and 15, and whenever an aircraft's tracking was terminated either because it has landed or because it has left the area, its track was reassigned to a new aircraft. Our method could be extended to tracking varying numbers of objects, but that is beyond the scope of our work at this time. The number 15 is significant because it is large enough to make storing a full distribution over permutations impossible ($15! \approx 1.3 \cdot 10^{12}$), yet with up to 4 Fourier components our algorithm can still run on an entire day's worth of data in a matter of hours. The original data comes pre-labeled with track/target assignment information. We introduce uncertainty by at each time-step t for each pair (j_1, j_2) of tracks, swapping their assignment with probability $p_{\text{mix}} \exp\left(-\| \boldsymbol{x}_{j_1}(t) - \boldsymbol{x}_{j_2}(t) \|^2 / \left(2s^2\right)\right)$, where $\boldsymbol{x}_j(t)$ is the position of track j at time t ($s = 0.1$ and $p_{\text{mix}} = 0.1$). This models noisy data, where the assignment breaks down when targets approach each other.

The algorithm has access to track locations, from which it can model this shifting distribution over assignments using the pairwise diffusion updates discussed in Section 6.1.1. In addition, with probability p_{obs} for each t and each j, the algorithm is provided with a first order observation of

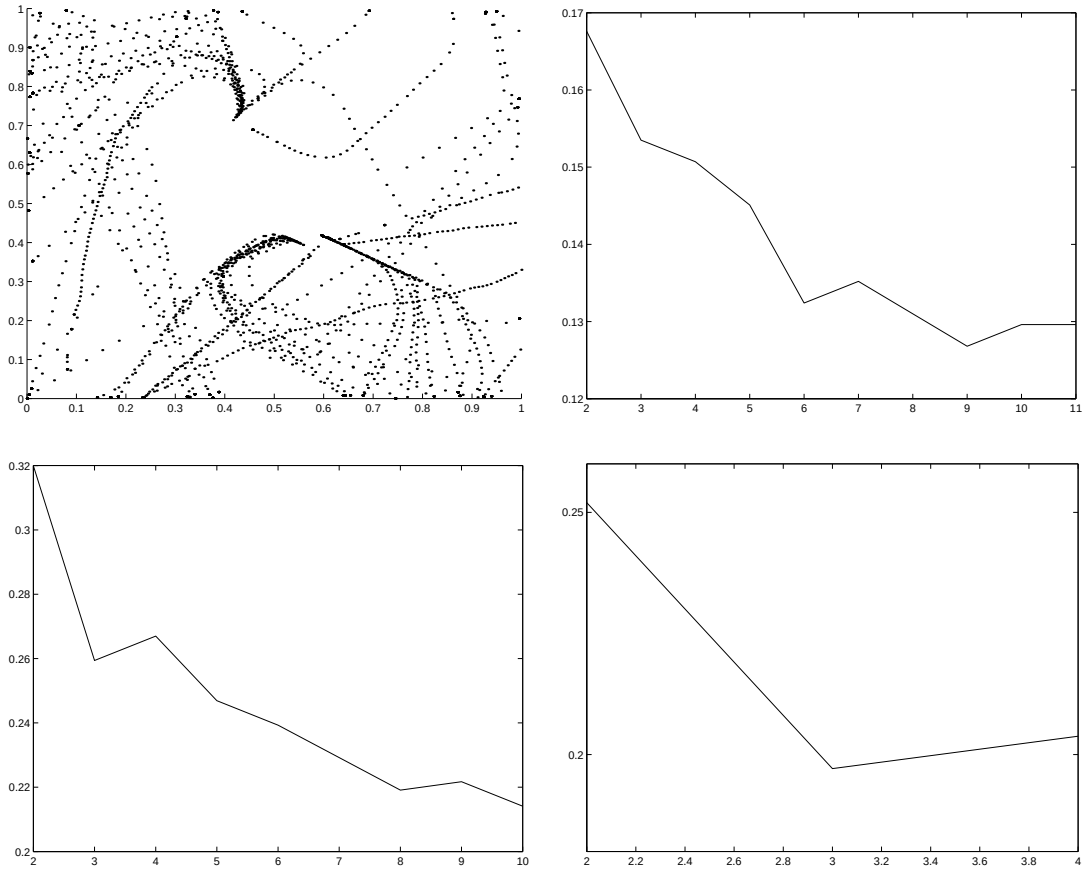


Figure 6.2: Trajectories of some of the flight tracks in the data (runways of LGA and JFK are clearly visible) and the error of our tracking algorithm for $n = 6, 10$ and 15 for varying number of Fourier components ($p_{\text{obs}} = 0.05, 0.1$ and 0.3)

the form $O_{i \to j}$, with $\pi = 1$. The task of the algorithm is to predict the target currently assigned to track j by computing $\arg\max_i p([\sigma(i) = j])$. The evaluation measure is the fraction of incorrect predictions.

Experimental results are promising (Figure 6.2) and clearly show that performance generally improves with the number of irreducibles. It is difficult to compare our algorithm directly with other identity management and data association methods, since most of them do not accept probabilistic observation and diffusion updates in the form supplied by the data. However, the best performance we could achieve with the information-form filter Schumitsch et al. [2005] for $n = 6$ was 0.2996 error and for $n = 10$ was 0.2885, which is significantly worse than our own result. Time limitations prevented us from more extensive comparisons, but our preliminary results indicate that the information form filter is better suited to domains where the distribution is highly peaked, whereas our method performs better when there is more uncertainty and the distribution is smooth. We are engaged in a thorough evaluation of our method in different domains, varying the observation frequency, π, n and the types of observations.

Part III

Invariance

Chapter 7

Invariant functionals

Since groups ultimately capture symmetries, the study of invariants has been inseparable from group theory from the start. In this chapter we concentrate on the case when G acts transitively on a set $\mathcal{X}$, and we wish to find invariants of functions $f\colon \mathcal{X} \to \mathbb{C}$ with respect to the induced (translation) action $f \mapsto f^g$ defined $f^g(x) = f(g^{-1}(x)),\ g \in G$. The simplest example is $G = \mathcal{X}$ acting on itself by left-multiplication, in which case f^g is the familiar left-translate of f.

We are particularly interested in constructing sets of invariants $\{(q_1(f), q_2(f), \dots, q_m(f))\}$ which are provably **complete**, in the sense that up to translation they uniquely determine f. Formally, this means that $q_1(f) = q_1(f'), \dots, q_m(f) = q_m(f')$ implies $f' = f^g$ for some $g \in G$.

This type of problem occurs in machine learning in the representation of individual data instances. The translation- rotation- and scale-invariance of images already mentioned in Section 4.4 is perhaps the most obvious case, but there are many other situations where such invariances appear in learning problems.

In contrast to Section 4.4 we now address the invariance problem in its most general form, independent of the RKHS framework, or any specific learning algorithm. While the symmetrization arguments in Chapter 4 were essentially linear operations, our quest for completeness will lead us to quadratic and cubic invariants.

Unless stated otherwise, the results of this chapter pertain to any finite group or compact Lie group. While for notational simplicity we often present our arguments in terms of the former, it should be understood that the arguments apply equally to the latter by substituting integration with respect to Haar measure for summation over group elements.

Non-compact Lie groups, while important from the point of view of applications, are more problematic for two reasons: having to deal with infinite dimensional irreducible representations, and the breakdown of so-called Tannaka–Krein duality, which is crucial for certain completeness results. For these reasons, except for a short discussion at the end of Section 7.1, we will confine ourselves to the compact case. As we will see in Chapter 8, one of the key issues in applying our methods to real world problems is to find the right way of compactifying problems.

7.1 The power spectrum and the bispectrum

Our starting point for developing invariants is the translation property of the Fourier transform, $\widehat{f^t}(\rho) = \rho(t)\,\widehat{f}(\rho)$, which we derived in Section 2.2. In general, any matrix valued functional $s\colon L(G) \to \mathbb{C}^{d_\rho \times d_\rho}$ which obeys $s(f^t) = \rho(t)\,s(f)$ we call ρ-**covariant**. We also introduce ρ-**contravariant** functionals, which transform according to $s'(f^t) = s'(f)\,\rho(t)^\dagger$. A trivial example of a ρ-contravariant function is $s'\colon f \mapsto \widehat{f}(\rho)^\dagger$.

The power spectrum

Assuming as always that the irreducible representations $\rho \in \mathcal{R}$ are unitary, it is clear that the product of a ρ-contravariant and a ρ-covariant function is invariant to translation: $s'(f^t)\,s(f^t) = s'(f)\,\rho(t)^\dagger\rho(t)\,s(f) = s'(f)\,s(f)$. In particular,

$$\widehat{a}_f(\rho) = \widehat{f}(\rho)^\dagger \widehat{f}(\rho) \qquad \rho \in \mathcal{R} \tag{7.1}$$

furnishes a system of translation invariant matrices, called the **power spectrum** of f.

For $G = \mathbb{R}$ or $G = \mathbb{Z}_n$ (7.1) does indeed reduce to the classical power spectrum, $\widehat{a}(k) = |\,\widehat{f}(k)\,|^2$. One property inherited from these classical cases is that the power spectrum is the Fourier transform of the **autocorrelation**, which on a group is defined

$$a_f(x) = \sum_{y \in G} f(yx^{-1})^* f(y), \tag{7.2}$$

where * denotes complex conjugation. To see this, rewrite (7.2) in the form $\sum_{y\in G} f^-(xy^{-1})\,f(y)$, where f^- is the **reflection** of f, defined $f^-(x) = f(x^{-1})$. By unitarity, $\widehat{f^-}(\rho) = \sum_{x\in G} f(x^{-1})\,\rho(x) = \sum_{x\in G} f(x)\,\rho(x^{-1}) = \widehat{f}(\rho)^\dagger$. Hence, by the convolution theorem $\widehat{a}_f(\rho) = \widehat{f^-}(\rho)\cdot\widehat{f}(\rho) = \widehat{f}(\rho)^\dagger\cdot\widehat{f}(\rho)$.

Another respect in which the non-commutative power spectrum is similar to its classical counterpart is that it is only a very impoverished representation of the original signal f. In the classical cases $\widehat{a}(k) = |\widehat{f}(k)|^2$ measures the energy in each frequency mode, and is oblivious to phase information. While translation-invariance does make the overall phase of the signal irrelevant, the relative phases of the different components contain important information which is all lost.

In the non-commutative case we face a similar situation. By construction, the matrices $\widehat{a}_f(\rho)$ are positive definite, and hence, by the results of Section 4.5, a_f is restricted to the cone of positive definite functions on G. Hence, the mapping $f \mapsto a_f$ is necessarily lossy. In particular, for any collection $(U_\rho)_{\rho\in\mathcal{R}}$ of unitary matrices, the function with Fourier transform $\widehat{f^U}(\rho) = U_\rho \cdot \widehat{f}(\rho)$ will have the same power spectrum as f. This is the non-commutative analog of losing the "relative phase information" between the different isotypals.

The bispectrum

The **bispectrum** addresses the issue of lost phase information by coupling different Fourier components. We first describe the classical case, then derive its generalization to the non-commutative setting. Without question, the key piece of work in this domain is Kakarala's PhD thesis [Kakarala, 1992], and in particular, his completeness result, which we present below as Theorem 7.1.1. While the material of this subsection mirrors Kakarala's work, the angle of the presentation is somewhat different, and the notations have been changed to conform to the rest of the present thesis.

In the classical case $f\colon \mathbb{Z}_n \to \mathbb{C}$, the definition of the bispectrum is

$$\widehat{b}_f(k_1, k_2) = \widehat{f}(k_1)^* \, \widehat{f}(k_2)^* \, \widehat{f}(k_1 + k_2). \tag{7.3}$$

Note that $\widehat{b}$ takes two real arguments and not just one, so it is a much larger object than the original signal. In general, $\widehat{b}_f$ is a highly redundant representation of f.

The invariance of $\widehat{b}$ under translation is immediate:

$$\widehat{b}_{f^t} = (e^{-i2\pi k_1 x} \, \widehat{f}(k_1))^* \, (e^{-i2\pi k_1 x} \, \widehat{f}(k_2))^* \, (e^{-i2\pi (k_1+k_2) x} \, \widehat{f}(k_1 + k_2)) = \widehat{b}_f.$$

It is also easy to see that $\widehat{b}$ is the (two dimensional) Fourier transform of the so called **triple correlation**

$$b_f(x_1, x_2) = \sum_{y=0}^{n-1} f(y - x_1)^* \, f(y - x_2)^* \, f(y),$$

which is manifestly shift-invariant. The most important property of the bispectrum, however, is that (provided that $|\, \widehat{f}(k) \,| > 0$ for all k) it is complete, in the sense that it uniquely determines f, up to translation.

The completeness property is easiest to prove by giving an explicit algorithm to reconstruct f from $\widehat{b}_f$. The algorithm proceeds by iteratively computing $\widehat{f}(0), \widehat{f}(1), \ldots$, starting with $\widehat{f}(0) = (\widehat{b}_f(0,0))^{1/3}$. Now $\widehat{f}(1)$ can be set to $(b_f(0,1)/\widehat{f}(0))^{1/2}$, but we can equally well take $\widehat{f}(1) = e^{i\phi}(b_f(0,1)/\widehat{f}(0))^{1/2}$ for any phase factor $\phi \in [0, 2\pi)$. This indeterminacy is inevitable, since it is a reflection of translation invariance. Once we have set $\widehat{f}(1)$, however, all the other coefficients are uniquely determined by

$$\widehat{f}(k) = \frac{b_f(1, k-1)}{\widehat{f}(k-1)^2}, \qquad k = 2, 3, \ldots. \tag{7.4}$$

It is easy to see that the bispectrum of f computed in this manner is indeed $\widehat{b}_f$, and that any pair of functions f_{ϕ_1} and f_{ϕ_2} corresponding to different choices of ϕ are left-translates of one-another. The only circumstance under which the reconstruction algorithm would break down is if one of the $\widehat{f}(k-1)$ coefficients that we divide by in (7.4) were zero. Since beyond ϕ, there is no other freedom in the reconstruction process, we conclude that the original f must be equal to f_ϕ for some ϕ.

The generalization of the bispectrum to non-commutative compact groups similarly involves multiplying together different Fourier components, but now these components are matrices. While the individual $\widehat{f}(\rho)$ transform according to $\widehat{f}(\rho) \mapsto \rho(t) \, \widehat{f}(\rho)$, the matrix product of two such matrices does not follow any such simple transformation rule. Instead of the matrix product, we therefore take the tensor product, which transforms according to

$$\widehat{f^t}(\rho_1) \otimes \widehat{f^t}(\rho_2) = (\rho_1(t) \otimes \rho_2(t)) \, (\widehat{f}(\rho_1) \otimes \widehat{f}(\rho_2)).$$

To form cubic invariants as in (7.5), we must multiply this product with a linear function of f in such a way that the various $\rho(t)$ factors appearing on translation will cancel.

The way this may be done is revealed by the general way in which tensor product representations decompose into irreducible representations. First note that since

$$\rho_1(xy) \otimes \rho_2(xy) = (\rho_1(x) \otimes \rho_2(x)) \cdot (\rho_1(y) \otimes \rho_2(y)), \qquad x, y \in G,$$

the tensor product of any two representations is always a representation. In general, $\rho_1 \otimes \rho_2$ is not irreducible. By the theorem of complete decomposability (Theorem 1.2.1), however, it must decompose into a direct sum of irreducibles in an essentially unique way

$$(\rho_1 \otimes \rho_2)(x) = C_{\rho_1,\rho_2} \left[\bigoplus_{\rho \in \mathcal{R}} \rho(x)^{\oplus m_\rho} \right] C_{\rho_1,\rho_2}^\dagger \qquad x \in G.$$

This is called the **Clebsch-Gordan** decomposition and the C_{ρ_1,ρ_2} unitary matrices are called Clebsch-Gordan matrices. The $m_\rho \geq 0$ multiplicities are well defined and $M^{\oplus m}$ is a shorthand for $\bigoplus_{i=1}^m M$.

Assuming that the Clebsch-Gordan decomposition is known, it is evident that multiplying the conjugate of $\widehat{f}(\rho_1) \otimes \widehat{f}(\rho_2)$ by $C_{\rho_1,\rho_2}\big[\bigoplus_{\rho \in \mathcal{R}} \widehat{f}(\rho)^{\oplus m_\rho} \big] C_{\rho_1,\rho_2}^\dagger$ will yield an invariant matrix. The **bispectrum** on a non-commutative group is the system of matrices

$$\widehat{b}_f(\rho_1,\rho_2) = \big(\widehat{f}(\rho_1) \otimes \widehat{f}(\rho_2)\big)^\dagger \, C_{\rho_1,\rho_2} \left[\bigoplus_{\rho \in \mathcal{R}} \widehat{f}(\rho)^{\oplus m_\rho} \right] C_{\rho_1,\rho_2}^\dagger, \qquad \rho_1,\rho_2 \in \mathcal{R}. \tag{7.5}$$

When there is no risk of ambiguity, we shall sometimes omit the index f from $\widehat{b}_f$. In numerical computations it can save some computational time to omit the final multiplication by $C_{\rho_1,\rho_2}^\dagger$, and use the modified bispectrum

$$\widehat{b'}(\rho_1,\rho_2) = \big(\widehat{f}(\rho_1) \otimes \widehat{f}(\rho_2)\big)^\dagger \, C_{\rho_1,\rho_2} \bigoplus_{\rho \in \mathcal{R}} \widehat{f}(\rho)^{\oplus m_\rho}, \qquad \rho_1,\rho_2 \in \mathcal{R}, \tag{7.6}$$

which is unitarily equivalent to the original, and hence has the same invariance and completeness properties.

Non-commutative bispectra have been extensively investigated by Ramakrishna Kakarala in the article [Kakarala, 1993] and in his thesis [Kakarala, 1992]. To find what form the non-commutative triple correlation takes on non-commutative groups, i.e., what invariant function $b\colon G \times G \to \mathbb{C}$ the bispectrum is the Fourier transform of, first recall from general representation theory that the irreducible representations of $G \times G$ are tensor products of the irreducible representations of G. It follows that

$$\widehat{f}(\rho_1) \otimes \widehat{f}(\rho_2) = \sum_{x_1 \in G} \sum_{x_2 \in G} (\rho_1(x_1) \otimes \rho_2(x_2)) \, f(x_1) \, f(x_2)$$

in (7.5) is the Fourier transform of $u(x_1,x_2) = f(x_1)\, f(x_2)$, while

$$C_{\rho_1\rho_2} \left[\bigoplus_{\rho \in \mathcal{R}} \widehat{f}(\rho)^{\oplus m_\rho} \right] C_{\rho_1\rho_2}^\dagger = \sum_{x \in G} C_{\rho_1\rho_2} \left[\bigoplus_{\rho \in \mathcal{R}} \widehat{\rho}(x)^{\oplus m_\rho} \right] C_{\rho_1\rho_2}^\dagger \; f(x) = \sum_{x \in G} (\rho_1(x) \otimes \rho_2(x)) \, f(x)$$

is the Fourier transform of

$$v(x_1,x_2) = \begin{cases} f(x_1) & \text{if } x_1 = x_2 \\ 0 & \text{otherwise .} \end{cases}$$

By the convolution theorem (on $G \times G$), (7.5) must therefore be the Fourier transform of $u^- * v$

(recall that $u^-(x_1, x_2) = u(x_1^{-1}, x_2^{-1})$), hence the non-commutative **triple correlation** is

$$b(x_1, x_2) = \sum_{y_1 \in G} \sum_{y_2 \in G} u^-(x_1 y_1^{-1}, x_2 y_2^{-1})^* \, v(y_1, y_2) = \sum_{y_1 \in G} \sum_{y_2 \in G} u(y_1 x_1^{-1}, y_2 x_2^{-1})^* \, v(y_1, y_2) = \sum_{y \in G} f(y x_1^{-1})^* \, f(y x_2^{-1})^* \, f(y). \quad (7.7)$$

Once again, this function is manifestly translation invariant.

All of the above results can be found in Kakarala's thesis, up to some slight differences in the exact definition of the triple correlation and the bispectrum. Kakarala's most significant result, however, is that at least for compact groups, the non-commutative bispectrum exhibits the same completeness property as its classical counterpart:

Theorem 7.1.1 *([Kakarala, 1992, Theorem 3.2.4]) Let f and f' be a pair of complex valued integrable functions on a compact group G, and let the bispectrum be defined as in (7.5). Assume that $\widehat{f}(\rho)$ is invertible for each $\rho \in \mathcal{R}$. Then $f' = f^z$ for some $z \in G$ if and only if $b_f(\rho_1, \rho_2) = b_{f'}(\rho_1, \rho_2)$ for all $\rho_1, \rho_2 \in \mathcal{R}$.*

This is a remarkable and non-trivial result, the proof of which we shall not attempt to reproduce here. The proof hinges on Tannaka–Krein duality, which in loose terms states that given a compact group G with irreducible representations $\mathcal{R}$ and a collection of non-zero unitary matrices $\left\{U_\rho \in \mathbb{C}^{d_\rho \times d_\rho}\right\}_{\rho \in \mathcal{R}}$ obeying the same decomposition rule

$$U_{\rho_1} \otimes U_{\rho_2} = C_{\rho_1, \rho_2} \left[\bigoplus_{\rho \in \mathcal{R}} U_\rho^{\oplus m_\rho} \right] C_{\rho_1, \rho_2}^\dagger$$

as the representation matrices, there is an $x \in G$ such that $U_\rho = \rho(x)$ for each $\rho \in \mathcal{R}$ [Kakarala, 1992, Theorem 3.2.2].

Kakarala manages to generalize Theorem 7.1.1 to a specific subset of globally non-compact but separable and locally compact groups, which he calls Tatsuuma duality groups. Significantly, the affine transformation group of $\mathbb{R}$, the Euclidean groups $\mathrm{ISO}(n)$, and the n-dimensional proper inhomogeneous Lorentz group all belong to this class. Note that non-compact groups have infinite dimensional irreducible representations, therefore the Fourier transform and the bispectrum need to be generalized to the linear operator realm.

Theorem 7.1.2 *([Kakarala, 1992, Theorem 3.4.5]) Let G be a Tatsuuma duality group and let f, f' be complex valued functions on G in $L_1(G)$. Let the bispectrum be defined as in (7.5), and assume that $\widehat{f}(\rho)$ is invertible for each $\rho \in \mathcal{R}$. Then $f' = f^z$ for some $z \in G$ if and only if $b_f(\rho_1, \rho_2) = b_{f'}(\rho_1, \rho_2)$ for all $\rho_1, \rho_2 \in \mathcal{R}$.*

7.2 The skew spectrum

From a computational point of view, both the bispectrum and the triple correlation can be problematic. The former demands Clebsch-Gordan matrices, which are often surprisingly hard to compute, even for such fundamental groups as $\mathbb{S}_n$, and computing (7.5) involves multiplying very large matrices. The latter involves summing (integrating) over the whole group, and cannot easily be specialized to band-limited functions.

These are the considerations that motivated us to develop a third, unitarily equivalent form of the bispectrum, which we named the **skew spectrum**. The skew spectrum is midway between the triple correlation and the bispectrum in the sense that each of its components is indexed by a pair (z, ρ), where z is an element of G and ρ is an irreducible.

Definition 7.2.1 *Let G be a compact group, f a function $f\colon G \to \mathbb{C}$, and let $r_z(x) = f(xz)\, f(x)$. The skew spectrum of f is then defined as the collection of matrices*

$$\widehat{q}_z(\rho) = \widehat{r}_z(\rho)^\dagger \cdot \widehat{f}(\rho), \qquad z \in G,\ \rho \in \mathcal{R}. \tag{7.8}$$

To see that the skew spectrum and the triple correlation are unitarily equivalent, it is sufficient to observe that by the convolution theorem

$$q_z(x) = \sum_{y\in G} r_z^-(xy^{-1})^*\, f(y) = \sum_{y\in G} r_z(yx^{-1})^*\, f(y) = \sum_{y\in G} f(yx^{-1})^*\, f(yx^{-1}z)^*\, f(y) = b(x, z^{-1}x),$$

and that as z sweeps out G, $(b(x, z^{-1}x))_{z,x}$ covers the entire triple correlation b. We thus have the following theorem.

Theorem 7.2.2 *Let f and f' be complex valued integrable functions on a compact group G with skew spectra $\widehat{q}$ and $\widehat{q}'$, with respect to a complete set of inequivalent irreducible representations $\mathcal{R}$ of G. Assume that $\widehat{f}(\rho)$ is invertible for all $\rho \in \mathcal{R}$. Then $f' = f^t$ for some $t \in G$ if and only if $\widehat{q}'_z(\rho) = \widehat{q}_z(\rho)$ for all $z \in G$ and all $\rho \in \mathcal{R}$.*

Since the bispectrum, triple correlation, and skew spectrum are unitarily equivalent sets of invariants, in a computational setting the choice between them will largely be determined by their respective computational and storage complexities. We now present an analysis of these factors for finite groups.

The triple correlation consists of $|G|^2$ scalars, and by unitarity the total number of entries in the bispectrum matrices must be the same. The symmetry $b(x_1, x_2) = b(x_2, x_1)$ and its counterpart $\widehat{b}(\rho_1, \rho_2) = \widehat{b}(\rho_2, \rho_1)$ (up to reordering of rows and columns) reduce the number of relevant components to $|G|(|G|+1)/2$, but to keep the analysis as simple as possible we ignore this constant factor. Another technical detail that we ignore for now is that if all we need is a complete set of invariants, then it is sufficient to compute $\widehat{b}\,(\rho_1, \rho_2)\, C_{\rho_1\rho_2}$, avoiding the final multiplication by $C^\dagger_{\rho_1\rho_2}$ in (7.5).

The cost of computing the bispectrum involves two factors: the cost of the Fourier transform, and the cost of multiplying the various matrices in (7.5). As discussed in Chapter 3, fast Fourier transforms typically reduce the complexity of Fourier transformation (and inverse Fourier transformation) from $O(|G|^2)$ to $O(|G| \log^k |G|)$ scalar operations for some small integer k. We assume that for whatever group we are working on, an $O(|G| \log^k |G|)$ transform is available. This makes matrix multiplication the dominant factor in the cost of computing the bispectrum.

The complexity of the naive approach to multiplying two D-dimensional matrices is D^3, hence the cost of computing (7.5) for given (ρ_1, ρ_2) is $O(d^3_{\rho_1} d^3_{\rho_2})$. Summing over all $\rho_1, \rho_2 \in \mathcal{R}$ and assuming that $\sum_{\rho\in\mathcal{R}} d^3_\rho$ grows with some power $1 < \theta < 2$ of $|G|$ (note that since $\sum_{\rho\in\mathcal{R}} d^2_\rho = |G|$, even $\sum_{\rho\in\mathcal{R}} d^4_\rho$ grows with at most $|G|^2$) gives an overall complexity bound of $O(|G|^{2\theta})$. Thus, the critical factor in computing the bispectrum is matrix multiplication and not the fast Fourier transform.

The complexity analysis of the triple correlation is much simpler. Equation (7.7) involves an explicit sum over G, which is to be computed for all $|G|^2$ possible values of (x_1, x_2). Hence, the total time complexity is $O(|G|^3)$.

In contrast, the skew spectrum involves $|G|+1$ separate Fourier transforms followed by $|G|$ sequences of multiplying $\{\mathbb{C}^{d_\rho \times d_\rho}\}_{\rho \in \mathcal{R}}$ matrices. The cost of the former is $O(|G|^2 \log^k |G|)$, while the cost of the latter is $O(|G|^{\theta+1})$, potentially improving on both the triple correlation and the bispectrum. Further advantages of the skew spectrum become apparent when extending our discussion to functions on homogeneous spaces.

7.3 Generalization to homogeneous spaces

In machine learning the objects that want to find invariants of with respect to G are not necessarily functions on G itself. Often f is defined on some other space $\mathcal{X}$, which G is acting on. In this case we still have a natural interpretation of left-translation by $t \in G$, namely $f^t(x) = (t^{-1}(x))$. In this section we investigate the way bispectral techniques generalize to this case.

Clearly, the interesting case is when the action of G on $\mathcal{X}$ is transitive. Otherwise we can decompose $\mathcal{X}$ into orbits (on which, by definition, G acts transitively) and find invariants of f restricted to each orbit. Accordingly, in the following we can assume that $\mathcal{X}$ is a homogeneous space of G, and by the results of Section 1.1.5 we can regard f as a function on the quotient space G/H where H is the isotropy subgroup. The generalization of the Fourier transform to this case is

$$\widehat{f}(\rho) = \sum_{x \in G} \rho(x)\, f(xH) \qquad \rho \in \mathcal{R}, \tag{7.9}$$

which factors in the form

$$\widehat{f}(\rho) = \sum_{x \in G/H} f(xH)\, \rho(x) \sum_{h \in H} \rho(h) \qquad \rho \in \mathcal{R}. \tag{7.10}$$

Any $f\colon G/H \to \mathbb{C}$ induces a function $f\uparrow^G\colon G \to \mathbb{C}$ by $f\uparrow^G(x) = f(xH)$, which is constant on each xH coset. Such functions we call G/H-**coset functions**. Equation (7.9) tells us that $\widehat{f}(\rho) = \widehat{f\uparrow^G}(\rho)$, so harmonic analysis on G/H is essentially just the harmonic analysis of G/H-coset functions. Consequently, all the important properties of Fourier transformation, including unitarity, the translation property $\widehat{f^t}(\rho) = \rho(t)\,\widehat{f}(\rho)$, etc., generalize without change (note, however, that the right-translate of G/H-coset function is no longer necessarily a G/H-coset function).

An important feature of the Fourier transform of coset functions is that they are potentially very sparse. In particular, if the irreducible representation ρ in (7.10) when restricted to H splits into irreducible representations of H in the form

$$\rho\downarrow_H(h) = T^\dagger \left[\bigoplus_{\rho'} \rho'(h) \right] T, \qquad h \in H, \tag{7.11}$$

then the Fourier transform can be rewritten as

$$\widehat{f}(\rho) = \left[\sum_{x \in G/H} f(x)\, \rho(x) \right] M_\rho\,, \tag{7.12}$$

where M_ρ is the constant matrix

$$M_\rho = T^\dagger \left[\bigoplus_{\rho'} \sum_{h \in H} \rho'(h) \right] T.$$

If ρ' is the trivial representation ρ^H_{triv} of H, then $\sum_{h \in H} \rho'(h) = |H|$. However, for any other irreducible, by the unitarity of Fourier transformation over H we must have $\sum_{h \in H} \rho'(h) = 0$, so the rank of M (and hence of $\widehat{f}(\rho)$) is at most equal to the multiplicity of the trivial representation of H in (7.11).

In practice this means that depending on the choice of G and H we will often find that many $\widehat{f}(\rho)$ components are identically zero, because $\rho \downarrow_H$ does not contain any copies of ρ^H_{triv} at all. Furthermore, if $\mathcal{R}$ is adapted to $G > H$, then M_ρ will be block diagonal, so even if $\widehat{f}(\rho)$ does not identically vanish, it will only have as many non-zero columns as the multiplicity of ρ^H_{triv} in $\rho \downarrow_H$.

From a computational point of view these restrictions on the form of $\widehat{f}(\rho)$ are a blessing, reducing both the time complexity and the storage requirements associated with computing the Fourier transform of functions on G/H. On the other hand, they also mean that the $\widehat{f}(\rho)$ matrices become singular, which blatantly violates the conditions of Theorems 7.1.1 and 7.2.2. The bispectrum and the skew spectrum will still be invariants of f with respect to left-translation, however, since that property only depends on the translation property of the Fourier transform. To be explicit, we state the following theorem.

Theorem 7.3.1 *Let H be a subgroup of a compact group G, and let $f, f' \in L_1(G/H)$. Assume that the Fourier transform of f is defined as in (7.12), the bispectrum is defined as in (7.5), and the skew spectrum is defined as in (7.8). Then if $f' = f^t$ for some $t \in G$, then $\widehat{b}_{f'} = \widehat{b}_f$ and $\widehat{q}_{f'} = \widehat{q}_f$.*

The potential computational savings involved in computing the bispectrum of functions on G/H arise from the special form of their Fourier transform. In the case of the skew spectrum this is compounded by redundancies in the z index. The following theorem shows that we may restrict z to a complete set of $H\backslash G/H$ double coset representatives without losing any information.

Theorem 7.3.2 *Let H be subgroup of a compact group G and let $f \in L_1(G/H)$. Then the skew spectrum $\widehat{q}_f$ is uniquely determined by its subset of components $\{\, \widehat{q}_z(\rho) \mid \rho \in \mathcal{R},\ z \in H\backslash G/H \,\}$.*

Proof. For any $h \in H$,

$$r_{zh}(x) = f(xzhH)f(xhH) = f(xzH)f(xH) = r_z(x),$$

so $\widehat{q}_{zh}(\rho) = \widehat{r}_z(\rho)^\dagger \widehat{f}(\rho) = \widehat{r}(\rho)^\dagger \widehat{f}(\rho) = q_z(\rho)$. Now for $h' \in H$

$$r_{h'z}(x) = f(xh'zH)f(xH) = f(xh'zH)f(xh'H) = r_z^{(h'^{-1})}(x),$$

and thus by the translation property of the Fourier transform,

$$\widehat{q}_{h'z}(\rho) = (\widehat{r}_z(\rho)\, \rho(h'^{-1}))^\dagger \widehat{f}(\rho) = \rho(h')\, \widehat{r}_z(\rho)^\dagger\, \widehat{f}(\rho) = \rho(h')\, \widehat{q}_z(\rho),$$

so $\widehat{q}_{h'z}(\rho)$ and $\widehat{q}_z(\rho)$ albeit not equal, are related by an invertible linear mapping. ■

Before closing this section we mention that Kakarala presents a completeness result for the bispectrum of functions on homogeneous spaces, but one that is only applicable to functions on $G\backslash H$ and not G/H. One crucial difference between these cases is the fact that if f is a $G\backslash H$-coset function, that does not imply that its left-translates will also be in the same class. Unlike the G/H case, the left translation of such functions does not have a clear interpretation in terms of G acting on a space $\mathcal{X}$ on which f is defined. Consequently, this case is of less relevance to capturing real-world invariances. At the same time, if $f \in L(G\backslash H)$, then any $g, g' \in G$ related by $g' = gh$ for some $h \in H$ will have the same action on f, effectively reducing the set of possible translations to members of G/H. In this sense the $G\backslash H$-coset case is easier.

Below we reproduce Kakarala's result paraphrased to conform to our notation. Note that in contrast to our previous theorems, this result is asymmetric: it compares $f \in L_1(G\backslash H)$ to $f' \in L_1(G)$.

Theorem 7.3.3 *[Kakarala, 1992, Theorem 3.3.6] Let H be any closed subgroup of a compact group G, and let $f \in L_1(G\backslash H)$ be such that for all $\rho \in \mathcal{R}$, the matrix rank of $\widehat{f}(\rho)$ is equal to the multiplicity of ρ^H_{triv} in the decomposition of ρ into irreducible representations of H. Then $\widehat{b}_f = \widehat{b}_{f'}$ for some $f' \in L_1(G)$ if and only if there exists some $t \in G$ such that $f' = (f\uparrow^G)^t$.*

To summarize this section, the machinery of bispectra and skew spectra does carry over to homogeneous spaces of compact groups, but the completeness property is lost. Nonetheless, as we shall see in the following chapter, in machine learning applications we often find that the bispectrum and the skew spectrum contains more than enough information to solve the practical problem at hand.

Chapter 8

Invariant features for images

The most immediate application of the invariants of the previous chapter is in pattern recognition and computer vision, since natural images exhibit a wide range of invariances, including translation- rotation- and to some extent scale invariance. In this chapter we focus on just the first two of these.

The difficulty with directly applying the bispectral methods of the previous chapter to this problem is that the relevant group, the group $\mathrm{ISO}^+(2)$ of rigid body motions of $\mathbb{R}^2$, is not compact. Although $\mathrm{ISO}^+(2)$ belongs to the class of special non-compact groups to which Theorem 7.1.2 applies, non-compactness leads to various computational complications, not the least of which is that the irreducible representations, and hence the Fourier components, become operator-valued. The key idea of this chapter is to compactify the translation/rotation problem by exploiting a local isomorphism between the action of $\mathrm{ISO}^+(2)$ on $\mathbb{R}^2$ and the action of $\mathrm{SO}(3)$ on the two-sphere S_2.

8.1 A projection onto the sphere

Our approach to compactifying the $\mathrm{ISO}^+(2)$ invariance problem is based on the observation that while $\mathrm{ISO}^+(2)$ itself is not compact, the images that we are operating on are confined to a compact region, and the translations/rotations that we care about are also restricted to a compact subset of the full transformation group. Asymptotically, any group that is locally isomorphic to $\mathrm{ISO}^+(2)$ acting on any space locally isomorphic to $\mathbb{R}^2$ leads to the same invariance problem. Here we take the simplest case of $\mathrm{SO}(3)$ acting on the sphere S_2.

To find the local isomorphism we need to fix a way of parametrizing both groups as well as the spaces that they are acting on. Taking advantage of the fact that any $T \in \mathrm{ISO}^+(2)$ may be decomposed into a rotation followed by a translation, we use the parametrization $T = (t_x, t_y, \alpha)$ given by

$$\begin{pmatrix} x \\ y \end{pmatrix} \stackrel{H}{\longmapsto} \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}.$$

We use the notation $B_\epsilon = \left\{ (x, y) \in \mathbb{R}^2 \mid \| (x, y) \| < \epsilon \right\}$ to denote an ϵ-radius circular patch in $\mathbb{R}^2$ around the origin, and the convention $x = r\cos\phi$ and $y = r\sin\phi$ to label points in $\mathbb{R}^2$ in polar coordinates (r, ϕ).

The surface of S_2 will be parametrized by spherical polar coordinates (θ, ϕ), where $0 \leq \theta \leq 2\pi$ measures the polar angle from the positive z axis, and $0 \leq \phi < 2\pi$ measures the azimuthal angle

from the positive x axis:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \sin\theta\,\cos\phi \\ \sin\theta\,\sin\phi \\ \cos\theta \end{pmatrix}.$$

For $R \in \mathrm{SO}(3)$ we use **Euler angles** (θ, ϕ, ψ), which correspond to decomposing R into a sequence of three rotations about the z axis, the y axis, and again the z axis:

$$R(\theta,\phi,\psi) = \begin{pmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix} \begin{pmatrix} \cos(\psi-\phi) & -\sin(\psi-\phi) & 0 \\ \sin(\psi-\phi) & \cos(\psi-\phi) & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{8.1}$$

The intuitive picture is that $R = (\theta, \phi, \psi)$ rotates the z axis to spherical coordinates (θ, ϕ), and then rotates the other two basis vectors around the rotated image of e_z by ψ. We also introduce the notation $\mathrm{ISO}^+_\epsilon(2) = \{\, (t_x, t_y, \alpha) \in \mathrm{ISO}^+(2) \mid (t_x, t_y) \in B_\epsilon \,\}$.

We identify points in B_1 with points on the S_2^+ (the Northern hemisphere of the sphere) by the projection

$$\omega\colon (r, \phi_{\mathbb{R}^2}) \mapsto (\theta, \phi_{S_2}) \qquad \text{with} \quad r = \sin\theta \quad \text{and} \quad \phi_{\mathbb{R}^2} = \phi_{S_2}. \tag{8.2}$$

The $\mathbb{R}_2$ and S_2 subscripts are to clarify whether we are referring to spherical or planar polars. We also set up a correspondence $\Upsilon\colon (\alpha, t_x, t_y) \mapsto (\theta, \phi, \psi)$ between rigid body motions and three dimensional rotations defined by

$$\alpha = \psi, \qquad t_x = \sin\theta\cos\phi, \qquad t_y = \sin\theta\sin\phi. \tag{8.3}$$

Note that the domain of Υ must be restricted to $\mathrm{ISO}^+_1(2)$. The following theorem establishes that these correspondences indeed lead to a local isomorphism.

Theorem 8.1.1 *If $\boldsymbol{x} = (x, y) \in B_\epsilon$ and $T \in ISO^+_\epsilon(2)$ for some $0 < \epsilon < 1/2$, and $R = \Upsilon(T)$, then*

$$\left\| \; T(\boldsymbol{x}) - \omega^{-1}(R(\omega(\boldsymbol{x}))) \; \right\| = O(\epsilon^2).$$

Proof. Multiplying the second two matrices in (8.1) gives

$$R(\theta,\phi,\psi) = \begin{pmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos(\psi-\phi) & -\sin(\psi-\phi) & \sin\theta \\ \sin(\psi-\phi) & \cos(\psi-\phi) & 0 \\ -\sin\theta\cos(\psi-\phi) & \sin\theta\sin(\psi-\phi) & \cos\theta \end{pmatrix}.$$

Performing the multiplication and using $\epsilon^2 > t_x^2 + t_t^2 = \sin^2\theta(\sin^2\phi + \cos^2\phi) = \sin^2\theta$, and the Taylor series approximations $\sin^2\theta = O(\epsilon^2)$ and $\cos\theta = 1 - O(\epsilon^2)$ gives

$$R(\theta,\phi,\psi) = \begin{pmatrix} \cos\psi & -\sin\psi & \cos\phi\sin\theta \\ \sin\psi & -\cos\psi & \sin\phi\sin\theta \\ -\sin\theta\cos(\psi-\phi) & \sin\theta\sin(\psi-\phi) & 1 \end{pmatrix} + O(\epsilon^2).$$

Now in Cartesian coordinates $\omega(\boldsymbol{x})$ is just $\boldsymbol{v} = (x, y, \sqrt{1-x^2-y^2})^\top = (x, y, 1)^\top + O(\epsilon^2)$, and both $x\sin\theta$ and $y\sin\theta$ are $O(\epsilon^2)$, whence

$$R(\theta,\phi,\psi)\cdot\boldsymbol{v} = \begin{pmatrix} \cos\psi & -\sin\psi & \cos\phi\sin\theta \\ \sin\psi & -\cos\psi & \sin\phi\sin\theta \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} + O(\epsilon^2).$$

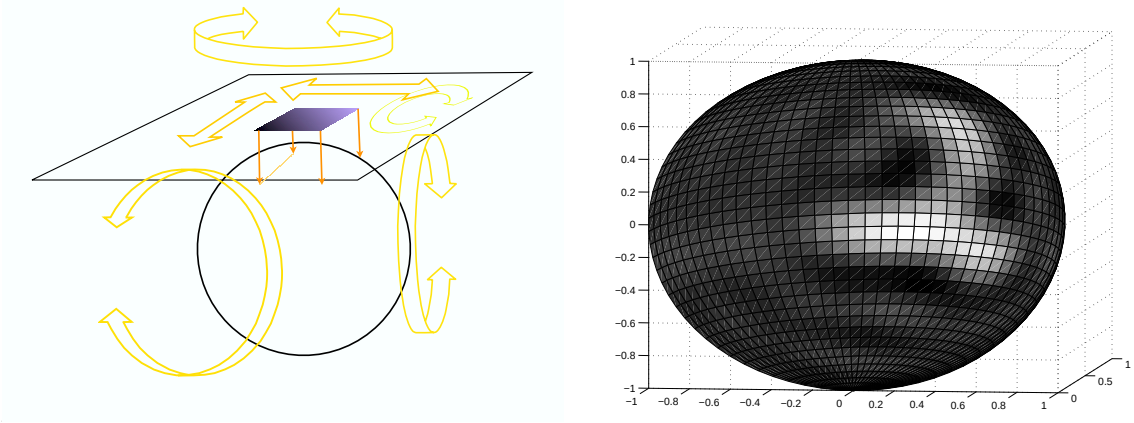


Figure 8.1: (a) The local isomorphism between the action of rotations and translations on functions on the plane and the action of 3D rotations on the same functions projected on the sphere. (b) A NIST handwritten digit projected onto the sphere. The band-limit is $L = 15$. Note that there is a small amount of "ringing".

Taking advantage of the fact that in Cartesian coordinates $\omega^{-1}\colon (v_x, v_y, v_z) \mapsto (v_x, v_y)$, and using (8.3), this gives

$$\omega^{-1}(R(\omega(\boldsymbol{x}))) = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix} + O(\epsilon^2).$$

■

If in our original problem images are represented by functions $h\colon \mathbb{R}^2 \to \mathbb{R}$ with support confined to B_ϵ, Theorem 8.1.1 suggests projecting them onto the sphere by $\Omega\colon h \mapsto f$, where $f(\omega(\boldsymbol{x})) = h(\boldsymbol{x})$.

We say that a homeomorphism $\Phi\colon \mathbb{R}^2 \to \mathbb{R}^2$ is a δ-homeomorphism, if $\| x - \Phi(x) \| \leq \delta$ for all $x \in \mathbb{R}^2$. Theorem 8.1.1 then immediately gives the following result.

Corollary 8.1.2 *For any $T \in ISO^+_\epsilon(2)$ with $0 < \epsilon < 1/2$ there is a δ-homeomorphism Φ_T, such that for any image h with support confined to B_ϵ,*

$$(\Omega^{-1}((\Omega f)^{\Upsilon T}))(\boldsymbol{x}) = f(\Phi_T(\boldsymbol{x})), \tag{8.4}$$

with $\delta = O(\epsilon^2)$.

The somewhat cumbersome notation of (8.4) belies the intuitive nature of this corollary. In plain words, it simply states that up to $O(\epsilon^2)$ distortion, translations/rotations no greater than ϵ of images no larger than ϵ can be imitated by projecting the image down to the sphere and rotating the sphere (Figure 8.1). Although Theorem 8.1.1 and Corollary 8.1.2 are asymptotic results, in practice the distortion is tolerable even for $\epsilon = 1/2$, while for $\epsilon < 0.1$, its effect is negligible: it is legitimate to say that we have reduced our problem to finding invariants of f to rotations.

8.2 Bispectral invariants of functions on S_2

Regarding rotations as "translation of functions on S_2 by elements of SO(3)" makes the framework of bispectral invariants (bispectrum, triple correlation, skew spectrum and tangent spectra) directly applicable to finding rotational invariants of the projected images. In fact, the action of SO(3) on S_2 is probably the simplest non-trivial case of the theory of non-commutative bispectral invariants, and has already been briefly described in [Healy et al., 1996]. To make the method as transparent as possible, we derive the invariants from first principles and then show that they are indeed a special case of the bispectrum framework.

Our starting point is to expand f as

$$f(\theta,\phi) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} \widehat{f}_{l,m}\, Y_l^m(\theta,\phi) \tag{8.5}$$

in terms of the **spherical harmonics**

$$Y_l^m(\theta,\phi) = \sqrt{\frac{2l+1}{4\pi}\frac{(l-m)!}{(l+m)!}}\;\; P_l^m(\cos\theta)\; e^{im\phi},$$

where P_l^m are the associated Legendre polynomials. It can be shown that for any sufficiently smooth integrable function on S_2 such an expansion exists, and the expansion coefficients are given by $\widehat{f}_{l,m} = \langle f, Y_l^m\rangle$, where $\langle\cdot,\cdot\rangle$ is the inner product

$$\langle f, g\rangle = \frac{1}{4\pi}\int_0^{\pi}\int_0^{2\pi} f^*(\theta,\phi)\, g(\theta,\phi)\sin\theta\, d\phi\, d\theta.$$

In numerical applications we will truncate (8.5) at some finite L, corresponding to a high frequency cutoff. We also introduce the vectors $\widehat{\boldsymbol{f}}_l = (\widehat{f}_{l,-l}, \widehat{f}_{l,-l+1}, \ldots, \widehat{f}_{l,l})$.

In the image invariants application, starting with an $n\times n$ pixel image M of width $0 < w < 1/\sqrt{2}$, the vectors $\widehat{\boldsymbol{f}}_0, \widehat{\boldsymbol{f}}_1, \ldots, \widehat{\boldsymbol{f}}_L$ can be computed in $O(L^3n^2)$ time by

$$\widehat{f}_{l,m} = \sum_{i,j=1}^{n} M_{i,j} Y_l^m(\theta,\phi), \tag{8.6}$$

where by (8.2) for $x = w\,((i-1)/(n-1) - 0.5)$ and $x = w\,((i-1)/(n-1) - 0.5)$,

$$\theta = \arcsin\sqrt{x^2+y^2}, \qquad \phi = \begin{cases} \arctan(y/x) & \text{if } \; y > 0 \\ 2\pi - \arctan(y/x) & \text{if } \; y < 0 \end{cases}.$$

The key to constructing invariants from $\widehat{\boldsymbol{f}}_0, \widehat{\boldsymbol{f}}_1, \ldots, \widehat{\boldsymbol{f}}_L$ is that Y_l^m are the $-l^2$ eigenvalue eigenfunctions of the **Laplacian**

$$\Delta = \frac{1}{\sin^2\theta}\frac{\partial}{\partial\phi^2} + \frac{1}{\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\,\frac{\partial}{\partial\theta}\right),$$

which is a rotation invariant operator, hence under rotation any given Y_l^m must transform into a linear combination of $Y_l^{-l}, Y_l^{-l+1}, \ldots, Y_l^l$. In other words, if for some $R \in \mathrm{SO}(3)$, f' is the rotated

function f^R, then

$$\begin{pmatrix} \widehat{f}'_{l,-l} \\ \vdots \\ \widehat{f}'_{l,l} \end{pmatrix} = D^{(l)}(R) \begin{pmatrix} \widehat{f}_{l,-l} \\ \vdots \\ \widehat{f}_{l,l} \end{pmatrix} \tag{8.7}$$

for some matrix $D^{(l)}(R) \in \mathbb{C}^{(2l+1)\times(2l+1)}$. Clearly $D^{(l)}$ must be a representation of SO(3); in fact, $D^{(0)}, D^{(1)}, D^{(2)}, \ldots$ turn out to be exactly the unitary irreducible representations of SO(3). Thus, in the language of section 7.1, the vector $\widehat{\boldsymbol{f}}_l$ is $D^{(l)}$-covariant.

As before, this give rise to the concept of **power spectrum**

$$p_l = \sum_{m=-l}^{l} \left| \, \widehat{f}_{l,m} \, \right|^2 = \widehat{\boldsymbol{f}}_l^\dagger \cdot \widehat{\boldsymbol{f}}_l, \qquad l = 0, 1, 2, \ldots$$

which is invariant. Since each p_l only measures the total energy in each Fourier mode, the power spectrum by itself would not be a very good rotation invariant representation of f, however.

As in section 7.1, we remedy the problem by considering the Clebsch-Gordan decomposition, which for SO(3) takes the particularly simple form

$$D^{(l_1)}(R) \otimes D^{(l_2)}(R) = \left(C^{l_1,l_2}\right)^\dagger \left[\bigoplus_{l=|\,l_1-l_2\,|}^{l_1+l_2} D^{(l)}(R) \right] C^{l_1,l_2}. \tag{8.8}$$

It is convenient to split C^{l_1,l_2} into a collection of tall and thin matrices $C^{l_1,l_2,|\,l_1-l_2\,|}, \ldots, C^{l_1,l_2,l_1+l_2}$ collecting those columns of C^{l_1,l_2} which match up with the same l index in (8.8). With this notation, each $\left(\widehat{\boldsymbol{f}}_{l_1} \otimes \widehat{\boldsymbol{f}}_{l_2}\right)^\dagger C^{l_1,l_2,l}$ is $D^{(l)}$-contravariant, giving rise to the invariants

$$p_{l_1,l_2,l} = \left(\widehat{\boldsymbol{f}}_{l_1} \otimes \widehat{\boldsymbol{f}}_{l_2}\right)^\dagger C^{l_1,l_2,l} \, \widehat{\boldsymbol{f}}_l. \tag{8.9}$$

This is the appropriate generalization of the bispectrum to S_2, and these are the features that we will use to characterize 2D images in a rotation and translation invariant manner.

For SO(3) the Clebsch-Gordan coefficients $C^{l_1,l_2,l}_{m_1,m_2,m} = [C^{l_1,l_2}]_{(l,m),(m_1,m_2)}$ can be computed relatively easily by using recursion relations. The final form of our bispectral invariants is

$$p_{l_1,l_2,l} = \sum_{m=-l}^{l} \sum_{m_1=-l_1}^{l_1} C^{l_1,l_2,l}_{m_1,m-m_1,m} \, \widehat{f}^*_{l_1,m_1} \, \widehat{f}^*_{l_2,m-m_1} \, \widehat{f}_{l,m}, \qquad 0 \le l_1, l_2 \le L, \quad |\,l_1 - l_2\,| \le l \le l_1 + l_2. \tag{8.10}$$

In total this gives $O(L^3)$ invariants. If the original images were $n \times n$ pixels in size, Nyquist-type considerations dictate that L scale linearly with n, and lead to a computation time of $O(L^5) = O(n^5)$ per image. Of course, if the width w is very small, the constant of proportionality might be large. In our practical experiments, we had success with w values as large as 0.5, and $L \approx n$.

8.2.1 Relation to the bispectrum on SO(3)

Since the sphere can be identified with SO(3)/SO(2), the general formula for the Fourier transform of $f\colon S_2 \to \mathbb{C}$ would be

$$\widehat{f}(D^{(l)}) = \int_{R \in \mathrm{SO}(3)} f(R\, e_z) D^{(l)}(R) \, d\mu(R),$$

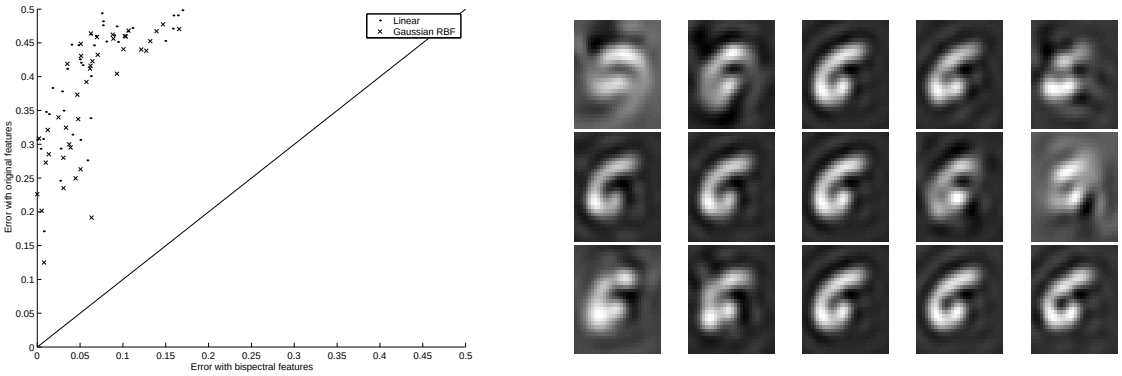


Figure 8.2: (a) Classification error for each pair of digits using the bispectral vs. baseline representation of NIST digits. (b) The first three principal components in bispectrum space of 100 images of the digit "6" mapped back to image space by BFGS. Row i shows the perturbation of the mean image by $-2, 1, 0, 1, 2$ "standard deviations" along the i'th eigenvector.

where e_z is the unit vector pointing in the positive z direction. Changing to Euler angles makes it clear that this expression factors just like (7.10), since $R\,e_z$ is insensitive to the ψ-parameter of R:

$$\widehat{f}(D^{(l)}) = \left[\frac{1}{4\pi}\int_{S_2} f(\theta,\phi)\, D^{(l)}(\theta,\phi,\psi)\, \sin\theta\, d\theta d\phi\right] \cdot \left[\frac{1}{2\pi}\int_0^{2\pi} D^{(l)}(0,0,\psi)\, d\psi\right].$$

However, the irreducible representations of SO(3) are closely related to the spherical harmonics. In particular, indexing the rows of $D^{(l)}(R)$ by $-l \le m \le l$ and its columns by $l \le m' \le l$, the most popular choice for the actual form of $D^{(l)}$ is $[D^{(l)}]_{m,m'} = e^{-il\psi} Y_l^m(\theta,\phi)$, which immediately makes it clear that

$$\left[\frac{1}{2\pi}\int_0^{2\pi} D^{(l)}(0,0,\psi)\, d\psi\right]_{m,m'} = \delta_{m',0}.$$

In other words, $\widehat{f}(D^{(l)})$ has but one non-zero column, and that column is exactly $\widehat{\boldsymbol{f}}_l$ from (8.5).

Substituting (8.8) into (7.6), the SO(3)-bispectrum formed from these matrices would be

$$\widehat{b}'(l_1,l_2) = \left(\widehat{f}(D^{(l_1)}) \otimes \widehat{f}(D^{(l_2)})\right)^\dagger\ C^{l_1,l_2} \bigoplus_{l=|\,l_1-l_2\,|}^{l_1+l_2} \widehat{f}(D^{(l)}), \qquad l_1,l_2 \in \mathbb{N}.$$

It is convenient to index the rows of these matrices by (m_1', m_2') corresponding to the rows of the two matrices in the tensor product, and to label the columns by (l, m') specifying which column of which element in the direct sum is relevant. Given this convention, when we compute (8.2.1), the sparsity pattern of the $\widehat{f}(D^{(l)})$ matrices zeros out every element of $\widehat{b}'(l_1,l)$ except $([\widehat{b}'(l_1,l)]_{(0,0),(l,0)})_{l=|\,l_1-l_2\,|}^{l_1+l_2}$. Comparing with (8.9) shows that these elements are exactly the same as the $p_{l_1,l_2,l}$ invariants that we had before.

8.3 Experiments

We conducted experiments on randomly translated and rotated versions of hand-written digits from the well known NIST dataset [LeCun and Cortes, 2007]. The original images are 28×28 pixels in size, but most of them only occupy a fraction of the image patch. The characters are rotated by a random angle between 0 and 2π, clipped, and embedded at a random position in a 30×30 patch.

The experiments consisted of comparing linear and Gaussian RBF SVMs trained on the bispectral representations of images to similar SVMs trained on the original 900-dimensional pixel intensity vectors. We used $L = 15$, which is a relatively low resolution for images of this size. The width parameter was set to $w = 1/2$. We trained separate 2-class SVMs for all $\binom{10}{2}$ digit pairings, and cross-validated the regularization parameter and the kernel width σ independently for each such task. We used 10-fold cross validation to set the parameters for the linear kernels, but to save time only 3-fold cross validation for the Gaussian kernels. Testing and training was conducted on the relevant digits from the second one thousand images in the NIST dataset. The results we report are averages of error for 10 random even splits of this data. Since there are on average 100 digits of each type amongst the 1000 images in the data, our average training set and test set consisted of just 50 digits of each class. Given that the images also suffered random translations and rotations this is an artificially difficult learning problem.

The results are shown graphically in Figure 8.2 and in tabular form in the table below. For both the linear kernels and the Gaussian kernels, the bispectral features far outperform the baseline bitmap representation. Indeed, it seems that in many cases the baseline cannot do better than what is essentially random guessing. In contrast, the bispectrum can effectively discriminate even in the hard cases such as 8 vs. 9 and reaches almost 100% accuracy on the easy cases such as 0 vs. 1. Surprisingly, to some extent the bispectrum can even discriminate between 6 and 9, which in some fonts are exact rotated versions of each other. However, in handwriting, 9's often have a straight leg and/or a protrusion at the top where right handed scribes reverse the direction of the pen.

The results make it clear that the bispectral features are able to capture position and orientation invariant characteristics of handwritten figures. Short of a handwriting-specific algorithm which extracts explicit landmarks we do not expect other methods to yield a comparable degree of position and rotation invariance.

In contrast to most other invariant features employed in vision, the richness of the bispectral representation makes it possible to reconstruct images. However, since the bispectrum consists of a collection of cubic polynomials, inverting the mapping, i.e., finding the image the bispectrum of which is closest to a given bispectrum requires gradient descent (with BFGS). The PCA results shown in Figure 8.2 are notable for two reasons: (a) the mean is a remarkably good "average" 6; (b) the eigenvectors capture much more interesting forms of variation than the Gabor wavelet-type patterns which often appear in such experiments, but are to a large extent just a artifact of translations and rotations. Reconstructibility suggests that the bispectral representation is close to complete and opens up a host of other possible applications beyond classification.

	1	2	3	4	5	6	7	8	9	
	0.77(0.41) 17.12(3.67)	6.22(2.41) 33.87(3.59)	5.09(1.54) 42.06(3.59)	5.03(1.07) 30.64(2.53)	2.90(1.53) 37.82(3.51)	4.11(2.39) 31.42(5.85)	2.73(1.11) 29.36(3.83)	4.98(1.64) 42.58(4.33)	5.86(2.88) 27.61(3.16)	0
		0.68(0.81) 30.78(2.90)	0.39(0.98) 29.34(4.50)	3.07(1.30) 34.96(3.41)	0.00(0.00) 30.66(2.85)	1.37(0.88) 34.46(4.47)	1.77(1.48) 38.32(4.05)	2.68(2.02) 24.60(2.57)	1.02(1.00) 34.78(3.57)	1
8	10.06(2.04) 44.06(3.93)		15.89(5.79) 49.06(4.18)	15.82(3.22) 47.12(4.72)	8.06(3.60) 45.20(4.26)	9.64(2.00) 51.44(5.21)	11.11(2.29) 47.20(5.54)	9.26(1.63) 47.44(6.23)	10.55(2.95) 46.70(2.95)	2
7	8.81(2.81) 46.22(4.13)	4.68(2.30) 37.33(2.21)		4.81(1.68) 44.64(3.03)	16.42(5.69) 49.07(4.81)	7.54(2.75) 49.38(5.26)	4.00(1.13) 44.74(4.42)	10.70(3.79) 50.37(4.21)	7.66(3.01) 47.60(5.55)	3
6	21.37(3.81) 50.73(4.31)	6.16(2.30) 41.19(4.47)	9.30(3.33) 40.43(5.16)		6.26(1.90) 40.08(6.67)	10.94(4.09) 50.11(5.26)	14.95(2.89) 45.30(3.30)	6.27(3.57) 46.26(2.63)	16.95(1.84) 49.82(4.68)	4
5	7.07(2.93) 43.23(2.46)	6.23(3.05) 41.63(3.29)	6.26(1.54) 46.39(3.41)	16.56(1.66) 47.04(4.21)		14.63(2.42) 50.00(4.02)	5.31(2.27) 41.70(4.09)	6.62(2.72) 44.63(3.31)	6.84(2.23) 46.01(4.37)	5
4	12.09(2.47) 52.73(3.65)	6.45(2.26) 42.29(4.44)	13.92(2.63) 46.73(6.47)	10.67(1.47) 46.82(5.32)	5.75(1.22) 39.21(4.29)		7.68(4.05) 48.19(4.10)	9.00(2.93) 46.13(5.82)	20.15(3.62) 53.75(2.69)	6
3	5.08(1.50) 44.87(3.91)	10.21(3.89) 46.00(4.97)	3.50(1.48) 41.90(4.09)	6.98(3.46) 45.86(5.27)	16.88(2.73) 52.53(3.39)	5.12(2.35) 43.07(9.05)		3.50(2.28) 41.16(5.18)	8.06(3.49) 53.21(5.01)	7
2	10.34(2.51) 45.95(4.84)	12.14(2.27) 44.02(3.14)	12.73(3.39) 43.84(4.38)	8.89(3.09) 45.63(5.49)	8.83(4.22) 50.09(4.78)	13.20(2.56) 45.26(5.11)	14.68(4.60) 47.75(3.46)		9.43(2.14) 45.13(2.87)	8
1	3.05(0.88) 28.01(4.56)	0.52(0.55) 20.16(2.93)	1.22(1.09) 32.12(6.34)	1.35(0.43) 28.52(9.47)	0.21(0.45) 30.86(9.99)	2.48(0.97) 33.98(9.44)	0.00(0.00) 22.61(8.82)	0.99(0.48) 27.29(4.00)		
0	6.34(2.57) 19.16(2.65)	3.74(2.23) 29.99(4.20)	4.48(1.39) 24.96(3.73)	3.07(1.77) 23.51(4.93)	3.90(2.25) 29.52(3.99)	3.35(1.69) 32.45(12.63)	4.78(1.08) 33.72(4.58)	5.06(1.52) 26.30(4.32)	0.80(0.42) 12.50(3.60)	
	9	8	7	6	5	4	3	2	1	

Table 8.1: Classification error in percent for each pair of digits for the linear kernels (top wedge, refer to column labels on top and row labels on right) and the Gaussian kernels (bottom wedge, refer to column labels on bottom and row labels on left). In each cell, the performance of the bispectrum-based classifier is shown on top, and the baseline on bottom; standard errors are in parentheses.

Conclusions

This thesis presented a framework for harnessing some of the power of group theory and non-commutative harmonic analysis in machine learning. Some of our main results were the following:

- the symmetrization theorem for kernels invariant under the action of a group (Theorem 4.4.3 on page 62);
- the characterization of translation invariant positive definite kernels on groups in terms of their Fourier spectra (Theorem 4.5.4 on page 65);
- the introduction of the concept of diffusion kernels on groups and their interpretation in terms of spectral graph theory (Section 5.2);
- the PerMceptron algorithm for learning permutations;
- the mathematical framework set forth in Chapter 6 for identity management in multi-object tracking, in particular, our algorithm for efficient inference based on the twisted Fourier transform of Theorem 3.3.2;
- the introduction of the skew spectrum and showing its equivalence to the bispectrum (Section 7.2);
- the new bispectral invariants of Chapter 8 for representing images in a rotation and translation invariant way.

Clearly, there is room for more work, both along the lines begun in this thesis, and in exploring other connections between learning and abstract algebra. Some of the most immediate goals might be the following:

- getting the PerMceptron to work on real problems and exploring its connection to ranking;
- extending the identity management algorithm to the case when some of the targets are exchangeable (red team vs. blue team);
- developing the distributed version of the identity management algorithm;
- giving a better description of exactly what information is lost when the bispectrum is restricted to a homogeneous space, and finding extensions of the bispectrum to minimize this loss of uniqueness;
- devising general recovery algorithms for the bispectrm/skew spectrum;
- exploring other bispectral invariants, in particular scale and homeomorphism invariant representations of images.

Between the defense of this thesis and its final submission some related publications have already appeared. In Chapter 6 we already mentioned that Huang et al. [2008] have proposed an identity

management model essentially identical to ours, but also added some new features. Regarding invariants, we have a new paper taking the skew spectrum to the discrete domain, and showing how it can be used to construct practical graph invariants [Kondor and Borgwardt, 2008].

On the longer term, my hope is that many more connections between algebra and machine learning will be discovered, increasing the intensity of interaction between the two communities.

Bibliography

Nir Ailon, Moses Charikar, and Alantha Newman. Aggregating inconsistent information: Ranking and clustering. In *Proceedings of STOC 2005*, 2005.

Yaakov Bar-Shalom and Thomas E. Formann. *Tracking and Data Association*. Academic Press, 1988.

Asim O. Barut and Ryszard Rączka. *Theory of Group Representation and Applications*. PWN Polish Scientific Publishers, 1977.

S. Bochner. *Vorlesungen uber Fouriersche Integrale*. Akademische Verlagesellschaft, 1932.

S. Bochner. Monotone funktionen, Stieljessche integrale und harmonische analyse. *Math. Ann.*, 108:378–410, 1933.

Hermann Boerner. *Representations of Groups*. North-Holland, 1970.

Peter Bürgisser. The computational complexity to evaluate representations of general linear groups. *SIAM J. Comput.*, 30(3):1010–1022, 2000.

Peter Bürgisser, Michael Clausen, and M. Amin Shokrollahi. *Algebraic Complexity Theory*. Springer, 1997.

M. Clausen. Fast generalized Fourier transforms. *Theor. Comput. Sci.*, pages 55–63, 1989.

Michael Clausen and Ulrich Baum. *Fast Fourier Transforms*. Wissenschaftsverlag, 1993.

Stéphan Clemençon, Gábor Lugosi, and Nicolas Vayatis. Ranking and scoring using empirical risk minimization. In *Proc. 18th Annual Conference on Computational Learning Theory*, 2005.

J. W. Cooley and J. W. Tukey. An algorithm for machine calculation of complex Fourier series. *Math. Comp.*, 19:297–301, 1965.

K. Crammer and Y. Singer. Pranking with ranking. In *Advances in Neural Information Processing Systems NIPS'2002*, volume 13. MIT Press, 2002.

K. Crammer and Y. Singer. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research (JMLR)*, 3:951–991, 2003.

Douglas E. Critchlow. *Metric Methods for Analyzing Partially Ranked Data*, volume 34 of *Lecture Notes in Statistics*. Springer-Verlag, 1985.

P. Diaconis. *Group Representation in Probability and Statistics*, volume 11 of *IMS Lecture Series*. Institute of Mathematical Statistics, 1988.

Y. Freund and R. E. Schapire. Large margin classification using the perceptron algorithm. *Machine Learning*, 37(3):277 – 296, 1999.

Yoav Freund, Raj Iyer, Robert E. Schapire, and Yoram Singer. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 4:933–969, 2003.

William Fulton. *Young Tableaux*. Student Texts. London Mathematical Society, 1997.

William Fulton and Joe Harris. *Representation Theory*. Graduate texts in mathematics. Springer Verlag, 1991.

C. F. Gauss. Theoria interpolationis methodo nova tracta. In *Gauss' Collected Works*, volume 3, pages 265–303. Royal Society of Göttingen, 1886.

F. Girosi, M. Jones, and T. Poggio. Regularization theory and neural networks architectures. *Neural Computation*, 7(2):219 – 269, 1995.

Dennis M. Healy, Daniel N. Rockmore, and Sean S. B. Moore. FFTs for the 2-sphere – improvements and variations. Technical Report PCS-TR96-292, Department of Computer Science, Dartmouth College, 1996.

Jonathan Huang, Carlos Guestrin, and Leonidas Guibas. Efficient inference for distributions on permutations. In *Advances in Neural Information Processing Systems NIPS'2007*, 2008.

G. D. James. *The Representation Theory of the Symmetric Groups*. Springer, 1978.

Gordon James and Adalbert Kerber. *The Representation Theory of the Symmetric Group*. Addison-Wesley, 1981.

Mark Jerrum, Alistair Sinclair, and Eric Vigoda. A polynomial-time approximation algorithm for the permanent of a matrix with non-negative entries. In *ACM Symposium on Theory of Computing*, pages 712–721, 2001.

Ramakrishna Kakarala. *Triple correlation on groups*. PhD thesis, Department of Mathematics, UC Irvine, 1992.

Ramakrishna Kakarala. A group theoretic approach to the triple correlation. In *IEEE Workshop on higher order statistics*, pages 28–32, 1993.

Adalbert Kerber. *Representations of Permutation Groups I-II*. Springer, 1971.

G. S. Kimeldorf and G. Wahba. Some results on Tchebycheffian spline functions. *J. Math. Anal. Applic.*, 33:82–95, 1971.

Risi Kondor. The skew spectrum of functions on groups and their homogeneous spaces, 2007a. `http://arXiv.org/abs/0712.4259`.

Risi Kondor. A novel set of rotationally and translationally invariant features for images based on the non-commutative bispectrum, 2007b. `http://arxiv.org/abs/cs.CV/0701127`.

Risi Kondor and Karsten M. Borgwardt. The skew spectrum of graphs. In *Proceedings of the International Conference on Machine Learning*, 2008.

Risi Kondor and John Lafferty. Diffusion kernels on graphs and other discrete structures. In *Proceedings of the International Conference on Machine Learning*, 2002.

Risi Kondor, Andrew Howard, and Tony Jebara. Multi-object tracking with representations of the symmetric group. In *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, 2007.

Thomas W. Körner. *Fourier Analysis*. Cambridge University Press, 1988.

M. Krein. Hermitian-positive definite kernels on homogeneous spaces I and II (in translation). *Amer. Math. Soc. Translations*, 34:69–164, 1950.

Serge Lang. *Algebra*. Addison-Wesley, 1984.

G. Lebanon and J. Lafferty. Cranking: Combining rankings using conditional probability models on permutations. In *Proceedings of the International Conference on Machine Learning*, 2002.

Yann LeCun and Corinna Cortes. The NIST dataset, 2007. `http://yann.lecun.com/db/mnist/`.

David J. C. Mackay. Gaussian processes: A replacement for neural networks? *Tutorial at the Tenth Annual Conference on Neural Information Processing Systems*, 1997. Available from `http://wol.ra.phy.cam.ac.uk/pub/mackay/`.

D. Maslen and D. Rockmore. Generalized FFTs – a survey of some recent results. In *Groups and Computation II*, volume 28 of *DIMACS Ser. Discrete Math. Theor. Comput. Sci.*, pages 183–287. AMS, Providence, RI, 1997.

David K. Maslen. The efficient computation of Fourier transforms on the symmetric group. *Mathematics of Computation*, 67(223):1121–1147, 1998.

David K. Maslen and Daniel N. Rockmore. Double coset decompositions and computational harmonic analysis on groups. *Journal of Fourier Analysis and Applications*, 6(4), 2000.

M. Mathias. Über positive Fourier-integrale. *Math Zeit*, 16:103–125, 1923.

A. Povzner. Über positive funktionen auf einer Abelsche gruppe. *Dokl. Akad. Nauk. SSSR*, 28: 294–295, 1940.

D. A. Raikov. Positive definite functions on commutative groups with an invariant measure. *Dokl. Akad. Nauk. SSSR*, 28:296–300, 1940.

F. Riesz. Über sätze von Stone und Bochner. *Acta Sci. Math.*, 6:184–198, 1933.

Daniel N. Rockmore. Some applications of generalized FFTs. *Proceedings of the DIMACS workshop on groups and computation*, 1997.

Cynthia Rudin, Corinna Cortes, Mehryar Mohri, and Robert E. Schapire. Margin-based ranking meets boosting in the middle. In *Proc. 18th Annual Conference on Computational Learning Theory*, 2005.

Walter Rudin. *Fourier Analysis on Groups*. Interscience Publishers, 1962.

H. J. Ryser. *Combinatorial Mathematics*, volume 14 of *Carus Math. Monog.* Math. Assoc. America, New York, 1963.

Bruce E. Sagan. *The Symmetric Group*. Graduate Texts in Mathematics. Springer, 2001.

Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels*. MIT Press, 2002.

Brad Schumitsch, Sebastian Thrun, Gary R. Bradski, and Kunle Olukotun. The information-form data association filter. In *Advances in Neural Information Processing Systems (NIPS)*, 2005.

Jean-Pierre Serre. *Linear Representations of Finite Groups*, volume 42 of *Graduate Texts in Mathamatics*. Springer-Verlag, 1977.

Jaewon Shin, Nelso Lee, Sebastian Thrun, and Leonidas Gubias. Lazy inference on object identities in wireless sensor networks. Technical report, Stanford AI laboratory, submitted for publication, 2006.

Alexander J. Smola and Risi Kondor. Kernels and regularization on graphs. In Manfred Warmuth and Bernhard Schökopf, editors, *Proceedings of the Conference on Learning Theory and Kernels Workshop*, 2003.

Lawrence Stone, Carl Barlow, and Thomas Corwin. *Bayesian Multiple Target Tracking*. Artech House, 2000.

Audrey Terras. *Fourier analysis on finite groups and applications*, volume 43 of *London Mathematical Society Student Texts*. Cambridge Univ. Press, 1999.

Audrey Terras. *Harmonic Analysis on Symmetric Spaces and Applications I-II*. Springer-Verlag, 1985.

Vladimir Vapnik. *The nature of statsitical learning theory*. Springer, 1995.

A. Weil. L'integration dans le groupes topologiques et ses applications. *Actualités Sci. Indust.*, 869, 1940.

Hermann Weyl. *The Classical Groups*. Princeton University Press, 1946.

F. Yates. The design and analysis of factorial experiments. *Imp. Bur. Soil Sci. Tech. Comm.*, 35, 1937.

A. Yuille and N. Grzywacz. The motion coherence theory. In *Proceedings of the International Conference on Computer Vision*, pages 344 – 354, Washington, D.C., December 1988. IEEE Computer Society Press.

List of Symbols

A_n, alternating group, **20**
$[A, B]$, Lie bracket, **15**
$\mathrm{Aut}(G)$, automorphism group, **12**
$\widehat{a}_f(\rho)$, power spectrum, **82**
a_f, autocorrelation, **82**

$\widehat{b}_f$, bispectrum, **83**
b_f, triple correlation, **83**

$\mathbb{C}^*$, $\mathbb{C} \setminus \{0\}$, **11**
C^{l_1,l_2}, Clebsch-Gordan matrices for SO(3), **94**
C_t, column stabilizer, **22**
$\mathbb{C}$, the complex numbers, **13**

$D^{(l)}$, irreducible representations of $SO(3)$, **94**
$\mathcal{D}$, true distribution of examples, **50**

$\mathcal{E}_{\mathcal{D}}[f]$, true error, **50**
$\mathcal{E}_{\mathrm{emp}}[f]$, empirical error, **50**
$\mathcal{E}_{\mathrm{gen}}[f]$, generalization error, **50**
$\boldsymbol{e}_{\mathbf{t}}$, polytabloid, **22**

$\mathbb{F}[G]$, group algebra, **13**
$\mathbb{F}$, field, **13**
$\mathcal{F}$, Fourier transform, **29**
$\mathcal{F}$, hypothesis space, **50**
$\langle f|$, Dirac bra, **56**
$\widehat{f}(\lambda)$, Fourier components on $\mathbb{S}_n$, **41**
$\widehat{f}(k)$, Fourier coefficient or component, **29**
$|f\rangle$, Dirac ket, **56**
$f * g$, convolution, **29**
f^t, left-translate of f, **14**
$f^{(t)}$, right-translate of f, **14**
f^-, reflected function, **82**

G/H, left quotient space, **11**
$G \cong G'$, isomorphism, **9**
$G \times H$, direct product, **12**
$G \rtimes H$, semi-direct product, **12**
$|G|$, order of G, **10**
$\widehat{G}$, dual group, **17**
$\mathrm{GL}(V)$, general linear group, **10**
GL_n, $\equiv \mathrm{GL}(\mathbb{C}^n)$, **25**

$H\backslash G$, right quotient space, **11**
$H_1\backslash G/H_2$, double coset, **12**
Hx, right coset, **11**

Im_λ, the λ-immanant, **71**
$[\![i, n]\!]$, contiguous cycle, **41**
$\mathrm{ISO}^+(n)$, rigid body motions group, **12**
$\mathrm{ISO}^+_\epsilon(2)$, subset of $\mathrm{ISO}^+(2)$, **91**

$\mathcal{K}$, kernel operator, **54**

$L(\widehat{y}, y)$, loss function, **50**
$\mathcal{L}$, Lie algebra, **15**
$\ell(\widehat{y}, y)$, loss function, **55**
$\lambda' \leq \lambda$, inclusion order on partitions, **24**
λ/λ', skew shape, **24**
$\lambda \vdash n$, integer partition, **19**

M^λ, permutation module, **22**

$\nabla_T f$, directional derivative, **15**

$\mathbb{Q}$, the rational numbers, **13**

$R(\lambda)$, branching on $\mathbb{S}_n$, **40**
R_t, row stabilizer, **22**
$\mathbb{R}^n$, Euclidean space, **11**
$\mathbb{R}$, the real numbers, **13**
$\rho(x)$, representation, **16**
$\rho\downarrow^G_H$, restricted representation, **18**
$\rho\uparrow^G_H$, induced representation, **18**

S^λ, Specht module, **22**

G, **9**
S_n, the n-dimensional unit sphere, **14**
$\mathrm{SO}(n)$, special orthogonal group, **11**
$\mathbb{S}_{\lambda_1} \times \mathbb{S}_{\lambda_2} \times \ldots \times \mathbb{S}_{\lambda_k}$, Young subgroup, **22**
$\mathbb{S}_n$, symmetric group, **19**
$\mathrm{sgn}(\sigma)$, sign of permutation, **20**

$\boldsymbol{\{t\}}$, tabloid vector in $\mathbb{C}[\mathbb{S}_n]$, **22**
$\{t\}$, tabloid, **21**
τ_k, adjacent transposition, **19**
$\mathbb{T}$, circle group, **11**

Υ, regularization operator, **54**
$\langle u, w \rangle$, inner product, **51**
$\| u \|$, norm of u, **51**

$\mathcal{X}$, input space, **49**
xH, left coset, **11**
$x_1 H x_2$, two-sided coset, **12**

Y_l^m, spherical harmonics, **93**
$\mathcal{Y}$, output space, **49**

$\mathbb{Z}_n$, cyclic group, **10**

Index

Abelian, **12**
action, **15**
    transitive, **15**
algebra, **14**
    irreducible, **15**
    semi-simple, 31
alternating group, **21**
associativity, **10**
autocorrelation, **83**
automorphism, **13**

Baker-Campbell-Hausdorff formula, **16**
band-limited functions
    on $\mathbb{S}_n$, **45**
bispectrum
    on S_2, 95
    classical, **83**
    on groups, **85**
Bratelli diagram, **41**

capacity control, **51**
Cauchy sequence, **52**
Cayley graph, **69**
character, **18**
    irreducible, **18**
chart, **15**
circle group, **12**
circulant matrix, **69**
class functions, **12**
classification, **51**
Clausen's algorithm, **40**
Clebsch-Gordan
    coefficients, 95
    decomposition, **85**, 95
    matrices, **85**
column stabilizer, **23**
commutativity, **12**
commutator, **16**
complete reducibility, theorem, **17**
conjugacy, **12**
conjugacy classes, **12**
convolution, **30**, 31
    theorem, **30**
Cooley-Tukey algorithm, **37**
coset
    double, **13**, 89
    left, **12**
    representatives, **12**
    right, **12**
    two-sided, **13**
coset function, **88**
cranking, **73**
cycle, **20**
    contiguous, **42**
    notation, 20
cycle-type, **20**

determinant, 72
DFT, *see* Fourier transform, on $\mathbb{Z}_n$
diffusion kernel, **69**
direct product, **13**
dual group, **18**
dual space, **18**

empirical risk minimization, **51**
ERM, *see* empirical risk minimization
error
    generalization, **51**
    training, **51**
    true, **51**
Euler angles, **92**
evaluation map, **54**

factorial design, **33**
fast Fourier transform, **36**

on $\mathbb{S}_n$ (Clausen), **40**
on $\mathbb{Z}_n$ (Cooley-Tukey), **37**
on $\mathbb{Z}_n$ (Rader), **37**
Ferres diagram, **21**
field, **14**
Fourier series, **30**
Fourier transform, **30**
discrete, **30**, 37
generalized, 31
on $\mathbb{R}$, **30**
on $\mathbb{T}$, **30**
on $\mathbb{Z}_n$, **30**
on Abelian groups, **31**
on homogeneous space, **88**
twisted, 46
frame, **21**

G-module, **14**
Gaussian process, **56**
Gel'fand-Tsetlin basis, **39**
general linear group, **11**
generalization, **50**
generator, **13**
Gram matrix, **56**
group, **10**
Abelian, **12**, 18
action, **15**
transitive, **15**
algebra, **14**, 31
alternating, **21**
automorphism, **13**
commutative, **12**
compact, **11**
countable, **11**
cyclic, **11**
finite, **11**
isometry, **13**
Lie, *see* Lie, group
linear, **11**, 16
locally compact, **11**
locally compact Abelian, 19
order of, **11**
symmetric, *see* symmetric group
topological, **11**

Haar measure, **15**, 31
harmonic analysis, *see* Fourier transform
Hilbert space, **52**
homogeneous space, **15**, 88
homomorphism, **12**
hook, **24**
hypothesis, **50**

identity element, **10**
immanant, **72**
inner product, **52**
input space, **50**
invariant
kernel, **61**
invariants, 82
complete, **82**
inverse, **10**
$\mathrm{ISO}^+(2)$, 91
isomorphism, **10**, 12
isotypal, 31

kernel
bi-invariant, **63**
conditionally positive definite, **53**
Gaussian, **55**
invariant, **61**
left-invariant, **63**
negative definite, **53**
operator, **55**
positive (semi-)definite, **53**
reproducing, **53**
right-invariant, **63**

Lagrange's theorem, **12**
Laplacian
on S_2, **94**
on graphs, 69
large margin algorithms, **60**
LCA groups, *see* locally compact Abelian group, 19
Lie
algebra, **16**
bracket, **16**
group, 11, **15**
locally compact Abelian group, **12**
loss
function, 51

hinge, 59
squared error, **51**, 56
zero-one, **51**

Maschke's theorem, **17**
Murnaghan-Nakayama rule, **25**

neural networks, 50
norm, **52**

orbit, **15**, 88
output space, **50**
overfitting, **50**

Parseval's theorem, **30**
partition
length of, **20**
of integers, **20**
perceptron, **73**
permanent, 72
perMceptron, **73**
permutation, **20**
even, **21**
matrix, **21**
module, **23**
odd, **21**
representation, **19**
sign of, **21**
permutation learning, **66**
Plancherel's theorem, **30**
polytabloid, **23**
positive definite
function, **63**
kernel, *see* kernel
power spectrum, **83**
on S_2, **95**
pranking, **73**

quotient space, **12**

Rader's algorithm, **37**
ranking, **66**
reflection
of function, **83**
regression, **51**
regularization
operator, **55**, 69
parameter, **52**
regularizer, **52**
representation, **17**
adapted, **39**, 89
equivalence of, **17**
induced, **19**
infinite dimensional, **19**
irreducible, **17**
order of, **17**
permutation, **19**
reducible, **17**
regular, **18**, 19
restriction of, **19**
trivial, **18**
unitary, 18
representer theorem, **54**
reproducing kernel Hilbert space, **53**
ridge regression, **59**
rigid body motions group, **13**, 91
risk
empirical, **51**
regularized, **52**
true, **51**
RKHS, *see* reproducing kernel Hilbert space
row stabilizer, **23**

S_2, 15, 91
semi-direct product, **13**
shape, **21**
skew shape, **25**
skew spectrum, **87**, 89
$\mathbb{S}_n$`ob` , **46**
SO(3), **12**, 15, 16, 91
SO(n), **12**
Specht module, **23**
spherical harmonics, **94**
subalgebra, **15**
subgroup, **12**
conjugate, **12**
isotropy, **15**
normal, **12**
proper, **12**
support vector machine, **59**
symmetric group, 11, **20**
characters of, **25**
degree of, **20**

representations of, **21**
alternating, **21**
defining, **21**
permutation, **22**
standard, **24**
Young's orthogonal (YOR), **24**, 40

tableau, **22**
border strip, **25, 26**
hook, **24**
semi-standard, **28**
standard, **24**
Young, **22**
tabloid, **22**
tangent space, **16**
tangent vector, **16**
testing set, **50**
topological space
compact, **11**
training set, **50**
translation
of functions, 15
on $\mathbb{R}$, $\mathbb{T}$ and $\mathbb{Z}_n$, **30**
property of Fourier transform, **30**
transposition, **20**
adjacent, **20**
transversal, **12**
triple correlation
classical, 84
on groups, 86
two-line notation, **20**

vector field, **16**
vector space, **14**
Viergruppe, **11**

Wedderburn's theorem, **17**
Weyl module, **27**

YOR, *see* symmetric group, repr.'s of, orthogonal
Young subgroup, **23**
Young's
orthogonal representation, *see* symmetric group, repr.'s of, orthogonal
rule, **41**